

The Open Toolkit library

1.0

Generated by Doxygen 1.7.1

Wed Oct 6 2010 19:27:17

Contents

1	Namespace Index	1
1.1	Package List	1
2	Class Index	3
2.1	Class Hierarchy	3
3	Class Index	7
3.1	Class List	7
4	Namespace Documentation	13
4.1	Package OpenTK.Audio	13
4.2	Package OpenTK.Audio.OpenAL	14
4.2.1	Enumeration Type Documentation	20
4.2.1.1	ALBufferState	20
4.2.1.2	ALCapability	20
4.2.1.3	AlcContextAttributes	21
4.2.1.4	AlcError	21
4.2.1.5	AlcGetInteger	21
4.2.1.6	AlcGetString	22
4.2.1.7	AlcGetStringList	23
4.2.1.8	ALDistanceModel	23
4.2.1.9	ALError	23
4.2.1.10	ALFormat	24
4.2.1.11	ALGetBufferi	25

4.2.1.12	ALGetFloat	26
4.2.1.13	ALGetInteger	26
4.2.1.14	ALGetSourceci	26
4.2.1.15	ALGetString	27
4.2.1.16	ALListener3f	27
4.2.1.17	ALListenerf	27
4.2.1.18	ALListenerfv	28
4.2.1.19	ALSource3f	28
4.2.1.20	ALSource3i	28
4.2.1.21	ALSourceb	29
4.2.1.22	ALSourcef	29
4.2.1.23	ALSourceci	30
4.2.1.24	ALSourceState	31
4.2.1.25	ALSourceType	31
4.2.1.26	EfxAuxiliaryf	31
4.2.1.27	EfxAuxiliaryi	31
4.2.1.28	EfxEffect3f	32
4.2.1.29	EfxEffectf	32
4.2.1.30	EfxEffecti	38
4.2.1.31	EfxEffectType	40
4.2.1.32	EfxFilterf	41
4.2.1.33	EfxFilteri	41
4.2.1.34	EfxFilterType	41
4.2.1.35	EfxFormantFilterSettings	42
4.3	Package OpenTK.Compute	43
4.4	Package OpenTK.Graphics	43
4.4.1	Enumeration Type Documentation	44
4.4.1.1	GraphicsContextFlags	44
4.5	Package OpenTK.Graphics.ES10	45
4.6	Package OpenTK.Graphics.ES11	52
4.7	Package OpenTK.Graphics.ES20	68

4.8	Package OpenTK.Graphics.OpenGL	85
4.9	Package OpenTK.Input	244
4.9.1	Enumeration Type Documentation	247
4.9.1.1	InputDeviceType	247
4.9.1.2	JoystickAxis	248
4.9.1.3	JoystickButton	248
4.9.1.4	Key	249
4.9.1.5	MouseButton	253
4.10	Package OpenTK.Platform	254
4.11	Package OpenTK.Platform.Dummy	254
4.12	Package OpenTK.Platform.Egl	254
4.13	Package OpenTK.Platform.MacOS	254
4.14	Package OpenTK.Platform.MacOS.Carbon	255
4.15	Package OpenTK.Platform.Windows	258
4.15.1	Enumeration Type Documentation	275
4.15.1.1	GdiCharset	275
4.15.1.2	GWL	275
4.15.1.3	MapVirtualKeyType	275
4.15.1.4	MouseKeys	275
4.15.1.5	QueueStatusFlags	276
4.15.1.6	RawInputDeviceFlags	276
4.15.1.7	RawMouseFlags	277
4.15.1.8	SetWindowPosFlags	277
4.15.1.9	ShGetFileIconFlags	278
4.15.1.10	ShowWindowCommand	279
4.15.1.11	ShowWindowMessageIdentifiers	280
4.15.1.12	WindowMessage	280
4.16	Package OpenTK.Platform.X11	280
4.16.1	Enumeration Type Documentation	294
4.16.1.1	XKey	294
4.17	Package OpenTK.Properties	294

5	Class Documentation	295
5.1	OpenTK.Audio.AudioCapture Class Reference	295
5.1.1	Detailed Description	296
5.1.2	Constructor & Destructor Documentation	297
5.1.2.1	AudioCapture	297
5.1.2.2	AudioCapture	297
5.1.3	Member Function Documentation	297
5.1.3.1	CheckErrors	297
5.1.3.2	Dispose	297
5.1.3.3	ReadSamples	297
5.1.3.4	ReadSamples< TBuffer >	298
5.1.3.5	Start	298
5.1.3.6	Stop	298
5.1.4	Property Documentation	298
5.1.4.1	AvailableDevices	298
5.1.4.2	AvailableSamples	299
5.1.4.3	CurrentDevice	299
5.1.4.4	CurrentError	299
5.1.4.5	DefaultDevice	299
5.1.4.6	IsRunning	299
5.1.4.7	SampleFormat	299
5.1.4.8	SampleFrequency	299
5.2	OpenTK.Audio.AudioContext Class Reference	299
5.2.1	Detailed Description	302
5.2.2	Member Enumeration Documentation	302
5.2.2.1	MaxAuxiliarySends	302
5.2.3	Constructor & Destructor Documentation	302
5.2.3.1	AudioContext	302
5.2.3.2	AudioContext	302
5.2.3.3	AudioContext	302
5.2.3.4	AudioContext	303

5.2.3.5	AudioContext	303
5.2.3.6	AudioContext	303
5.2.3.7	AudioContext	304
5.2.4	Member Function Documentation	305
5.2.4.1	CheckErrors	305
5.2.4.2	Dispose	305
5.2.4.3	Equals	305
5.2.4.4	GetHashCode	306
5.2.4.5	MakeCurrent	306
5.2.4.6	Process	306
5.2.4.7	SupportsExtension	306
5.2.4.8	Suspend	307
5.2.4.9	ToString	307
5.2.5	Property Documentation	307
5.2.5.1	AvailableDevices	307
5.2.5.2	CurrentContext	308
5.2.5.3	CurrentDevice	308
5.2.5.4	CurrentError	308
5.2.5.5	DefaultDevice	308
5.2.5.6	IsProcessing	308
5.2.5.7	IsSynchronized	308
5.3	OpenTK.Audio.AudioContextException Class Reference	308
5.3.1	Detailed Description	309
5.3.2	Constructor & Destructor Documentation	309
5.3.2.1	AudioContextException	309
5.3.2.2	AudioContextException	309
5.4	OpenTK.Audio.AudioDeviceException Class Reference	309
5.4.1	Detailed Description	310
5.4.2	Constructor & Destructor Documentation	310
5.4.2.1	AudioDeviceException	310
5.4.2.2	AudioDeviceException	310

5.5	OpenTK.Audio.AudioException Class Reference	310
5.5.1	Detailed Description	311
5.5.2	Constructor & Destructor Documentation	311
5.5.2.1	AudioException	311
5.5.2.2	AudioException	311
5.6	OpenTK.Audio.AudioValueException Class Reference	311
5.6.1	Detailed Description	312
5.6.2	Constructor & Destructor Documentation	312
5.6.2.1	AudioValueException	312
5.6.2.2	AudioValueException	312
5.7	OpenTK.Audio.OpenAL.EffectsExtension Class Reference	312
5.7.1	Detailed Description	319
5.7.2	Constructor & Destructor Documentation	319
5.7.2.1	EffectsExtension	319
5.7.3	Member Function Documentation	319
5.7.3.1	AuxiliaryEffectSlot	319
5.7.3.2	AuxiliaryEffectSlot	319
5.7.3.3	AuxiliaryEffectSlot	320
5.7.3.4	AuxiliaryEffectSlot	320
5.7.3.5	BindEffect	320
5.7.3.6	BindEffect	321
5.7.3.7	BindEffectToAuxiliarySlot	321
5.7.3.8	BindEffectToAuxiliarySlot	321
5.7.3.9	BindFilterToSource	321
5.7.3.10	BindFilterToSource	322
5.7.3.11	BindSourceToAuxiliarySlot	322
5.7.3.12	BindSourceToAuxiliarySlot	322
5.7.3.13	DeleteAuxiliaryEffectSlot	323
5.7.3.14	DeleteAuxiliaryEffectSlot	323
5.7.3.15	DeleteAuxiliaryEffectSlots	323
5.7.3.16	DeleteAuxiliaryEffectSlots	323

5.7.3.17	DeleteAuxiliaryEffectSlots	324
5.7.3.18	DeleteAuxiliaryEffectSlots	324
5.7.3.19	DeleteEffect	324
5.7.3.20	DeleteEffect	324
5.7.3.21	DeleteEffects	325
5.7.3.22	DeleteEffects	325
5.7.3.23	DeleteEffects	325
5.7.3.24	DeleteEffects	325
5.7.3.25	DeleteFilter	326
5.7.3.26	DeleteFilter	326
5.7.3.27	DeleteFilters	326
5.7.3.28	DeleteFilters	326
5.7.3.29	DeleteFilters	327
5.7.3.30	DeleteFilters	327
5.7.3.31	Effect	327
5.7.3.32	Effect	327
5.7.3.33	Effect	328
5.7.3.34	Effect	328
5.7.3.35	Effect	328
5.7.3.36	Effect	328
5.7.3.37	Filter	329
5.7.3.38	Filter	329
5.7.3.39	Filter	329
5.7.3.40	Filter	329
5.7.3.41	GenAuxiliaryEffectSlot	330
5.7.3.42	GenAuxiliaryEffectSlot	330
5.7.3.43	GenAuxiliaryEffectSlots	330
5.7.3.44	GenAuxiliaryEffectSlots	331
5.7.3.45	GenAuxiliaryEffectSlots	331
5.7.3.46	GenEffect	331
5.7.3.47	GenEffect	332

5.7.3.48	GenEffects	332
5.7.3.49	GenEffects	332
5.7.3.50	GenEffects	333
5.7.3.51	GenFilter	333
5.7.3.52	GenFilter	333
5.7.3.53	GenFilters	333
5.7.3.54	GenFilters	334
5.7.3.55	GenFilters	334
5.7.3.56	GetAuxiliaryEffectSlot	335
5.7.3.57	GetAuxiliaryEffectSlot	335
5.7.3.58	GetAuxiliaryEffectSlot	335
5.7.3.59	GetAuxiliaryEffectSlot	335
5.7.3.60	GetEffect	336
5.7.3.61	GetEffect	336
5.7.3.62	GetEffect	336
5.7.3.63	GetEffect	336
5.7.3.64	GetEffect	337
5.7.3.65	GetEffect	337
5.7.3.66	GetFilter	337
5.7.3.67	GetFilter	337
5.7.3.68	GetFilter	338
5.7.3.69	GetFilter	338
5.7.3.70	IsAuxiliaryEffectSlot	338
5.7.3.71	IsAuxiliaryEffectSlot	338
5.7.3.72	IsEffect	339
5.7.3.73	IsEffect	339
5.7.3.74	IsFilter	339
5.7.3.75	IsFilter	340
5.7.4	Property Documentation	340
5.7.4.1	IsInitialized	340
5.8	OpenTK.Audio.OpenAL.XRamExtension Class Reference	340

5.8.1	Detailed Description	341
5.8.2	Member Enumeration Documentation	341
5.8.2.1	XRamStorage	341
5.8.3	Constructor & Destructor Documentation	342
5.8.3.1	XRamExtension	342
5.8.4	Member Function Documentation	342
5.8.4.1	GetBufferMode	342
5.8.4.2	GetBufferMode	342
5.8.4.3	SetBufferMode	342
5.8.4.4	SetBufferMode	343
5.8.5	Property Documentation	343
5.8.5.1	GetRamFree	343
5.8.5.2	GetRamSize	343
5.8.5.3	IsInitialized	343
5.9	OpenTK.AutoGeneratedAttribute Class Reference	344
5.9.1	Detailed Description	344
5.9.2	Constructor & Destructor Documentation	344
5.9.2.1	AutoGeneratedAttribute	344
5.9.3	Member Data Documentation	344
5.9.3.1	Category	344
5.9.3.2	EntryPoint	345
5.9.3.3	Version	345
5.10	OpenTK.BezierCurve Struct Reference	345
5.10.1	Detailed Description	346
5.10.2	Constructor & Destructor Documentation	346
5.10.2.1	BezierCurve	346
5.10.2.2	BezierCurve	346
5.10.2.3	BezierCurve	347
5.10.2.4	BezierCurve	347
5.10.3	Member Function Documentation	347
5.10.3.1	CalculateLength	347

5.10.3.2	CalculateLength	347
5.10.3.3	CalculateLength	348
5.10.3.4	CalculatePoint	348
5.10.3.5	CalculatePoint	348
5.10.3.6	CalculatePoint	349
5.10.4	Member Data Documentation	349
5.10.4.1	Parallel	349
5.10.5	Property Documentation	349
5.10.5.1	Points	349
5.11	OpenTK.BezierCurveCubic Struct Reference	350
5.11.1	Detailed Description	350
5.11.2	Constructor & Destructor Documentation	351
5.11.2.1	BezierCurveCubic	351
5.11.2.2	BezierCurveCubic	351
5.11.3	Member Function Documentation	351
5.11.3.1	CalculateLength	351
5.11.3.2	CalculatePoint	352
5.11.4	Member Data Documentation	352
5.11.4.1	EndAnchor	352
5.11.4.2	FirstControlPoint	352
5.11.4.3	Parallel	352
5.11.4.4	SecondControlPoint	352
5.11.4.5	StartAnchor	352
5.12	OpenTK.BezierCurveQuadric Struct Reference	352
5.12.1	Detailed Description	353
5.12.2	Constructor & Destructor Documentation	353
5.12.2.1	BezierCurveQuadric	353
5.12.2.2	BezierCurveQuadric	354
5.12.3	Member Function Documentation	354
5.12.3.1	CalculateLength	354
5.12.3.2	CalculatePoint	354

5.12.4	Member Data Documentation	355
5.12.4.1	ControlPoint	355
5.12.4.2	EndAnchor	355
5.12.4.3	Parallel	355
5.12.4.4	StartAnchor	355
5.13	OpenTK.BindingsBase Class Reference	355
5.13.1	Detailed Description	356
5.13.2	Constructor & Destructor Documentation	356
5.13.2.1	BindingsBase	356
5.13.3	Member Function Documentation	357
5.13.3.1	GetAddress	357
5.13.4	Member Data Documentation	357
5.13.4.1	CoreClass	357
5.13.4.2	CoreFunctionMap	357
5.13.4.3	DelegatesClass	357
5.13.5	Property Documentation	357
5.13.5.1	RebuildExtensionList	357
5.13.5.2	SyncRoot	358
5.14	OpenTK.Box2 Struct Reference	358
5.14.1	Detailed Description	359
5.14.2	Constructor & Destructor Documentation	359
5.14.2.1	Box2	359
5.14.2.2	Box2	359
5.14.3	Member Function Documentation	360
5.14.3.1	FromTLRB	360
5.14.3.2	ToString	360
5.14.4	Member Data Documentation	360
5.14.4.1	Bottom	360
5.14.4.2	Left	360
5.14.4.3	Right	360
5.14.4.4	Top	360

5.14.5	Property Documentation	361
5.14.5.1	Height	361
5.14.5.2	Width	361
5.15	OpenTK.ContextExistsException Class Reference	361
5.15.1	Detailed Description	361
5.15.2	Constructor & Destructor Documentation	361
5.15.2.1	ContextExistsException	361
5.15.3	Property Documentation	362
5.15.3.1	Message	362
5.16	OpenTK.ContextHandle Struct Reference	362
5.16.1	Detailed Description	363
5.16.2	Constructor & Destructor Documentation	363
5.16.2.1	ContextHandle	363
5.16.3	Member Function Documentation	364
5.16.3.1	CompareTo	364
5.16.3.2	Equals	364
5.16.3.3	Equals	364
5.16.3.4	GetHashCode	364
5.16.3.5	operator ContextHandle	365
5.16.3.6	operator IntPtr	365
5.16.3.7	operator !=	365
5.16.3.8	operator ==	365
5.16.3.9	ToString	366
5.16.4	Member Data Documentation	366
5.16.4.1	Zero	366
5.16.5	Property Documentation	366
5.16.5.1	Handle	366
5.17	OpenTK.DisplayDevice Class Reference	366
5.17.1	Detailed Description	368
5.17.2	Member Function Documentation	368
5.17.2.1	ChangeResolution	368

5.17.2.2	ChangeResolution	368
5.17.2.3	RestoreResolution	369
5.17.2.4	SelectResolution	369
5.17.2.5	ToString	369
5.17.3	Property Documentation	370
5.17.3.1	AvailableDisplays	370
5.17.3.2	AvailableResolutions	370
5.17.3.3	BitsPerPixel	370
5.17.3.4	Bounds	370
5.17.3.5	Default	370
5.17.3.6	Height	370
5.17.3.7	IsPrimary	370
5.17.3.8	RefreshRate	370
5.17.3.9	Width	371
5.18	OpenTK.DisplayResolution Class Reference	371
5.18.1	Detailed Description	372
5.18.2	Member Function Documentation	372
5.18.2.1	Equals	372
5.18.2.2	GetHashCode	372
5.18.2.3	operator!=	372
5.18.2.4	operator==	373
5.18.2.5	ToString	373
5.18.3	Property Documentation	373
5.18.3.1	BitsPerPixel	373
5.18.3.2	Bounds	373
5.18.3.3	Height	373
5.18.3.4	RefreshRate	374
5.18.3.5	Width	374
5.19	OpenTK.FrameEventArgs Class Reference	374
5.19.1	Detailed Description	374
5.19.2	Constructor & Destructor Documentation	375

5.19.2.1	FrameEventArgs	375
5.19.2.2	FrameEventArgs	375
5.19.3	Property Documentation	375
5.19.3.1	Time	375
5.20	OpenTK.GameWindow Class Reference	375
5.20.1	Detailed Description	379
5.20.2	Constructor & Destructor Documentation	380
5.20.2.1	GameWindow	380
5.20.2.2	GameWindow	380
5.20.2.3	GameWindow	380
5.20.2.4	GameWindow	380
5.20.2.5	GameWindow	381
5.20.2.6	GameWindow	381
5.20.2.7	GameWindow	381
5.20.2.8	GameWindow	382
5.20.3	Member Function Documentation	382
5.20.3.1	Dispose	382
5.20.3.2	Dispose	382
5.20.3.3	Exit	383
5.20.3.4	MakeCurrent	383
5.20.3.5	OnClosing	383
5.20.3.6	OnLoad	383
5.20.3.7	OnRenderFrame	383
5.20.3.8	OnUnload	384
5.20.3.9	OnUpdateFrame	384
5.20.3.10	OnWindowInfoChanged	384
5.20.3.11	Run	384
5.20.3.12	Run	385
5.20.3.13	Run	385
5.20.3.14	SwapBuffers	385
5.20.4	Property Documentation	385

5.20.4.1	Context	385
5.20.4.2	IsExiting	385
5.20.4.3	Joysticks	385
5.20.4.4	Keyboard	385
5.20.4.5	Mouse	386
5.20.4.6	RenderFrequency	386
5.20.4.7	RenderPeriod	386
5.20.4.8	RenderTime	386
5.20.4.9	TargetRenderFrequency	386
5.20.4.10	TargetRenderPeriod	386
5.20.4.11	TargetUpdateFrequency	386
5.20.4.12	TargetUpdatePeriod	387
5.20.4.13	UpdateFrequency	387
5.20.4.14	UpdatePeriod	387
5.20.4.15	UpdateTime	387
5.20.4.16	VSync	387
5.20.4.17	WindowState	387
5.20.5	Event Documentation	388
5.20.5.1	Load	388
5.20.5.2	RenderFrame	388
5.20.5.3	Unload	388
5.20.5.4	UpdateFrame	388
5.21	OpenTK.GLControl Class Reference	388
5.21.1	Detailed Description	390
5.21.2	Constructor & Destructor Documentation	390
5.21.2.1	GLControl	390
5.21.2.2	GLControl	390
5.21.2.3	GLControl	390
5.21.3	Member Function Documentation	391
5.21.3.1	Dispose	391
5.21.3.2	GrabScreenshot	391

5.21.3.3	MakeCurrent	391
5.21.3.4	OnHandleCreated	391
5.21.3.5	OnHandleDestroyed	391
5.21.3.6	OnPaint	392
5.21.3.7	OnParentChanged	392
5.21.3.8	OnResize	392
5.21.3.9	SwapBuffers	392
5.21.4	Property Documentation	392
5.21.4.1	AspectRatio	392
5.21.4.2	Context	393
5.21.4.3	GraphicsMode	393
5.21.4.4	IsIdle	393
5.21.4.5	VSync	393
5.21.4.6	WindowInfo	393
5.22	OpenTK.Graphics.Color4 Struct Reference	393
5.22.1	Detailed Description	405
5.22.2	Constructor & Destructor Documentation	405
5.22.2.1	Color4	405
5.22.2.2	Color4	405
5.22.2.3	Color4	405
5.22.3	Member Function Documentation	406
5.22.3.1	Equals	406
5.22.3.2	Equals	406
5.22.3.3	GetHashCode	406
5.22.3.4	operator Color4	406
5.22.3.5	operator System.Drawing.Color	407
5.22.3.6	operator !=	407
5.22.3.7	operator ==	407
5.22.3.8	ToArgb	408
5.22.3.9	ToString	408
5.22.4	Member Data Documentation	408

5.22.4.1	A	408
5.22.4.2	B	408
5.22.4.3	G	408
5.22.4.4	R	408
5.22.5	Property Documentation	408
5.22.5.1	AliceBlue	408
5.22.5.2	AntiqueWhite	409
5.22.5.3	Aqua	409
5.22.5.4	Aquamarine	409
5.22.5.5	Azure	409
5.22.5.6	Beige	409
5.22.5.7	Bisque	409
5.22.5.8	Black	409
5.22.5.9	BlanchedAlmond	409
5.22.5.10	Blue	409
5.22.5.11	BlueViolet	409
5.22.5.12	Brown	410
5.22.5.13	BurlyWood	410
5.22.5.14	CadetBlue	410
5.22.5.15	Chartreuse	410
5.22.5.16	Chocolate	410
5.22.5.17	Coral	410
5.22.5.18	CornflowerBlue	410
5.22.5.19	Cornsilk	410
5.22.5.20	Crimson	410
5.22.5.21	Cyan	410
5.22.5.22	DarkBlue	411
5.22.5.23	DarkCyan	411
5.22.5.24	DarkGoldenrod	411
5.22.5.25	DarkGray	411
5.22.5.26	DarkGreen	411

5.22.5.27 DarkKhaki	411
5.22.5.28 DarkMagenta	411
5.22.5.29 DarkOliveGreen	411
5.22.5.30 DarkOrange	411
5.22.5.31 DarkOrchid	412
5.22.5.32 DarkRed	412
5.22.5.33 DarkSalmon	412
5.22.5.34 DarkSeaGreen	412
5.22.5.35 DarkSlateBlue	412
5.22.5.36 DarkSlateGray	412
5.22.5.37 DarkTurquoise	412
5.22.5.38 DarkViolet	412
5.22.5.39 DeepPink	412
5.22.5.40 DeepSkyBlue	412
5.22.5.41 DimGray	413
5.22.5.42 DodgerBlue	413
5.22.5.43 Firebrick	413
5.22.5.44 FloralWhite	413
5.22.5.45 ForestGreen	413
5.22.5.46 Fuchsia	413
5.22.5.47 Gainsboro	413
5.22.5.48 GhostWhite	413
5.22.5.49 Gold	413
5.22.5.50 Goldenrod	413
5.22.5.51 Gray	414
5.22.5.52 Green	414
5.22.5.53 GreenYellow	414
5.22.5.54 Honeydew	414
5.22.5.55 HotPink	414
5.22.5.56 IndianRed	414
5.22.5.57 Indigo	414

5.22.5.58 Ivory	414
5.22.5.59 Khaki	414
5.22.5.60 Lavender	414
5.22.5.61 LavenderBlush	415
5.22.5.62 LawnGreen	415
5.22.5.63 LemonChiffon	415
5.22.5.64 LightBlue	415
5.22.5.65 LightCoral	415
5.22.5.66 LightCyan	415
5.22.5.67 LightGoldenrodYellow	415
5.22.5.68 LightGray	415
5.22.5.69 LightGreen	415
5.22.5.70 LightPink	416
5.22.5.71 LightSalmon	416
5.22.5.72 LightSeaGreen	416
5.22.5.73 LightSkyBlue	416
5.22.5.74 LightSlateGray	416
5.22.5.75 LightSteelBlue	416
5.22.5.76 LightYellow	416
5.22.5.77 Lime	416
5.22.5.78 LimeGreen	416
5.22.5.79 Linen	416
5.22.5.80 Magenta	417
5.22.5.81 Maroon	417
5.22.5.82 MediumAquaMarine	417
5.22.5.83 MediumBlue	417
5.22.5.84 MediumOrchid	417
5.22.5.85 MediumPurple	417
5.22.5.86 MediumSeaGreen	417
5.22.5.87 MediumSlateBlue	417
5.22.5.88 MediumSpringGreen	417

5.22.5.89 MediumTurquoise	418
5.22.5.90 MediumVioletRed	418
5.22.5.91 MidnightBlue	418
5.22.5.92 MintCream	418
5.22.5.93 MistyRose	418
5.22.5.94 Moccasin	418
5.22.5.95 NavajoWhite	418
5.22.5.96 Navy	418
5.22.5.97 OldLace	418
5.22.5.98 Olive	419
5.22.5.99 OliveDrab	419
5.22.5.100Orange	419
5.22.5.101OrangeRed	419
5.22.5.102Orchid	419
5.22.5.103PaleGoldenrod	419
5.22.5.104PaleGreen	419
5.22.5.105PaleTurquoise	419
5.22.5.106PaleVioletRed	419
5.22.5.107PapayaWhip	420
5.22.5.108PeachPuff	420
5.22.5.109Peru	420
5.22.5.110Pink	420
5.22.5.111Plum	420
5.22.5.112PowderBlue	420
5.22.5.113Purple	420
5.22.5.114Red	420
5.22.5.115RosyBrown	420
5.22.5.116RoyalBlue	420
5.22.5.117SaddleBrown	421
5.22.5.118Salmon	421
5.22.5.119SandyBrown	421

5.22.5.120SeaGreen	421
5.22.5.121SeaShell	421
5.22.5.122Sienna	421
5.22.5.123Silver	421
5.22.5.124SkyBlue	421
5.22.5.125SlateBlue	421
5.22.5.126SlateGray	421
5.22.5.127Snow	422
5.22.5.128SpringGreen	422
5.22.5.129SteelBlue	422
5.22.5.130Tan	422
5.22.5.131Teal	422
5.22.5.132Thistle	422
5.22.5.133Tomato	422
5.22.5.134Transparent	422
5.22.5.135Turquoise	422
5.22.5.136Violet	422
5.22.5.137Wheat	423
5.22.5.138White	423
5.22.5.139WhiteSmoke	423
5.22.5.140Yellow	423
5.22.5.141YellowGreen	423
5.23 OpenTK.Graphics.ColorFormat Struct Reference	423
5.23.1 Detailed Description	425
5.23.2 Constructor & Destructor Documentation	425
5.23.2.1 ColorFormat	425
5.23.2.2 ColorFormat	425
5.23.3 Member Function Documentation	425
5.23.3.1 Equals	425
5.23.3.2 GetHashCode	426
5.23.3.3 operator ColorFormat	426

5.23.3.4	operator!=	426
5.23.3.5	operator==	427
5.23.3.6	ToString	427
5.23.4	Property Documentation	427
5.23.4.1	Alpha	427
5.23.4.2	BitsPerPixel	427
5.23.4.3	Blue	427
5.23.4.4	Green	427
5.23.4.5	IsIndexed	427
5.23.4.6	Red	428
5.24	OpenTK.Graphics.ES10.GL Class Reference	428
5.24.1	Detailed Description	442
5.24.2	Member Function Documentation	442
5.24.2.1	ActiveTexture	442
5.24.2.2	AlphaFunc	442
5.24.2.3	BindTexture	443
5.24.2.4	BindTexture	443
5.24.2.5	BlendFunc	443
5.24.2.6	Clear	444
5.24.2.7	Clear	444
5.24.2.8	ClearColor	444
5.24.2.9	ClearDepth	445
5.24.2.10	ClearStencil	445
5.24.2.11	ClientActiveTexture	445
5.24.2.12	Color4	445
5.24.2.13	ColorMask	446
5.24.2.14	ColorPointer	446
5.24.2.15	ColorPointer< T3 >	446
5.24.2.16	ColorPointer< T3 >	447
5.24.2.17	ColorPointer< T3 >	447
5.24.2.18	ColorPointer< T3 >	448

5.24.2.19 CompressedTexImage2D	448
5.24.2.20 CompressedTexImage2D< T7 >	449
5.24.2.21 CompressedTexImage2D< T7 >	450
5.24.2.22 CompressedTexImage2D< T7 >	451
5.24.2.23 CompressedTexImage2D< T7 >	452
5.24.2.24 CompressedTexSubImage2D	453
5.24.2.25 CompressedTexSubImage2D< T8 >	453
5.24.2.26 CompressedTexSubImage2D< T8 >	454
5.24.2.27 CompressedTexSubImage2D< T8 >	455
5.24.2.28 CompressedTexSubImage2D< T8 >	456
5.24.2.29 CopyTexImage2D	456
5.24.2.30 CopyTexSubImage2D	457
5.24.2.31 CullFace	458
5.24.2.32 DeleteTextures	458
5.24.2.33 DeleteTextures	458
5.24.2.34 DeleteTextures	459
5.24.2.35 DeleteTextures	459
5.24.2.36 DeleteTextures	459
5.24.2.37 DeleteTextures	459
5.24.2.38 DepthFunc	460
5.24.2.39 DepthMask	460
5.24.2.40 DepthRange	460
5.24.2.41 DrawArrays	460
5.24.2.42 DrawElements	461
5.24.2.43 DrawElements< T3 >	461
5.24.2.44 DrawElements< T3 >	462
5.24.2.45 DrawElements< T3 >	462
5.24.2.46 DrawElements< T3 >	463
5.24.2.47 Enable	463
5.24.2.48 EnableClientState	464
5.24.2.49 Finish	464

5.24.2.50 Flush	464
5.24.2.51 Fog	464
5.24.2.52 Fog	464
5.24.2.53 Fog	465
5.24.2.54 FrontFace	465
5.24.2.55 Frustum	465
5.24.2.56 GenTextures	466
5.24.2.57 GenTextures	466
5.24.2.58 GenTextures	466
5.24.2.59 GenTextures	466
5.24.2.60 GenTextures	467
5.24.2.61 GenTextures	467
5.24.2.62 GetError	467
5.24.2.63 GetString	467
5.24.2.64 Hint	467
5.24.2.65 Light	468
5.24.2.66 Light	468
5.24.2.67 Light	469
5.24.2.68 LightModel	469
5.24.2.69 LightModel	469
5.24.2.70 LightModel	470
5.24.2.71 LineWidth	470
5.24.2.72 LoadIdentity	470
5.24.2.73 LoadMatrix	470
5.24.2.74 LoadMatrix	471
5.24.2.75 LoadMatrix	471
5.24.2.76 LogicOp	471
5.24.2.77 Material	471
5.24.2.78 Material	472
5.24.2.79 Material	472
5.24.2.80 MatrixMode	472

5.24.2.81 MultiTexCoord4	473
5.24.2.82 MultMatrix	473
5.24.2.83 MultMatrix	473
5.24.2.84 MultMatrix	473
5.24.2.85 Normal3	474
5.24.2.86 NormalPointer	474
5.24.2.87 NormalPointer< T2 >	474
5.24.2.88 NormalPointer< T2 >	475
5.24.2.89 NormalPointer< T2 >	475
5.24.2.90 NormalPointer< T2 >	476
5.24.2.91 Ortho	476
5.24.2.92 PixelStore	476
5.24.2.93 PointSize	477
5.24.2.94 PolygonOffset	477
5.24.2.95 PushMatrix	477
5.24.2.96 ReadPixels	477
5.24.2.97 ReadPixels< T6 >	478
5.24.2.98 ReadPixels< T6 >	479
5.24.2.99 ReadPixels< T6 >	480
5.24.2.100ReadPixels< T6 >	480
5.24.2.101Rotate	481
5.24.2.102SampleCoverage	481
5.24.2.103Scale	482
5.24.2.104Scissor	482
5.24.2.105ShadeModel	482
5.24.2.106StencilFunc	483
5.24.2.107StencilFunc	483
5.24.2.108StencilMask	483
5.24.2.109StencilMask	484
5.24.2.110StencilOp	484
5.24.2.111TexCoordPointer	484

5.24.2.112	TexCoordPointer< T3 >	485
5.24.2.113	TexCoordPointer< T3 >	485
5.24.2.114	TexCoordPointer< T3 >	486
5.24.2.115	TexCoordPointer< T3 >	486
5.24.2.116	TexEnv	487
5.24.2.117	TexEnv	488
5.24.2.118	TexEnv	488
5.24.2.119	TexImage2D	489
5.24.2.120	TexImage2D< T8 >	490
5.24.2.121	TexImage2D< T8 >	492
5.24.2.122	TexImage2D< T8 >	493
5.24.2.123	TexImage2D< T8 >	495
5.24.2.124	TexParameter	496
5.24.2.125	TexSubImage2D	497
5.24.2.126	TexSubImage2D< T8 >	498
5.24.2.127	TexSubImage2D< T8 >	499
5.24.2.128	TexSubImage2D< T8 >	500
5.24.2.129	TexSubImage2D< T8 >	501
5.24.2.130	Translate	502
5.24.2.131	VertexPointer	502
5.24.2.132	VertexPointer< T3 >	502
5.24.2.133	VertexPointer< T3 >	503
5.24.2.134	VertexPointer< T3 >	503
5.24.2.135	VertexPointer< T3 >	504
5.24.2.136	Viewport	504
5.24.3	Property Documentation	505
5.24.3.1	SyncRoot	505
5.25	OpenTK.Graphics.ES11.GL Class Reference	505
5.25.1	Detailed Description	527
5.25.2	Member Function Documentation	527
5.25.2.1	ActiveTexture	527

5.25.2.2	AlphaFunc	528
5.25.2.3	BindBuffer	528
5.25.2.4	BindBuffer	528
5.25.2.5	BindTexture	529
5.25.2.6	BindTexture	529
5.25.2.7	BlendFunc	529
5.25.2.8	BufferData	530
5.25.2.9	BufferData< T2 >	530
5.25.2.10	BufferData< T2 >	531
5.25.2.11	BufferData< T2 >	531
5.25.2.12	BufferData< T2 >	532
5.25.2.13	BufferSubData	533
5.25.2.14	BufferSubData< T3 >	533
5.25.2.15	BufferSubData< T3 >	533
5.25.2.16	BufferSubData< T3 >	534
5.25.2.17	BufferSubData< T3 >	534
5.25.2.18	Clear	535
5.25.2.19	Clear	535
5.25.2.20	ClearColor	535
5.25.2.21	ClearDepth	536
5.25.2.22	ClearStencil	536
5.25.2.23	ClientActiveTexture	536
5.25.2.24	ClipPlane	536
5.25.2.25	ClipPlane	537
5.25.2.26	ClipPlane	537
5.25.2.27	Color4	537
5.25.2.28	Color4	537
5.25.2.29	ColorMask	538
5.25.2.30	ColorPointer	538
5.25.2.31	ColorPointer< T3 >	538
5.25.2.32	ColorPointer< T3 >	539

5.25.2.33 ColorPointer< T3 >	539
5.25.2.34 ColorPointer< T3 >	540
5.25.2.35 CompressedTexImage2D	541
5.25.2.36 CompressedTexImage2D< T7 >	541
5.25.2.37 CompressedTexImage2D< T7 >	542
5.25.2.38 CompressedTexImage2D< T7 >	543
5.25.2.39 CompressedTexImage2D< T7 >	544
5.25.2.40 CompressedTexSubImage2D	545
5.25.2.41 CompressedTexSubImage2D< T8 >	546
5.25.2.42 CompressedTexSubImage2D< T8 >	546
5.25.2.43 CompressedTexSubImage2D< T8 >	547
5.25.2.44 CompressedTexSubImage2D< T8 >	548
5.25.2.45 CopyTexImage2D	549
5.25.2.46 CopyTexSubImage2D	550
5.25.2.47 CullFace	550
5.25.2.48 DeleteBuffers	550
5.25.2.49 DeleteBuffers	551
5.25.2.50 DeleteBuffers	551
5.25.2.51 DeleteBuffers	551
5.25.2.52 DeleteBuffers	551
5.25.2.53 DeleteBuffers	552
5.25.2.54 DeleteTextures	552
5.25.2.55 DeleteTextures	552
5.25.2.56 DeleteTextures	552
5.25.2.57 DeleteTextures	553
5.25.2.58 DeleteTextures	553
5.25.2.59 DeleteTextures	553
5.25.2.60 DepthFunc	553
5.25.2.61 DepthMask	554
5.25.2.62 DepthRange	554
5.25.2.63 DrawArrays	554

5.25.2.64 DrawElements	555
5.25.2.65 DrawElements< T3 >	555
5.25.2.66 DrawElements< T3 >	556
5.25.2.67 DrawElements< T3 >	556
5.25.2.68 DrawElements< T3 >	557
5.25.2.69 Enable	557
5.25.2.70 EnableClientState	557
5.25.2.71 Finish	558
5.25.2.72 Flush	558
5.25.2.73 Fog	558
5.25.2.74 Fog	558
5.25.2.75 Fog	558
5.25.2.76 FrontFace	559
5.25.2.77 Frustum	559
5.25.2.78 GenBuffers	559
5.25.2.79 GenBuffers	559
5.25.2.80 GenBuffers	560
5.25.2.81 GenBuffers	560
5.25.2.82 GenBuffers	560
5.25.2.83 GenBuffers	560
5.25.2.84 GenTextures	561
5.25.2.85 GenTextures	561
5.25.2.86 GenTextures	561
5.25.2.87 GenTextures	561
5.25.2.88 GenTextures	562
5.25.2.89 GenTextures	562
5.25.2.90 GetBufferParameter	562
5.25.2.91 GetBufferParameter	562
5.25.2.92 GetBufferParameter	563
5.25.2.93 GetClipPlane	563
5.25.2.94 GetClipPlane	564

5.25.2.95 GetClipPlane	564
5.25.2.96 GetError	564
5.25.2.97 GetLight	564
5.25.2.98 GetLight	565
5.25.2.99 GetLight	565
5.25.2.100GetMaterial	566
5.25.2.101GetMaterial	566
5.25.2.102GetMaterial	566
5.25.2.103GetPointer	567
5.25.2.104GetPointer< T1 >	567
5.25.2.105GetPointer< T1 >	568
5.25.2.106GetPointer< T1 >	568
5.25.2.107GetPointer< T1 >	569
5.25.2.108GetString	569
5.25.2.109GetTexEnv	569
5.25.2.110GetTexEnv	570
5.25.2.111GetTexEnv	570
5.25.2.112GetTexEnv	571
5.25.2.113GetTexEnv	571
5.25.2.114GetTexEnv	572
5.25.2.115GetTexParameter	572
5.25.2.116GetTexParameter	573
5.25.2.117GetTexParameter	573
5.25.2.118GetTexParameter	574
5.25.2.119GetTexParameter	574
5.25.2.120GetTexParameter	575
5.25.2.121Hint	575
5.25.2.122IsBuffer	576
5.25.2.123IsBuffer	576
5.25.2.124IsEnabled	576
5.25.2.125IsTexture	576

5.25.2.126IsTexture	577
5.25.2.127Light	577
5.25.2.128Light	577
5.25.2.129Light	578
5.25.2.130LightModel	578
5.25.2.131LightModel	578
5.25.2.132LightModel	579
5.25.2.133LineWidth	579
5.25.2.134LoadIdentity	579
5.25.2.135LoadMatrix	579
5.25.2.136LoadMatrix	580
5.25.2.137LoadMatrix	580
5.25.2.138LogicOp	580
5.25.2.139Material	580
5.25.2.140Material	581
5.25.2.141Material	581
5.25.2.142MatrixMode	581
5.25.2.143MultiTexCoord4	582
5.25.2.144MultMatrix	582
5.25.2.145MultMatrix	582
5.25.2.146MultMatrix	582
5.25.2.147Normal3	583
5.25.2.148NormalPointer	583
5.25.2.149NormalPointer< T2 >	583
5.25.2.150NormalPointer< T2 >	584
5.25.2.151NormalPointer< T2 >	584
5.25.2.152NormalPointer< T2 >	585
5.25.2.153Ortho	585
5.25.2.154PixelStore	585
5.25.2.155PointParameter	586
5.25.2.156PointParameter	586

5.25.2.157	PointParameter	586
5.25.2.158	PointSize	587
5.25.2.159	PolygonOffset	587
5.25.2.160	PushMatrix	587
5.25.2.161	ReadPixels	587
5.25.2.162	ReadPixels< T6 >	588
5.25.2.163	ReadPixels< T6 >	589
5.25.2.164	ReadPixels< T6 >	590
5.25.2.165	ReadPixels< T6 >	590
5.25.2.166	Rotate	591
5.25.2.167	SampleCoverage	591
5.25.2.168	Scale	592
5.25.2.169	Scissor	592
5.25.2.170	ShadeModel	592
5.25.2.171	StencilFunc	592
5.25.2.172	StencilFunc	593
5.25.2.173	StencilMask	593
5.25.2.174	StencilMask	593
5.25.2.175	StencilOp	594
5.25.2.176	TexCoordPointer	594
5.25.2.177	TexCoordPointer< T3 >	595
5.25.2.178	TexCoordPointer< T3 >	595
5.25.2.179	TexCoordPointer< T3 >	596
5.25.2.180	TexCoordPointer< T3 >	596
5.25.2.181	TexEnv	597
5.25.2.182	TexEnv	597
5.25.2.183	TexEnv	598
5.25.2.184	TexEnv	599
5.25.2.185	TexEnv	599
5.25.2.186	TexEnv	600
5.25.2.187	TexImage2D	601

5.25.2.188TexImage2D< T8 >	602
5.25.2.189TexImage2D< T8 >	604
5.25.2.190TexImage2D< T8 >	605
5.25.2.191TexImage2D< T8 >	607
5.25.2.192TexParameter	608
5.25.2.193TexParameter	609
5.25.2.194TexParameter	609
5.25.2.195TexParameter	610
5.25.2.196TexParameter	610
5.25.2.197TexParameter	611
5.25.2.198TexSubImage2D	611
5.25.2.199TexSubImage2D< T8 >	612
5.25.2.200TexSubImage2D< T8 >	613
5.25.2.201TexSubImage2D< T8 >	614
5.25.2.202TexSubImage2D< T8 >	615
5.25.2.203Translate	616
5.25.2.204VertexPointer	616
5.25.2.205VertexPointer< T3 >	617
5.25.2.206VertexPointer< T3 >	617
5.25.2.207VertexPointer< T3 >	618
5.25.2.208VertexPointer< T3 >	618
5.25.2.209Viewport	619
5.25.3 Property Documentation	619
5.25.3.1 SyncRoot	619
5.26 OpenTK.Graphics.ES20.GL Class Reference	619
5.26.1 Detailed Description	656
5.26.2 Member Function Documentation	656
5.26.2.1 ActiveTexture	656
5.26.2.2 AttachShader	657
5.26.2.3 AttachShader	657
5.26.2.4 BindAttribLocation	657

5.26.2.5 BindAttribLocation	658
5.26.2.6 BindBuffer	658
5.26.2.7 BindBuffer	658
5.26.2.8 BindTexture	659
5.26.2.9 BindTexture	659
5.26.2.10 BlendColor	659
5.26.2.11 BlendEquation	659
5.26.2.12 BlendEquationSeparate	660
5.26.2.13 BlendFunc	660
5.26.2.14 BlendFuncSeparate	661
5.26.2.15 BufferData	662
5.26.2.16 BufferData< T2 >	662
5.26.2.17 BufferData< T2 >	663
5.26.2.18 BufferData< T2 >	663
5.26.2.19 BufferData< T2 >	664
5.26.2.20 BufferSubData	664
5.26.2.21 BufferSubData< T3 >	665
5.26.2.22 BufferSubData< T3 >	665
5.26.2.23 BufferSubData< T3 >	666
5.26.2.24 BufferSubData< T3 >	666
5.26.2.25 Clear	667
5.26.2.26 ClearColor	667
5.26.2.27 ClearDepth	667
5.26.2.28 ClearStencil	667
5.26.2.29 ColorMask	668
5.26.2.30 CompileShader	668
5.26.2.31 CompileShader	668
5.26.2.32 CompressedTexImage2D	668
5.26.2.33 CompressedTexImage2D< T7 >	669
5.26.2.34 CompressedTexImage2D< T7 >	670
5.26.2.35 CompressedTexImage2D< T7 >	671

5.26.2.36 CompressedTexImage2D< T7 >	672
5.26.2.37 CompressedTexSubImage2D	673
5.26.2.38 CompressedTexSubImage2D< T8 >	673
5.26.2.39 CompressedTexSubImage2D< T8 >	674
5.26.2.40 CompressedTexSubImage2D< T8 >	675
5.26.2.41 CompressedTexSubImage2D< T8 >	676
5.26.2.42 CopyTexImage2D	676
5.26.2.43 CopyTexSubImage2D	677
5.26.2.44 CreateProgram	678
5.26.2.45 CreateShader	678
5.26.2.46 CullFace	678
5.26.2.47 DeleteBuffers	678
5.26.2.48 DeleteBuffers	679
5.26.2.49 DeleteBuffers	679
5.26.2.50 DeleteBuffers	679
5.26.2.51 DeleteBuffers	679
5.26.2.52 DeleteBuffers	680
5.26.2.53 DeleteProgram	680
5.26.2.54 DeleteProgram	680
5.26.2.55 DeleteShader	680
5.26.2.56 DeleteShader	680
5.26.2.57 DeleteTextures	681
5.26.2.58 DeleteTextures	681
5.26.2.59 DeleteTextures	681
5.26.2.60 DeleteTextures	681
5.26.2.61 DeleteTextures	682
5.26.2.62 DeleteTextures	682
5.26.2.63 DepthFunc	682
5.26.2.64 DepthMask	682
5.26.2.65 DepthRange	683
5.26.2.66 DetachShader	683

5.26.2.67 DetachShader	683
5.26.2.68 DrawArrays	683
5.26.2.69 DrawElements	684
5.26.2.70 DrawElements< T3 >	684
5.26.2.71 DrawElements< T3 >	685
5.26.2.72 DrawElements< T3 >	685
5.26.2.73 DrawElements< T3 >	686
5.26.2.74 Enable	686
5.26.2.75 EnableVertexAttribArray	686
5.26.2.76 EnableVertexAttribArray	687
5.26.2.77 Finish	687
5.26.2.78 Flush	687
5.26.2.79 FrontFace	687
5.26.2.80 GenBuffers	687
5.26.2.81 GenBuffers	687
5.26.2.82 GenBuffers	688
5.26.2.83 GenBuffers	688
5.26.2.84 GenBuffers	688
5.26.2.85 GenBuffers	688
5.26.2.86 GenTextures	689
5.26.2.87 GenTextures	689
5.26.2.88 GenTextures	689
5.26.2.89 GenTextures	689
5.26.2.90 GenTextures	690
5.26.2.91 GenTextures	690
5.26.2.92 GetActiveAttrib	690
5.26.2.93 GetActiveAttrib	691
5.26.2.94 GetActiveAttrib	691
5.26.2.95 GetActiveAttrib	692
5.26.2.96 GetActiveAttrib	692
5.26.2.97 GetActiveAttrib	693

5.26.2.98 GetActiveUniform	693
5.26.2.99 GetActiveUniform	694
5.26.2.100GetActiveUniform	694
5.26.2.101GetActiveUniform	695
5.26.2.102GetActiveUniform	695
5.26.2.103GetActiveUniform	696
5.26.2.104GetAttachedShaders	696
5.26.2.105GetAttachedShaders	696
5.26.2.106GetAttachedShaders	697
5.26.2.107GetAttachedShaders	697
5.26.2.108GetAttachedShaders	697
5.26.2.109GetAttachedShaders	698
5.26.2.110GetAttribLocation	698
5.26.2.111GetAttribLocation	698
5.26.2.112GetBufferParameter	699
5.26.2.113GetBufferParameter	699
5.26.2.114GetBufferParameter	699
5.26.2.115GetError	700
5.26.2.116GetProgram	700
5.26.2.117GetProgram	700
5.26.2.118GetProgram	701
5.26.2.119GetProgram	701
5.26.2.120GetProgram	702
5.26.2.121GetProgram	702
5.26.2.122GetProgramInfoLog	702
5.26.2.123GetProgramInfoLog	703
5.26.2.124GetProgramInfoLog	703
5.26.2.125GetProgramInfoLog	703
5.26.2.126GetProgramInfoLog	704
5.26.2.127GetProgramInfoLog	704
5.26.2.128GetShader	705

5.26.2.129GetShader	705
5.26.2.130GetShader	705
5.26.2.131GetShader	706
5.26.2.132GetShader	706
5.26.2.133GetShader	706
5.26.2.134GetShaderInfoLog	707
5.26.2.135GetShaderInfoLog	707
5.26.2.136GetShaderInfoLog	707
5.26.2.137GetShaderInfoLog	708
5.26.2.138GetShaderInfoLog	708
5.26.2.139GetShaderInfoLog	708
5.26.2.140GetShaderSource	709
5.26.2.141GetShaderSource	709
5.26.2.142GetShaderSource	709
5.26.2.143GetShaderSource	710
5.26.2.144GetShaderSource	710
5.26.2.145GetShaderSource	710
5.26.2.146GetString	711
5.26.2.147GetTexParameter	711
5.26.2.148GetTexParameter	711
5.26.2.149GetTexParameter	712
5.26.2.150GetTexParameter	713
5.26.2.151GetTexParameter	713
5.26.2.152GetTexParameter	714
5.26.2.153GetUniform	714
5.26.2.154GetUniform	714
5.26.2.155GetUniform	715
5.26.2.156GetUniform	715
5.26.2.157GetUniform	715
5.26.2.158GetUniform	716
5.26.2.159GetUniform	716

5.26.2.160GetUniform	716
5.26.2.161GetUniform	716
5.26.2.162GetUniform	717
5.26.2.163GetUniform	717
5.26.2.164GetUniform	717
5.26.2.165GetUniformLocation	718
5.26.2.166GetUniformLocation	718
5.26.2.167GetVertexAttrib	718
5.26.2.168GetVertexAttrib	719
5.26.2.169GetVertexAttrib	719
5.26.2.170GetVertexAttrib	719
5.26.2.171GetVertexAttrib	720
5.26.2.172GetVertexAttrib	720
5.26.2.173GetVertexAttrib	721
5.26.2.174GetVertexAttrib	721
5.26.2.175GetVertexAttrib	721
5.26.2.176GetVertexAttrib	722
5.26.2.177GetVertexAttrib	722
5.26.2.178GetVertexAttrib	723
5.26.2.179GetVertexAttribPointer	723
5.26.2.180GetVertexAttribPointer	723
5.26.2.181GetVertexAttribPointer< T2 >	724
5.26.2.182GetVertexAttribPointer< T2 >	724
5.26.2.183GetVertexAttribPointer< T2 >	724
5.26.2.184GetVertexAttribPointer< T2 >	725
5.26.2.185GetVertexAttribPointer< T2 >	725
5.26.2.186GetVertexAttribPointer< T2 >	726
5.26.2.187GetVertexAttribPointer< T2 >	726
5.26.2.188GetVertexAttribPointer< T2 >	726
5.26.2.189Hint	727
5.26.2.190IsBuffer	727

5.26.2.191IsBuffer	727
5.26.2.192IsEnabled	728
5.26.2.193IsProgram	728
5.26.2.194IsProgram	728
5.26.2.195IsShader	728
5.26.2.196IsShader	728
5.26.2.197IsTexture	729
5.26.2.198IsTexture	729
5.26.2.199LineWidth	729
5.26.2.200LinkProgram	729
5.26.2.201LinkProgram	729
5.26.2.202PixelStore	730
5.26.2.203PolygonOffset	730
5.26.2.204ReadPixels	730
5.26.2.205ReadPixels< T6 >	731
5.26.2.206ReadPixels< T6 >	732
5.26.2.207ReadPixels< T6 >	733
5.26.2.208ReadPixels< T6 >	733
5.26.2.209SampleCoverage	734
5.26.2.210Scissor	734
5.26.2.211ShaderSource	735
5.26.2.212ShaderSource	735
5.26.2.213ShaderSource	735
5.26.2.214ShaderSource	736
5.26.2.215ShaderSource	736
5.26.2.216ShaderSource	737
5.26.2.217StencilFunc	737
5.26.2.218StencilFunc	737
5.26.2.219StencilFuncSeparate	738
5.26.2.220StencilFuncSeparate	738
5.26.2.221StencilMask	739

5.26.2.222StencilMask	739
5.26.2.223StencilMaskSeparate	739
5.26.2.224StencilMaskSeparate	740
5.26.2.225StencilOp	740
5.26.2.226StencilOpSeparate	740
5.26.2.227TexImage2D	741
5.26.2.228TexImage2D< T8 >	743
5.26.2.229TexImage2D< T8 >	744
5.26.2.230TexImage2D< T8 >	746
5.26.2.231TexImage2D< T8 >	747
5.26.2.232TexParameter	749
5.26.2.233TexParameter	749
5.26.2.234TexParameter	750
5.26.2.235TexParameter	750
5.26.2.236TexParameter	751
5.26.2.237TexParameter	751
5.26.2.238TexSubImage2D	752
5.26.2.239TexSubImage2D< T8 >	753
5.26.2.240TexSubImage2D< T8 >	754
5.26.2.241TexSubImage2D< T8 >	755
5.26.2.242TexSubImage2D< T8 >	756
5.26.2.243Uniform1	757
5.26.2.244Uniform1	757
5.26.2.245Uniform1	757
5.26.2.246Uniform1	757
5.26.2.247Uniform1	758
5.26.2.248Uniform1	758
5.26.2.249Uniform1	758
5.26.2.250Uniform1	758
5.26.2.251Uniform2	759
5.26.2.252Uniform2	759

5.26.2.253Uniform2	759
5.26.2.254Uniform2	759
5.26.2.255Uniform2	760
5.26.2.256Uniform2	760
5.26.2.257Uniform2	760
5.26.2.258Uniform3	760
5.26.2.259Uniform3	761
5.26.2.260Uniform3	761
5.26.2.261Uniform3	761
5.26.2.262Uniform3	761
5.26.2.263Uniform3	762
5.26.2.264Uniform3	762
5.26.2.265Uniform3	762
5.26.2.266Uniform4	762
5.26.2.267Uniform4	763
5.26.2.268Uniform4	763
5.26.2.269Uniform4	763
5.26.2.270Uniform4	763
5.26.2.271Uniform4	764
5.26.2.272Uniform4	764
5.26.2.273Uniform4	764
5.26.2.274UseProgram	764
5.26.2.275UseProgram	765
5.26.2.276ValidateProgram	765
5.26.2.277ValidateProgram	765
5.26.2.278VertexAttrib1	765
5.26.2.279VertexAttrib1	766
5.26.2.280VertexAttrib1	766
5.26.2.281VertexAttrib1	766
5.26.2.282VertexAttrib1	766
5.26.2.283VertexAttrib1	767

5.26.2.284VertexAttrib2	767
5.26.2.285VertexAttrib2	767
5.26.2.286VertexAttrib2	767
5.26.2.287VertexAttrib2	768
5.26.2.288VertexAttrib2	768
5.26.2.289VertexAttrib2	768
5.26.2.290VertexAttrib2	768
5.26.2.291VertexAttrib2	769
5.26.2.292VertexAttrib3	769
5.26.2.293VertexAttrib3	769
5.26.2.294VertexAttrib3	769
5.26.2.295VertexAttrib3	770
5.26.2.296VertexAttrib3	770
5.26.2.297VertexAttrib3	770
5.26.2.298VertexAttrib3	770
5.26.2.299VertexAttrib3	771
5.26.2.300VertexAttrib4	771
5.26.2.301VertexAttrib4	771
5.26.2.302VertexAttrib4	771
5.26.2.303VertexAttrib4	772
5.26.2.304VertexAttrib4	772
5.26.2.305VertexAttrib4	772
5.26.2.306VertexAttrib4	772
5.26.2.307VertexAttrib4	773
5.26.2.308VertexAttribPointer	773
5.26.2.309VertexAttribPointer	773
5.26.2.310VertexAttribPointer< T5 >	774
5.26.2.311VertexAttribPointer< T5 >	775
5.26.2.312VertexAttribPointer< T5 >	775
5.26.2.313VertexAttribPointer< T5 >	776
5.26.2.314VertexAttribPointer< T5 >	777

5.26.2.315VertexAttribPointer< T5 >	778
5.26.2.316VertexAttribPointer< T5 >	778
5.26.2.317VertexAttribPointer< T5 >	779
5.26.2.318Viewport	780
5.26.3 Property Documentation	780
5.26.3.1 SyncRoot	780
5.27 OpenTK.Graphics.GraphicsBindingsBase Class Reference	780
5.27.1 Detailed Description	781
5.27.2 Member Function Documentation	781
5.27.2.1 GetAddress	781
5.28 OpenTK.Graphics.GraphicsContext Class Reference	781
5.28.1 Detailed Description	783
5.28.2 Constructor & Destructor Documentation	784
5.28.2.1 GraphicsContext	784
5.28.2.2 GraphicsContext	784
5.28.2.3 GraphicsContext	784
5.28.2.4 GraphicsContext	785
5.28.3 Member Function Documentation	785
5.28.3.1 Assert	785
5.28.3.2 CreateDummyContext	786
5.28.3.3 CreateDummyContext	786
5.28.3.4 Dispose	786
5.28.3.5 LoadAll	786
5.28.3.6 MakeCurrent	787
5.28.3.7 SwapBuffers	787
5.28.3.8 Update	787
5.28.4 Property Documentation	787
5.28.4.1 CurrentContext	787
5.28.4.2 DirectRendering	787
5.28.4.3 ErrorChecking	788
5.28.4.4 GraphicsMode	788

5.28.4.5	IsCurrent	788
5.28.4.6	IsDisposed	788
5.28.4.7	ShareContexts	788
5.28.4.8	VSync	789
5.29	OpenTK.Graphics.GraphicsContextException Class Reference	789
5.29.1	Detailed Description	789
5.29.2	Constructor & Destructor Documentation	789
5.29.2.1	GraphicsContextException	789
5.29.2.2	GraphicsContextException	790
5.30	OpenTK.Graphics.GraphicsContextMissingException Class Reference	790
5.30.1	Detailed Description	790
5.30.2	Constructor & Destructor Documentation	790
5.30.2.1	GraphicsContextMissingException	790
5.31	OpenTK.Graphics.GraphicsContextVersion Class Reference	791
5.31.1	Detailed Description	791
5.31.2	Property Documentation	791
5.31.2.1	Major	791
5.31.2.2	Minor	791
5.31.2.3	Renderer	791
5.31.2.4	Vendor	792
5.32	OpenTK.Graphics.GraphicsErrorException Class Reference	792
5.32.1	Detailed Description	792
5.32.2	Constructor & Destructor Documentation	792
5.32.2.1	GraphicsErrorException	792
5.33	OpenTK.Graphics.GraphicsMode Class Reference	793
5.33.1	Detailed Description	795
5.33.2	Constructor & Destructor Documentation	795
5.33.2.1	GraphicsMode	795
5.33.2.2	GraphicsMode	795
5.33.2.3	GraphicsMode	795
5.33.2.4	GraphicsMode	795

5.33.2.5	GraphicsMode	796
5.33.2.6	GraphicsMode	796
5.33.2.7	GraphicsMode	796
5.33.2.8	GraphicsMode	797
5.33.3	Member Function Documentation	797
5.33.3.1	Equals	797
5.33.3.2	Equals	797
5.33.3.3	GetHashCode	798
5.33.3.4	ToString	798
5.33.4	Property Documentation	798
5.33.4.1	AccumulatorFormat	798
5.33.4.2	Buffers	798
5.33.4.3	ColorFormat	798
5.33.4.4	Default	798
5.33.4.5	Depth	799
5.33.4.6	Index	799
5.33.4.7	Samples	799
5.33.4.8	Stencil	799
5.33.4.9	Stereo	799
5.34	OpenTK.Graphics.GraphicsModeException Class Reference	799
5.34.1	Detailed Description	800
5.34.2	Constructor & Destructor Documentation	800
5.34.2.1	GraphicsModeException	800
5.34.2.2	GraphicsModeException	800
5.35	OpenTK.Graphics.IGraphicsContext Interface Reference	800
5.35.1	Detailed Description	801
5.35.2	Member Function Documentation	801
5.35.2.1	LoadAll	801
5.35.2.2	MakeCurrent	801
5.35.2.3	SwapBuffers	802
5.35.2.4	Update	802

5.35.3	Property Documentation	802
5.35.3.1	ErrorChecking	802
5.35.3.2	GraphicsMode	802
5.35.3.3	IsCurrent	802
5.35.3.4	IsDisposed	803
5.35.3.5	VSync	803
5.36	OpenTK.Graphics.IGraphicsContextInternal Interface Reference . . .	803
5.36.1	Detailed Description	804
5.36.2	Member Function Documentation	804
5.36.2.1	GetAddress	804
5.36.2.2	LoadAll	804
5.36.3	Property Documentation	804
5.36.3.1	Context	804
5.36.3.2	Implementation	804
5.37	OpenTK.Graphics.OpenGL.GL Class Reference	805
5.37.1	Detailed Description	953
5.37.2	Member Function Documentation	953
5.37.2.1	Accum	953
5.37.2.2	ActiveTexture	954
5.37.2.3	AlphaFunc	954
5.37.2.4	AreTexturesResident	954
5.37.2.5	AreTexturesResident	955
5.37.2.6	AreTexturesResident	955
5.37.2.7	AreTexturesResident	955
5.37.2.8	AreTexturesResident	956
5.37.2.9	AreTexturesResident	956
5.37.2.10	ArrayElement	956
5.37.2.11	AttachShader	956
5.37.2.12	AttachShader	957
5.37.2.13	Begin	957
5.37.2.14	BeginQuery	957

5.37.2.15 BeginQuery	958
5.37.2.16 BindAttribLocation	958
5.37.2.17 BindAttribLocation	958
5.37.2.18 BindBuffer	959
5.37.2.19 BindBuffer	959
5.37.2.20 BindTexture	959
5.37.2.21 BindTexture	960
5.37.2.22 Bitmap	960
5.37.2.23 Bitmap	960
5.37.2.24 Bitmap	961
5.37.2.25 BlendColor	961
5.37.2.26 BlendEquation	961
5.37.2.27 BlendEquation	961
5.37.2.28 BlendEquation	962
5.37.2.29 BlendEquationSeparate	962
5.37.2.30 BlendEquationSeparate	962
5.37.2.31 BlendEquationSeparate	963
5.37.2.32 BlendFunc	963
5.37.2.33 BlendFunc	964
5.37.2.34 BlendFunc	965
5.37.2.35 BlendFuncSeparate	965
5.37.2.36 BlendFuncSeparate	966
5.37.2.37 BlendFuncSeparate	967
5.37.2.38 BufferData	968
5.37.2.39 BufferData< T2 >	968
5.37.2.40 BufferData< T2 >	969
5.37.2.41 BufferData< T2 >	969
5.37.2.42 BufferData< T2 >	970
5.37.2.43 BufferSubData	970
5.37.2.44 BufferSubData< T3 >	971
5.37.2.45 BufferSubData< T3 >	971

5.37.2.46 BufferSubData< T3 >	972
5.37.2.47 BufferSubData< T3 >	972
5.37.2.48 CallList	973
5.37.2.49 CallList	973
5.37.2.50 CallLists	973
5.37.2.51 CallLists< T2 >	973
5.37.2.52 CallLists< T2 >	974
5.37.2.53 CallLists< T2 >	974
5.37.2.54 CallLists< T2 >	975
5.37.2.55 Clear	975
5.37.2.56 ClearAccum	976
5.37.2.57 ClearColor	976
5.37.2.58 ClearDepth	976
5.37.2.59 ClearIndex	976
5.37.2.60 ClearStencil	977
5.37.2.61 ClientActiveTexture	977
5.37.2.62 ClipPlane	977
5.37.2.63 ClipPlane	977
5.37.2.64 ClipPlane	978
5.37.2.65 Color3	978
5.37.2.66 Color3	978
5.37.2.67 Color3	979
5.37.2.68 Color3	979
5.37.2.69 Color3	979
5.37.2.70 Color3	979
5.37.2.71 Color3	980
5.37.2.72 Color3	980
5.37.2.73 Color3	980
5.37.2.74 Color3	980
5.37.2.75 Color3	981
5.37.2.76 Color3	981

5.37.2.77 Color3	981
5.37.2.78 Color3	981
5.37.2.79 Color3	982
5.37.2.80 Color3	982
5.37.2.81 Color3	982
5.37.2.82 Color3	982
5.37.2.83 Color3	983
5.37.2.84 Color3	983
5.37.2.85 Color3	983
5.37.2.86 Color3	983
5.37.2.87 Color3	984
5.37.2.88 Color3	984
5.37.2.89 Color3	984
5.37.2.90 Color3	984
5.37.2.91 Color3	985
5.37.2.92 Color3	985
5.37.2.93 Color3	985
5.37.2.94 Color3	985
5.37.2.95 Color3	986
5.37.2.96 Color3	986
5.37.2.97 Color4	986
5.37.2.98 Color4	986
5.37.2.99 Color4	987
5.37.2.100Color4	987
5.37.2.101Color4	987
5.37.2.102Color4	987
5.37.2.103Color4	988
5.37.2.104Color4	988
5.37.2.105Color4	988
5.37.2.106Color4	988
5.37.2.107Color4	989

5.37.2.108Color4	989
5.37.2.109Color4	989
5.37.2.110Color4	989
5.37.2.111Color4	990
5.37.2.112Color4	990
5.37.2.113Color4	990
5.37.2.114Color4	990
5.37.2.115Color4	991
5.37.2.116Color4	991
5.37.2.117Color4	991
5.37.2.118Color4	991
5.37.2.119Color4	992
5.37.2.120Color4	992
5.37.2.121Color4	992
5.37.2.122Color4	992
5.37.2.123Color4	993
5.37.2.124Color4	993
5.37.2.125Color4	993
5.37.2.126Color4	993
5.37.2.127Color4	994
5.37.2.128Color4	994
5.37.2.129ColorMask	994
5.37.2.130ColorMask	994
5.37.2.131ColorMask	995
5.37.2.132ColorMaterial	995
5.37.2.133ColorPointer	995
5.37.2.134ColorPointer< T3 >	996
5.37.2.135ColorPointer< T3 >	996
5.37.2.136ColorPointer< T3 >	997
5.37.2.137ColorPointer< T3 >	997
5.37.2.138ColorSubTable	998

5.37.2.139ColorSubTable< T5 >	999
5.37.2.140ColorSubTable< T5 >	999
5.37.2.141ColorSubTable< T5 >	1000
5.37.2.142ColorSubTable< T5 >	1001
5.37.2.143ColorTable	1002
5.37.2.144ColorTable< T5 >	1003
5.37.2.145ColorTable< T5 >	1004
5.37.2.146ColorTable< T5 >	1005
5.37.2.147ColorTable< T5 >	1006
5.37.2.148ColorTableParameter	1007
5.37.2.149ColorTableParameter	1008
5.37.2.150ColorTableParameter	1008
5.37.2.151ColorTableParameter	1008
5.37.2.152ColorTableParameter	1009
5.37.2.153ColorTableParameter	1009
5.37.2.154CompileShader	1009
5.37.2.155CompileShader	1010
5.37.2.156CompressedTexImage1D	1010
5.37.2.157CompressedTexImage1D< T6 >	1011
5.37.2.158CompressedTexImage1D< T6 >	1011
5.37.2.159CompressedTexImage1D< T6 >	1012
5.37.2.160CompressedTexImage1D< T6 >	1013
5.37.2.161CompressedTexImage2D	1014
5.37.2.162CompressedTexImage2D< T7 >	1014
5.37.2.163CompressedTexImage2D< T7 >	1015
5.37.2.164CompressedTexImage2D< T7 >	1016
5.37.2.165CompressedTexImage2D< T7 >	1017
5.37.2.166CompressedTexImage3D	1018
5.37.2.167CompressedTexImage3D< T8 >	1019
5.37.2.168CompressedTexImage3D< T8 >	1020
5.37.2.169CompressedTexImage3D< T8 >	1021

5.37.2.170CompressedTexImage3D< T8 >	1022
5.37.2.171CompressedTexSubImage1D	1023
5.37.2.172CompressedTexSubImage1D< T6 >	1023
5.37.2.173CompressedTexSubImage1D< T6 >	1024
5.37.2.174CompressedTexSubImage1D< T6 >	1024
5.37.2.175CompressedTexSubImage1D< T6 >	1025
5.37.2.176CompressedTexSubImage2D	1025
5.37.2.177CompressedTexSubImage2D< T8 >	1026
5.37.2.178CompressedTexSubImage2D< T8 >	1027
5.37.2.179CompressedTexSubImage2D< T8 >	1028
5.37.2.180CompressedTexSubImage2D< T8 >	1028
5.37.2.181CompressedTexSubImage3D	1029
5.37.2.182CompressedTexSubImage3D< T10 >	1030
5.37.2.183CompressedTexSubImage3D< T10 >	1030
5.37.2.184CompressedTexSubImage3D< T10 >	1031
5.37.2.185CompressedTexSubImage3D< T10 >	1032
5.37.2.186ConvolutionFilter1D	1032
5.37.2.187ConvolutionFilter1D< T5 >	1033
5.37.2.188ConvolutionFilter1D< T5 >	1034
5.37.2.189ConvolutionFilter1D< T5 >	1035
5.37.2.190ConvolutionFilter1D< T5 >	1036
5.37.2.191ConvolutionFilter2D	1037
5.37.2.192ConvolutionFilter2D< T6 >	1038
5.37.2.193ConvolutionFilter2D< T6 >	1039
5.37.2.194ConvolutionFilter2D< T6 >	1040
5.37.2.195ConvolutionFilter2D< T6 >	1041
5.37.2.196ConvolutionParameter	1042
5.37.2.197ConvolutionParameter	1043
5.37.2.198ConvolutionParameter	1043
5.37.2.199ConvolutionParameter	1044
5.37.2.200ConvolutionParameter	1044

5.37.2.201ConvolutionParameter	1044
5.37.2.202CopyColorSubTable	1045
5.37.2.203CopyColorTable	1045
5.37.2.204CopyConvolutionFilter1D	1046
5.37.2.205CopyConvolutionFilter2D	1047
5.37.2.206CopyPixels	1047
5.37.2.207CopyTexImage1D	1048
5.37.2.208CopyTexImage2D	1049
5.37.2.209CopyTexSubImage1D	1050
5.37.2.210CopyTexSubImage2D	1050
5.37.2.211CopyTexSubImage3D	1051
5.37.2.212CreateProgram	1051
5.37.2.213CreateShader	1051
5.37.2.214CullFace	1051
5.37.2.215DeleteBuffers	1052
5.37.2.216DeleteBuffers	1052
5.37.2.217DeleteBuffers	1052
5.37.2.218DeleteBuffers	1052
5.37.2.219DeleteBuffers	1053
5.37.2.220DeleteBuffers	1053
5.37.2.221DeleteLists	1053
5.37.2.222DeleteLists	1053
5.37.2.223DeleteProgram	1054
5.37.2.224DeleteProgram	1054
5.37.2.225DeleteQueries	1054
5.37.2.226DeleteQueries	1054
5.37.2.227DeleteQueries	1055
5.37.2.228DeleteQueries	1055
5.37.2.229DeleteQueries	1055
5.37.2.230DeleteQueries	1055
5.37.2.231DeleteShader	1056

5.37.2.232DeleteShader	1056
5.37.2.233DeleteTextures	1056
5.37.2.234DeleteTextures	1056
5.37.2.235DeleteTextures	1057
5.37.2.236DeleteTextures	1057
5.37.2.237DeleteTextures	1057
5.37.2.238DeleteTextures	1057
5.37.2.239DepthFunc	1058
5.37.2.240DepthMask	1058
5.37.2.241DepthRange	1058
5.37.2.242DetachShader	1058
5.37.2.243DetachShader	1059
5.37.2.244DrawArrays	1059
5.37.2.245DrawBuffer	1059
5.37.2.246DrawBuffers	1060
5.37.2.247DrawBuffers	1060
5.37.2.248DrawBuffers	1060
5.37.2.249DrawElements	1060
5.37.2.250DrawElements< T3 >	1061
5.37.2.251DrawElements< T3 >	1061
5.37.2.252DrawElements< T3 >	1062
5.37.2.253DrawElements< T3 >	1062
5.37.2.254DrawPixels	1063
5.37.2.255DrawPixels< T4 >	1064
5.37.2.256DrawPixels< T4 >	1064
5.37.2.257DrawPixels< T4 >	1065
5.37.2.258DrawPixels< T4 >	1066
5.37.2.259DrawRangeElements	1067
5.37.2.260DrawRangeElements	1067
5.37.2.261DrawRangeElements< T5 >	1068
5.37.2.262DrawRangeElements< T5 >	1068

5.37.2.263	DrawRangeElements< T5 >	1069
5.37.2.264	DrawRangeElements< T5 >	1069
5.37.2.265	DrawRangeElements< T5 >	1070
5.37.2.266	DrawRangeElements< T5 >	1071
5.37.2.267	DrawRangeElements< T5 >	1071
5.37.2.268	DrawRangeElements< T5 >	1072
5.37.2.269	EdgeFlag	1072
5.37.2.270	EdgeFlag	1073
5.37.2.271	EdgeFlagPointer	1073
5.37.2.272	EdgeFlagPointer< T1 >	1073
5.37.2.273	EdgeFlagPointer< T1 >	1073
5.37.2.274	EdgeFlagPointer< T1 >	1074
5.37.2.275	EdgeFlagPointer< T1 >	1074
5.37.2.276	Enable	1075
5.37.2.277	Enable	1075
5.37.2.278	Enable	1075
5.37.2.279	EnableClientState	1075
5.37.2.280	EnableVertexAttribArray	1076
5.37.2.281	EnableVertexAttribArray	1076
5.37.2.282	EvalCoord1	1076
5.37.2.283	EvalCoord1	1076
5.37.2.284	EvalCoord1	1077
5.37.2.285	EvalCoord1	1077
5.37.2.286	EvalCoord2	1077
5.37.2.287	EvalCoord2	1078
5.37.2.288	EvalCoord2	1078
5.37.2.289	EvalCoord2	1078
5.37.2.290	EvalCoord2	1079
5.37.2.291	EvalCoord2	1079
5.37.2.292	EvalCoord2	1079
5.37.2.293	EvalCoord2	1080

5.37.2.294EvalMesh1	1080
5.37.2.295EvalMesh2	1080
5.37.2.296EvalPoint1	1080
5.37.2.297EvalPoint2	1081
5.37.2.298FeedbackBuffer	1081
5.37.2.299FeedbackBuffer	1081
5.37.2.300FeedbackBuffer	1082
5.37.2.301Finish	1082
5.37.2.302Flush	1082
5.37.2.303Fog	1082
5.37.2.304Fog	1082
5.37.2.305Fog	1083
5.37.2.306Fog	1083
5.37.2.307Fog	1083
5.37.2.308Fog	1084
5.37.2.309FogCoord	1084
5.37.2.310FogCoord	1084
5.37.2.311IFogCoord	1084
5.37.2.312FogCoord	1085
5.37.2.313FogCoordPointer	1085
5.37.2.314FogCoordPointer< T2 >	1085
5.37.2.315FogCoordPointer< T2 >	1086
5.37.2.316FogCoordPointer< T2 >	1086
5.37.2.317FogCoordPointer< T2 >	1086
5.37.2.318FrontFace	1087
5.37.2.319Frustum	1087
5.37.2.320GenBuffers	1087
5.37.2.321GenBuffers	1088
5.37.2.322GenBuffers	1088
5.37.2.323GenBuffers	1088
5.37.2.324GenBuffers	1088

5.37.2.325GenBuffers	1089
5.37.2.326GenLists	1089
5.37.2.327GenQueries	1089
5.37.2.328GenQueries	1089
5.37.2.329GenQueries	1090
5.37.2.330GenQueries	1090
5.37.2.331GenQueries	1090
5.37.2.332GenQueries	1090
5.37.2.333GenTextures	1091
5.37.2.334GenTextures	1091
5.37.2.335GenTextures	1091
5.37.2.336GenTextures	1091
5.37.2.337GenTextures	1092
5.37.2.338GenTextures	1092
5.37.2.339GetActiveAttrib	1092
5.37.2.340GetActiveAttrib	1093
5.37.2.341GetActiveAttrib	1093
5.37.2.342GetActiveAttrib	1094
5.37.2.343GetActiveUniform	1094
5.37.2.344GetActiveUniform	1095
5.37.2.345GetActiveUniform	1095
5.37.2.346GetActiveUniform	1096
5.37.2.347GetAttachedShaders	1096
5.37.2.348GetAttachedShaders	1097
5.37.2.349GetAttachedShaders	1097
5.37.2.350GetAttachedShaders	1097
5.37.2.351GetAttachedShaders	1098
5.37.2.352GetAttachedShaders	1098
5.37.2.353GetAttribLocation	1098
5.37.2.354GetAttribLocation	1099
5.37.2.355GetBufferParameter	1099

5.37.2.356GetBufferParameter	1099
5.37.2.357GetBufferParameter	1100
5.37.2.358GetBufferPointer	1100
5.37.2.359GetBufferPointer< T2 >	1100
5.37.2.360GetBufferPointer< T2 >	1101
5.37.2.361GetBufferPointer< T2 >	1101
5.37.2.362GetBufferPointer< T2 >	1102
5.37.2.363GetBufferSubData	1102
5.37.2.364GetBufferSubData< T3 >	1103
5.37.2.365GetBufferSubData< T3 >	1103
5.37.2.366GetBufferSubData< T3 >	1104
5.37.2.367GetBufferSubData< T3 >	1104
5.37.2.368GetClipPlane	1105
5.37.2.369GetClipPlane	1105
5.37.2.370GetClipPlane	1105
5.37.2.371GetColorTable	1106
5.37.2.372GetColorTable< T3 >	1106
5.37.2.373GetColorTable< T3 >	1107
5.37.2.374GetColorTable< T3 >	1108
5.37.2.375GetColorTable< T3 >	1108
5.37.2.376GetColorTableParameter	1109
5.37.2.377GetColorTableParameter	1110
5.37.2.378GetColorTableParameter	1110
5.37.2.379GetColorTableParameter	1111
5.37.2.380GetColorTableParameter	1111
5.37.2.381GetColorTableParameter	1112
5.37.2.382GetCompressedTexImage	1112
5.37.2.383GetCompressedTexImage< T2 >	1113
5.37.2.384GetCompressedTexImage< T2 >	1113
5.37.2.385GetCompressedTexImage< T2 >	1114
5.37.2.386GetCompressedTexImage< T2 >	1114

5.37.2.387GetConvolutionFilter	1115
5.37.2.388GetConvolutionFilter< T3 >	1116
5.37.2.389GetConvolutionFilter< T3 >	1116
5.37.2.390GetConvolutionFilter< T3 >	1117
5.37.2.391GetConvolutionFilter< T3 >	1118
5.37.2.392GetConvolutionParameter	1118
5.37.2.393GetConvolutionParameter	1119
5.37.2.394GetConvolutionParameter	1119
5.37.2.395GetConvolutionParameter	1120
5.37.2.396GetConvolutionParameter	1120
5.37.2.397GetConvolutionParameter	1121
5.37.2.398GetError	1121
5.37.2.399GetHistogram	1121
5.37.2.400GetHistogram< T4 >	1122
5.37.2.401GetHistogram< T4 >	1123
5.37.2.402GetHistogram< T4 >	1124
5.37.2.403GetHistogram< T4 >	1124
5.37.2.404GetHistogramParameter	1125
5.37.2.405GetHistogramParameter	1126
5.37.2.406GetHistogramParameter	1126
5.37.2.407GetHistogramParameter	1126
5.37.2.408GetHistogramParameter	1127
5.37.2.409GetHistogramParameter	1127
5.37.2.410GetLight	1128
5.37.2.411GetLight	1128
5.37.2.412GetLight	1129
5.37.2.413GetLight	1129
5.37.2.414GetLight	1130
5.37.2.415GetLight	1130
5.37.2.416GetMap	1131
5.37.2.417GetMap	1131

5.37.2.418GetMap	1132
5.37.2.419GetMap	1132
5.37.2.420GetMap	1133
5.37.2.421GetMap	1133
5.37.2.422GetMap	1134
5.37.2.423GetMap	1134
5.37.2.424GetMap	1135
5.37.2.425GetMaterial	1135
5.37.2.426GetMaterial	1135
5.37.2.427GetMaterial	1136
5.37.2.428GetMaterial	1136
5.37.2.429GetMaterial	1137
5.37.2.430GetMaterial	1137
5.37.2.431GetMinmax	1137
5.37.2.432GetMinmax< T4 >	1138
5.37.2.433GetMinmax< T4 >	1139
5.37.2.434GetMinmax< T4 >	1140
5.37.2.435GetMinmax< T4 >	1140
5.37.2.436GetMinmaxParameter	1141
5.37.2.437GetMinmaxParameter	1141
5.37.2.438GetMinmaxParameter	1142
5.37.2.439GetMinmaxParameter	1142
5.37.2.440GetMinmaxParameter	1142
5.37.2.441GetMinmaxParameter	1143
5.37.2.442GetPixelMap	1143
5.37.2.443GetPixelMap	1143
5.37.2.444GetPixelMap	1144
5.37.2.445GetPixelMap	1144
5.37.2.446GetPixelMap	1144
5.37.2.447GetPixelMap	1145
5.37.2.448GetPixelMap	1145

5.37.2.449GetPixelMap	1145
5.37.2.450GetPixelMap	1146
5.37.2.451GetPixelMap	1146
5.37.2.452GetPixelMap	1146
5.37.2.453GetPixelMap	1147
5.37.2.454GetPixelMap	1147
5.37.2.455GetPixelMap	1147
5.37.2.456GetPixelMap	1148
5.37.2.457GetPointer	1148
5.37.2.458GetPointer< T1 >	1149
5.37.2.459GetPointer< T1 >	1149
5.37.2.460GetPointer< T1 >	1150
5.37.2.461GetPointer< T1 >	1150
5.37.2.462GetPolygonStipple	1151
5.37.2.463GetPolygonStipple	1151
5.37.2.464GetPolygonStipple	1151
5.37.2.465GetProgram	1151
5.37.2.466GetProgram	1152
5.37.2.467GetProgram	1152
5.37.2.468GetProgram	1152
5.37.2.469GetProgram	1153
5.37.2.470GetProgram	1153
5.37.2.471GetProgramInfoLog	1154
5.37.2.472GetProgramInfoLog	1154
5.37.2.473GetProgramInfoLog	1154
5.37.2.474GetProgramInfoLog	1155
5.37.2.475GetQuery	1155
5.37.2.476GetQuery	1155
5.37.2.477GetQuery	1156
5.37.2.478GetQueryObject	1156
5.37.2.479GetQueryObject	1156

5.37.2.480GetQueryObject	1157
5.37.2.481GetQueryObject	1157
5.37.2.482GetQueryObject	1157
5.37.2.483GetQueryObject	1158
5.37.2.484GetQueryObject	1158
5.37.2.485GetQueryObject	1158
5.37.2.486GetQueryObject	1159
5.37.2.487GetSeparableFilter	1159
5.37.2.488GetSeparableFilter< T3, T4, T5 >	1160
5.37.2.489GetSeparableFilter< T3, T4, T5 >	1160
5.37.2.490GetSeparableFilter< T3, T4, T5 >	1161
5.37.2.491GetSeparableFilter< T3, T4, T5 >	1162
5.37.2.492GetSeparableFilter< T4, T5 >	1163
5.37.2.493GetSeparableFilter< T4, T5 >	1164
5.37.2.494GetSeparableFilter< T4, T5 >	1164
5.37.2.495GetSeparableFilter< T4, T5 >	1165
5.37.2.496GetSeparableFilter< T5 >	1166
5.37.2.497GetSeparableFilter< T5 >	1167
5.37.2.498GetSeparableFilter< T5 >	1168
5.37.2.499GetSeparableFilter< T5 >	1168
5.37.2.500GetShader	1169
5.37.2.501GetShader	1169
5.37.2.502GetShader	1170
5.37.2.503GetShader	1170
5.37.2.504GetShader	1170
5.37.2.505GetShader	1171
5.37.2.506GetShaderInfoLog	1171
5.37.2.507GetShaderInfoLog	1171
5.37.2.508GetShaderInfoLog	1172
5.37.2.509GetShaderInfoLog	1172
5.37.2.510GetShaderSource	1173

5.37.2.511GetShaderSource	1173
5.37.2.512GetShaderSource	1173
5.37.2.513GetShaderSource	1174
5.37.2.514GetString	1174
5.37.2.515GetString	1174
5.37.2.516GetString	1174
5.37.2.517GetTexEnv	1175
5.37.2.518GetTexEnv	1175
5.37.2.519GetTexEnv	1176
5.37.2.520GetTexEnv	1176
5.37.2.521GetTexEnv	1177
5.37.2.522GetTexEnv	1177
5.37.2.523GetTexGen	1178
5.37.2.524GetTexGen	1178
5.37.2.525GetTexGen	1179
5.37.2.526GetTexGen	1179
5.37.2.527GetTexGen	1179
5.37.2.528GetTexGen	1180
5.37.2.529GetTexGen	1180
5.37.2.530GetTexGen	1180
5.37.2.531GetTexGen	1181
5.37.2.532GetTexImage	1181
5.37.2.533GetTexImage< T4 >	1182
5.37.2.534GetTexImage< T4 >	1183
5.37.2.535GetTexImage< T4 >	1184
5.37.2.536GetTexImage< T4 >	1185
5.37.2.537GetTexLevelParameter	1185
5.37.2.538GetTexLevelParameter	1186
5.37.2.539GetTexLevelParameter	1187
5.37.2.540GetTexLevelParameter	1187
5.37.2.541GetTexLevelParameter	1188

5.37.2.542GetTexLevelParameter	1189
5.37.2.543GetTexParameter	1189
5.37.2.544GetTexParameter	1190
5.37.2.545GetTexParameter	1191
5.37.2.546GetTexParameter	1191
5.37.2.547GetTexParameter	1192
5.37.2.548GetTexParameter	1192
5.37.2.549GetUniform	1193
5.37.2.550GetUniform	1193
5.37.2.551GetUniform	1193
5.37.2.552GetUniform	1193
5.37.2.553GetUniform	1194
5.37.2.554GetUniform	1194
5.37.2.555GetUniform	1194
5.37.2.556GetUniform	1195
5.37.2.557GetUniform	1195
5.37.2.558GetUniform	1195
5.37.2.559GetUniform	1195
5.37.2.560GetUniform	1196
5.37.2.561GetUniform	1196
5.37.2.562GetUniform	1196
5.37.2.563GetUniform	1197
5.37.2.564GetUniformLocation	1197
5.37.2.565GetUniformLocation	1197
5.37.2.566GetVertexAttrib	1197
5.37.2.567GetVertexAttrib	1198
5.37.2.568GetVertexAttrib	1198
5.37.2.569GetVertexAttrib	1199
5.37.2.570GetVertexAttrib	1199
5.37.2.571GetVertexAttrib	1199
5.37.2.572GetVertexAttrib	1200

5.37.2.573GetVertexAttrib	1200
5.37.2.574GetVertexAttrib	1201
5.37.2.575GetVertexAttrib	1201
5.37.2.576GetVertexAttrib	1201
5.37.2.577GetVertexAttrib	1202
5.37.2.578GetVertexAttrib	1202
5.37.2.579GetVertexAttrib	1203
5.37.2.580GetVertexAttrib	1203
5.37.2.581GetVertexAttrib	1203
5.37.2.582GetVertexAttrib	1204
5.37.2.583GetVertexAttrib	1204
5.37.2.584GetVertexAttribPointer	1205
5.37.2.585GetVertexAttribPointer	1205
5.37.2.586GetVertexAttribPointer< T2 >	1205
5.37.2.587GetVertexAttribPointer< T2 >	1206
5.37.2.588GetVertexAttribPointer< T2 >	1206
5.37.2.589GetVertexAttribPointer< T2 >	1207
5.37.2.590GetVertexAttribPointer< T2 >	1207
5.37.2.591GetVertexAttribPointer< T2 >	1207
5.37.2.592GetVertexAttribPointer< T2 >	1208
5.37.2.593GetVertexAttribPointer< T2 >	1208
5.37.2.594Hint	1209
5.37.2.595Histogram	1209
5.37.2.596Index	1210
5.37.2.597Index	1210
5.37.2.598Index	1210
5.37.2.599Index	1210
5.37.2.600Index	1211
5.37.2.601Index	1211
5.37.2.602Index	1211
5.37.2.603Index	1211

5.37.2.604Index	1211
5.37.2.605Index	1212
5.37.2.606IndexMask	1212
5.37.2.607IndexMask	1212
5.37.2.608IndexPointer	1212
5.37.2.609IndexPointer< T2 >	1213
5.37.2.610IndexPointer< T2 >	1213
5.37.2.611IndexPointer< T2 >	1213
5.37.2.612IndexPointer< T2 >	1214
5.37.2.613InitNames	1214
5.37.2.614InterleavedArrays	1214
5.37.2.615InterleavedArrays< T2 >	1215
5.37.2.616InterleavedArrays< T2 >	1215
5.37.2.617InterleavedArrays< T2 >	1216
5.37.2.618InterleavedArrays< T2 >	1216
5.37.2.619IsBuffer	1217
5.37.2.620IsBuffer	1217
5.37.2.621IsEnabled	1217
5.37.2.622IsEnabled	1217
5.37.2.623IsEnabled	1217
5.37.2.624IsList	1218
5.37.2.625IsList	1218
5.37.2.626IsProgram	1218
5.37.2.627IsProgram	1218
5.37.2.628IsQuery	1219
5.37.2.629IsQuery	1219
5.37.2.630IsShader	1219
5.37.2.631IsShader	1219
5.37.2.632IsTexture	1219
5.37.2.633IsTexture	1220
5.37.2.634Light	1220

5.37.2.635Light	1220
5.37.2.636Light	1221
5.37.2.637Light	1221
5.37.2.638Light	1222
5.37.2.639Light	1222
5.37.2.640LightModel	1223
5.37.2.641LightModel	1223
5.37.2.642LightModel	1223
5.37.2.643LightModel	1224
5.37.2.644LightModel	1224
5.37.2.645LightModel	1224
5.37.2.646LineStipple	1225
5.37.2.647LineStipple	1225
5.37.2.648LineWidth	1225
5.37.2.649LinkProgram	1225
5.37.2.650LinkProgram	1226
5.37.2.651ListBase	1226
5.37.2.652ListBase	1226
5.37.2.653LoadAll	1226
5.37.2.654LoadIdentity	1226
5.37.2.655LoadMatrix	1227
5.37.2.656LoadMatrix	1227
5.37.2.657LoadMatrix	1227
5.37.2.658LoadMatrix	1227
5.37.2.659LoadMatrix	1228
5.37.2.660LoadMatrix	1228
5.37.2.661LoadName	1228
5.37.2.662LoadName	1228
5.37.2.663LoadTransposeMatrix	1228
5.37.2.664LoadTransposeMatrix	1229
5.37.2.665LoadTransposeMatrix	1229

5.37.2.666LoadTransposeMatrix	1229
5.37.2.667LoadTransposeMatrix	1229
5.37.2.668LoadTransposeMatrix	1230
5.37.2.669LogicOp	1230
5.37.2.670Map1	1230
5.37.2.671Map1	1231
5.37.2.672Map1	1231
5.37.2.673Map1	1232
5.37.2.674Map1	1233
5.37.2.675Map1	1233
5.37.2.676Map2	1234
5.37.2.677Map2	1235
5.37.2.678Map2	1236
5.37.2.679Map2	1237
5.37.2.680Map2	1238
5.37.2.681Map2	1239
5.37.2.682MapBuffer	1240
5.37.2.683MapGrid1	1240
5.37.2.684MapGrid1	1240
5.37.2.685MapGrid2	1241
5.37.2.686MapGrid2	1241
5.37.2.687Material	1242
5.37.2.688Material	1242
5.37.2.689Material	1242
5.37.2.690Material	1243
5.37.2.691Material	1243
5.37.2.692Material	1243
5.37.2.693MatrixMode	1244
5.37.2.694Minmax	1244
5.37.2.695MultiDrawArrays	1244
5.37.2.696MultiDrawArrays	1245

5.37.2.697MultiDrawArrays	1245
5.37.2.698MultiDrawElements	1246
5.37.2.699MultiDrawElements	1246
5.37.2.700MultiDrawElements	1247
5.37.2.701MultiDrawElements< T3 >	1247
5.37.2.702MultiDrawElements< T3 >	1248
5.37.2.703MultiDrawElements< T3 >	1248
5.37.2.704MultiDrawElements< T3 >	1249
5.37.2.705MultiDrawElements< T3 >	1249
5.37.2.706MultiDrawElements< T3 >	1250
5.37.2.707MultiDrawElements< T3 >	1250
5.37.2.708MultiDrawElements< T3 >	1251
5.37.2.709MultiDrawElements< T3 >	1252
5.37.2.710MultiDrawElements< T3 >	1252
5.37.2.711MultiDrawElements< T3 >	1253
5.37.2.712MultiDrawElements< T3 >	1253
5.37.2.713MultiTexCoord1	1254
5.37.2.714MultiTexCoord1	1254
5.37.2.715MultiTexCoord1	1255
5.37.2.716MultiTexCoord1	1255
5.37.2.717MultiTexCoord1	1255
5.37.2.718MultiTexCoord1	1256
5.37.2.719MultiTexCoord1	1256
5.37.2.720MultiTexCoord1	1256
5.37.2.721MultiTexCoord2	1257
5.37.2.722MultiTexCoord2	1257
5.37.2.723MultiTexCoord2	1258
5.37.2.724MultiTexCoord2	1258
5.37.2.725MultiTexCoord2	1258
5.37.2.726MultiTexCoord2	1259
5.37.2.727MultiTexCoord2	1259

5.37.2.728MultiTexCoord2	1259
5.37.2.729MultiTexCoord2	1260
5.37.2.730MultiTexCoord2	1260
5.37.2.731MultiTexCoord2	1261
5.37.2.732MultiTexCoord2	1261
5.37.2.733MultiTexCoord2	1261
5.37.2.734MultiTexCoord2	1262
5.37.2.735MultiTexCoord2	1262
5.37.2.736MultiTexCoord2	1262
5.37.2.737MultiTexCoord3	1263
5.37.2.738MultiTexCoord3	1263
5.37.2.739MultiTexCoord3	1264
5.37.2.740MultiTexCoord3	1264
5.37.2.741MultiTexCoord3	1264
5.37.2.742MultiTexCoord3	1265
5.37.2.743MultiTexCoord3	1265
5.37.2.744MultiTexCoord3	1265
5.37.2.745MultiTexCoord3	1266
5.37.2.746MultiTexCoord3	1266
5.37.2.747MultiTexCoord3	1267
5.37.2.748MultiTexCoord3	1267
5.37.2.749MultiTexCoord3	1267
5.37.2.750MultiTexCoord3	1268
5.37.2.751MultiTexCoord3	1268
5.37.2.752MultiTexCoord3	1268
5.37.2.753MultiTexCoord4	1269
5.37.2.754MultiTexCoord4	1269
5.37.2.755MultiTexCoord4	1270
5.37.2.756MultiTexCoord4	1270
5.37.2.757MultiTexCoord4	1270
5.37.2.758MultiTexCoord4	1271

5.37.2.759MultiTexCoord4	1271
5.37.2.760MultiTexCoord4	1271
5.37.2.761MultiTexCoord4	1272
5.37.2.762MultiTexCoord4	1272
5.37.2.763MultiTexCoord4	1273
5.37.2.764MultiTexCoord4	1273
5.37.2.765MultiTexCoord4	1273
5.37.2.766MultiTexCoord4	1274
5.37.2.767MultiTexCoord4	1274
5.37.2.768MultiTexCoord4	1274
5.37.2.769MultMatrix	1275
5.37.2.770MultMatrix	1275
5.37.2.771MultMatrix	1275
5.37.2.772MultMatrix	1276
5.37.2.773MultMatrix	1276
5.37.2.774MultMatrix	1276
5.37.2.775MultTransposeMatrix	1276
5.37.2.776MultTransposeMatrix	1277
5.37.2.777MultTransposeMatrix	1277
5.37.2.778MultTransposeMatrix	1277
5.37.2.779MultTransposeMatrix	1277
5.37.2.780MultTransposeMatrix	1278
5.37.2.781NewList	1278
5.37.2.782NewList	1278
5.37.2.783Normal3	1278
5.37.2.784Normal3	1279
5.37.2.785Normal3	1279
5.37.2.786Normal3	1279
5.37.2.787Normal3	1279
5.37.2.788Normal3	1280
5.37.2.789Normal3	1280

5.37.2.790Normal3	1280
5.37.2.791Normal3	1280
5.37.2.792Normal3	1281
5.37.2.793Normal3	1281
5.37.2.794Normal3	1281
5.37.2.795Normal3	1281
5.37.2.796Normal3	1282
5.37.2.797Normal3	1282
5.37.2.798Normal3	1282
5.37.2.799Normal3	1282
5.37.2.800Normal3	1283
5.37.2.801Normal3	1283
5.37.2.802Normal3	1283
5.37.2.803Normal3	1283
5.37.2.804Normal3	1284
5.37.2.805Normal3	1284
5.37.2.806Normal3	1284
5.37.2.807NormalPointer	1284
5.37.2.808NormalPointer< T2 >	1285
5.37.2.809NormalPointer< T2 >	1285
5.37.2.810NormalPointer< T2 >	1286
5.37.2.811NormalPointer< T2 >	1286
5.37.2.812Ortho	1287
5.37.2.813PassThrough	1287
5.37.2.814PixelMap	1287
5.37.2.815PixelMap	1288
5.37.2.816PixelMap	1288
5.37.2.817PixelMap	1288
5.37.2.818PixelMap	1289
5.37.2.819PixelMap	1289
5.37.2.820PixelMap	1289

5.37.2.821PixelMap	1290
5.37.2.822PixelMap	1290
5.37.2.823PixelMap	1291
5.37.2.824PixelMap	1291
5.37.2.825PixelMap	1291
5.37.2.826PixelMap	1292
5.37.2.827PixelMap	1292
5.37.2.828PixelMap	1292
5.37.2.829PixelStore	1293
5.37.2.830PixelStore	1293
5.37.2.831PixelTransfer	1294
5.37.2.832PixelTransfer	1294
5.37.2.833PixelZoom	1295
5.37.2.834PointParameter	1295
5.37.2.835PointParameter	1296
5.37.2.836PointParameter	1296
5.37.2.837PointParameter	1296
5.37.2.838PointParameter	1297
5.37.2.839PointParameter	1297
5.37.2.840PointParameter	1297
5.37.2.841PointSize	1297
5.37.2.842PolygonMode	1298
5.37.2.843PolygonOffset	1298
5.37.2.844PolygonStipple	1298
5.37.2.845PolygonStipple	1298
5.37.2.846PolygonStipple	1299
5.37.2.847PrioritizeTextures	1299
5.37.2.848PrioritizeTextures	1299
5.37.2.849PrioritizeTextures	1300
5.37.2.850PrioritizeTextures	1300
5.37.2.851PrioritizeTextures	1300

5.37.2.852PrioritizeTextures	1301
5.37.2.853PushAttrib	1301
5.37.2.854PushClientAttrib	1301
5.37.2.855PushMatrix	1301
5.37.2.856PushName	1301
5.37.2.857PushName	1302
5.37.2.858RasterPos2	1302
5.37.2.859RasterPos2	1302
5.37.2.860RasterPos2	1302
5.37.2.861RasterPos2	1303
5.37.2.862RasterPos2	1303
5.37.2.863RasterPos2	1303
5.37.2.864RasterPos2	1303
5.37.2.865RasterPos2	1303
5.37.2.866RasterPos2	1304
5.37.2.867RasterPos2	1304
5.37.2.868RasterPos2	1304
5.37.2.869RasterPos2	1304
5.37.2.870RasterPos2	1304
5.37.2.871RasterPos2	1305
5.37.2.872RasterPos2	1305
5.37.2.873RasterPos2	1305
5.37.2.874RasterPos3	1305
5.37.2.875RasterPos3	1305
5.37.2.876RasterPos3	1306
5.37.2.877RasterPos3	1306
5.37.2.878RasterPos3	1306
5.37.2.879RasterPos3	1306
5.37.2.880RasterPos3	1306
5.37.2.881RasterPos3	1307
5.37.2.882RasterPos3	1307

5.37.2.883RasterPos3	1307
5.37.2.884RasterPos3	1307
5.37.2.885RasterPos3	1307
5.37.2.886RasterPos3	1308
5.37.2.887RasterPos3	1308
5.37.2.888RasterPos3	1308
5.37.2.889RasterPos3	1308
5.37.2.890RasterPos4	1308
5.37.2.891RasterPos4	1309
5.37.2.892RasterPos4	1309
5.37.2.893RasterPos4	1309
5.37.2.894RasterPos4	1309
5.37.2.895RasterPos4	1309
5.37.2.896RasterPos4	1310
5.37.2.897RasterPos4	1310
5.37.2.898RasterPos4	1310
5.37.2.899RasterPos4	1310
5.37.2.900RasterPos4	1310
5.37.2.901RasterPos4	1311
5.37.2.902RasterPos4	1311
5.37.2.903RasterPos4	1311
5.37.2.904RasterPos4	1311
5.37.2.905RasterPos4	1311
5.37.2.906ReadBuffer	1312
5.37.2.907ReadPixels	1312
5.37.2.908ReadPixels< T6 >	1313
5.37.2.909ReadPixels< T6 >	1313
5.37.2.910ReadPixels< T6 >	1314
5.37.2.911ReadPixels< T6 >	1315
5.37.2.912Rect	1316
5.37.2.913Rect	1316

5.37.2.914Rect	1316
5.37.2.915Rect	1316
5.37.2.916Rect	1317
5.37.2.917Rect	1317
5.37.2.918Rect	1317
5.37.2.919Rect	1317
5.37.2.920Rect	1318
5.37.2.921Rect	1318
5.37.2.922Rect	1318
5.37.2.923Rect	1318
5.37.2.924Rect	1319
5.37.2.925Rect	1319
5.37.2.926Rect	1319
5.37.2.927RenderMode	1319
5.37.2.928ResetHistogram	1320
5.37.2.929ResetMinmax	1320
5.37.2.930Rotate	1320
5.37.2.931Rotate	1320
5.37.2.932SampleCoverage	1321
5.37.2.933Scale	1321
5.37.2.934Scale	1321
5.37.2.935Scissor	1321
5.37.2.936SecondaryColor3	1322
5.37.2.937SecondaryColor3	1322
5.37.2.938SecondaryColor3	1322
5.37.2.939SecondaryColor3	1322
5.37.2.940SecondaryColor3	1322
5.37.2.941SecondaryColor3	1323
5.37.2.942SecondaryColor3	1323
5.37.2.943SecondaryColor3	1323
5.37.2.944SecondaryColor3	1323

5.37.2.945SecondaryColor3	1323
5.37.2.946SecondaryColor3	1324
5.37.2.947SecondaryColor3	1324
5.37.2.948SecondaryColor3	1324
5.37.2.949SecondaryColor3	1324
5.37.2.950SecondaryColor3	1324
5.37.2.951SecondaryColor3	1325
5.37.2.952SecondaryColor3	1325
5.37.2.953SecondaryColor3	1325
5.37.2.954SecondaryColor3	1325
5.37.2.955SecondaryColor3	1325
5.37.2.956SecondaryColor3	1326
5.37.2.957SecondaryColor3	1326
5.37.2.958SecondaryColor3	1326
5.37.2.959SecondaryColor3	1326
5.37.2.960SecondaryColor3	1326
5.37.2.961SecondaryColor3	1327
5.37.2.962SecondaryColor3	1327
5.37.2.963SecondaryColor3	1327
5.37.2.964SecondaryColor3	1327
5.37.2.965SecondaryColor3	1327
5.37.2.966SecondaryColor3	1328
5.37.2.967SecondaryColor3	1328
5.37.2.968SecondaryColorPointer	1328
5.37.2.969SecondaryColorPointer< T3 >	1328
5.37.2.970SecondaryColorPointer< T3 >	1329
5.37.2.971SecondaryColorPointer< T3 >	1330
5.37.2.972SecondaryColorPointer< T3 >	1330
5.37.2.973SelectBuffer	1331
5.37.2.974SelectBuffer	1331
5.37.2.975SelectBuffer	1331

5.37.2.976SelectBuffer	1331
5.37.2.977SelectBuffer	1332
5.37.2.978SelectBuffer	1332
5.37.2.979SeparableFilter2D	1332
5.37.2.980SeparableFilter2D< T6, T7 >	1333
5.37.2.981SeparableFilter2D< T6, T7 >	1334
5.37.2.982SeparableFilter2D< T6, T7 >	1336
5.37.2.983SeparableFilter2D< T6, T7 >	1337
5.37.2.984SeparableFilter2D< T7 >	1338
5.37.2.985SeparableFilter2D< T7 >	1339
5.37.2.986SeparableFilter2D< T7 >	1340
5.37.2.987SeparableFilter2D< T7 >	1342
5.37.2.988ShadeModel	1343
5.37.2.989ShaderSource	1343
5.37.2.990ShaderSource	1343
5.37.2.991ShaderSource	1344
5.37.2.992ShaderSource	1344
5.37.2.993StencilFunc	1344
5.37.2.994StencilFunc	1345
5.37.2.995StencilFuncSeparate	1345
5.37.2.996StencilFuncSeparate	1346
5.37.2.997StencilMask	1346
5.37.2.998StencilMask	1346
5.37.2.999StencilMaskSeparate	1347
5.37.2.1000StencilMaskSeparate	1347
5.37.2.1001StencilOp	1347
5.37.2.1002StencilOpSeparate	1348
5.37.2.1003TexCoord1	1348
5.37.2.1004TexCoord1	1349
5.37.2.1005TexCoord1	1349
5.37.2.1006TexCoord1	1349

5.37.2.100TexCoord1	1349
5.37.2.100TexCoord1	1350
5.37.2.100TexCoord1	1350
5.37.2.101TexCoord1	1350
5.37.2.101TexCoord2	1350
5.37.2.101TexCoord2	1351
5.37.2.101TexCoord2	1351
5.37.2.101TexCoord2	1351
5.37.2.101TexCoord2	1351
5.37.2.101TexCoord2	1351
5.37.2.101TexCoord2	1352
5.37.2.101TexCoord2	1352
5.37.2.101TexCoord2	1352
5.37.2.101TexCoord2	1352
5.37.2.102TexCoord2	1353
5.37.2.102TexCoord2	1353
5.37.2.102TexCoord2	1353
5.37.2.102TexCoord2	1353
5.37.2.102TexCoord2	1354
5.37.2.102TexCoord2	1354
5.37.2.102TexCoord2	1354
5.37.2.102TexCoord3	1354
5.37.2.102TexCoord3	1355
5.37.2.102TexCoord3	1355
5.37.2.103TexCoord3	1355
5.37.2.103TexCoord3	1355
5.37.2.103TexCoord3	1356
5.37.2.103TexCoord3	1356
5.37.2.103TexCoord3	1356
5.37.2.103TexCoord3	1356
5.37.2.103TexCoord3	1357
5.37.2.103TexCoord3	1357

5.37.2.1033TexCoord3	1357
5.37.2.1034TexCoord3	1357
5.37.2.1040TexCoord3	1358
5.37.2.1041TexCoord3	1358
5.37.2.1042TexCoord3	1358
5.37.2.1043TexCoord4	1358
5.37.2.1044TexCoord4	1359
5.37.2.1045TexCoord4	1359
5.37.2.1046TexCoord4	1359
5.37.2.1047TexCoord4	1359
5.37.2.1048TexCoord4	1360
5.37.2.1049TexCoord4	1360
5.37.2.1050TexCoord4	1360
5.37.2.1051TexCoord4	1360
5.37.2.1052TexCoord4	1361
5.37.2.1053TexCoord4	1361
5.37.2.1054TexCoord4	1361
5.37.2.1055TexCoord4	1361
5.37.2.1056TexCoord4	1362
5.37.2.1057TexCoord4	1362
5.37.2.1058TexCoord4	1362
5.37.2.1059TexCoordPointer	1362
5.37.2.1060TexCoordPointer< T3 >	1363
5.37.2.1061TexCoordPointer< T3 >	1363
5.37.2.1062TexCoordPointer< T3 >	1364
5.37.2.1063TexCoordPointer< T3 >	1364
5.37.2.1064TexEnv	1365
5.37.2.1065TexEnv	1366
5.37.2.1066TexEnv	1366
5.37.2.1067TexEnv	1367
5.37.2.1068TexEnv	1368

5.37.2.106	TexEnv	1368
5.37.2.107	TexGen	1369
5.37.2.107	TexGen	1369
5.37.2.107	TexGen	1370
5.37.2.107	TexGen	1370
5.37.2.107	TexGen	1371
5.37.2.107	TexGen	1371
5.37.2.107	TexGen	1371
5.37.2.107	TexGen	1372
5.37.2.107	TexGen	1372
5.37.2.107	TexImage1D	1373
5.37.2.108	TexImage1D < T7 >	1374
5.37.2.108	TexImage1D < T7 >	1375
5.37.2.108	TexImage1D < T7 >	1377
5.37.2.108	TexImage1D < T7 >	1378
5.37.2.108	TexImage2D	1380
5.37.2.108	TexImage2D < T8 >	1381
5.37.2.108	TexImage2D < T8 >	1383
5.37.2.108	TexImage2D < T8 >	1384
5.37.2.108	TexImage2D < T8 >	1386
5.37.2.108	TexImage3D	1387
5.37.2.109	TexImage3D < T9 >	1389
5.37.2.109	TexImage3D < T9 >	1390
5.37.2.109	TexImage3D < T9 >	1392
5.37.2.109	TexImage3D < T9 >	1393
5.37.2.109	TexParameter	1395
5.37.2.109	TexParameter	1395
5.37.2.109	TexParameter	1396
5.37.2.109	TexParameter	1396
5.37.2.109	TexParameter	1397
5.37.2.109	TexParameter	1397

5.37.2.110TexSubImage1D	1398
5.37.2.110TexSubImage1D< T6 >	1399
5.37.2.110TexSubImage1D< T6 >	1399
5.37.2.110TexSubImage1D< T6 >	1400
5.37.2.110TexSubImage1D< T6 >	1401
5.37.2.110TexSubImage2D	1402
5.37.2.110TexSubImage2D< T8 >	1403
5.37.2.110TexSubImage2D< T8 >	1404
5.37.2.110TexSubImage2D< T8 >	1405
5.37.2.110TexSubImage2D< T8 >	1406
5.37.2.111TexSubImage3D	1407
5.37.2.111TexSubImage3D< T10 >	1408
5.37.2.111TexSubImage3D< T10 >	1409
5.37.2.111TexSubImage3D< T10 >	1410
5.37.2.111TexSubImage3D< T10 >	1411
5.37.2.111Translate	1412
5.37.2.111Translate	1412
5.37.2.111Uniform1	1412
5.37.2.111Uniform1	1412
5.37.2.111Uniform1	1413
5.37.2.112Uniform1	1413
5.37.2.112Uniform1	1413
5.37.2.112Uniform1	1413
5.37.2.112Uniform1	1414
5.37.2.112Uniform1	1414
5.37.2.112Uniform1	1414
5.37.2.112Uniform1	1414
5.37.2.112Uniform1	1415
5.37.2.112Uniform1	1415
5.37.2.112Uniform2	1415
5.37.2.113Uniform2	1415

5.37.2.113Uniform2	1416
5.37.2.113Uniform2	1416
5.37.2.113Uniform2	1416
5.37.2.113Uniform2	1416
5.37.2.113Uniform2	1417
5.37.2.113Uniform2	1417
5.37.2.113Uniform2	1417
5.37.2.113Uniform2	1417
5.37.2.113Uniform2	1418
5.37.2.114Uniform3	1418
5.37.2.114Uniform3	1418
5.37.2.114Uniform3	1418
5.37.2.114Uniform3	1419
5.37.2.114Uniform3	1419
5.37.2.114Uniform3	1419
5.37.2.114Uniform3	1419
5.37.2.114Uniform3	1420
5.37.2.114Uniform3	1420
5.37.2.114Uniform3	1420
5.37.2.115Uniform3	1420
5.37.2.115Uniform3	1421
5.37.2.115Uniform4	1421
5.37.2.115Uniform4	1421
5.37.2.115Uniform4	1421
5.37.2.115Uniform4	1422
5.37.2.115Uniform4	1422
5.37.2.115Uniform4	1422
5.37.2.115Uniform4	1422
5.37.2.115Uniform4	1423
5.37.2.116Uniform4	1423
5.37.2.116Uniform4	1423

5.37.2.1160	Uniform4	1423
5.37.2.1161	Uniform4	1424
5.37.2.1162	UseProgram	1424
5.37.2.1163	UseProgram	1424
5.37.2.1164	ValidateProgram	1424
5.37.2.1165	ValidateProgram	1425
5.37.2.1166	Vertex2	1425
5.37.2.1167	Vertex2	1425
5.37.2.1168	Vertex2	1425
5.37.2.1169	Vertex2	1426
5.37.2.1170	Vertex2	1426
5.37.2.1171	Vertex2	1426
5.37.2.1172	Vertex2	1426
5.37.2.1173	Vertex2	1426
5.37.2.1174	Vertex2	1426
5.37.2.1175	Vertex2	1427
5.37.2.1176	Vertex2	1427
5.37.2.1177	Vertex2	1427
5.37.2.1178	Vertex2	1427
5.37.2.1179	Vertex2	1428
5.37.2.1180	Vertex2	1428
5.37.2.1181	Vertex2	1428
5.37.2.1182	Vertex2	1428
5.37.2.1183	Vertex2	1429
5.37.2.1184	Vertex3	1429
5.37.2.1185	Vertex3	1429
5.37.2.1186	Vertex3	1429
5.37.2.1187	Vertex3	1430
5.37.2.1188	Vertex3	1430
5.37.2.1189	Vertex3	1430
5.37.2.1190	Vertex3	1430
5.37.2.1191	Vertex3	1431
5.37.2.1192	Vertex3	1431

5.37.2.119Vertex3	1431
5.37.2.119Vertex3	1431
5.37.2.119Vertex3	1432
5.37.2.119Vertex3	1432
5.37.2.119Vertex3	1432
5.37.2.119Vertex3	1432
5.37.2.119Vertex3	1433
5.37.2.120Vertex4	1433
5.37.2.120Vertex4	1433
5.37.2.120Vertex4	1433
5.37.2.120Vertex4	1434
5.37.2.120Vertex4	1434
5.37.2.120Vertex4	1434
5.37.2.120Vertex4	1434
5.37.2.120Vertex4	1435
5.37.2.120Vertex4	1435
5.37.2.120Vertex4	1435
5.37.2.121Vertex4	1435
5.37.2.121Vertex4	1436
5.37.2.121Vertex4	1436
5.37.2.121Vertex4	1436
5.37.2.121Vertex4	1436
5.37.2.121Vertex4	1437
5.37.2.121VertexAttrib1	1437
5.37.2.121VertexAttrib1	1437
5.37.2.121VertexAttrib1	1437
5.37.2.121VertexAttrib1	1438
5.37.2.122VertexAttrib1	1438
5.37.2.122VertexAttrib1	1438
5.37.2.122VertexAttrib1	1438
5.37.2.122VertexAttrib1	1439

5.37.2.122	VertexAttrib1	1439
5.37.2.122	VertexAttrib1	1439
5.37.2.122	VertexAttrib1	1439
5.37.2.122	VertexAttrib1	1440
5.37.2.122	VertexAttrib2	1440
5.37.2.122	VertexAttrib2	1440
5.37.2.123	VertexAttrib2	1440
5.37.2.123	VertexAttrib2	1441
5.37.2.123	VertexAttrib2	1441
5.37.2.123	VertexAttrib2	1441
5.37.2.123	VertexAttrib2	1441
5.37.2.123	VertexAttrib2	1442
5.37.2.123	VertexAttrib2	1442
5.37.2.123	VertexAttrib2	1442
5.37.2.123	VertexAttrib2	1442
5.37.2.123	VertexAttrib2	1443
5.37.2.124	VertexAttrib2	1443
5.37.2.124	VertexAttrib2	1443
5.37.2.124	VertexAttrib2	1443
5.37.2.124	VertexAttrib2	1444
5.37.2.124	VertexAttrib2	1444
5.37.2.124	VertexAttrib2	1444
5.37.2.124	VertexAttrib2	1445
5.37.2.124	VertexAttrib2	1445
5.37.2.124	VertexAttrib2	1445
5.37.2.125	VertexAttrib2	1445
5.37.2.125	VertexAttrib2	1446
5.37.2.125	VertexAttrib3	1446
5.37.2.125	VertexAttrib3	1446
5.37.2.125	VertexAttrib3	1446

5.37.2.1255VertexAttrib3	1447
5.37.2.1256VertexAttrib3	1447
5.37.2.1257VertexAttrib3	1447
5.37.2.1258VertexAttrib3	1447
5.37.2.1259VertexAttrib3	1448
5.37.2.1260VertexAttrib3	1448
5.37.2.1261VertexAttrib3	1448
5.37.2.1262VertexAttrib3	1448
5.37.2.1263VertexAttrib3	1449
5.37.2.1264VertexAttrib3	1449
5.37.2.1265VertexAttrib3	1449
5.37.2.1266VertexAttrib3	1449
5.37.2.1267VertexAttrib3	1450
5.37.2.1268VertexAttrib3	1450
5.37.2.1269VertexAttrib3	1450
5.37.2.1270VertexAttrib3	1450
5.37.2.1271VertexAttrib3	1451
5.37.2.1272VertexAttrib3	1451
5.37.2.1273VertexAttrib3	1451
5.37.2.1274VertexAttrib3	1451
5.37.2.1275VertexAttrib3	1452
5.37.2.1276VertexAttrib4	1452
5.37.2.1277VertexAttrib4	1452
5.37.2.1278VertexAttrib4	1452
5.37.2.1279VertexAttrib4	1453
5.37.2.1280VertexAttrib4	1453
5.37.2.1281VertexAttrib4	1453
5.37.2.1282VertexAttrib4	1453
5.37.2.1283VertexAttrib4	1454
5.37.2.1284VertexAttrib4	1454
5.37.2.1285VertexAttrib4	1454

5.37.2.1286VertexAttrib4	1454
5.37.2.1287VertexAttrib4	1455
5.37.2.1288VertexAttrib4	1455
5.37.2.1289VertexAttrib4	1455
5.37.2.1290VertexAttrib4	1455
5.37.2.1291VertexAttrib4	1456
5.37.2.1292VertexAttrib4	1456
5.37.2.1293VertexAttrib4	1456
5.37.2.1294VertexAttrib4	1456
5.37.2.1295VertexAttrib4	1457
5.37.2.1296VertexAttrib4	1457
5.37.2.1297VertexAttrib4	1457
5.37.2.1298VertexAttrib4	1457
5.37.2.1299VertexAttrib4	1458
5.37.2.1300VertexAttrib4	1458
5.37.2.1301VertexAttrib4	1458
5.37.2.1302VertexAttrib4	1459
5.37.2.1303VertexAttrib4	1459
5.37.2.1304VertexAttrib4	1459
5.37.2.1305VertexAttrib4	1459
5.37.2.1306VertexAttrib4	1460
5.37.2.1307VertexAttrib4	1460
5.37.2.1308VertexAttrib4	1460
5.37.2.1309VertexAttrib4	1460
5.37.2.1310VertexAttrib4	1460
5.37.2.1311VertexAttrib4	1461
5.37.2.1312VertexAttrib4	1461
5.37.2.1313VertexAttrib4	1461
5.37.2.1314VertexAttrib4	1461
5.37.2.1315VertexAttrib4	1462
5.37.2.1316VertexAttrib4	1462

5.37.2.131VertexAttrib4	1462
5.37.2.131VertexAttrib4	1462
5.37.2.131VertexAttrib4	1463
5.37.2.132VertexAttrib4	1463
5.37.2.132VertexAttribPointer	1463
5.37.2.132VertexAttribPointer	1464
5.37.2.132VertexAttribPointer< T5 >	1464
5.37.2.132VertexAttribPointer< T5 >	1465
5.37.2.132VertexAttribPointer< T5 >	1466
5.37.2.132VertexAttribPointer< T5 >	1466
5.37.2.132VertexAttribPointer< T5 >	1467
5.37.2.132VertexAttribPointer< T5 >	1468
5.37.2.132VertexAttribPointer< T5 >	1468
5.37.2.133VertexAttribPointer< T5 >	1469
5.37.2.133VertexPointer	1470
5.37.2.133VertexPointer< T3 >	1470
5.37.2.133VertexPointer< T3 >	1471
5.37.2.133VertexPointer< T3 >	1471
5.37.2.133VertexPointer< T3 >	1472
5.37.2.133Viewport	1472
5.37.2.133WindowPos2	1473
5.37.2.133WindowPos2	1473
5.37.2.133WindowPos2	1473
5.37.2.134WindowPos2	1473
5.37.2.134WindowPos2	1474
5.37.2.134WindowPos2	1474
5.37.2.134WindowPos2	1474
5.37.2.134WindowPos2	1474
5.37.2.134WindowPos2	1474
5.37.2.134WindowPos2	1475
5.37.2.134WindowPos2	1475

5.37.2.134WindowPos2	1475
5.37.2.134WindowPos2	1475
5.37.2.135WindowPos2	1475
5.37.2.135WindowPos2	1476
5.37.2.135WindowPos2	1476
5.37.2.135WindowPos3	1476
5.37.2.135WindowPos3	1476
5.37.2.135WindowPos3	1476
5.37.2.135WindowPos3	1477
5.37.2.135WindowPos3	1477
5.37.2.135WindowPos3	1477
5.37.2.135WindowPos3	1477
5.37.2.136WindowPos3	1477
5.37.2.136WindowPos3	1478
5.37.2.136WindowPos3	1478
5.37.2.136WindowPos3	1478
5.37.2.136WindowPos3	1478
5.37.2.136WindowPos3	1478
5.37.2.136WindowPos3	1479
5.37.2.136WindowPos3	1479
5.37.2.136WindowPos3	1479
5.37.3 Property Documentation	1479
5.37.3.1 SyncRoot	1479
5.38 OpenTK.GraphicsException Class Reference	1479
5.38.1 Detailed Description	1480
5.38.2 Constructor & Destructor Documentation	1480
5.38.2.1 GraphicsException	1480
5.38.2.2 GraphicsException	1480
5.39 OpenTK.Half Struct Reference	1480
5.39.1 Detailed Description	1484
5.39.2 Constructor & Destructor Documentation	1484

5.39.2.1	Half	1484
5.39.2.2	Half	1484
5.39.2.3	Half	1484
5.39.2.4	Half	1485
5.39.2.5	Half	1485
5.39.3	Member Function Documentation	1485
5.39.3.1	CompareTo	1485
5.39.3.2	Equals	1486
5.39.3.3	FromBinaryStream	1486
5.39.3.4	FromBytes	1486
5.39.3.5	GetBytes	1486
5.39.3.6	GetObjectData	1487
5.39.3.7	operator double	1487
5.39.3.8	operator float	1487
5.39.3.9	operator Half	1487
5.39.3.10	operator Half	1488
5.39.3.11	Parse	1488
5.39.3.12	Parse	1488
5.39.3.13	ToBinaryStream	1489
5.39.3.14	ToSingle	1489
5.39.3.15	ToString	1489
5.39.3.16	ToString	1489
5.39.3.17	TryParse	1490
5.39.3.18	TryParse	1490
5.39.4	Member Data Documentation	1490
5.39.4.1	Epsilon	1490
5.39.4.2	MaxValue	1490
5.39.4.3	MinNormalizedValue	1491
5.39.4.4	MinValue	1491
5.39.4.5	SizeInBytes	1491
5.39.5	Property Documentation	1491

5.39.5.1	IsNaN	1491
5.39.5.2	IsNegativeInfinity	1491
5.39.5.3	IsPositiveInfinity	1491
5.39.5.4	IsZero	1491
5.40	OpenTK.INativeWindow Interface Reference	1491
5.40.1	Detailed Description	1495
5.40.2	Member Function Documentation	1495
5.40.2.1	Close	1495
5.40.2.2	PointToClient	1495
5.40.2.3	PointToScreen	1495
5.40.2.4	ProcessEvents	1496
5.40.3	Property Documentation	1496
5.40.3.1	Bounds	1496
5.40.3.2	ClientRectangle	1496
5.40.3.3	ClientSize	1496
5.40.3.4	Exists	1496
5.40.3.5	Focused	1496
5.40.3.6	Height	1497
5.40.3.7	Icon	1497
5.40.3.8	InputDriver	1497
5.40.3.9	Location	1497
5.40.3.10	Size	1497
5.40.3.11	Title	1497
5.40.3.12	Visible	1497
5.40.3.13	Width	1498
5.40.3.14	WindowBorder	1498
5.40.3.15	WindowInfo	1498
5.40.3.16	WindowState	1498
5.40.3.17	X	1498
5.40.3.18	Y	1498
5.40.4	Event Documentation	1498

5.40.4.1	Closed	1498
5.40.4.2	Closing	1499
5.40.4.3	Disposed	1499
5.40.4.4	FocusedChanged	1499
5.40.4.5	IconChanged	1499
5.40.4.6	KeyPress	1499
5.40.4.7	MouseEnter	1499
5.40.4.8	MouseLeave	1499
5.40.4.9	Move	1500
5.40.4.10	Resize	1500
5.40.4.11	TitleChanged	1500
5.40.4.12	VisibleChanged	1500
5.40.4.13	WindowBorderChanged	1500
5.40.4.14	WindowStateChanged	1500
5.41	OpenTK.Input.GamePad Class Reference	1500
5.41.1	Detailed Description	1501
5.42	OpenTK.Input.GamePadState Struct Reference	1501
5.42.1	Detailed Description	1501
5.43	OpenTK.Input.IInputDevice Interface Reference	1501
5.43.1	Detailed Description	1501
5.43.2	Property Documentation	1502
5.43.2.1	Description	1502
5.43.2.2	DeviceType	1502
5.44	OpenTK.Input.IInputDriver Interface Reference	1502
5.44.1	Detailed Description	1502
5.44.2	Member Function Documentation	1503
5.44.2.1	Poll	1503
5.45	OpenTK.Input.IJoystickDriver Interface Reference	1503
5.45.1	Detailed Description	1503
5.45.2	Property Documentation	1503
5.45.2.1	Joysticks	1503

5.46	OpenTK.Input.IKeyboardDriver Interface Reference	1504
5.46.1	Detailed Description	1504
5.46.2	Property Documentation	1504
5.46.2.1	Keyboard	1504
5.47	OpenTK.Input.IMouseDriver Interface Reference	1504
5.47.1	Detailed Description	1505
5.47.2	Property Documentation	1505
5.47.2.1	Mouse	1505
5.48	OpenTK.Input.JoystickAxisCollection Class Reference	1505
5.48.1	Detailed Description	1506
5.48.2	Property Documentation	1506
5.48.2.1	Count	1506
5.48.2.2	this	1506
5.49	OpenTK.Input.JoystickButtonCollection Class Reference	1506
5.49.1	Detailed Description	1507
5.49.2	Property Documentation	1507
5.49.2.1	Count	1507
5.49.2.2	this	1507
5.50	OpenTK.Input.JoystickButtonEventArgs Class Reference	1507
5.50.1	Detailed Description	1508
5.50.2	Property Documentation	1508
5.50.2.1	Button	1508
5.50.2.2	Pressed	1508
5.51	OpenTK.Input.JoystickDevice Class Reference	1508
5.51.1	Detailed Description	1509
5.51.2	Member Data Documentation	1509
5.51.2.1	ButtonDown	1509
5.51.2.2	ButtonUp	1510
5.51.2.3	Move	1510
5.51.3	Property Documentation	1510
5.51.3.1	Axis	1510

5.51.3.2	Button	1510
5.51.3.3	Description	1510
5.51.3.4	DeviceType	1510
5.52	OpenTK.Input.JoystickEventArgs Class Reference	1511
5.52.1	Detailed Description	1511
5.53	OpenTK.Input.JoystickMoveEventArgs Class Reference	1511
5.53.1	Detailed Description	1512
5.53.2	Constructor & Destructor Documentation	1512
5.53.2.1	JoystickMoveEventArgs	1512
5.53.3	Property Documentation	1512
5.53.3.1	Axis	1512
5.53.3.2	Delta	1512
5.53.3.3	Value	1512
5.54	OpenTK.Input.KeyboardDevice Class Reference	1513
5.54.1	Detailed Description	1514
5.54.2	Member Function Documentation	1514
5.54.2.1	GetHashCode	1514
5.54.2.2	ToString	1514
5.54.3	Property Documentation	1515
5.54.3.1	Description	1515
5.54.3.2	DeviceID	1515
5.54.3.3	DeviceType	1515
5.54.3.4	KeyRepeat	1515
5.54.3.5	NumberOfFunctionKeys	1515
5.54.3.6	NumberOfKeys	1515
5.54.3.7	NumberOfLeds	1516
5.54.3.8	this	1516
5.54.4	Event Documentation	1516
5.54.4.1	KeyDown	1516
5.54.4.2	KeyUp	1516
5.55	OpenTK.Input.KeyboardKeyEventArgs Class Reference	1516

5.55.1 Detailed Description	1517
5.55.2 Constructor & Destructor Documentation	1517
5.55.2.1 KeyboardKeyEventArgs	1517
5.55.2.2 KeyboardKeyEventArgs	1517
5.55.3 Property Documentation	1517
5.55.3.1 Key	1517
5.56 OpenTK.Input.KeyboardState Struct Reference	1517
5.56.1 Detailed Description	1518
5.56.2 Member Function Documentation	1518
5.56.2.1 Equals	1518
5.56.2.2 IsKeyDown	1518
5.56.2.3 IsKeyUp	1519
5.57 OpenTK.Input.MouseButtonEventArgs Class Reference	1519
5.57.1 Detailed Description	1520
5.57.2 Constructor & Destructor Documentation	1520
5.57.2.1 MouseButtonEventArgs	1520
5.57.2.2 MouseButtonEventArgs	1520
5.57.2.3 MouseButtonEventArgs	1520
5.57.3 Property Documentation	1520
5.57.3.1 Button	1520
5.57.3.2 IsPressed	1521
5.58 OpenTK.Input.MouseDevice Class Reference	1521
5.58.1 Detailed Description	1522
5.58.2 Member Function Documentation	1523
5.58.2.1 GetHashCode	1523
5.58.2.2 ToString	1523
5.58.3 Property Documentation	1523
5.58.3.1 Description	1523
5.58.3.2 DeviceID	1523
5.58.3.3 DeviceType	1523
5.58.3.4 NumberOfButtons	1523

5.58.3.5	NumberOfWheels	1523
5.58.3.6	this	1524
5.58.3.7	Wheel	1524
5.58.3.8	WheelPrecise	1524
5.58.3.9	X	1524
5.58.3.10	Y	1524
5.58.4	Event Documentation	1524
5.58.4.1	ButtonDown	1524
5.58.4.2	ButtonUp	1524
5.58.4.3	Move	1525
5.58.4.4	WheelChanged	1525
5.59	OpenTK.Input.MouseEventArgs Class Reference	1525
5.59.1	Detailed Description	1526
5.59.2	Constructor & Destructor Documentation	1526
5.59.2.1	EventArgs	1526
5.59.2.2	EventArgs	1526
5.59.2.3	EventArgs	1526
5.59.3	Property Documentation	1526
5.59.3.1	Position	1526
5.59.3.2	X	1527
5.59.3.3	Y	1527
5.60	OpenTK.Input.MouseMoveEventArgs Class Reference	1527
5.60.1	Detailed Description	1528
5.60.2	Constructor & Destructor Documentation	1528
5.60.2.1	MouseMoveEventArgs	1528
5.60.2.2	MouseMoveEventArgs	1528
5.60.2.3	MouseMoveEventArgs	1528
5.60.3	Property Documentation	1528
5.60.3.1	XDelta	1528
5.60.3.2	YDelta	1528
5.61	OpenTK.Input.MouseState Struct Reference	1529

5.61.1 Detailed Description	1529
5.61.2 Member Function Documentation	1529
5.61.2.1 Equals	1529
5.62 OpenTK.Input.MouseWheelEventArgs Class Reference	1529
5.62.1 Detailed Description	1530
5.62.2 Constructor & Destructor Documentation	1530
5.62.2.1 MouseEventArgs	1530
5.62.2.2 MouseEventArgs	1531
5.62.2.3 MouseEventArgs	1531
5.62.3 Property Documentation	1531
5.62.3.1 Delta	1531
5.62.3.2 DeltaPrecise	1531
5.62.3.3 Value	1531
5.62.3.4 ValuePrecise	1531
5.63 OpenTK.KeyPressEventArgs Class Reference	1532
5.63.1 Detailed Description	1532
5.63.2 Constructor & Destructor Documentation	1532
5.63.2.1 KeyPressEventArgs	1532
5.63.3 Property Documentation	1532
5.63.3.1 KeyChar	1532
5.64 OpenTK.Matrix4 Struct Reference	1533
5.64.1 Detailed Description	1539
5.64.2 Constructor & Destructor Documentation	1539
5.64.2.1 Matrix4	1539
5.64.2.2 Matrix4	1539
5.64.3 Member Function Documentation	1540
5.64.3.1 CreateFromAxisAngle	1540
5.64.3.2 CreateFromAxisAngle	1540
5.64.3.3 CreateOrthographic	1541
5.64.3.4 CreateOrthographic	1541
5.64.3.5 CreateOrthographicOffCenter	1541

5.64.3.6	CreateOrthographicOffCenter	1542
5.64.3.7	CreatePerspectiveFieldOfView	1542
5.64.3.8	CreatePerspectiveFieldOfView	1543
5.64.3.9	CreatePerspectiveOffCenter	1543
5.64.3.10	CreatePerspectiveOffCenter	1544
5.64.3.11	CreateRotationX	1544
5.64.3.12	CreateRotationX	1545
5.64.3.13	CreateRotationY	1545
5.64.3.14	CreateRotationY	1545
5.64.3.15	CreateRotationZ	1545
5.64.3.16	CreateRotationZ	1546
5.64.3.17	CreateTranslation	1546
5.64.3.18	CreateTranslation	1546
5.64.3.19	CreateTranslation	1546
5.64.3.20	CreateTranslation	1547
5.64.3.21	Equals	1547
5.64.3.22	Equals	1547
5.64.3.23	Frustum	1548
5.64.3.24	GetHashCode	1548
5.64.3.25	Invert	1548
5.64.3.26	Invert	1549
5.64.3.27	LookAt	1549
5.64.3.28	LookAt	1549
5.64.3.29	Mult	1550
5.64.3.30	Mult	1550
5.64.3.31	operator!=	1550
5.64.3.32	operator*	1551
5.64.3.33	operator==	1551
5.64.3.34	Perspective	1551
5.64.3.35	Rotate	1552
5.64.3.36	Rotate	1552

5.64.3.37 RotateX	1552
5.64.3.38 RotateY	1552
5.64.3.39 RotateZ	1553
5.64.3.40 Scale	1553
5.64.3.41 Scale	1553
5.64.3.42 Scale	1554
5.64.3.43 ToString	1554
5.64.3.44 Translation	1554
5.64.3.45 Translation	1554
5.64.3.46 Transpose	1555
5.64.3.47 Transpose	1555
5.64.3.48 Transpose	1555
5.64.4 Member Data Documentation	1555
5.64.4.1 Identity	1555
5.64.4.2 Row0	1555
5.64.4.3 Row1	1555
5.64.4.4 Row2	1556
5.64.4.5 Row3	1556
5.64.5 Property Documentation	1556
5.64.5.1 Column0	1556
5.64.5.2 Column1	1556
5.64.5.3 Column2	1556
5.64.5.4 Column3	1556
5.64.5.5 Determinant	1556
5.64.5.6 M11	1556
5.64.5.7 M12	1556
5.64.5.8 M13	1557
5.64.5.9 M14	1557
5.64.5.10 M21	1557
5.64.5.11 M22	1557
5.64.5.12 M23	1557

5.64.5.13 M24	1557
5.64.5.14 M31	1557
5.64.5.15 M32	1557
5.64.5.16 M33	1557
5.64.5.17 M34	1557
5.64.5.18 M41	1558
5.64.5.19 M42	1558
5.64.5.20 M43	1558
5.64.5.21 M44	1558
5.65 OpenTK.Matrix4d Struct Reference	1558
5.65.1 Detailed Description	1564
5.65.2 Constructor & Destructor Documentation	1564
5.65.2.1 Matrix4d	1564
5.65.2.2 Matrix4d	1565
5.65.3 Member Function Documentation	1565
5.65.3.1 CreateFromAxisAngle	1565
5.65.3.2 CreateFromAxisAngle	1566
5.65.3.3 CreateOrthographic	1566
5.65.3.4 CreateOrthographic	1566
5.65.3.5 CreateOrthographicOffCenter	1567
5.65.3.6 CreateOrthographicOffCenter	1567
5.65.3.7 CreatePerspectiveFieldOfView	1568
5.65.3.8 CreatePerspectiveFieldOfView	1568
5.65.3.9 CreatePerspectiveOffCenter	1569
5.65.3.10 CreatePerspectiveOffCenter	1569
5.65.3.11 CreateRotationX	1570
5.65.3.12 CreateRotationX	1570
5.65.3.13 CreateRotationY	1571
5.65.3.14 CreateRotationY	1571
5.65.3.15 CreateRotationZ	1571
5.65.3.16 CreateRotationZ	1571

5.65.3.17 CreateTranslation	1572
5.65.3.18 CreateTranslation	1572
5.65.3.19 CreateTranslation	1572
5.65.3.20 CreateTranslation	1572
5.65.3.21 Equals	1573
5.65.3.22 Equals	1573
5.65.3.23 Frustum	1573
5.65.3.24 GetHashCode	1574
5.65.3.25 Invert	1574
5.65.3.26 Invert	1574
5.65.3.27 LookAt	1574
5.65.3.28 LookAt	1575
5.65.3.29 Mult	1575
5.65.3.30 Mult	1576
5.65.3.31 operator!=	1576
5.65.3.32 operator*	1576
5.65.3.33 operator==	1577
5.65.3.34 Perspective	1577
5.65.3.35 Rotate	1577
5.65.3.36 Rotate	1578
5.65.3.37 RotateX	1578
5.65.3.38 RotateY	1578
5.65.3.39 RotateZ	1579
5.65.3.40 Scale	1579
5.65.3.41 Scale	1579
5.65.3.42 Scale	1580
5.65.3.43 ToString	1580
5.65.3.44 Translation	1580
5.65.3.45 Translation	1580
5.65.3.46 Transpose	1581
5.65.3.47 Transpose	1581

5.65.3.48 Transpose	1581
5.65.4 Member Data Documentation	1581
5.65.4.1 Identity	1581
5.65.4.2 Row0	1582
5.65.4.3 Row1	1582
5.65.4.4 Row2	1582
5.65.4.5 Row3	1582
5.65.5 Property Documentation	1582
5.65.5.1 Column0	1582
5.65.5.2 Column1	1582
5.65.5.3 Column2	1582
5.65.5.4 Column3	1582
5.65.5.5 Determinant	1582
5.65.5.6 M11	1583
5.65.5.7 M12	1583
5.65.5.8 M13	1583
5.65.5.9 M14	1583
5.65.5.10 M21	1583
5.65.5.11 M22	1583
5.65.5.12 M23	1583
5.65.5.13 M24	1583
5.65.5.14 M31	1583
5.65.5.15 M32	1583
5.65.5.16 M33	1584
5.65.5.17 M34	1584
5.65.5.18 M41	1584
5.65.5.19 M42	1584
5.65.5.20 M43	1584
5.65.5.21 M44	1584
5.66 OpenTK.NativeWindow Class Reference	1584
5.66.1 Detailed Description	1589

5.66.2	Constructor & Destructor Documentation	1589
5.66.2.1	NativeWindow	1589
5.66.2.2	NativeWindow	1590
5.66.2.3	NativeWindow	1590
5.66.3	Member Function Documentation	1591
5.66.3.1	Close	1591
5.66.3.2	Dispose	1591
5.66.3.3	EnsureUndisposed	1591
5.66.3.4	OnClosed	1591
5.66.3.5	OnClosing	1591
5.66.3.6	OnDisposed	1592
5.66.3.7	OnFocusedChanged	1592
5.66.3.8	OnIconChanged	1592
5.66.3.9	OnKeyPress	1592
5.66.3.10	OnMouseEnter	1593
5.66.3.11	OnMouseLeave	1593
5.66.3.12	OnMove	1593
5.66.3.13	OnResize	1593
5.66.3.14	OnTitleChanged	1593
5.66.3.15	OnVisibleChanged	1594
5.66.3.16	OnWindowBorderChanged	1594
5.66.3.17	OnWindowStateChanged	1594
5.66.3.18	PointToClient	1594
5.66.3.19	PointToScreen	1595
5.66.3.20	ProcessEvents	1595
5.66.3.21	ProcessEvents	1595
5.66.4	Property Documentation	1595
5.66.4.1	Bounds	1595
5.66.4.2	ClientRectangle	1595
5.66.4.3	ClientSize	1596
5.66.4.4	Exists	1596

5.66.4.5	Focused	1596
5.66.4.6	Height	1596
5.66.4.7	Icon	1596
5.66.4.8	InputDriver	1596
5.66.4.9	IsDisposed	1596
5.66.4.10	Location	1597
5.66.4.11	Size	1597
5.66.4.12	Title	1597
5.66.4.13	Visible	1597
5.66.4.14	Width	1597
5.66.4.15	WindowBorder	1597
5.66.4.16	WindowInfo	1597
5.66.4.17	WindowState	1598
5.66.4.18	X	1598
5.66.4.19	Y	1598
5.66.5	Event Documentation	1598
5.66.5.1	Closed	1598
5.66.5.2	Closing	1598
5.66.5.3	Disposed	1598
5.66.5.4	FocusedChanged	1598
5.66.5.5	IconChanged	1599
5.66.5.6	KeyPress	1599
5.66.5.7	MouseEnter	1599
5.66.5.8	MouseLeave	1599
5.66.5.9	Move	1599
5.66.5.10	Resize	1599
5.66.5.11	TitleChanged	1599
5.66.5.12	VisibleChanged	1600
5.66.5.13	WindowBorderChanged	1600
5.66.5.14	WindowStateChanged	1600
5.67	OpenTK.Platform.IGameWindow Interface Reference	1600

5.67.1 Detailed Description	1601
5.67.2 Member Function Documentation	1601
5.67.2.1 MakeCurrent	1601
5.67.2.2 Run	1601
5.67.2.3 Run	1602
5.67.2.4 SwapBuffers	1602
5.67.3 Event Documentation	1602
5.67.3.1 Load	1602
5.67.3.2 RenderFrame	1602
5.67.3.3 Unload	1602
5.67.3.4 UpdateFrame	1602
5.68 OpenTK.Platform.IWindowInfo Interface Reference	1603
5.68.1 Detailed Description	1603
5.69 OpenTK.PlatformException Class Reference	1603
5.69.1 Detailed Description	1603
5.69.2 Constructor & Destructor Documentation	1603
5.69.2.1 PlatformException	1603
5.70 OpenTK.Properties.Resources Class Reference	1603
5.70.1 Detailed Description	1604
5.71 OpenTK.Quaternion Struct Reference	1604
5.71.1 Detailed Description	1607
5.71.2 Constructor & Destructor Documentation	1607
5.71.2.1 Quaternion	1607
5.71.2.2 Quaternion	1608
5.71.3 Member Function Documentation	1608
5.71.3.1 Add	1608
5.71.3.2 Add	1608
5.71.3.3 Conjugate	1609
5.71.3.4 Conjugate	1609
5.71.3.5 Conjugate	1609
5.71.3.6 Equals	1609

5.71.3.7	Equals	1610
5.71.3.8	FromAxisAngle	1610
5.71.3.9	GetHashCode	1610
5.71.3.10	Invert	1610
5.71.3.11	Invert	1611
5.71.3.12	Mult	1611
5.71.3.13	Mult	1611
5.71.3.14	Multiply	1611
5.71.3.15	Multiply	1612
5.71.3.16	Multiply	1612
5.71.3.17	Multiply	1612
5.71.3.18	Normalize	1613
5.71.3.19	Normalize	1613
5.71.3.20	Normalize	1613
5.71.3.21	operator!=	1613
5.71.3.22	operator*	1614
5.71.3.23	operator*	1614
5.71.3.24	operator*	1614
5.71.3.25	operator+	1615
5.71.3.26	operator-	1615
5.71.3.27	operator==	1615
5.71.3.28	Slerp	1616
5.71.3.29	Sub	1616
5.71.3.30	Sub	1616
5.71.3.31	ToAxisAngle	1617
5.71.3.32	ToAxisAngle	1617
5.71.3.33	ToString	1617
5.71.4	Member Data Documentation	1617
5.71.4.1	Identity	1617
5.71.5	Property Documentation	1617
5.71.5.1	Length	1617

5.71.5.2	LengthSquared	1618
5.71.5.3	W	1618
5.71.5.4	X	1618
5.71.5.5	Xyz	1618
5.71.5.6	XYZ	1618
5.71.5.7	Y	1618
5.71.5.8	Z	1618
5.72	OpenTK.Quaterniond Struct Reference	1618
5.72.1	Detailed Description	1622
5.72.2	Constructor & Destructor Documentation	1622
5.72.2.1	Quaterniond	1622
5.72.2.2	Quaterniond	1622
5.72.3	Member Function Documentation	1623
5.72.3.1	Add	1623
5.72.3.2	Add	1623
5.72.3.3	Conjugate	1623
5.72.3.4	Conjugate	1624
5.72.3.5	Conjugate	1624
5.72.3.6	Equals	1624
5.72.3.7	Equals	1624
5.72.3.8	FromAxisAngle	1625
5.72.3.9	GetHashCode	1625
5.72.3.10	Invert	1625
5.72.3.11	Invert	1625
5.72.3.12	Mult	1626
5.72.3.13	Mult	1626
5.72.3.14	Multiply	1626
5.72.3.15	Multiply	1627
5.72.3.16	Multiply	1627
5.72.3.17	Multiply	1627
5.72.3.18	Normalize	1627

5.72.3.19 Normalize	1628
5.72.3.20 Normalize	1628
5.72.3.21 operator!=	1628
5.72.3.22 operator*	1628
5.72.3.23 operator*	1629
5.72.3.24 operator*	1629
5.72.3.25 operator+	1629
5.72.3.26 operator-	1630
5.72.3.27 operator==	1630
5.72.3.28 Slerp	1630
5.72.3.29 Sub	1631
5.72.3.30 Sub	1631
5.72.3.31 ToAxisAngle	1631
5.72.3.32 ToAxisAngle	1631
5.72.3.33 ToString	1632
5.72.4 Member Data Documentation	1632
5.72.4.1 Identity	1632
5.72.5 Property Documentation	1632
5.72.5.1 Length	1632
5.72.5.2 LengthSquared	1632
5.72.5.3 W	1632
5.72.5.4 X	1632
5.72.5.5 Xyz	1632
5.72.5.6 XYZ	1633
5.72.5.7 Y	1633
5.72.5.8 Z	1633
5.73 OpenTK.Toolkit Class Reference	1633
5.73.1 Detailed Description	1633
5.73.2 Member Function Documentation	1633
5.73.2.1 Init	1633
5.74 OpenTK.Vector2 Struct Reference	1634

5.74.1 Detailed Description	1640
5.74.2 Constructor & Destructor Documentation	1640
5.74.2.1 Vector2	1640
5.74.2.2 Vector2	1640
5.74.2.3 Vector2	1640
5.74.2.4 Vector2	1640
5.74.3 Member Function Documentation	1641
5.74.3.1 Add	1641
5.74.3.2 Add	1641
5.74.3.3 Add	1641
5.74.3.4 Add	1641
5.74.3.5 BaryCentric	1642
5.74.3.6 BaryCentric	1642
5.74.3.7 Clamp	1642
5.74.3.8 Clamp	1643
5.74.3.9 ComponentMax	1643
5.74.3.10 ComponentMax	1643
5.74.3.11 ComponentMin	1644
5.74.3.12 ComponentMin	1644
5.74.3.13 Div	1644
5.74.3.14 Div	1644
5.74.3.15 Div	1645
5.74.3.16 Divide	1645
5.74.3.17 Divide	1645
5.74.3.18 Divide	1646
5.74.3.19 Divide	1646
5.74.3.20 Dot	1646
5.74.3.21 Dot	1646
5.74.3.22 Equals	1647
5.74.3.23 Equals	1647
5.74.3.24 GetHashCode	1647

5.74.3.25 Lerp	1648
5.74.3.26 Lerp	1648
5.74.3.27 Max	1648
5.74.3.28 Min	1649
5.74.3.29 Mult	1649
5.74.3.30 Mult	1649
5.74.3.31 Mult	1649
5.74.3.32 Multiply	1650
5.74.3.33 Multiply	1650
5.74.3.34 Multiply	1650
5.74.3.35 Multiply	1651
5.74.3.36 Normalize	1651
5.74.3.37 Normalize	1651
5.74.3.38 Normalize	1651
5.74.3.39 NormalizeFast	1651
5.74.3.40 NormalizeFast	1652
5.74.3.41 NormalizeFast	1652
5.74.3.42 operator!=	1652
5.74.3.43 operator*	1652
5.74.3.44 operator*	1653
5.74.3.45 operator+	1653
5.74.3.46 operator-	1653
5.74.3.47 operator-	1654
5.74.3.48 operator/	1654
5.74.3.49 operator==	1654
5.74.3.50 Scale	1655
5.74.3.51 Scale	1655
5.74.3.52 Scale	1655
5.74.3.53 Sub	1655
5.74.3.54 Sub	1655
5.74.3.55 Sub	1656

5.74.3.56 Sub	1656
5.74.3.57 Subtract	1656
5.74.3.58 Subtract	1656
5.74.3.59 ToString	1657
5.74.3.60 Transform	1657
5.74.3.61 Transform	1657
5.74.4 Member Data Documentation	1657
5.74.4.1 One	1657
5.74.4.2 SizeInBytes	1657
5.74.4.3 UnitX	1658
5.74.4.4 UnitY	1658
5.74.4.5 X	1658
5.74.4.6 Y	1658
5.74.4.7 Zero	1658
5.74.5 Property Documentation	1658
5.74.5.1 Length	1658
5.74.5.2 LengthFast	1658
5.74.5.3 LengthSquared	1659
5.74.5.4 PerpendicularLeft	1659
5.74.5.5 PerpendicularRight	1659
5.75 OpenTK.Vector2d Struct Reference	1659
5.75.1 Detailed Description	1665
5.75.2 Constructor & Destructor Documentation	1665
5.75.2.1 Vector2d	1665
5.75.3 Member Function Documentation	1665
5.75.3.1 Add	1665
5.75.3.2 Add	1666
5.75.3.3 Add	1666
5.75.3.4 Add	1666
5.75.3.5 BaryCentric	1666
5.75.3.6 BaryCentric	1667

5.75.3.7 Clamp	1667
5.75.3.8 Clamp	1668
5.75.3.9 Div	1668
5.75.3.10 Div	1668
5.75.3.11 Div	1668
5.75.3.12 Divide	1669
5.75.3.13 Divide	1669
5.75.3.14 Divide	1669
5.75.3.15 Divide	1670
5.75.3.16 Dot	1670
5.75.3.17 Dot	1670
5.75.3.18 Equals	1670
5.75.3.19 Equals	1671
5.75.3.20 GetHashCode	1671
5.75.3.21 Lerp	1671
5.75.3.22 Lerp	1672
5.75.3.23 Max	1672
5.75.3.24 Max	1672
5.75.3.25 Min	1672
5.75.3.26 Min	1673
5.75.3.27 Mult	1673
5.75.3.28 Mult	1673
5.75.3.29 Mult	1674
5.75.3.30 Multiply	1674
5.75.3.31 Multiply	1674
5.75.3.32 Multiply	1675
5.75.3.33 Multiply	1675
5.75.3.34 Normalize	1675
5.75.3.35 Normalize	1675
5.75.3.36 Normalize	1676
5.75.3.37 NormalizeFast	1676

5.75.3.38 NormalizeFast	1676
5.75.3.39 operator Vector2	1676
5.75.3.40 operator Vector2d	1677
5.75.3.41 operator!=	1677
5.75.3.42 operator*	1677
5.75.3.43 operator*	1678
5.75.3.44 operator+	1678
5.75.3.45 operator-	1678
5.75.3.46 operator-	1679
5.75.3.47 operator/	1679
5.75.3.48 operator==	1679
5.75.3.49 Scale	1680
5.75.3.50 Scale	1680
5.75.3.51 Scale	1680
5.75.3.52 Sub	1680
5.75.3.53 Sub	1680
5.75.3.54 Sub	1681
5.75.3.55 Sub	1681
5.75.3.56 Subtract	1681
5.75.3.57 Subtract	1681
5.75.3.58 ToString	1682
5.75.3.59 Transform	1682
5.75.3.60 Transform	1682
5.75.4 Member Data Documentation	1683
5.75.4.1 One	1683
5.75.4.2 SizeInBytes	1683
5.75.4.3 UnitX	1683
5.75.4.4 UnitY	1683
5.75.4.5 X	1683
5.75.4.6 Y	1683
5.75.4.7 Zero	1683

5.75.5	Property Documentation	1683
5.75.5.1	Length	1683
5.75.5.2	LengthSquared	1684
5.75.5.3	PerpendicularLeft	1684
5.75.5.4	PerpendicularRight	1684
5.76	OpenTK.Vector2h Struct Reference	1684
5.76.1	Detailed Description	1687
5.76.2	Constructor & Destructor Documentation	1687
5.76.2.1	Vector2h	1687
5.76.2.2	Vector2h	1687
5.76.2.3	Vector2h	1687
5.76.2.4	Vector2h	1688
5.76.2.5	Vector2h	1688
5.76.2.6	Vector2h	1688
5.76.2.7	Vector2h	1688
5.76.2.8	Vector2h	1689
5.76.2.9	Vector2h	1689
5.76.2.10	Vector2h	1689
5.76.2.11	Vector2h	1689
5.76.2.12	Vector2h	1690
5.76.3	Member Function Documentation	1690
5.76.3.1	Equals	1690
5.76.3.2	FromBinaryStream	1690
5.76.3.3	FromBytes	1690
5.76.3.4	GetBytes	1691
5.76.3.5	GetObjectData	1691
5.76.3.6	operator Vector2	1691
5.76.3.7	operator Vector2d	1691
5.76.3.8	operator Vector2h	1692
5.76.3.9	operator Vector2h	1692
5.76.3.10	ToBinaryStream	1692

5.76.3.11 ToString	1692
5.76.3.12 ToVector2	1692
5.76.3.13 ToVector2d	1693
5.76.4 Member Data Documentation	1693
5.76.4.1 SizeInBytes	1693
5.76.4.2 X	1693
5.76.4.3 Y	1693
5.77 OpenTK.Vector3 Struct Reference	1693
5.77.1 Detailed Description	1700
5.77.2 Constructor & Destructor Documentation	1701
5.77.2.1 Vector3	1701
5.77.2.2 Vector3	1701
5.77.2.3 Vector3	1701
5.77.2.4 Vector3	1701
5.77.3 Member Function Documentation	1701
5.77.3.1 Add	1701
5.77.3.2 Add	1702
5.77.3.3 Add	1702
5.77.3.4 Add	1702
5.77.3.5 BaryCentric	1702
5.77.3.6 BaryCentric	1703
5.77.3.7 CalculateAngle	1703
5.77.3.8 CalculateAngle	1704
5.77.3.9 Clamp	1704
5.77.3.10 Clamp	1704
5.77.3.11 ComponentMax	1705
5.77.3.12 ComponentMax	1705
5.77.3.13 ComponentMin	1705
5.77.3.14 ComponentMin	1706
5.77.3.15 Cross	1706
5.77.3.16 Cross	1706

5.77.3.17 Div	1707
5.77.3.18 Div	1707
5.77.3.19 Div	1707
5.77.3.20 Divide	1707
5.77.3.21 Divide	1708
5.77.3.22 Divide	1708
5.77.3.23 Divide	1708
5.77.3.24 Dot	1708
5.77.3.25 Dot	1709
5.77.3.26 Equals	1709
5.77.3.27 Equals	1709
5.77.3.28 GetHashCode	1710
5.77.3.29 Lerp	1710
5.77.3.30 Lerp	1710
5.77.3.31 Max	1710
5.77.3.32 Min	1711
5.77.3.33 Mult	1711
5.77.3.34 Mult	1711
5.77.3.35 Mult	1712
5.77.3.36 Multiply	1712
5.77.3.37 Multiply	1712
5.77.3.38 Multiply	1712
5.77.3.39 Multiply	1713
5.77.3.40 Normalize	1713
5.77.3.41 Normalize	1713
5.77.3.42 Normalize	1714
5.77.3.43 NormalizeFast	1714
5.77.3.44 NormalizeFast	1714
5.77.3.45 NormalizeFast	1714
5.77.3.46 operator!=	1714
5.77.3.47 operator*	1715

5.77.3.48 operator*	1715
5.77.3.49 operator+	1715
5.77.3.50 operator-	1716
5.77.3.51 operator-	1716
5.77.3.52 operator/	1716
5.77.3.53 operator==	1717
5.77.3.54 Scale	1717
5.77.3.55 Scale	1717
5.77.3.56 Scale	1717
5.77.3.57 Sub	1718
5.77.3.58 Sub	1718
5.77.3.59 Sub	1718
5.77.3.60 Sub	1718
5.77.3.61 Subtract	1719
5.77.3.62 Subtract	1719
5.77.3.63 ToString	1719
5.77.3.64 Transform	1719
5.77.3.65 Transform	1720
5.77.3.66 Transform	1720
5.77.3.67 Transform	1720
5.77.3.68 TransformNormal	1720
5.77.3.69 TransformNormal	1721
5.77.3.70 TransformNormalInverse	1721
5.77.3.71 TransformNormalInverse	1721
5.77.3.72 TransformPerspective	1722
5.77.3.73 TransformPerspective	1722
5.77.3.74 TransformPosition	1722
5.77.3.75 TransformPosition	1723
5.77.3.76 TransformVector	1723
5.77.3.77 TransformVector	1723
5.77.4 Member Data Documentation	1723

5.77.4.1	One	1723
5.77.4.2	SizeInBytes	1724
5.77.4.3	UnitX	1724
5.77.4.4	UnitY	1724
5.77.4.5	UnitZ	1724
5.77.4.6	X	1724
5.77.4.7	Y	1724
5.77.4.8	Z	1724
5.77.4.9	Zero	1724
5.77.5	Property Documentation	1725
5.77.5.1	Length	1725
5.77.5.2	LengthFast	1725
5.77.5.3	LengthSquared	1725
5.77.5.4	Xy	1725
5.78	OpenTK.Vector3d Struct Reference	1725
5.78.1	Detailed Description	1733
5.78.2	Constructor & Destructor Documentation	1733
5.78.2.1	Vector3d	1733
5.78.2.2	Vector3d	1734
5.78.2.3	Vector3d	1734
5.78.2.4	Vector3d	1734
5.78.3	Member Function Documentation	1734
5.78.3.1	Add	1734
5.78.3.2	Add	1734
5.78.3.3	Add	1735
5.78.3.4	Add	1735
5.78.3.5	BaryCentric	1735
5.78.3.6	BaryCentric	1736
5.78.3.7	CalculateAngle	1736
5.78.3.8	CalculateAngle	1736
5.78.3.9	Clamp	1737

5.78.3.10 Clamp	1737
5.78.3.11 ComponentMax	1737
5.78.3.12 ComponentMax	1738
5.78.3.13 ComponentMin	1738
5.78.3.14 ComponentMin	1738
5.78.3.15 Cross	1738
5.78.3.16 Cross	1739
5.78.3.17 Div	1739
5.78.3.18 Div	1739
5.78.3.19 Div	1740
5.78.3.20 Divide	1740
5.78.3.21 Divide	1740
5.78.3.22 Divide	1741
5.78.3.23 Divide	1741
5.78.3.24 Dot	1741
5.78.3.25 Dot	1741
5.78.3.26 Equals	1742
5.78.3.27 Equals	1742
5.78.3.28 GetHashCode	1742
5.78.3.29 Lerp	1743
5.78.3.30 Lerp	1743
5.78.3.31 Max	1743
5.78.3.32 Min	1744
5.78.3.33 Mult	1744
5.78.3.34 Mult	1744
5.78.3.35 Mult	1744
5.78.3.36 Multiply	1745
5.78.3.37 Multiply	1745
5.78.3.38 Multiply	1745
5.78.3.39 Multiply	1745
5.78.3.40 Normalize	1746

5.78.3.41 Normalize	1746
5.78.3.42 Normalize	1746
5.78.3.43 NormalizeFast	1746
5.78.3.44 NormalizeFast	1747
5.78.3.45 NormalizeFast	1747
5.78.3.46 operator Vector3	1747
5.78.3.47 operator Vector3d	1747
5.78.3.48 operator!=	1748
5.78.3.49 operator*	1748
5.78.3.50 operator*	1748
5.78.3.51 operator+	1749
5.78.3.52 operator-	1749
5.78.3.53 operator-	1749
5.78.3.54 operator/	1750
5.78.3.55 operator==	1750
5.78.3.56 Scale	1750
5.78.3.57 Scale	1750
5.78.3.58 Scale	1751
5.78.3.59 Sub	1751
5.78.3.60 Sub	1751
5.78.3.61 Sub	1751
5.78.3.62 Sub	1752
5.78.3.63 Subtract	1752
5.78.3.64 Subtract	1752
5.78.3.65 ToString	1752
5.78.3.66 Transform	1753
5.78.3.67 Transform	1753
5.78.3.68 Transform	1753
5.78.3.69 Transform	1753
5.78.3.70 TransformNormal	1754
5.78.3.71 TransformNormal	1754

5.78.3.72 TransformNormalInverse	1754
5.78.3.73 TransformNormalInverse	1755
5.78.3.74 TransformPerspective	1755
5.78.3.75 TransformPerspective	1756
5.78.3.76 TransformPosition	1756
5.78.3.77 TransformPosition	1756
5.78.3.78 TransformVector	1756
5.78.3.79 TransformVector	1757
5.78.4 Member Data Documentation	1757
5.78.4.1 One	1757
5.78.4.2 SizeInBytes	1757
5.78.4.3 UnitX	1757
5.78.4.4 UnitY	1757
5.78.4.5 UnitZ	1758
5.78.4.6 X	1758
5.78.4.7 Y	1758
5.78.4.8 Z	1758
5.78.4.9 Zero	1758
5.78.5 Property Documentation	1758
5.78.5.1 Length	1758
5.78.5.2 LengthFast	1758
5.78.5.3 LengthSquared	1759
5.78.5.4 Xy	1759
5.79 OpenTK.Vector3h Struct Reference	1759
5.79.1 Detailed Description	1762
5.79.2 Constructor & Destructor Documentation	1762
5.79.2.1 Vector3h	1762
5.79.2.2 Vector3h	1762
5.79.2.3 Vector3h	1763
5.79.2.4 Vector3h	1763
5.79.2.5 Vector3h	1763

5.79.2.6	Vector3h	1763
5.79.2.7	Vector3h	1764
5.79.2.8	Vector3h	1764
5.79.2.9	Vector3h	1764
5.79.2.10	Vector3h	1764
5.79.2.11	Vector3h	1765
5.79.2.12	Vector3h	1765
5.79.3	Member Function Documentation	1765
5.79.3.1	Equals	1765
5.79.3.2	FromBinaryStream	1765
5.79.3.3	FromBytes	1766
5.79.3.4	GetBytes	1766
5.79.3.5	GetObjectData	1766
5.79.3.6	operator Vector3	1766
5.79.3.7	operator Vector3d	1767
5.79.3.8	operator Vector3h	1767
5.79.3.9	operator Vector3h	1767
5.79.3.10	ToBinaryStream	1768
5.79.3.11	ToString	1768
5.79.3.12	ToVector3	1768
5.79.3.13	ToVector3d	1768
5.79.4	Member Data Documentation	1768
5.79.4.1	SizeInBytes	1768
5.79.4.2	X	1768
5.79.4.3	Y	1768
5.79.4.4	Z	1768
5.79.5	Property Documentation	1769
5.79.5.1	Xy	1769
5.80	OpenTK.Vector4 Struct Reference	1769
5.80.1	Detailed Description	1775
5.80.2	Constructor & Destructor Documentation	1775

5.80.2.1	Vector4	1775
5.80.2.2	Vector4	1776
5.80.2.3	Vector4	1776
5.80.2.4	Vector4	1776
5.80.2.5	Vector4	1776
5.80.3	Member Function Documentation	1777
5.80.3.1	Add	1777
5.80.3.2	Add	1777
5.80.3.3	Add	1777
5.80.3.4	Add	1777
5.80.3.5	BaryCentric	1778
5.80.3.6	BaryCentric	1778
5.80.3.7	Clamp	1778
5.80.3.8	Clamp	1779
5.80.3.9	Div	1779
5.80.3.10	Div	1779
5.80.3.11	Div	1780
5.80.3.12	Divide	1780
5.80.3.13	Divide	1780
5.80.3.14	Divide	1780
5.80.3.15	Divide	1781
5.80.3.16	Dot	1781
5.80.3.17	Dot	1781
5.80.3.18	Equals	1781
5.80.3.19	Equals	1782
5.80.3.20	GetHashCode	1782
5.80.3.21	Lerp	1782
5.80.3.22	Lerp	1783
5.80.3.23	Max	1783
5.80.3.24	Max	1783
5.80.3.25	Min	1783

5.80.3.26 Min	1784
5.80.3.27 Mult	1784
5.80.3.28 Mult	1784
5.80.3.29 Mult	1785
5.80.3.30 Multiply	1785
5.80.3.31 Multiply	1785
5.80.3.32 Multiply	1786
5.80.3.33 Multiply	1786
5.80.3.34 Normalize	1786
5.80.3.35 Normalize	1786
5.80.3.36 Normalize	1787
5.80.3.37 NormalizeFast	1787
5.80.3.38 NormalizeFast	1787
5.80.3.39 NormalizeFast	1787
5.80.3.40 operator float *	1787
5.80.3.41 operator IntPtr	1788
5.80.3.42 operator!=	1788
5.80.3.43 operator*	1788
5.80.3.44 operator*	1789
5.80.3.45 operator+	1789
5.80.3.46 operator-	1789
5.80.3.47 operator-	1790
5.80.3.48 operator/	1790
5.80.3.49 operator==	1790
5.80.3.50 Scale	1791
5.80.3.51 Scale	1791
5.80.3.52 Scale	1791
5.80.3.53 Sub	1791
5.80.3.54 Sub	1791
5.80.3.55 Sub	1792
5.80.3.56 Sub	1792

5.80.3.57 Subtract	1792
5.80.3.58 Subtract	1793
5.80.3.59 ToString	1793
5.80.3.60 Transform	1793
5.80.3.61 Transform	1793
5.80.3.62 Transform	1794
5.80.3.63 Transform	1794
5.80.4 Member Data Documentation	1794
5.80.4.1 One	1794
5.80.4.2 SizeInBytes	1794
5.80.4.3 UnitW	1794
5.80.4.4 UnitX	1795
5.80.4.5 UnitY	1795
5.80.4.6 UnitZ	1795
5.80.4.7 W	1795
5.80.4.8 X	1795
5.80.4.9 Y	1795
5.80.4.10 Z	1795
5.80.4.11 Zero	1795
5.80.5 Property Documentation	1795
5.80.5.1 Length	1795
5.80.5.2 LengthFast	1796
5.80.5.3 LengthSquared	1796
5.80.5.4 Xy	1796
5.80.5.5 Xyz	1796
5.81 OpenTK.Vector4d Struct Reference	1796
5.81.1 Detailed Description	1803
5.81.2 Constructor & Destructor Documentation	1803
5.81.2.1 Vector4d	1803
5.81.2.2 Vector4d	1804
5.81.2.3 Vector4d	1804

5.81.2.4	Vector4d	1804
5.81.2.5	Vector4d	1804
5.81.3	Member Function Documentation	1804
5.81.3.1	Add	1804
5.81.3.2	Add	1805
5.81.3.3	Add	1805
5.81.3.4	Add	1805
5.81.3.5	BaryCentric	1805
5.81.3.6	BaryCentric	1806
5.81.3.7	Clamp	1806
5.81.3.8	Clamp	1807
5.81.3.9	Div	1807
5.81.3.10	Div	1807
5.81.3.11	Div	1807
5.81.3.12	Divide	1808
5.81.3.13	Divide	1808
5.81.3.14	Divide	1808
5.81.3.15	Divide	1809
5.81.3.16	Dot	1809
5.81.3.17	Dot	1809
5.81.3.18	Equals	1809
5.81.3.19	Equals	1810
5.81.3.20	GetHashCode	1810
5.81.3.21	Lerp	1810
5.81.3.22	Lerp	1811
5.81.3.23	Max	1811
5.81.3.24	Max	1811
5.81.3.25	Min	1811
5.81.3.26	Min	1812
5.81.3.27	Mult	1812
5.81.3.28	Mult	1812

5.81.3.29 Mult	1813
5.81.3.30 Multiply	1813
5.81.3.31 Multiply	1813
5.81.3.32 Multiply	1813
5.81.3.33 Multiply	1814
5.81.3.34 Normalize	1814
5.81.3.35 Normalize	1814
5.81.3.36 Normalize	1815
5.81.3.37 NormalizeFast	1815
5.81.3.38 NormalizeFast	1815
5.81.3.39 NormalizeFast	1815
5.81.3.40 operator double *	1815
5.81.3.41 operator IntPtr	1816
5.81.3.42 operator Vector4	1816
5.81.3.43 operator Vector4d	1816
5.81.3.44 operator!=	1816
5.81.3.45 operator*	1817
5.81.3.46 operator*	1817
5.81.3.47 operator+	1817
5.81.3.48 operator-	1818
5.81.3.49 operator-	1818
5.81.3.50 operator/	1818
5.81.3.51 operator==	1819
5.81.3.52 Scale	1819
5.81.3.53 Scale	1819
5.81.3.54 Scale	1819
5.81.3.55 Sub	1820
5.81.3.56 Sub	1820
5.81.3.57 Sub	1820
5.81.3.58 Sub	1820
5.81.3.59 Subtract	1821

5.81.3.60 Subtract	1821
5.81.3.61 ToString	1821
5.81.3.62 Transform	1821
5.81.3.63 Transform	1822
5.81.3.64 Transform	1822
5.81.3.65 Transform	1822
5.81.4 Member Data Documentation	1822
5.81.4.1 One	1822
5.81.4.2 SizeInBytes	1823
5.81.4.3 UnitW	1823
5.81.4.4 UnitX	1823
5.81.4.5 UnitY	1823
5.81.4.6 UnitZ	1823
5.81.4.7 W	1823
5.81.4.8 X	1823
5.81.4.9 Y	1823
5.81.4.10 Z	1823
5.81.4.11 Zero	1824
5.81.5 Property Documentation	1824
5.81.5.1 Length	1824
5.81.5.2 LengthFast	1824
5.81.5.3 LengthSquared	1824
5.81.5.4 Xy	1824
5.81.5.5 Xyz	1824
5.82 OpenTK.Vector4h Struct Reference	1825
5.82.1 Detailed Description	1828
5.82.2 Constructor & Destructor Documentation	1828
5.82.2.1 Vector4h	1828
5.82.2.2 Vector4h	1828
5.82.2.3 Vector4h	1828
5.82.2.4 Vector4h	1829

5.82.2.5	Vector4h	1829
5.82.2.6	Vector4h	1829
5.82.2.7	Vector4h	1829
5.82.2.8	Vector4h	1830
5.82.2.9	Vector4h	1830
5.82.2.10	Vector4h	1830
5.82.2.11	Vector4h	1830
5.82.2.12	Vector4h	1831
5.82.3	Member Function Documentation	1831
5.82.3.1	Equals	1831
5.82.3.2	FromBinaryStream	1831
5.82.3.3	FromBytes	1831
5.82.3.4	GetBytes	1832
5.82.3.5	GetObjectData	1832
5.82.3.6	operator Vector4	1832
5.82.3.7	operator Vector4d	1832
5.82.3.8	operator Vector4h	1833
5.82.3.9	operator Vector4h	1833
5.82.3.10	ToBinaryStream	1833
5.82.3.11	ToString	1833
5.82.3.12	ToVector4	1833
5.82.3.13	ToVector4d	1834
5.82.4	Member Data Documentation	1834
5.82.4.1	SizeInBytes	1834
5.82.4.2	W	1834
5.82.4.3	X	1834
5.82.4.4	Y	1834
5.82.4.5	Z	1834
5.82.5	Property Documentation	1834
5.82.5.1	Xy	1834
5.82.5.2	Xyz	1834

Chapter 1

Namespace Index

1.1 Package List

Here are the packages with brief descriptions (if available):

OpenTK.Audio	13
OpenTK.Audio.OpenAL	14
OpenTK.Compute	43
OpenTK.Graphics	43
OpenTK.Graphics.ES10	45
OpenTK.Graphics.ES11	52
OpenTK.Graphics.ES20	68
OpenTK.Graphics.OpenGL	85
OpenTK.Input	244
OpenTK.Platform	254
OpenTK.Platform.Dummy	254
OpenTK.Platform.Egl	254
OpenTK.Platform.MacOS	254
OpenTK.Platform.MacOS.Carbon	255
OpenTK.Platform.Windows	258
OpenTK.Platform.X11	280
OpenTK.Properties	294

Chapter 2

Class Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

OpenTK.Audio.AudioCapture	295
OpenTK.Audio.AudioContext	299
OpenTK.Audio.AudioException	310
OpenTK.Audio.AudioContextException	308
OpenTK.Audio.AudioDeviceException	309
OpenTK.Audio.AudioValueException	311
OpenTK.Audio.OpenAL.EffectsExtension	312
OpenTK.Audio.OpenAL.XRamExtension	340
OpenTK.AutoGeneratedAttribute	344
OpenTK.BezierCurve	345
OpenTK.BezierCurveCubic	350
OpenTK.BezierCurveQuadric	352
OpenTK.BindingsBase	355
OpenTK.Graphics.GraphicsBindingsBase	780
OpenTK.Graphics.ES10.GL	428
OpenTK.Graphics.ES11.GL	505
OpenTK.Graphics.ES20.GL	619
OpenTK.Graphics.OpenGL.GL	805
OpenTK.Box2	358
OpenTK.ContextExistsException	361
OpenTK.ContextHandle	362
OpenTK.DisplayDevice	366
OpenTK.DisplayResolution	371
OpenTK.FrameEventArgs	374
OpenTK.GLControl	388

OpenTK.Graphics.Color4	393
OpenTK.Graphics.ColorFormat	423
OpenTK.Graphics.GraphicsContextException	789
OpenTK.Graphics.GraphicsContextMissingException	790
OpenTK.Graphics.GraphicsContextVersion	791
OpenTK.Graphics.GraphicsMode	793
OpenTK.Graphics.GraphicsModeException	799
OpenTK.Graphics.IGraphicsContext	800
OpenTK.Graphics.GraphicsContext	781
OpenTK.Graphics.IGraphicsContextInternal	803
OpenTK.Graphics.GraphicsContext	781
OpenTK.GraphicsException	1479
OpenTK.Graphics.GraphicsErrorException	792
OpenTK.Half	1480
OpenTK.INativeWindow	1491
OpenTK.NativeWindow	1584
OpenTK.GameWindow	375
OpenTK.Platform.IGameWindow	1600
OpenTK.GameWindow	375
OpenTK.Input.GamePad	1500
OpenTK.Input.GamePadState	1501
OpenTK.Input.IInputDevice	1501
OpenTK.Input.JoystickDevice	1508
OpenTK.Input.KeyboardDevice	1513
OpenTK.Input.MouseDevice	1521
OpenTK.Input.IJoystickDriver	1503
OpenTK.Input.IInputDriver	1502
OpenTK.Input.IKeyboardDriver	1504
OpenTK.Input.IInputDriver	1502
OpenTK.Input.IMouseDriver	1504
OpenTK.Input.IInputDriver	1502
OpenTK.Input.JoystickAxisCollection	1505
OpenTK.Input.JoystickButtonCollection	1506
OpenTK.Input.JoystickButtonEventArgs	1507
OpenTK.Input.JoystickEventArgs	1511
OpenTK.Input.JoystickMoveEventArgs	1511
OpenTK.Input.KeyboardKeyEventArgs	1516
OpenTK.Input.KeyboardState	1517
OpenTK.Input.MouseEventArgs	1525
OpenTK.Input.MouseButtonEventArgs	1519
OpenTK.Input.MouseMoveEventArgs	1527
OpenTK.Input.MouseWheelEventArgs	1529

OpenTK.Input.MouseState	1529
OpenTK.KeyPressEventArgs	1532
OpenTK.Matrix4	1533
OpenTK.Matrix4d	1558
OpenTK.Platform.IWindowInfo	1603
OpenTK.PlatformException	1603
OpenTK.Properties.Resources	1603
OpenTK.Quaternion	1604
OpenTK.Quaterniond	1618
OpenTK.Toolkit	1633
OpenTK.Vector2	1634
OpenTK.Vector2d	1659
OpenTK.Vector2h	1684
OpenTK.Vector3	1693
OpenTK.Vector3d	1725
OpenTK.Vector3h	1759
OpenTK.Vector4	1769
OpenTK.Vector4d	1796
OpenTK.Vector4h	1825

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

OpenTK.Audio.AudioCapture (Provides methods to instantiate, use and destroy an audio device for recording. Static methods are provided to list available devices known by the driver)	295
OpenTK.Audio.AudioContext (Provides methods to instantiate, use and destroy an audio context for playback. Static methods are provided to list available devices known by the driver)	299
OpenTK.Audio.AudioContextException (Represents exceptions related to an OpenTK.Audio.AudioContext)	308
OpenTK.Audio.AudioDeviceException (Represents exceptions related to an OpenTK.Audio device)	309
OpenTK.Audio.AudioException (Represents exceptions related to the OpenTK.Audio subsystem)	310
OpenTK.Audio.AudioValueException (Represents exceptions related to invalid values)	311
OpenTK.Audio.OpenAL.EffectsExtension (Provides access to the OpenAL effects extension)	312
OpenTK.Audio.OpenAL.XRamExtension (The X-Ram Extension is provided on the top-end Sound Blaster X-Fi solutions (Sound Blaster X-Fi Fatal1ty, Sound Blaster X-Fi Elite Pro, or later). These products feature 64MB of X-Ram that can only be used for audio purposes, which can be controlled by this Extension. /summary>) . . .	340
OpenTK.AutoGeneratedAttribute (Indicates that this function is generated automatically by a tool)	344
OpenTK.BezierCurve (Represents a bezier curve with as many points as you want)	345

OpenTK.BezierCurveCubic (Represents a cubic bezier curve with two anchor and two control points)	350
OpenTK.BezierCurveQuadric (Represents a quadric bezier curve with two anchor and one control point)	352
OpenTK.BindingsBase (Provides a common foundation for all flat API bindings and implements the extension loading interface)	355
OpenTK.Box2 (Defines a 2d box (rectangle))	358
OpenTK.ContextExistsException (This exception is thrown when a GraphicsContext property cannot be changed after creation)	361
OpenTK.ContextHandle (Represents a handle to an OpenGL or OpenAL context)	362
OpenTK.DisplayDevice (Defines a display device on the underlying system, and provides methods to query and change its display parameters) .	366
OpenTK.DisplayResolution (Contains information regarding a monitor's display resolution)	371
OpenTK.FrameEventArgs (Defines the arguments for frame events. A FrameEventArgs instance is only valid for the duration of the relevant event; do not store references to FrameEventArgs outside this event)	374
OpenTK.GameWindow (The GameWindow class contains cross-platform methods to create and render on an OpenGL window, handle input and load resources)	375
OpenTK.GLControl (Defines a UserControl with OpenGL rendering capabilities)	388
OpenTK.Graphics.Color4 (Represents a color with 4 floating-point components (R, G, B, A))	393
OpenTK.Graphics.ColorFormat (Defines the ColorFormat component of a GraphicsMode)	423
OpenTK.Graphics.ES10.GL (Provides access to OpenGL ES 1.0 methods) .	428
OpenTK.Graphics.ES11.GL (Provides access to OpenGL ES 1.1 methods) .	505
OpenTK.Graphics.ES20.GL (Provides access to OpenGL ES 2.0 methods) .	619
OpenTK.Graphics.GraphicsBindingsBase (Implements BindingsBase for the OpenTK.Graphics namespace (OpenGL and OpenGL ES)) . .	780
OpenTK.Graphics.GraphicsContext (Represents and provides methods to manipulate an OpenGL render context)	781
OpenTK.Graphics.GraphicsContextException (Represents errors related to a GraphicsContext)	789
OpenTK.Graphics.GraphicsContextMissingException (Thrown when an operation that required GraphicsContext is performed, when no GraphicsContext is current in the calling thread)	790
OpenTK.Graphics.GraphicsContextVersion (Defines the version information of a GraphicsContext)	791

OpenTK.Graphics.GraphicsErrorException (Identifies a specific OpenGL or OpenGL ES error. Such exceptions are only thrown when OpenGL or OpenGL ES automatic error checking is enabled - GraphicsContext.ErrorChecking property. Important: Do <i>*not*</i> catch this exception. Rather, fix the underlying issue that caused the error)	792
OpenTK.Graphics.GraphicsMode (Defines the format for graphics operations)	793
OpenTK.Graphics.GraphicsModeException (Represents errors related to unavailable graphics parameters)	799
OpenTK.Graphics.IGraphicsContext (Provides methods for creating and interacting with an OpenGL context)	800
OpenTK.Graphics.IGraphicsContextInternal (Provides methods to create new GraphicsContexts . Should only be used for extending OpenTK)	803
OpenTK.Graphics.OpenGL.GL (OpenGL bindings for .NET, implementing the full OpenGL API, including extensions)	805
OpenTK.Graphics.Exception (Represents errors related to Graphics operations)	1479
OpenTK.Half (The name Half is derived from half-precision floating-point number. It occupies only 16 bits, which are split into 1 Sign bit, 5 Exponent bits and 10 Mantissa bits)	1480
OpenTK.INativeWindow (Defines the interface for a native window)	1491
OpenTK.Input.GamePad (Provides access to GamePad devices. Note: this API is not implemented yet)	1500
OpenTK.Input.GamePadState (Encapsulates the state of a GamePad device) .	1501
OpenTK.Input.IInputDevice (Defines a common interface for all input devices)	1501
OpenTK.Input.InputDriver (Defines the interface for an input driver)	1502
OpenTK.Input.IJoystickDriver (Defines the interface for JoystickDevice drivers)	1503
OpenTK.Input.IKeyboardDriver (Defines the interface for KeyboardDevice drivers)	1504
OpenTK.Input.IMouseDriver (Defines the interface for MouseDevice drivers)	1504
OpenTK.Input.JoystickAxisCollection (Defines a collection of JoystickAxes)	1505
OpenTK.Input.JoystickButtonCollection (Defines a collection of JoystickButtons)	1506
OpenTK.Input.JoystickButtonEventArgs (Provides data for the JoystickDevice.ButtonDown and JoystickDevice.ButtonUp events. This class is cached for performance reasons - avoid storing references outside the scope of the event)	1507
OpenTK.Input.JoystickDevice (Represents a joystick device and provides methods to query its status)	1508
OpenTK.Input.JoystickEventArgs (The base class for JoystickDevice event arguments)	1511
OpenTK.Input.JoystickMoveEventArgs (Provides data for the JoystickDevice.Move event. This class is cached for performance reasons - avoid storing references outside the scope of the event)	1511

OpenTK.Input.KeyboardDevice (Represents a keyboard device and provides methods to query its status)	1513
OpenTK.Input.KeyboardKeyEventArgs (Defines the event data for KeyboardDevice events)	1516
OpenTK.Input.KeyboardState (Encapsulates the state of a Keyboard device)	1517
OpenTK.Input.MouseButtonEventArgs (Defines the event data for MouseDevice.ButtonDown and MouseDevice.ButtonUp events)	1519
OpenTK.Input.MouseDevice (Represents a mouse device and provides methods to query its status)	1521
OpenTK.Input.MouseEventArgs (Defines the event data for MouseDevice events)	1525
OpenTK.Input.MouseMoveEventArgs (Defines the event data for MouseDevice.Move events)	1527
OpenTK.Input.MouseState (Encapsulates the state of a mouse device)	1529
OpenTK.Input.MouseWheelEventArgs (Defines the event data for MouseDevice.WheelChanged events)	1529
OpenTK.KeyPressEventArgs (Defines the event arguments for KeyPress events. Instances of this class are cached: KeyPressEventArgs should only be used inside the relevant event, unless manually cloned)	1532
OpenTK.Matrix4 (Represents a 4x4 Matrix)	1533
OpenTK.Matrix4d (Represents a 4x4 Matrix with double-precision components)	1558
OpenTK.NativeWindow (Instances of this class implement the OpenTK.INativeWindow interface on the current platform)	1584
OpenTK.Platform.IGameWindow (Defines the interface for a GameWindow)	1600
OpenTK.Platform.IWindowInfo (Describes an OS window)	1603
OpenTK.PlatformException (Defines a platform specific exception)	1603
OpenTK.Properties.Resources (A strongly-typed resource class, for looking up localized strings, etc)	1603
OpenTK.Quaternion (Represents a Quaternion)	1604
OpenTK.Quaterniond (Represents a double-precision Quaternion)	1618
OpenTK.Toolkit (Provides static methods to manage an OpenTK application)	1633
OpenTK.Vector2 (Represents a 2D vector using two single-precision floating-point numbers)	1634
OpenTK.Vector2d (Represents a 2D vector using two double-precision floating-point numbers)	1659
OpenTK.Vector2h (2-component Vector of the Half type. Occupies 4 Byte total)	1684
OpenTK.Vector3 (Represents a 3D vector using three single-precision floating-point numbers)	1693
OpenTK.Vector3d (Represents a 3D vector using three double-precision floating-point numbers)	1725
OpenTK.Vector3h (3-component Vector of the Half type. Occupies 6 Byte total)	1759

OpenTK.Vector4 (Represents a 4D vector using four single-precision floating-point numbers)	1769
OpenTK.Vector4d (Represents a 4D vector using four double-precision floating-point numbers)	1796
OpenTK.Vector4h (4-component Vector of the Half type. Occupies 8 Byte total)	1825

Chapter 4

Namespace Documentation

4.1 Package OpenTK.Audio

Packages

- package [OpenAL](#)

Classes

- class [AudioCapture](#)
Provides methods to instantiate, use and destroy an audio device for recording. Static methods are provided to list available devices known by the driver.
- class [AudioContext](#)
Provides methods to instantiate, use and destroy an audio context for playback. Static methods are provided to list available devices known by the driver.
- class [AudioContextException](#)
Represents exceptions related to an [OpenTK.Audio.AudioContext](#).
- class [AudioDeviceException](#)
Represents exceptions related to an [OpenTK.Audio](#) device.
- class [AudioException](#)
Represents exceptions related to the [OpenTK.Audio](#) subsystem.
- class [AudioValueException](#)
Represents exceptions related to invalid values.

4.2 Package OpenTK.Audio.OpenAL

Classes

- class [EffectsExtension](#)
Provides access to the [OpenAL](#) effects extension.
- class [XRamExtension](#)
The X-Ram Extension is provided on the top-end Sound Blaster X-Fi solutions (Sound Blaster X-Fi Fatal1ty, Sound Blaster X-Fi Elite Pro, or later). These products feature 64MB of X-Ram that can only be used for audio purposes, which can be controlled by this Extension. /summary>

Enumerations

- enum [ALCapability](#) { [Invalid](#) = -1 }
A list of valid Enable/Disable/IsEnabled parameters.
- enum [ALLListenerf](#) { [Gain](#) = 0x100A, [EfxMetersPerUnit](#) = 0x20004 }
A list of valid 32-bit Float Listener/GetListener parameters.
- enum [ALLListener3f](#) { [Position](#) = 0x1004, [Velocity](#) = 0x1006 }
A list of valid Math.Vector3 Listener/GetListener parameters.
- enum [ALLListenerfv](#) { [Orientation](#) = 0x100F }
A list of valid float[] Listener/GetListener parameters.
- enum [ALSourcef](#) {
[ReferenceDistance](#) = 0x1020, [MaxDistance](#) = 0x1023, [RolloffFactor](#) = 0x1021,
[Pitch](#) = 0x1003,
[Gain](#) = 0x100A, [MinGain](#) = 0x100D, [MaxGain](#) = 0x100E, [ConeInnerAngle](#) = 0x1001,
[ConeOuterAngle](#) = 0x1002, [ConeOuterGain](#) = 0x1022, [SecOffset](#) = 0x1024,
[EfxAirAbsorptionFactor](#) = 0x20007,
[EfxRoomRolloffFactor](#) = 0x20008, [EfxConeOuterGainHighFrequency](#) = 0x20009 }
A list of valid 32-bit Float Source/GetSource parameters.
- enum [ALSource3f](#) { [Position](#) = 0x1004, [Velocity](#) = 0x1006, [Direction](#) = 0x1005 }
A list of valid Math.Vector3 Source/GetSource parameters.

- enum **ALSourceb** {
SourceRelative = 0x202, **Looping** = 0x1007, **EfxDirectFilterGainHighFrequencyAuto** = 0x2000A, **EfxAuxiliarySendFilterGainAuto** = 0x2000B,
EfxAuxiliarySendFilterGainHighFrequencyAuto = 0x2000C }
A list of valid 8-bit boolean Source/GetSource parameters.
- enum **ALSourcei** {
ByteOffset = 0x1026, **SampleOffset** = 0x1025, **Buffer** = 0x1009, **SourceType** = 0x1027,
EfxDirectFilter = 0x20005 }
A list of valid Int32 Source parameters.
- enum **ALSource3i** { **EfxAuxiliarySendFilter** = 0x20006 }
A list of valid 3x Int32 Source/GetSource parameters.
- enum **ALGetSourcei** {
ByteOffset = 0x1026, **SampleOffset** = 0x1025, **Buffer** = 0x1009, **SourceState** = 0x1010,
BuffersQueued = 0x1015, **BuffersProcessed** = 0x1016, **SourceType** = 0x1027
}

A list of valid Int32 GetSource parameters.
- enum **ALSourceState** { **Initial** = 0x1011, **Playing** = 0x1012, **Paused** = 0x1013,
Stopped = 0x1014 }
Source state information, can be retrieved by AL.Source() with ALSourceci.SourceState.
- enum **ALSourceType** { **Static** = 0x1028, **Streaming** = 0x1029, **Undetermined** = 0x1030 }
Source type information, can be retrieved by AL.Source() with ALSourceci.SourceType.
- enum **ALFormat** {
Mono8 = 0x1100, **Mono16** = 0x1101, **Stereo8** = 0x1102, **Stereo16** = 0x1103,
MonoALawExt = 0x10016, **StereoALawExt** = 0x10017, **MonoMuLawExt** = 0x10014, **StereoMuLawExt** = 0x10015,
VorbisExt = 0x10003, **Mp3Ext** = 0x10020, **MonoIma4Ext** = 0x1300, **StereoIma4Ext** = 0x1301,
MonoFloat32Ext = 0x10010, **StereoFloat32Ext** = 0x10011, **MonoDoubleExt** = 0x10012, **StereoDoubleExt** = 0x10013,

[Multi51Chn16Ext](#) = 0x120B, [Multi51Chn32Ext](#) = 0x120C, [Multi51Chn8Ext](#) = 0x120A, [Multi61Chn16Ext](#) = 0x120E,
[Multi61Chn32Ext](#) = 0x120F, [Multi61Chn8Ext](#) = 0x120D, [Multi71Chn16Ext](#) = 0x1211, [Multi71Chn32Ext](#) = 0x1212,
[Multi71Chn8Ext](#) = 0x1210, [MultiQuad16Ext](#) = 0x1205, [MultiQuad32Ext](#) = 0x1206, [MultiQuad8Ext](#) = 0x1204,
[MultiRear16Ext](#) = 0x1208, [MultiRear32Ext](#) = 0x1209, [MultiRear8Ext](#) = 0x1207
 }

Sound samples: Format specifier.

- enum [ALGetBufferi](#) { [Frequency](#) = 0x2001, [Bits](#) = 0x2002, [Channels](#) = 0x2003, [Size](#) = 0x2004 }

A list of valid Int32 GetBuffer parameters.

- enum [ALBufferState](#) { [Unused](#) = 0x2010, [Pending](#) = 0x2011, [Processed](#) = 0x2012 }

Buffer state. Not supported for public use (yet).

- enum [ALError](#) {
[NoError](#) = 0, [InvalidName](#) = 0xA001, [IllegalEnum](#) = 0xA002, [InvalidEnum](#) = 0xA002,
[InvalidValue](#) = 0xA003, [IllegalCommand](#) = 0xA004, [InvalidOperation](#) = 0xA004, [OutOfMemory](#) = 0xA005 }

Returned by AL.GetError.

- enum [ALGetString](#) { [Vendor](#) = 0xB001, [Version](#) = 0xB002, [Renderer](#) = 0xB003, [Extensions](#) = 0xB004 }

A list of valid string AL.Get() parameters.

- enum [ALGetFloat](#) { [DopplerFactor](#) = 0xC000, [DopplerVelocity](#) = 0xC001, [SpeedOfSound](#) = 0xC003 }

A list of valid 32-bit Float AL.Get() parameters.

- enum [ALGetInteger](#) { [DistanceModel](#) = 0xD000 }

A list of valid Int32 AL.Get() parameters.

- enum [ALDistanceModel](#) {
[None](#) = 0, [InverseDistance](#) = 0xD001, [InverseDistanceClamped](#) = 0xD002, [LinearDistance](#) = 0xD003,
[LinearDistanceClamped](#) = 0xD004, [ExponentDistance](#) = 0xD005, [ExponentDistanceClamped](#) = 0xD006 }

Used by `AL.DistanceModel()`, the distance model can be retrieved by `AL.Get()` with `ALGetInteger.DistanceModel`.

- enum `EfxEffectf` {
 - `ReverbDensity` = 0x0001, `ReverbDiffusion` = 0x0002, `ReverbGain` = 0x0003, `ReverbGainHF` = 0x0004,
 - `ReverbDecayTime` = 0x0005, `ReverbDecayHFRatio` = 0x0006, `ReverbReflectionsGain` = 0x0007, `ReverbReflectionsDelay` = 0x0008,
 - `ReverbLateReverbGain` = 0x0009, `ReverbLateReverbDelay` = 0x000A, `ReverbAirAbsorptionGainHF` = 0x000B, `ReverbRoomRolloffFactor` = 0x000C,
 - `ChorusRate` = 0x0003, `ChorusDepth` = 0x0004, `ChorusFeedback` = 0x0005, `ChorusDelay` = 0x0006,
 - `DistortionEdge` = 0x0001, `DistortionGain` = 0x0002, `DistortionLowpassCutoff` = 0x0003, `DistortionEQCenter` = 0x0004,
 - `DistortionEQBandwidth` = 0x0005, `EchoDelay` = 0x0001, `EchoLRDelay` = 0x0002, `EchoDamping` = 0x0003,
 - `EchoFeedback` = 0x0004, `EchoSpread` = 0x0005, `FlangerRate` = 0x0003, `FlangerDepth` = 0x0004,
 - `FlangerFeedback` = 0x0005, `FlangerDelay` = 0x0006, `FrequencyShifterFrequency` = 0x0001, `VocalMorpherRate` = 0x0006,
 - `RingModulatorFrequency` = 0x0001, `RingModulatorHighpassCutoff` = 0x0002, `AutowahAttackTime` = 0x0001, `AutowahReleaseTime` = 0x0002,
 - `AutowahResonance` = 0x0003, `AutowahPeakGain` = 0x0004, `EqualizerLowGain` = 0x0001, `EqualizerLowCutoff` = 0x0002,
 - `EqualizerMid1Gain` = 0x0003, `EqualizerMid1Center` = 0x0004, `EqualizerMid1Width` = 0x0005, `EqualizerMid2Gain` = 0x0006,
 - `EqualizerMid2Center` = 0x0007, `EqualizerMid2Width` = 0x0008, `EqualizerHighGain` = 0x0009, `EqualizerHighCutoff` = 0x000A,
 - `EaxReverbDensity` = 0x0001, `EaxReverbDiffusion` = 0x0002, `EaxReverbGain` = 0x0003, `EaxReverbGainHF` = 0x0004,
 - `EaxReverbGainLF` = 0x0005, `EaxReverbDecayTime` = 0x0006, `EaxReverbDecayHFRatio` = 0x0007, `EaxReverbDecayLFRatio` = 0x0008,
 - `EaxReverbReflectionsGain` = 0x0009, `EaxReverbReflectionsDelay` = 0x000A, `EaxReverbLateReverbGain` = 0x000C, `EaxReverbLateReverbDelay` = 0x000D,
 - `EaxReverbEchoTime` = 0x000F, `EaxReverbEchoDepth` = 0x0010, `EaxReverbModulationTime` = 0x0011, `EaxReverbModulationDepth` = 0x0012,
 - `EaxReverbAirAbsorptionGainHF` = 0x0013, `EaxReverbHFReference` = 0x0014, `EaxReverbLFReference` = 0x0015, **`EaxReverbRoomRolloffFactor`** = 0x0016

A list of valid 32-bit Float Effect/GetEffect parameters.

- enum [EfxEffect3f](#) { [EaxReverbLateReverbPan](#) = 0x000E, [EaxReverbReflectionsPan](#) = 0x000B }

A list of valid [Math.Vector3](#) [Effect/GetEffect](#) parameters.

- enum [EfxEffecti](#) {
[ChorusWaveform](#) = 0x0001, [ChorusPhase](#) = 0x0002, [FlangerWaveform](#) = 0x0001, [FlangerPhase](#) = 0x0002,
[FrequencyShifterLeftDirection](#) = 0x0002, [FrequencyShifterRightDirection](#) = 0x0003, [VocalMorpherPhonemeA](#) = 0x0001, [VocalMorpherPhonemeACoarseTuning](#) = 0x0002,
[VocalMorpherPhonemeB](#) = 0x0003, [VocalMorpherPhonemeBCoarseTuning](#) = 0x0004, [VocalMorpherWaveform](#) = 0x0005, [PitchShifterCoarseTune](#) = 0x0001, [PitchShifterFineTune](#) = 0x0002, [RingModulatorWaveform](#) = 0x0003, [CompressorOnoff](#) = 0x0001, [ReverbDecayHFLimit](#) = 0x000D,
[EaxReverbDecayHFLimit](#) = 0x0017, [EffectType](#) = 0x8001 }

A list of valid [Int32](#) [Effect/GetEffect](#) parameters.

- enum [EfxFormantFilterSettings](#) {
[VocalMorpherPhonemeA](#) = 0, [VocalMorpherPhonemeE](#) = 1, [VocalMorpherPhonemeI](#) = 2, [VocalMorpherPhonemeO](#) = 3,
[VocalMorpherPhonemeU](#) = 4, [VocalMorpherPhonemeAA](#) = 5, [VocalMorpherPhonemeAE](#) = 6, [VocalMorpherPhonemeAH](#) = 7,
[VocalMorpherPhonemeAO](#) = 8, [VocalMorpherPhonemeEH](#) = 9, [VocalMorpherPhonemeER](#) = 10, [VocalMorpherPhonemeIH](#) = 11,
[VocalMorpherPhonemeIY](#) = 12, [VocalMorpherPhonemeUH](#) = 13, [VocalMorpherPhonemeUW](#) = 14, [VocalMorpherPhonemeB](#) = 15,
[VocalMorpherPhonemeD](#) = 16, [VocalMorpherPhonemeF](#) = 17, [VocalMorpherPhonemeG](#) = 18, [VocalMorpherPhonemeJ](#) = 19,
[VocalMorpherPhonemeK](#) = 20, [VocalMorpherPhonemeL](#) = 21, [VocalMorpherPhonemeM](#) = 22, [VocalMorpherPhonemeN](#) = 23,
[VocalMorpherPhonemeP](#) = 24, [VocalMorpherPhonemeR](#) = 25, [VocalMorpherPhonemeS](#) = 26, [VocalMorpherPhonemeT](#) = 27,
[VocalMorpherPhonemeV](#) = 28, [VocalMorpherPhonemeZ](#) = 29 }

Vocal morpher effect parameters. If both parameters are set to the same phoneme, that determines the filtering effect that will be heard. If these two parameters are set to different phonemes, the filtering effect will morph between the two settings at a rate specified by [EfxEffectf.VocalMorpherRate](#).

- enum **EfxEffectType** {
Null = 0x0000, **Reverb** = 0x0001, **Chorus** = 0x0002, **Distortion** = 0x0003,
Echo = 0x0004, **Flanger** = 0x0005, **FrequencyShifter** = 0x0006, **VocalMorpher**
= 0x0007,
PitchShifter = 0x0008, **RingModulator** = 0x0009, **Autowah** = 0x000A, **Compressor**
= 0x000B,
Equalizer = 0x000C, **EaxReverb** = 0x8000 }
Effect type definitions to be used with EfxEffecti.EffectType.
- enum **EfxAuxiliaryi** { **EffectsSlotEffect** = 0x0001, **EffectsSlotAuxiliarySendAuto**
= 0x0003 }
A list of valid Int32 AuxiliaryEffectSlot/GetAuxiliaryEffectSlot parameters.
- enum **EfxAuxiliaryf** { **EffectsSlotGain** = 0x0002 }
A list of valid 32-bits Float AuxiliaryEffectSlot/GetAuxiliaryEffectSlot parameters.
- enum **EfxFilterf** {
LowpassGain = 0x0001, **LowpassGainHF** = 0x0002, **HighpassGain** = 0x0001,
HighpassGainLF = 0x0002,
BandpassGain = 0x0001, **BandpassGainLF** = 0x0002, **BandpassGainHF** =
0x0003 }
A list of valid 32-bits Float Filter/GetFilter parameters.
- enum **EfxFilteri** { **FilterType** = 0x8001 }
A list of valid Int32 Filter/GetFilter parameters.
- enum **EfxFilterType** { **Null** = 0x0000, **Lowpass** = 0x0001, **Highpass** = 0x0002,
Bandpass = 0x0003 }
Filter type definitions to be used with EfxFilteri.FilterType.
- enum **AlcContextAttributes** {
Frequency = 0x1007, **Refresh** = 0x1008, **Sync** = 0x1009, **MonoSources** =
0x1010,
StereoSources = 0x1011, **EfxMaxAuxiliarySends** = 0x20003 }
Defines available context attributes.
- enum **AlcError** {
NoError = 0, **InvalidDevice** = 0xA001, **InvalidContext** = 0xA002, **InvalidEnum**
= 0xA003,
InvalidValue = 0xA004, **OutOfMemory** = 0xA005 }
Defines OpenAL context errors.

- enum [AlcGetString](#) {
[DefaultDeviceSpecifier](#) = 0x1004, [Extensions](#) = 0x1006, **CaptureDefaultDeviceSpecifier** = 0x311, [DefaultAllDevicesSpecifier](#) = 0x1012,
[CaptureDeviceSpecifier](#) = 0x310, **DeviceSpecifier** = 0x1005, [AllDevicesSpecifier](#) = 0x1013 }
Defines available parameters for `OpenTK.Audio.OpenAL.Alc.GetString(IntPtr, AlcGetString)`.
- enum [AlcGetStringList](#) { [CaptureDeviceSpecifier](#) = 0x310, **DeviceSpecifier** = 0x1005, [AllDevicesSpecifier](#) = 0x1013 }
Defines available parameters for `Alc.GetString(IntPtr, AlcGetStringList)`.
- enum [AlcGetInteger](#) {
[MajorVersion](#) = 0x1000, [MinorVersion](#) = 0x1001, [AttributesSize](#) = 0x1002, [AllAttributes](#) = 0x1003,
CaptureSamples = 0x312, [EfxMajorVersion](#) = 0x20001, [EfxMinorVersion](#) = 0x20002, [EfxMaxAuxiliarySends](#) = 0x20003 }
Defines available parameters for `Alc.GetInteger(IntPtr, AlcGetInteger, int, int[])`.

4.2.1 Enumeration Type Documentation

4.2.1.1 enum OpenTK::Audio::OpenAL::ALBufferState

Buffer state. Not supported for public use (yet).

Enumerator:

Unused summary>Buffer state. Not supported for public use (yet).

Pending summary>Buffer state. Not supported for public use (yet).

4.2.1.2 enum OpenTK::Audio::OpenAL::ALCapability

A list of valid Enable/Disable/IsEnabled parameters.

Enumerator:

Invalid summary>Currently no state toggles exist for vanilla [OpenAL](#) and no Extension uses it.

4.2.1.3 enum OpenTK::Audio::OpenAL::AlcContextAttributes

Defines available context attributes.

Enumerator:

Frequency summary>Followed by System.Int32 Hz

Refresh summary>Followed by AlBoolean.True, or AlBoolean.False

Sync summary>Followed by System.Int32 Num of requested Mono (3D) Sources

MonoSources summary>Followed by System.Int32 Num of requested Stereo Sources

EfxMaxAuxiliarySends (EFX Extension) This Context property can be passed to [OpenAL](#) during Context creation (alcCreateContext) to request a maximum number of Auxiliary Sends desired on each Source. It is not guaranteed that the desired number of sends will be available, so an application should query this property after creating the context using alcGetInterv. Default: 2

4.2.1.4 enum OpenTK::Audio::OpenAL::AlcError

Defines [OpenAL](#) context errors.

Enumerator:

NoError summary>There is no current error. summary>No Device. The device handle or specifier names an inaccessible driver/server.

InvalidDevice summary>Invalid context ID. The Context argument does not name a valid context.

InvalidContext summary>Bad enum. A token used is not valid, or not applicable.

InvalidEnum summary>Bad value. A value (e.g. Attribute) is not valid, or not applicable.

InvalidValue summary>Out of memory. Unable to allocate memory.

4.2.1.5 enum OpenTK::Audio::OpenAL::AlcGetInteger

Defines available parameters for Alc.GetInteger(IntPtr, AlcGetInteger, int, int[]).

Enumerator:

MajorVersion summary>The specification revision for this implementation (major version). NULL is an acceptable device. summary>The specification revision for this implementation (minor version). NULL is an acceptable device.

MinorVersion summary>The size (number of ALCint values) required for a zero-terminated attributes list, for the current context. NULL is an invalid device.

AttributesSize summary>Expects a destination of ALC_ATTRIBUTES_SIZE, and provides an attribute list for the current context of the specified device. NULL is an invalid device.

AllAttributes summary>The number of capture samples available. NULL is an invalid device.

EfxMajorVersion (EFX Extension) This property can be used by the application to retrieve the Major version number of the Effects Extension supported by this [OpenAL](#) implementation. As this is a Context property is should be retrieved using alcGetInterv.

EfxMinorVersion (EFX Extension) This property can be used by the application to retrieve the Minor version number of the Effects Extension supported by this [OpenAL](#) implementation. As this is a Context property is should be retrieved using alcGetInterv.

EfxMaxAuxiliarySends (EFX Extension) This Context property can be passed to [OpenAL](#) during Context creation (alcCreateContext) to request a maximum number of Auxiliary Sends desired on each Source. It is not guaranteed that the desired number of sends will be available, so an application should query this property after creating the context using alcGetInterv. Default: 2

4.2.1.6 enum OpenTK::Audio::OpenAL::AlcGetString

Defines available parameters for OpenTK.Audio.OpenAL.Alc.GetString(IntPtr, AlcGetString).

Enumerator:

DefaultDeviceSpecifier summary>The specifier string for the default device. summary>A list of available context extensions separated by spaces.

Extensions summary>The name of the default capture device

DefaultAllDevicesSpecifier a list of the default devices.

CaptureDeviceSpecifier summary>Will only return the first Device, not a list. Use AlcGetStringList.CaptureDeviceSpecifier. ALC_EXT_CAPTURE_EXT summary>Will only return the first Device, not a list. Use AlcGetStringList.DeviceSpecifier

AllDevicesSpecifier Will only return the first Device, not a list. Use AlcGetStringList.AllDevicesSpecifier.

4.2.1.7 enum OpenTK::Audio::OpenAL::AlcGetStringList

Defines available parameters for Alc.GetString(IntPtr, AlcGetStringList).

Enumerator:

CaptureDeviceSpecifier summary>The name of the specified capture device, or a list of all available capture devices if no capture device is specified. ALC_EXT_CAPTURE_EXT summary>The specifier strings for all available devices. ALC_ENUMERATION_EXT

AllDevicesSpecifier The specifier strings for all available devices. ALC_ENUMERATE_ALL_EXT.

4.2.1.8 enum OpenTK::Audio::OpenAL::ALDistanceModel

Used by AL.DistanceModel(), the distance model can be retrieved by AL.Get() with ALGetInteger.DistanceModel.

Enumerator:

None summary>Bypasses all distance attenuation calculation for all Sources. summary>InverseDistance is equivalent to the IASIG I3DL2 model with the exception that ALSourcef.ReferenceDistance does not imply any clamping.

InverseDistance summary>InverseDistanceClamped is the IASIG I3DL2 model, with ALSourcef.ReferenceDistance indicating both the reference distance and the distance below which gain will be clamped.

InverseDistanceClamped summary>AL_EXT_LINEAR_DISTANCE extension.

LinearDistance summary>AL_EXT_LINEAR_DISTANCE extension.

LinearDistanceClamped summary>AL_EXT_EXPONENT_DISTANCE extension.

ExponentDistance summary>AL_EXT_EXPONENT_DISTANCE extension.

4.2.1.9 enum OpenTK::Audio::OpenAL::ALError

Returned by AL.GetError.

Enumerator:

NoError summary>No [OpenAL](#) Error. summary>Invalid Name paramater passed to [OpenAL](#) call.

InvalidName summary>Invalid parameter passed to [OpenAL](#) call.

IllegalEnum summary>Invalid parameter passed to [OpenAL](#) call.

InvalidEnum summary>Invalid [OpenAL](#) enum parameter value.

InvalidValue summary>Illegal [OpenAL](#) call.

IllegalCommand summary>Illegal [OpenAL](#) call.

InvalidOperation summary>No [OpenAL](#) memory left.

4.2.1.10 enum OpenTK::Audio::OpenAL::ALFormat

Sound samples: Format specifier.

Enumerator:

Mono8 summary>1 Channel, 8 bits per sample. summary>1 Channel, 16 bits per sample.

Mono16 summary>2 Channels, 8 bits per sample each.

Stereo8 summary>2 Channels, 16 bits per sample each.

MonoALawExt 1 Channel, A-law encoded data. Requires Extension: AL_EXT_ALAW

StereoALawExt 2 Channels, A-law encoded data. Requires Extension: AL_EXT_ALAW

MonoMuLawExt 1 Channel, μ -law encoded data. Requires Extension: AL_EXT_MULAW

StereoMuLawExt 2 Channels, μ -law encoded data. Requires Extension: AL_EXT_MULAW

VorbisExt Ogg Vorbis encoded data. Requires Extension: AL_EXT_vorbis.

Mp3Ext MP3 encoded data. Requires Extension: AL_EXT_mp3.

MonoIma4Ext 1 Channel, IMA4 ADPCM encoded data. Requires Extension: AL_EXT_IMA4

StereoIma4Ext 2 Channels, IMA4 ADPCM encoded data. Requires Extension: AL_EXT_IMA4

MonoFloat32Ext 1 Channel, single-precision floating-point data. Requires Extension: AL_EXT_float32

StereoFloat32Ext 2 Channels, single-precision floating-point data. Requires Extension: AL_EXT_float32

- MonoDoubleExt*** 1 Channel, double-precision floating-point data. Requires Extension: AL_EXT_double
- StereoDoubleExt*** 2 Channels, double-precision floating-point data. Requires Extension: AL_EXT_double
- Multi51Chn16Ext*** Multichannel 5.1, 16-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi51Chn32Ext*** Multichannel 5.1, 32-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi51Chn8Ext*** Multichannel 5.1, 8-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi61Chn16Ext*** Multichannel 6.1, 16-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi61Chn32Ext*** Multichannel 6.1, 32-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi61Chn8Ext*** Multichannel 6.1, 8-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi71Chn16Ext*** Multichannel 7.1, 16-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi71Chn32Ext*** Multichannel 7.1, 32-bit data. Requires Extension: AL_EXT_MCFORMATS.
- Multi71Chn8Ext*** Multichannel 7.1, 8-bit data. Requires Extension: AL_EXT_MCFORMATS.
- MultiQuad16Ext*** Multichannel 4.0, 16-bit data. Requires Extension: AL_EXT_MCFORMATS.
- MultiQuad32Ext*** Multichannel 4.0, 32-bit data. Requires Extension: AL_EXT_MCFORMATS.
- MultiQuad8Ext*** Multichannel 4.0, 8-bit data. Requires Extension: AL_EXT_MCFORMATS.
- MultiRear16Ext*** 1 Channel rear speaker, 16-bit data. See Quadrophonic setups. Requires Extension: AL_EXT_MCFORMATS
- MultiRear32Ext*** 1 Channel rear speaker, 32-bit data. See Quadrophonic setups. Requires Extension: AL_EXT_MCFORMATS
- MultiRear8Ext*** 1 Channel rear speaker, 8-bit data. See Quadrophonic setups. Requires Extension: AL_EXT_MCFORMATS

4.2.1.11 enum OpenTK::Audio::OpenAL::ALGetBufferi

A list of valid Int32 GetBuffer parameters.

Enumerator:

Frequency summary>Sound sample's frequency, in units of hertz [Hz]. This is the number of samples per second. [Half](#) of the sample frequency marks the maximum significant frequency component.

Bits Bit depth of the buffer. Should be 8 or 16.

Channels Number of channels in buffer. > 1 is valid, but buffer won't be positioned when played. 1 for Mono, 2 for Stereo.

Size size of the Buffer in bytes.

4.2.1.12 enum OpenTK::Audio::OpenAL::ALGetFloat

A list of valid 32-bit Float AL.Get() parameters.

Enumerator:

DopplerFactor summary>Doppler scale. Default 1.0f summary>Tweaks speed of propagation. This functionality is deprecated.

DopplerVelocity summary>Speed of Sound in units per second. Default: 343.3f

4.2.1.13 enum OpenTK::Audio::OpenAL::ALGetInteger

A list of valid Int32 AL.Get() parameters.

Enumerator:

DistanceModel summary>See enum ALDistanceModel. [ALDistanceModel](#)

4.2.1.14 enum OpenTK::Audio::OpenAL::ALGetSourceci

A list of valid Int32 GetSource parameters.

Enumerator:

ByteOffset summary>The playback position, expressed in bytes. AL_EXT_OFFSET Extension. summary>The playback position, expressed in samples. AL_EXT_OFFSET Extension.

SampleOffset summary>Indicate the Buffer to provide sound samples. Type: uint Range: any valid Buffer Handle.

SourceState The state of the source (Stopped, Playing, etc.) Use the enum ALSourceState for comparison.

BuffersQueued The number of buffers queued on this source.

BuffersProcessed The number of buffers in the queue that have been processed.
summary>Source type (Static, Streaming or undetermined). Use enum AL-SourceType for comparison.

4.2.1.15 enum OpenTK::Audio::OpenAL::ALGetString

A list of valid string AL.Get() parameters.

Enumerator:

Vendor Gets the Vendor name.

Version Gets the driver version.

Renderer Gets the renderer mode.

Extensions Gets a list of all available Extensions, separated with spaces.

4.2.1.16 enum OpenTK::Audio::OpenAL::ALLListener3f

A list of valid Math.Vector3 Listener/GetListener parameters.

Enumerator:

Position summary>Specify the current location in three dimensional space.
[OpenAL](#), like OpenGL, uses a right handed coordinate system, where in a frontal default view X (thumb) points right, Y points up (index finger), and Z points towards the viewer/camera (middle finger). To switch from a left handed coordinate system, flip the sign on the Z coordinate. Listener position is always in the world coordinate system. summary>Specify the current velocity in three dimensional space.

4.2.1.17 enum OpenTK::Audio::OpenAL::ALLListenerf

A list of valid 32-bit Float Listener/GetListener parameters.

Enumerator:

Gain summary>Indicate the gain (Volume amplification) applied. Type: float
Range: [0.0f - ?] A value of 1.0 means un-attenuated/unchanged. Each division by 2 equals an attenuation of -6dB. Each multiplication with 2 equals an amplification of +6dB. A value of 0.0f is interpreted as zero volume and the channel is effectively disabled. summary>(EFX Extension) This setting

is critical if Air Absorption effects are enabled because the amount of Air Absorption applied is directly related to the real-world distance between the Source and the Listener. centimeters 0.01f meters 1.0f kilometers 1000.0f Range [float.MinValue .. float.MaxValue] Default: 1.0f

4.2.1.18 enum OpenTK::Audio::OpenAL::ALListenerfv

A list of valid float[] Listener/GetListener parameters.

Enumerator:

Orientation summary>Indicate Listener orientation. Expects two [Vector3](#), At followed by Up.

4.2.1.19 enum OpenTK::Audio::OpenAL::ALSource3f

A list of valid Math.Vector3 Source/GetSource parameters.

Enumerator:

Position summary>Specify the current location in three dimensional space. [OpenAL](#), like OpenGL, uses a right handed coordinate system, where in a frontal default view X (thumb) points right, Y points up (index finger), and Z points towards the viewer/camera (middle finger). To switch from a left handed coordinate system, flip the sign on the Z coordinate. Listener position is always in the world coordinate system. summary>Specify the current velocity in three dimensional space.

Velocity summary>Specify the current direction vector.

4.2.1.20 enum OpenTK::Audio::OpenAL::ALSource3i

A list of valid 3x Int32 Source/GetSource parameters.

Enumerator:

EfxAuxiliarySendFilter summary>(EFX Extension) This Source property is used to establish connections between Sources and Auxiliary Effect Slots. For a Source to feed an Effect that has been loaded into an Auxiliary Effect Slot the application must configure one of the Source's auxiliary sends. This process involves setting 3 variables – the destination Auxiliary Effect Slot ID, the Auxiliary Send number, and an optional Filter ID. Type: uint Range: any valid Filter Handle.

4.2.1.21 enum OpenTK::Audio::OpenAL::ALSourceb

A list of valid 8-bit boolean Source/GetSource parameters.

Enumerator:

SourceRelative summary>Indicate that the Source has relative coordinates. Type: bool Range: [True, False] summary>Indicate whether the Source is looping. Type: bool Range: [True, False] Default: False.

Looping summary>(EFX Extension) If this Source property is set to True, this Source's direct-path is automatically filtered according to the orientation of the source relative to the listener and the setting of the Source property Sourcef.ConeOuterGainHF. Type: bool Range [False, True] Default: True

EfxDirectFilterGainHighFrequencyAuto summary>(EFX Extension) If this Source property is set to True, the intensity of this Source's reflected sound is automatically attenuated according to source-listener distance and source directivity (as determined by the cone parameters). If it is False, the reflected sound is not attenuated according to distance and directivity. Type: bool Range [False, True] Default: True

EfxAuxiliarySendFilterGainAuto summary>(EFX Extension) If this Source property is AL_TRUE (its default value), the intensity of this Source's reflected sound at high frequencies will be automatically attenuated according to the high-frequency source directivity as set by the Sourcef.ConeOuterGainHF property. If this property is AL_FALSE, the Source's reflected sound is not filtered at all according to the Source's directivity. Type: bool Range [False, True] Default: True

4.2.1.22 enum OpenTK::Audio::OpenAL::ALSourcef

A list of valid 32-bit Float Source/GetSource parameters.

Enumerator:

ReferenceDistance summary>Source specific reference distance. Type: float Range: [0.0f - float.PositiveInfinity] At 0.0f, no distance attenuation occurs. Type: float Default: 1.0f. summary>Indicate distance above which Sources are not attenuated using the inverse clamped distance model. Default: float.PositiveInfinity Type: float Range: [0.0f - float.PositiveInfinity]

MaxDistance summary>Source specific rolloff factor. Type: float Range: [0.0f - float.PositiveInfinity]

RolloffFactor summary>Specify the pitch to be applied, either at Source, or on mixer results, at Listener. Range: [0.5f - 2.0f] Default: 1.0f

Pitch summary>Indicate the gain (volume amplification) applied. Type: float. Range: [0.0f - ?] A value of 1.0 means un-attenuated/unchanged. Each division by 2 equals an attenuation of -6dB. Each multiplication with 2 equals an amplification of +6dB. A value of 0.0f is meaningless with respect to a logarithmic scale; it is interpreted as zero volume - the channel is effectively disabled.

Gain summary>Indicate minimum Source attenuation. Type: float Range: [0.0f - 1.0f] (Logarithmic)

MinGain summary>Indicate maximum Source attenuation. Type: float Range: [0.0f - 1.0f] (Logarithmic)

MaxGain summary>Directional Source, inner cone angle, in degrees. Range: [0-360] Default: 360

ConeInnerAngle summary>Directional Source, outer cone angle, in degrees. Range: [0-360] Default: 360

ConeOuterAngle summary>Directional Source, outer cone gain. Default: 0.0f Range: [0.0f - 1.0] (Logarithmic)

SecOffset The playback position, expressed in seconds.

EfxAirAbsorptionFactor summary>(EFX Extension) This property is a multiplier on the amount of Air Absorption applied to the Source. The AL_AIR_ABSORPTION_FACTOR is multiplied by an internal Air Absorption Gain HF value of 0.994 (-0.05dB) per meter which represents normal atmospheric humidity and temperature. Range [0.0f .. 10.0f] Default: 0.0f
summary>(EFX Extension) This property is defined the same way as the Reverb Room Rolloff property: it is one of two methods available in the Effect Extension to attenuate the reflected sound (early reflections and reverberation) according to source-listener distance. Range [0.0f .. 10.0f] Default: 0.0f

EfxRoomRolloffFactor summary>(EFX Extension) A directed Source points in a specified direction. The Source sounds at full volume when the listener is directly in front of the source; it is attenuated as the listener circles the Source away from the front. Range [0.0f .. 1.0f] Default: 1.0f

4.2.1.23 enum OpenTK::Audio::OpenAL::ALSourcei

A list of valid Int32 Source parameters.

Enumerator:

ByteOffset summary>The playback position, expressed in bytes.

SampleOffset summary>The playback position, expressed in samples.

Buffer summary>Indicate the Buffer to provide sound samples. Type: uint Range: any valid Buffer Handle. summary>Source type (Static, Streaming or undetermined). Use enum ALSourceType for comparison

SourceType summary>(EFX Extension) This Source property is used to apply filtering on the direct-path (dry signal) of a Source.

4.2.1.24 enum OpenTK::Audio::OpenAL::ALSourceState

Source state information, can be retrieved by AL.Source() with ALSourceci.SourceState.

Enumerator:

Initial summary>Default State when loaded, can be manually set with AL.SourceRewind(). summary>The source is currently playing.

Playing summary>The source has paused playback.

Paused summary>The source is not playing.

4.2.1.25 enum OpenTK::Audio::OpenAL::ALSourceType

Source type information, can be retrieved by AL.Source() with ALSourceci.SourceType.

Enumerator:

Static summary>Source is Static if a Buffer has been attached using AL.Source with the parameter Sourceci.Buffer. summary>Source is Streaming if one or more Buffers have been attached using AL.SourceQueueBuffers

Streaming summary>Source is undetermined when it has a null Buffer attached

4.2.1.26 enum OpenTK::Audio::OpenAL::EfxAuxiliaryf

A list of valid 32-bits Float AuxiliaryEffectSlot/GetAuxiliaryEffectSlot parameters.

Enumerator:

EffectslotGain This property is used to specify an output level for the Auxiliary Effect Slot. Setting the gain to 0.0f mutes the output. Range [0.0f .. 1.0f] Default: 1.0f.

4.2.1.27 enum OpenTK::Audio::OpenAL::EfxAuxiliaryi

A list of valid Int32 AuxiliaryEffectSlot/GetAuxiliaryEffectSlot parameters.

Enumerator:

EffectslotEffect This property is used to attach an Effect object to the Auxiliary Effect Slot object. After the attachment, the Auxiliary Effect Slot object will contain the effect type and have the same effect parameters that were stored in the Effect object. Any Sources feeding the Auxiliary Effect Slot will immediately feed the new effect type and new effect parameters.

EffectslotAuxiliarySendAuto This property is used to enable or disable automatic send adjustments based on the physical positions of the sources and the listener. This property should be enabled when an application wishes to use a reverb effect to simulate the environment surrounding a listener or a collection of Sources. Range [False, True] Default: True.

4.2.1.28 enum OpenTK::Audio::OpenAL::EfxEffect3f

A list of valid Math.Vector3 Effect/GetEffect parameters.

Enumerator:

EaxReverbLateReverbPan Reverb Pan does for the Reverb what Reflections Pan does for the Reflections. Unit: [Vector3](#) of length 0f to 1f Default: {0.0f, 0.0f, 0.0f}.

EaxReverbReflectionsPan This [Vector3](#) controls the spatial distribution of the cluster of early reflections. The direction of this vector controls the global direction of the reflections, while its magnitude controls how focused the reflections are towards this direction. For legacy reasons this [Vector3](#) follows a left-handed co-ordinate system! Note that [OpenAL](#) uses a right-handed coordinate system. Unit: [Vector3](#) of length 0f to 1f Default: {0.0f, 0.0f, 0.0f}.

4.2.1.29 enum OpenTK::Audio::OpenAL::EfxEffectf

A list of valid 32-bit Float Effect/GetEffect parameters.

Enumerator:

ReverbDensity summary>Reverb Modal Density controls the coloration of the late reverb. Lowering the value adds more coloration to the late reverb. Range [0.0f .. 1.0f] Default: 1.0f summary>The Reverb Diffusion property controls the echo density in the reverberation decay. The default 1.0f provides the highest density. Reducing diffusion gives the reverberation a more "grainy" character that is especially noticeable with percussive sound sources. If you set a diffusion value of 0.0f, the later reverberation sounds like a succession of distinct echoes. Range [0.0f .. 1.0f] Default: 1.0f

ReverbDiffusion summary>The Reverb Gain property is the master volume control for the reflected sound - both early reflections and reverberation - that the reverb effect adds to all sound sources. Ranges from 1.0 (0db) (the maximum amount) to 0.0 (-100db) (no reflected sound at all) are accepted. Units: Linear gain Range [0.0f .. 1.0f] Default: 0.32f

ReverbGain summary>The Reverb Gain HF property further tweaks reflected sound by attenuating it at high frequencies. It controls a low-pass filter that applies globally to the reflected sound of all sound sources feeding the particular instance of the reverb effect. Ranges from 1.0f (0db) (no filter) to 0.0f (-100db) (virtually no reflected sound) are accepted. Units: Linear gain Range [0.0f .. 1.0f] Default: 0.89f

ReverbGainHF summary>The Decay Time property sets the reverberation decay time. It ranges from 0.1f (typically a small room with very dead surfaces) to 20.0 (typically a large room with very live surfaces). Unit: Seconds Range [0.1f .. 20.0f] Default: 1.49f

ReverbDecayTime summary>The Decay HF Ratio property sets the spectral quality of the Decay Time parameter. It is the ratio of high-frequency decay time relative to the time set by Decay Time.. Unit: linear multiplier Range [0.1f .. 2.0f] Default: 0.83f

ReverbDecayHFRatio summary>The Reflections Gain property controls the overall amount of initial reflections relative to the Gain property. The value of Reflections Gain ranges from a maximum of 3.16f (+10 dB) to a minimum of 0.0f (-100 dB) (no initial reflections at all), and is corrected by the value of the Gain property. Unit: Linear gain Range [0.0f .. 3.16f] Default: 0.05f

ReverbReflectionsGain summary>The Reflections Delay property is the amount of delay between the arrival time of the direct path from the source to the first reflection from the source. It ranges from 0 to 300 milliseconds. Unit: Seconds Range [0.0f .. 0.3f] Default: 0.007f

ReverbReflectionsDelay summary>The Late Reverb Gain property controls the overall amount of later reverberation relative to the Gain property. The value of Late Reverb Gain ranges from a maximum of 10.0f (+20 dB) to a minimum of 0.0f (-100 dB) (no late reverberation at all). Unit: Linear gain Range [0.0f .. 10.0f] Default: 1.26f

ReverbLateReverbGain summary>The Late Reverb Delay property defines the begin time of the late reverberation relative to the time of the initial reflection (the first of the early reflections). It ranges from 0 to 100 milliseconds. Unit: Seconds Range [0.0f .. 0.1f] Default: 0.011f

ReverbLateReverbDelay summary>The Air Absorption Gain HF property controls the distance-dependent attenuation at high frequencies caused by the propagation medium and applies to reflected sound only. Unit: Linear gain per meter Range [0.892f .. 1.0f] Default: 0.994f

ReverbAirAbsorptionGainHF summary>The Room Rolloff Factor property is one of two methods available to attenuate the reflected sound (containing

both reflections and reverberation) according to source-listener distance. It's defined the same way as OpenAL's Rolloff Factor, but operates on reverb sound instead of direct-path sound. Unit: Linear multiplier Range [0.0f .. 10.0f] Default: 0.0f

ReverbRoomRolloffFactor summary>This property sets the modulation rate of the low-frequency oscillator that controls the delay time of the delayed signals. Unit: Hz Range [0.0f .. 10.0f] Default: 1.1f

ChorusRate summary>This property controls the amount by which the delay time is modulated by the low-frequency oscillator. Range [0.0f .. 1.0f] Default: 0.1f

ChorusDepth summary>This property controls the amount of processed signal that is fed back to the input of the chorus effect. Negative values will reverse the phase of the feedback signal. At full magnitude the identical sample will repeat endlessly. Range [-1.0f .. +1.0f] Default: +0.25f

ChorusFeedback summary>This property controls the average amount of time the sample is delayed before it is played back, and with feedback, the amount of time between iterations of the sample. Larger values lower the pitch. Unit: Seconds Range [0.0f .. 0.016f] Default: 0.016f

ChorusDelay summary>This property controls the shape of the distortion. The higher the value for Edge, the "dirtier" and "fuzzier" the effect. Range [0.0f .. 1.0f] Default: 0.2f

DistortionEdge summary>This property allows you to attenuate the distorted sound. Range [0.01f .. 1.0f] Default: 0.05f

DistortionGain summary>[Input](#) signals can have a low pass filter applied, to limit the amount of high frequency signal feeding into the distortion effect. Unit: Hz Range [80.0f .. 24000.0f] Default: 8000.0f

DistortionLowpassCutoff summary>This property controls the frequency at which the post-distortion attenuation (Distortion Gain) is active. Unit: Hz Range [80.0f .. 24000.0f] Default: 3600.0f

DistortionEQCenter summary>This property controls the bandwidth of the post-distortion attenuation. Unit: Hz Range [80.0f .. 24000.0f] Default: 3600.0f

DistortionEQBandwidth summary>This property controls the delay between the original sound and the first "tap", or echo instance. Subsequently, the value for Echo Delay is used to determine the time delay between each "second tap" and the next "first tap". Unit: Seconds Range [0.0f .. 0.207f] Default: 0.1f

EchoDelay summary>This property controls the delay between the "first tap" and the "second tap". Subsequently, the value for Echo LR Delay is used to determine the time delay between each "first tap" and the next "second tap". Unit: Seconds Range [0.0f .. 0.404f] Default: 0.1f

EchoLRDelay summary>This property controls the amount of high frequency damping applied to each echo. As the sound is subsequently fed back for further echoes, damping results in an echo which progressively gets softer in tone as well as intensity. Range [0.0f .. 0.99f] Default: 0.5f

EchoDamping summary>This property controls the amount of feedback the output signal fed back into the input. Use this parameter to create "cascading" echoes. At full magnitude, the identical sample will repeat endlessly. Below full magnitude, the sample will repeat and fade. Range [0.0f .. 1.0f] Default: 0.5f

EchoFeedback summary>This property controls how hard panned the individual echoes are. With a value of 1.0f, the first "tap" will be panned hard left, and the second "tap" hard right. -1.0f gives the opposite result and values near to 0.0f result in less emphasized panning. Range [-1.0f .. +1.0f] Default: -1.0f

EchoSpread summary>The number of times per second the low-frequency oscillator controlling the amount of delay repeats. Range [0.0f .. 10.0f] Default: 0.27f

FlangerRate summary>The ratio by which the delay time is modulated by the low-frequency oscillator. Range [0.0f .. 1.0f] Default: 1.0f

FlangerDepth summary>This is the amount of the output signal level fed back into the effect's input. A negative value will reverse the phase of the feedback signal. Range [-1.0f .. +1.0f] Default: -0.5f

FlangerFeedback summary>The average amount of time the sample is delayed before it is played back. When used with the Feedback property it's the amount of time between iterations of the sample. Unit: Seconds Range [0.0f .. 0.004f] Default: 0.002f

FlangerDelay summary>This is the carrier frequency. For carrier frequencies below the audible range, the single sideband modulator may produce phaser effects, spatial effects or a slight pitch-shift. As the carrier frequency increases, the timbre of the sound is affected. Unit: Hz Range [0.0f .. 24000.0f] Default: 0.0f

FrequencyShifterFrequency summary>This controls the frequency of the low-frequency oscillator used to morph between the two phoneme filters. Unit: Hz Range [0.0f .. 10.0f] Default: 1.41f

VocalMorpherRate summary>This is the frequency of the carrier signal. If the carrier signal is slowly varying (less than 20 Hz), the result is a slow amplitude variation effect (tremolo). Unit: Hz Range [0.0f .. 8000.0f] Default: 440.0f

RingModulatorFrequency summary>This controls the cutoff frequency at which the input signal is high-pass filtered before being ring modulated. Unit: Hz Range [0.0f .. 24000.0f] Default: 800.0f

RingModulatorHighpassCutoff summary>This property controls the time the filtering effect takes to sweep from minimum to maximum center frequency

when it is triggered by input signal. Unit: Seconds Range [0.0001f .. 1.0f]
Default: 0.06f

AutowahAttackTime summary>This property controls the time the filtering effect takes to sweep from maximum back to base center frequency, when the input signal ends. Unit: Seconds Range [0.0001f .. 1.0f] Default: 0.06f

AutowahReleaseTime summary>This property controls the resonant peak, sometimes known as emphasis or Q, of the auto-wah band-pass filter. Range [2.0f .. 1000.0f] Default: 1000.0f

AutowahResonance summary>This property controls the input signal level at which the band-pass filter will be fully opened. Range [0.00003f .. 31621.0f] Default: 11.22f

AutowahPeakGain summary>This property controls amount of cut or boost on the low frequency range. Range [0.126f .. 7.943f] Default: 1.0f

EqualizerLowGain summary>This property controls the low frequency below which signal will be cut off. Unit: Hz Range [50.0f .. 800.0f] Default: 200.0f

EqualizerLowCutoff summary>This property allows you to cut/boost signal on the "mid1" range. Range [0.126f .. 7.943f] Default: 1.0f

EqualizerMid1Gain summary>This property sets the center frequency for the "mid1" range. Unit: Hz Range [200.0f .. 3000.0f] Default: 500.0f

EqualizerMid1Center summary>This property controls the width of the "mid1" range. Range [0.01f .. 1.0f] Default: 1.0f

EqualizerMid1Width summary>This property allows you to cut/boost signal on the "mid2" range. Range [0.126f .. 7.943f] Default: 1.0f

EqualizerMid2Gain summary>This property sets the center frequency for the "mid2" range. Unit: Hz Range [1000.0f .. 8000.0f] Default: 3000.0f

EqualizerMid2Center summary>This property controls the width of the "mid2" range. Range [0.01f .. 1.0f] Default: 1.0f

EqualizerMid2Width summary>This property allows to cut/boost the signal at high frequencies. Range [0.126f .. 7.943f] Default: 1.0f

EqualizerHighGain summary>This property controls the high frequency above which signal will be cut off. Unit: Hz Range [4000.0f .. 16000.0f] Default: 6000.0f

EqualizerHighCutoff summary>Reverb Modal Density controls the coloration of the late reverb. Range [0.0f .. 1.0f] Default: 1.0f

EaxReverbDensity summary>The Reverb Diffusion property controls the echo density in the reverberation decay. Range [0.0f .. 1.0f] Default: 1.0f

EaxReverbDiffusion summary>Reverb Gain controls the level of the reverberant sound in an environment. A high level of reverb is characteristic of rooms with highly reflective walls and/or small dimensions. Unit: Linear gain Range [0.0f .. 1.0f] Default: 0.32f

EaxReverbGain summary>Gain HF is used to attenuate the high frequency content of all the reflected sound in an environment. You can use this property to give a room specific spectral characteristic. Unit: Linear gain Range [0.0f .. 1.0f] Default: 0.89f

EaxReverbGainHF summary>Gain LF is the low frequency counterpart to Gain HF. Use this to reduce or boost the low frequency content in an environment. Unit: Linear gain Range [0.0f .. 1.0f] Default: 1.0f

EaxReverbGainLF summary>The Decay Time property sets the reverberation decay time. It ranges from 0.1f (typically a small room with very dead surfaces) to 20.0f (typically a large room with very live surfaces). Unit: Seconds Range [0.1f .. 20.0f] Default: 1.49f

EaxReverbDecayTime summary>Decay HF Ratio scales the decay time of high frequencies relative to the value of the Decay Time property. By changing this value, you are changing the amount of time it takes for the high frequencies to decay compared to the mid frequencies of the reverb. Range [0.1f .. 2.0f] Default: 0.83f

EaxReverbDecayHFRatio summary>Decay LF Ratio scales the decay time of low frequencies in the reverberation in the same manner that Decay HF Ratio handles high frequencies. Unit: Linear multiplier Range [0.1f .. 2.0f] Default: 1.0f

EaxReverbDecayLFRatio summary>Reflections Gain sets the level of the early reflections in an environment. Early reflections are used as a cue for determining the size of the environment we are in. Unit: Linear gain Range [0.0f .. 3.16f] Default: 0.05f

EaxReverbReflectionsGain summary>Reflections Delay controls the amount of time it takes for the first reflected wave front to reach the listener, relative to the arrival of the direct-path sound. Unit: Seconds Range [0.0f .. 0.3f] Default: 0.007f

EaxReverbReflectionsDelay summary>The Late Reverb Gain property controls the overall amount of later reverberation relative to the Gain property. Range [0.0f .. 10.0f] Default: 1.26f

EaxReverbLateReverbGain summary>The Late Reverb Delay property defines the begin time of the late reverberation relative to the time of the initial reflection (the first of the early reflections). It ranges from 0 to 100 milliseconds. Unit: Seconds Range [0.0f .. 0.1f] Default: 0.011f

EaxReverbLateReverbDelay summary>Echo Time controls the rate at which the cyclic echo repeats itself along the reverberation decay. Range [0.075f .. 0.25f] Default: 0.25f

EaxReverbEchoTime summary>Echo Depth introduces a cyclic echo in the reverberation decay, which will be noticeable with transient or percussive sounds. Range [0.0f .. 1.0f] Default: 0.0f

EaxReverbEchoDepth summary>Modulation Time controls the speed of the rate of periodic changes in pitch (vibrato). Range [0.04f .. 4.0f] Default: 0.25f

EaxReverbModulationTime summary>Modulation Depth controls the amount of pitch change. Low values of Diffusion will contribute to reinforcing the perceived effect by reducing the mixing of overlapping reflections in the reverb decay. Range [0.0f .. 1.0f] Default: 0.0f

EaxReverbModulationDepth summary>The Air Absorption Gain HF property controls the distance-dependent attenuation at high frequencies caused by the propagation medium. It applies to reflected sound only. Range [0.892f .. 1.0f] Default: 0.994f

EaxReverbAirAbsorptionGainHF summary>The property HF reference determines the frequency at which the high-frequency effects created by Reverb properties are measured. Unit: Hz Range [1000.0f .. 20000.0f] Default: 5000.0f

EaxReverbHFReference summary>The property LF reference determines the frequency at which the low-frequency effects created by Reverb properties are measured. Unit: Hz Range [20.0f .. 1000.0f] Default: 250.0f

EaxReverbLFReference summary>The Room Rolloff Factor property is one of two methods available to attenuate the reflected sound (containing both reflections and reverberation) according to source-listener distance. It's defined the same way as [OpenAL](#) Rolloff Factor, but operates on reverb sound instead of direct-path sound. Range [0.0f .. 10.0f] Default: 0.0f

4.2.1.30 enum OpenTK::Audio::OpenAL::EfxEffecti

A list of valid Int32 Effect/GetEffect parameters.

Enumerator:

ChorusWaveform summary>This property sets the waveform shape of the low-frequency oscillator that controls the delay time of the delayed signals. Unit: (0) Sinusoid, (1) Triangle Range [0 .. 1] Default: 1 summary>This property controls the phase difference between the left and right low-frequency oscillators. At zero degrees the two low-frequency oscillators are synchronized. Unit: Degrees Range [-180 .. 180] Default: 90

ChorusPhase summary>Selects the shape of the low-frequency oscillator waveform that controls the amount of the delay of the sampled signal. Unit: (0) Sinusoid, (1) Triangle Range [0 .. 1] Default: 1

FlangerWaveform summary>This changes the phase difference between the left and right low-frequency oscillator's. At zero degrees the two low-frequency oscillators are synchronized. Range [-180 .. +180] Default: 0

FlangerPhase summary>These select which internal signals are added together to produce the output. Unit: (0) Down, (1) Up, (2) Off Range [0 .. 2] Default: 0

FrequencyShifterLeftDirection summary>These select which internal signals are added together to produce the output. Unit: (0) Down, (1) Up, (2) Off Range [0 .. 2] Default: 0

FrequencyShifterRightDirection summary>Sets the vocal morpher 4-band formant filter A, used to impose vocal tract effects upon the input signal. The vocal morpher is not necessarily intended for use on voice signals; it is primarily intended for pitched noise effects, vocal-like wind effects, etc. Unit: Use enum EfxFormantFilterSettings Range [0 .. 29] Default: 0, "Phoneme A"

VocalMorpherPhonemeA summary>This is used to adjust the pitch of phoneme filter A in 1-semitone increments. Unit: Semitones Range [-24 .. +24] Default: 0

VocalMorpherPhonemeACoarseTuning summary>Sets the vocal morpher 4-band formant filter B, used to impose vocal tract effects upon the input signal. The vocal morpher is not necessarily intended for use on voice signals; it is primarily intended for pitched noise effects, vocal-like wind effects, etc. Unit: Use enum EfxFormantFilterSettings Range [0 .. 29] Default: 10, "Phoneme ER"

VocalMorpherPhonemeB summary>This is used to adjust the pitch of phoneme filter B in 1-semitone increments. Unit: Semitones Range [-24 .. +24] Default: 0

VocalMorpherPhonemeBCoarseTuning summary>This controls the shape of the low-frequency oscillator used to morph between the two phoneme filters. Unit: (0) Sinusoid, (1) Triangle, (2) Sawtooth Range [0 .. 2] Default: 0

VocalMorpherWaveform summary>This sets the number of semitones by which the pitch is shifted. There are 12 semitones per octave. Unit: Semitones Range [-12 .. +12] Default: +12

PitchShifterCoarseTune summary>This sets the number of cents between Semitones a pitch is shifted. A Cent is 1/100th of a Semitone. Unit: Cents Range [-50 .. +50] Default: 0

PitchShifterFineTune summary>This controls which waveform is used as the carrier signal. Traditional ring modulator and tremolo effects generally use a sinusoidal carrier. Unit: (0) Sinusoid, (1) Sawtooth, (2) Square Range [0 .. 2] Default: 0

RingModulatorWaveform summary>Enabling this will result in audio exhibiting smaller variation in intensity between the loudest and quietest portions. Unit: (0) Off, (1) On Range [0 .. 1] Default: 1

CompressorOnoff summary>When this flag is set, the high-frequency decay time automatically stays below a limit value that's derived from the setting of the property Air Absorption HF. Unit: (0) False, (1) True Range [False, True] Default: True

ReverbDecayHFLimit summary>When this flag is set, the high-frequency decay time automatically stays below a limit value that's derived from the setting of the property AirAbsorptionGainHF. Unit: (0) False, (1) True Range [False, True] Default: True

EffectType Used with the enum EfxEffectType as it's parameter.

4.2.1.31 enum OpenTK::Audio::OpenAL::EfxEffectType

Effect type definitions to be used with EfxEffecti.EffectType.

Enumerator:

Null summary>No Effect, disable. This Effect type is used when an Effect object is initially created. summary>The Reverb effect is the standard Effects Extension's environmental reverberation effect. It is available on all Generic Software and Generic Hardware devices.

Reverb summary>The Chorus effect essentially replays the input audio accompanied by another slightly delayed version of the signal, creating a "doubling" effect.

Chorus summary>The Distortion effect simulates turning up (overdriving) the gain stage on a guitar amplifier or adding a distortion pedal to an instrument's output.

Distortion summary>The Echo effect generates discrete, delayed instances of the input signal.

Echo summary>The Flanger effect creates a "tearing" or "whooshing" sound, like a jet flying overhead.

Flanger summary>The Frequency shifter is a single-sideband modulator, which translates all the component frequencies of the input signal by an equal amount.

FrequencyShifter summary>The Vocal morpher consists of a pair of 4-band formant filters, used to impose vocal tract effects upon the input signal.

VocalMorpher summary>The Pitch shifter applies time-invariant pitch shifting to the input signal, over a one octave range and controllable at a semi-tone and cent resolution.

PitchShifter summary>The Ring modulator multiplies an input signal by a carrier signal in the time domain, resulting in tremolo or inharmonic effects.

RingModulator summary>The Auto-wah effect emulates the sound of a wah-wah pedal used with an electric guitar, or a mute on a brass instrument.

Autowah summary>The Compressor will boost quieter portions of the audio, while louder portions will stay the same or may even be reduced.

Compressor summary>The Equalizer is very flexible, providing tonal control over four different adjustable frequency ranges.

Equalizer summary>The EAX Reverb has a more advanced parameter set than EfxEffectType.Reverb, but is only natively supported on devices that support the EAX 3.0 or above.

4.2.1.32 enum OpenTK::Audio::OpenAL::EfxFilterf

A list of valid 32-bits Float Filter/GetFilter parameters.

Enumerator:

LowpassGain summary>Range [0.0f .. 1.0f] Default: 1.0f

LowpassGainHF summary>Range [0.0f .. 1.0f] Default: 1.0f

HighpassGain summary>Range [0.0f .. 1.0f] Default: 1.0f

HighpassGainLF summary>Range [0.0f .. 1.0f] Default: 1.0f

BandpassGain summary>Range [0.0f .. 1.0f] Default: 1.0f

BandpassGainLF summary>Range [0.0f .. 1.0f] Default: 1.0f

4.2.1.33 enum OpenTK::Audio::OpenAL::EfxFilteri

A list of valid Int32 Filter/GetFilter parameters.

Enumerator:

FilterType Used with the enum EfxFilterType as Parameter to select a filter logic.

4.2.1.34 enum OpenTK::Audio::OpenAL::EfxFilterType

Filter type definitions to be used with EfxFilteri.FilterType.

Enumerator:

Null summary>No Filter, disable. This Filter type is used when a Filter object is initially created.

Lowpass A low-pass filter is used to remove high frequency content from a signal. summary>Currently not implemented. A high-pass filter is used to remove low frequency content from a signal.

Highpass summary>Currently not implemented. A band-pass filter is used to remove high and low frequency content from a signal.

4.2.1.35 enum OpenTK::Audio::OpenAL::EfxFormantFilterSettings

Vocal morpher effect parameters. If both parameters are set to the same phoneme, that determines the filtering effect that will be heard. If these two parameters are set to different phonemes, the filtering effect will morph between the two settings at a rate specified by `EfxEffectf.VocalMorpherRate`.

Enumerator:

<i>VocalMorpherPhonemeA</i>	The A phoneme of the vocal morpher.
<i>VocalMorpherPhonemeE</i>	The E phoneme of the vocal morpher.
<i>VocalMorpherPhonemeI</i>	The I phoneme of the vocal morpher.
<i>VocalMorpherPhonemeO</i>	The O phoneme of the vocal morpher.
<i>VocalMorpherPhonemeU</i>	The U phoneme of the vocal morpher.
<i>VocalMorpherPhonemeAA</i>	The AA phoneme of the vocal morpher.
<i>VocalMorpherPhonemeAE</i>	The AE phoneme of the vocal morpher.
<i>VocalMorpherPhonemeAH</i>	The AH phoneme of the vocal morpher.
<i>VocalMorpherPhonemeAO</i>	The AO phoneme of the vocal morpher.
<i>VocalMorpherPhonemeEH</i>	The EH phoneme of the vocal morpher.
<i>VocalMorpherPhonemeER</i>	The ER phoneme of the vocal morpher.
<i>VocalMorpherPhonemeIH</i>	The IH phoneme of the vocal morpher.
<i>VocalMorpherPhonemeIY</i>	The IY phoneme of the vocal morpher.
<i>VocalMorpherPhonemeUH</i>	The UH phoneme of the vocal morpher.
<i>VocalMorpherPhonemeUW</i>	The UW phoneme of the vocal morpher.
<i>VocalMorpherPhonemeB</i>	The B phoneme of the vocal morpher.
<i>VocalMorpherPhonemeD</i>	The D phoneme of the vocal morpher.
<i>VocalMorpherPhonemeF</i>	The F phoneme of the vocal morpher.
<i>VocalMorpherPhonemeG</i>	The G phoneme of the vocal morpher.
<i>VocalMorpherPhonemeJ</i>	The J phoneme of the vocal morpher.
<i>VocalMorpherPhonemeK</i>	The K phoneme of the vocal morpher.
<i>VocalMorpherPhonemeL</i>	The L phoneme of the vocal morpher.
<i>VocalMorpherPhonemeM</i>	The M phoneme of the vocal morpher.
<i>VocalMorpherPhonemeN</i>	The N phoneme of the vocal morpher.
<i>VocalMorpherPhonemeP</i>	The P phoneme of the vocal morpher.
<i>VocalMorpherPhonemeR</i>	The R phoneme of the vocal morpher.
<i>VocalMorpherPhonemeS</i>	The S phoneme of the vocal morpher.
<i>VocalMorpherPhonemeT</i>	The T phoneme of the vocal morpher.
<i>VocalMorpherPhonemeV</i>	The V phoneme of the vocal morpher.
<i>VocalMorpherPhonemeZ</i>	The Z phoneme of the vocal morpher.

4.3 Package OpenTK.Compute

4.4 Package OpenTK.Graphics

Packages

- package [ES10](#)
- package [ES11](#)
- package [ES20](#)
- package [OpenGL](#)

Classes

- struct [Color4](#)
Represents a color with 4 floating-point components (R, G, B, A).
- struct [ColorFormat](#)
Defines the [ColorFormat](#) component of a [GraphicsMode](#).
- class [GraphicsBindingsBase](#)
Implements [BindingsBase](#) for the [OpenTK.Graphics](#) namespace ([OpenGL](#) and [OpenGL|ES](#)).
- class [GraphicsContext](#)
Represents and provides methods to manipulate an [OpenGL](#) render context.
- class [GraphicsContextException](#)
Represents errors related to a [GraphicsContext](#).
- class [GraphicsContextMissingException](#)
Thrown when an operation that required [GraphicsContext](#) is performed, when no [GraphicsContext](#) is current in the calling thread.
- class [GraphicsContextVersion](#)
Defines the version information of a [GraphicsContext](#).
- class [GraphicsErrorException](#)
*Identifies a specific [OpenGL](#) or [OpenGL|ES](#) error. Such exceptions are only thrown when [OpenGL](#) or [OpenGL|ES](#) automatic error checking is enabled - [GraphicsContext.ErrorChecking](#) property. Important: Do **not** catch this exception. Rather, fix the underlying issue that caused the error.*

- class [GraphicsMode](#)

Defines the format for graphics operations.

- class [GraphicsModeException](#)

Represents errors related to unavailable graphics parameters.

- interface [IGraphicsContext](#)

Provides methods for creating and interacting with an [OpenGL](#) context.

- interface [IGraphicsContextInternal](#)

Provides methods to create new GraphicsContexts. Should only be used for extending OpenTK.

Enumerations

- enum [GraphicsContextFlags](#) { [Default](#) = 0x0000, [Debug](#) = 0x0001, [ForwardCompatible](#) = 0x0002, [Embedded](#) = 0x0004 }

Enumerates various flags that affect the creation of new GraphicsContexts.

4.4.1 Enumeration Type Documentation

4.4.1.1 enum OpenTK::Graphics::GraphicsContextFlags

Enumerates various flags that affect the creation of new GraphicsContexts.

Enumerator:

Default The default value of the GraphicsContextFlags enumeration.

Debug Indicates that this is a debug [GraphicsContext](#). Debug contexts may provide additional debugging information at the cost of performance.

ForwardCompatible Indicates that this is a forward compatible [GraphicsContext](#). Forward-compatible contexts do not support functionality marked as deprecated in the current [GraphicsContextVersion](#). Forward-compatible contexts are defined only for [OpenGL](#) versions 3.0 and later.

Embedded Indicates that this [GraphicsContext](#) is targeting OpenGL|ES.

4.5 Package OpenTK.Graphics.ES10

Classes

- class [GL](#)

Provides access to [OpenGL ES 1.0](#) methods.

Enumerations

- enum **All** {
False = ((int)0), **NoError** = ((int)0), **Zero** = ((int)0), **Points** = ((int)0x0000),
DepthBufferBit = ((int)0x00000100), **StencilBufferBit** = ((int)0x00000400),
ColorBufferBit = ((int)0x00004000), **Lines** = ((int)0x0001),
LineLoop = ((int)0x0002), **LineStrip** = ((int)0x0003), **Triangles** =
((int)0x0004), **TriangleStrip** = ((int)0x0005),
TriangleFan = ((int)0x0006), **Add** = ((int)0x0104), **Never** = ((int)0x0200), **Less**
= ((int)0x0201),
Equal = ((int)0x0202), **Lequal** = ((int)0x0203), **Greater** = ((int)0x0204), **Note-**
qual = ((int)0x0205),
Gequal = ((int)0x0206), **Always** = ((int)0x0207), **SrcColor** = ((int)0x0300),
OneMinusSrcColor = ((int)0x0301),
SrcAlpha = ((int)0x0302), **OneMinusSrcAlpha** = ((int)0x0303), **DstAlpha** =
((int)0x0304), **OneMinusDstAlpha** = ((int)0x0305),
DstColor = ((int)0x0306), **OneMinusDstColor** = ((int)0x0307), **SrcAlphaSat-**
urate = ((int)0x0308), **Front** = ((int)0x0404),
Back = ((int)0x0405), **FrontAndBack** = ((int)0x0408), **InvalidEnum** =
((int)0x0500), **InvalidValue** = ((int)0x0501),
InvalidOperation = ((int)0x0502), **StackOverflow** = ((int)0x0503), **StackUn-**
derflow = ((int)0x0504), **OutOfMemory** = ((int)0x0505),
Exp = ((int)0x0800), **Exp2** = ((int)0x0801), **Cw** = ((int)0x0900), **Ccw** =
((int)0x0901),
PointSmooth = ((int)0x0B10), **SmoothPointSizeRange** = ((int)0x0B12),
LineSmooth = ((int)0x0B20), **SmoothLineWidthRange** = ((int)0x0B22),
CullFace = ((int)0x0B44), **Lighting** = ((int)0x0B50), **LightModelTwoSide** =
((int)0x0B52), **LightModelAmbient** = ((int)0x0B53),
ColorMaterial = ((int)0x0B57), **Fog** = ((int)0x0B60), **FogDensity** =
((int)0x0B62), **FogStart** = ((int)0x0B63),
FogEnd = ((int)0x0B64), **FogMode** = ((int)0x0B65), **FogColor** =
((int)0x0B66), **DepthTest** = ((int)0x0B71),

StencilTest = ((int)0x0B90), **Normalize** = ((int)0x0BA1), **AlphaTest** = ((int)0x0BC0), **Dither** = ((int)0x0BD0),

Blend = ((int)0x0BE2), **ColorLogicOp** = ((int)0x0BF2), **ScissorTest** = ((int)0x0C11), **PerspectiveCorrectionHint** = ((int)0x0C50),

PointSmoothHint = ((int)0x0C51), **LineSmoothHint** = ((int)0x0C52), **PolygonSmoothHint** = ((int)0x0C53), **FogHint** = ((int)0x0C54),

UnpackAlignment = ((int)0x0CF5), **PackAlignment** = ((int)0x0D05), **MaxLights** = ((int)0x0D31), **MaxTextureSize** = ((int)0x0D33),

MaxModelviewStackDepth = ((int)0x0D36), **MaxProjectionStackDepth** = ((int)0x0D38), **MaxTextureStackDepth** = ((int)0x0D39), **MaxViewportDims** = ((int)0x0D3A),

SubpixelBits = ((int)0x0D50), **RedBits** = ((int)0x0D52), **GreenBits** = ((int)0x0D53), **BlueBits** = ((int)0x0D54),

AlphaBits = ((int)0x0D55), **DepthBits** = ((int)0x0D56), **StencilBits** = ((int)0x0D57), **Texture2D** = ((int)0x0DE1),

DontCare = ((int)0x1100), **Fastest** = ((int)0x1101), **Nicest** = ((int)0x1102), **Ambient** = ((int)0x1200),

Diffuse = ((int)0x1201), **Specular** = ((int)0x1202), **Position** = ((int)0x1203), **SpotDirection** = ((int)0x1204),

SpotExponent = ((int)0x1205), **SpotCutoff** = ((int)0x1206), **ConstantAttenuation** = ((int)0x1207), **LinearAttenuation** = ((int)0x1208),

QuadraticAttenuation = ((int)0x1209), **Byte** = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402),

UnsignedShort = ((int)0x1403), **Float** = ((int)0x1406), **Fixed** = ((int)0x140C), **Clear** = ((int)0x1500),

And = ((int)0x1501), **AndReverse** = ((int)0x1502), **Copy** = ((int)0x1503), **AndInverted** = ((int)0x1504),

Noop = ((int)0x1505), **Xor** = ((int)0x1506), **Or** = ((int)0x1507), **Nor** = ((int)0x1508),

Equiv = ((int)0x1509), **Invert** = ((int)0x150A), **OrReverse** = ((int)0x150B), **CopyInverted** = ((int)0x150C),

OrInverted = ((int)0x150D), **Nand** = ((int)0x150E), **Set** = ((int)0x150F), **Emission** = ((int)0x1600),

Shininess = ((int)0x1601), **AmbientAndDiffuse** = ((int)0x1602), **Modelview** = ((int)0x1700), **Projection** = ((int)0x1701),

Texture = ((int)0x1702), **Alpha** = ((int)0x1906), **Rgb** = ((int)0x1907), **Rgba** = ((int)0x1908),

Luminance = ((int)0x1909), **LuminanceAlpha** = ((int)0x190A), **Flat** = ((int)0x1D00), **Smooth** = ((int)0x1D01),

Keep = ((int)0x1E00), **Replace** = ((int)0x1E01), **Incr** = ((int)0x1E02), **Decr** = ((int)0x1E03),

Vendor = ((int)0x1F00), **Renderer** = ((int)0x1F01), **Version** = ((int)0x1F02), **Extensions** = ((int)0x1F03),

Modulate = ((int)0x2100), **Decal** = ((int)0x2101), **TextureEnvMode** = ((int)0x2200), **TextureEnvColor** = ((int)0x2201),

TextureEnv = ((int)0x2300), **Nearest** = ((int)0x2600), **Linear** = ((int)0x2601), **NearestMipmapNearest** = ((int)0x2700),

LinearMipmapNearest = ((int)0x2701), **NearestMipmapLinear** = ((int)0x2702), **LinearMipmapLinear** = ((int)0x2703), **TextureMagFilter** = ((int)0x2800),

TextureMinFilter = ((int)0x2801), **TextureWrapS** = ((int)0x2802), **TextureWrapT** = ((int)0x2803), **Repeat** = ((int)0x2901),

Light0 = ((int)0x4000), **Light1** = ((int)0x4001), **Light2** = ((int)0x4002), **Light3** = ((int)0x4003),

Light4 = ((int)0x4004), **Light5** = ((int)0x4005), **Light6** = ((int)0x4006), **Light7** = ((int)0x4007),

UnsignedShort4444 = ((int)0x8033), **UnsignedShort5551** = ((int)0x8034), **PolygonOffsetFill** = ((int)0x8037), **RescaleNormal** = ((int)0x803A),

VertexArray = ((int)0x8074), **NormalArray** = ((int)0x8075), **ColorArray** = ((int)0x8076), **TextureCoordArray** = ((int)0x8078),

Multisample = ((int)0x809D), **SampleAlphaToCoverage** = ((int)0x809E), **SampleAlphaToOne** = ((int)0x809F), **SampleCoverage** = ((int)0x80A0),

MaxElementsVertices = ((int)0x80E8), **MaxElementsIndices** = ((int)0x80E9), **ClampToEdge** = ((int)0x812F), **UnsignedShort565** = ((int)0x8363),

AliasedPointSizeRange = ((int)0x846D), **AliasedLineWidthRange** = ((int)0x846E), **Texture0** = ((int)0x84C0), **Texture1** = ((int)0x84C1),

Texture2 = ((int)0x84C2), **Texture3** = ((int)0x84C3), **Texture4** = ((int)0x84C4), **Texture5** = ((int)0x84C5),

Texture6 = ((int)0x84C6), **Texture7** = ((int)0x84C7), **Texture8** = ((int)0x84C8), **Texture9** = ((int)0x84C9),

Texture10 = ((int)0x84CA), **Texture11** = ((int)0x84CB), **Texture12** = ((int)0x84CC), **Texture13** = ((int)0x84CD),

Texture14 = ((int)0x84CE), **Texture15** = ((int)0x84CF), **Texture16** = ((int)0x84D0), **Texture17** = ((int)0x84D1),

Texture18 = ((int)0x84D2), **Texture19** = ((int)0x84D3), **Texture20** = ((int)0x84D4), **Texture21** = ((int)0x84D5),

Texture22 = ((int)0x84D6), **Texture23** = ((int)0x84D7), **Texture24** = ((int)0x84D8), **Texture25** = ((int)0x84D9),

```

Texture26 = ((int)0x84DA), Texture27 = ((int)0x84DB), Texture28 =
((int)0x84DC), Texture29 = ((int)0x84DD),
Texture30 = ((int)0x84DE), Texture31 = ((int)0x84DF), MaxTextureUnits =
((int)0x84E2), NumCompressedTextureFormats = ((int)0x86A2),
CompressedTextureFormats = ((int)0x86A3), Palette4Rgb8Oes =
((int)0x8B90), Palette4Rgba8Oes = ((int)0x8B91), Palette4R5G6B5Oes
= ((int)0x8B92),
Palette4Rgba4Oes = ((int)0x8B93), Palette4Rgb5A1Oes = ((int)0x8B94),
Palette8Rgb8Oes = ((int)0x8B95), Palette8Rgba8Oes = ((int)0x8B96),
Palette8R5G6B5Oes = ((int)0x8B97), Palette8Rgba4Oes = ((int)0x8B98),
Palette8Rgb5A1Oes = ((int)0x8B99), ImplementationColorReadTypeOes =
((int)0x8B9A),
ImplementationColorReadFormatOes = ((int)0x8B9B), OesCompressed-
PalettedTexture = ((int)1), OesReadFormat = ((int)1), OesVersion10 =
((int)1),
One = ((int)1), True = ((int)1) }
• enum AlphaFunction {
Never = ((int)0x0200), Less = ((int)0x0201), Equal = ((int)0x0202), Lequal =
((int)0x0203),
Greater = ((int)0x0204), Notequal = ((int)0x0205), Gequal = ((int)0x0206),
Always = ((int)0x0207) }
• enum BeginMode {
Points = ((int)0x0000), Lines = ((int)0x0001), LineLoop = ((int)0x0002),
LineStrip = ((int)0x0003),
Triangles = ((int)0x0004), TriangleStrip = ((int)0x0005), TriangleFan =
((int)0x0006) }
• enum BlendingFactorDest {
Zero = ((int)0), SrcColor = ((int)0x0300), OneMinusSrcColor = ((int)0x0301),
SrcAlpha = ((int)0x0302),
OneMinusSrcAlpha = ((int)0x0303), DstAlpha = ((int)0x0304), OneMinusD-
stAlpha = ((int)0x0305), One = ((int)1) }
• enum BlendingFactorSrc { DstColor = ((int)0x0306), OneMinusDstColor =
((int)0x0307), SrcAlphaSaturate = ((int)0x0308) }
• enum Boolean { False = ((int)0), True = ((int)1) }
• enum ClearBufferMask { DepthBufferBit = ((int)0x00000100), Stencil-
BufferBit = ((int)0x00000400), ColorBufferBit = ((int)0x00004000) }
• enum CullFaceMode { Front = ((int)0x0404), Back = ((int)0x0405),
FrontAndBack = ((int)0x0408) }
• enum DataType {
Byte = ((int)0x1400), UnsignedByte = ((int)0x1401), Short = ((int)0x1402),
UnsignedShort = ((int)0x1403),
Float = ((int)0x1406), Fixed = ((int)0x140C) }

```

- enum **EnableCap** {
 - PointSmooth** = ((int)0x0B10), **LineSmooth** = ((int)0x0B20), **CullFace** = ((int)0x0B44), **Lighting** = ((int)0x0B50),
 - ColorMaterial** = ((int)0x0B57), **Fog** = ((int)0x0B60), **DepthTest** = ((int)0x0B71), **StencilTest** = ((int)0x0B90),
 - Normalize** = ((int)0x0BA1), **AlphaTest** = ((int)0x0BC0), **Dither** = ((int)0x0BD0), **Blend** = ((int)0x0BE2),
 - ColorLogicOp** = ((int)0x0BF2), **ScissorTest** = ((int)0x0C11), **Texture2D** = ((int)0x0DE1), **PolygonOffsetFill** = ((int)0x8037),
 - RescaleNormal** = ((int)0x803A), **VertexArray** = ((int)0x8074), **NormalArray** = ((int)0x8075), **ColorArray** = ((int)0x8076),
 - TextureCoordArray** = ((int)0x8078), **Multisample** = ((int)0x809D), **SampleAlphaToCoverage** = ((int)0x809E), **SampleAlphaToOne** = ((int)0x809F),
 - SampleCoverage** = ((int)0x80A0) }
- enum **ErrorCode** {
 - NoError** = ((int)0), **InvalidEnum** = ((int)0x0500), **InvalidValue** = ((int)0x0501), **InvalidOperation** = ((int)0x0502),
 - StackOverflow** = ((int)0x0503), **StackUnderflow** = ((int)0x0504), **Out-OfMemory** = ((int)0x0505) }
- enum **Extensions** { **OesCompressedPalettedTexture** = ((int)1), **OesReadFormat** = ((int)1), **OesVersion10** = ((int)1) }
- enum **FogMode** { **Exp** = ((int)0x0800), **Exp2** = ((int)0x0801) }
- enum **FogParameter** {
 - FogDensity** = ((int)0x0B62), **FogStart** = ((int)0x0B63), **FogEnd** = ((int)0x0B64), **FogMode** = ((int)0x0B65),
 - FogColor** = ((int)0x0B66) }
- enum **FrontFaceDirection** { **Cw** = ((int)0x0900), **Ccw** = ((int)0x0901) }
- enum **GetPName** {
 - SmoothPointSizeRange** = ((int)0x0B12), **SmoothLineWidthRange** = ((int)0x0B22), **MaxLights** = ((int)0x0D31), **MaxTextureSize** = ((int)0x0D33),
 - MaxModelviewStackDepth** = ((int)0x0D36), **MaxProjectionStackDepth** = ((int)0x0D38), **MaxTextureStackDepth** = ((int)0x0D39), **MaxViewportDims** = ((int)0x0D3A),
 - SubpixelBits** = ((int)0x0D50), **RedBits** = ((int)0x0D52), **GreenBits** = ((int)0x0D53), **BlueBits** = ((int)0x0D54),
 - AlphaBits** = ((int)0x0D55), **DepthBits** = ((int)0x0D56), **StencilBits** = ((int)0x0D57), **MaxElementsVertices** = ((int)0x80E8),
 - MaxElementsIndices** = ((int)0x80E9), **AliasedPointSizeRange** = ((int)0x846D), **AliasedLineWidthRange** = ((int)0x846E), **MaxTextureUnits** = ((int)0x84E2),

```

NumCompressedTextureFormats = ((int)0x86A2), CompressedTextureFormats = ((int)0x86A3), ImplementationColorReadTypeOes = ((int)0x8B9A), ImplementationColorReadFormatOes = ((int)0x8B9B) }
• enum HintMode { DontCare = ((int)0x1100), Fastest = ((int)0x1101), Nicest = ((int)0x1102) }
• enum HintTarget {
    PerspectiveCorrectionHint = ((int)0x0C50), PointSmoothHint = ((int)0x0C51), LineSmoothHint = ((int)0x0C52), PolygonSmoothHint = ((int)0x0C53),
    FogHint = ((int)0x0C54) }
• enum LightModelParameter { LightModelTwoSide = ((int)0x0B52), LightModelAmbient = ((int)0x0B53) }
• enum LightName {
    Light0 = ((int)0x4000), Light1 = ((int)0x4001), Light2 = ((int)0x4002), Light3 = ((int)0x4003),
    Light4 = ((int)0x4004), Light5 = ((int)0x4005), Light6 = ((int)0x4006), Light7 = ((int)0x4007) }
• enum LightParameter {
    Ambient = ((int)0x1200), Diffuse = ((int)0x1201), Specular = ((int)0x1202), Position = ((int)0x1203),
    SpotDirection = ((int)0x1204), SpotExponent = ((int)0x1205), SpotCutoff = ((int)0x1206), ConstantAttenuation = ((int)0x1207),
    LinearAttenuation = ((int)0x1208), QuadraticAttenuation = ((int)0x1209) }
• enum LogicOp {
    Clear = ((int)0x1500), And = ((int)0x1501), AndReverse = ((int)0x1502), Copy = ((int)0x1503),
    AndInverted = ((int)0x1504), Noop = ((int)0x1505), Xor = ((int)0x1506), Or = ((int)0x1507),
    Nor = ((int)0x1508), Equiv = ((int)0x1509), Invert = ((int)0x150A), OrReverse = ((int)0x150B),
    CopyInverted = ((int)0x150C), OrInverted = ((int)0x150D), Nand = ((int)0x150E), Set = ((int)0x150F) }
• enum MaterialParameter { Emission = ((int)0x1600), Shininess = ((int)0x1601), AmbientAndDiffuse = ((int)0x1602) }
• enum MatrixMode { Modelview = ((int)0x1700), Projection = ((int)0x1701), Texture = ((int)0x1702) }
• enum PixelFormat {
    Alpha = ((int)0x1906), Rgb = ((int)0x1907), Rgba = ((int)0x1908), Luminance = ((int)0x1909),
    LuminanceAlpha = ((int)0x190A) }

```

- enum **PixelFormatInternalFormat** {
Palette4Rgb8Oes = ((int)0x8B90), **Palette4Rgba8Oes** = ((int)0x8B91),
Palette4R5G6B5Oes = ((int)0x8B92), **Palette4Rgba4Oes** = ((int)0x8B93),
Palette4Rgb5A1Oes = ((int)0x8B94), **Palette8Rgb8Oes** = ((int)0x8B95),
Palette8Rgba8Oes = ((int)0x8B96), **Palette8R5G6B5Oes** = ((int)0x8B97),
Palette8Rgba4Oes = ((int)0x8B98), **Palette8Rgb5A1Oes** = ((int)0x8B99) }
- enum **PixelStoreParameter** { **UnpackAlignment** = ((int)0x0CF5), **PackAlignment** = ((int)0x0D05) }
- enum **PixelType** { **UnsignedShort4444** = ((int)0x8033), **UnsignedShort5551** = ((int)0x8034), **UnsignedShort565** = ((int)0x8363) }
- enum **ShadingModel** { **Flat** = ((int)0x1D00), **Smooth** = ((int)0x1D01) }
- enum **StencilOp** { **Keep** = ((int)0x1E00), **Replace** = ((int)0x1E01), **Incr** = ((int)0x1E02), **Decr** = ((int)0x1E03) }
- enum **StringName** { **Vendor** = ((int)0x1F00), **Renderer** = ((int)0x1F01), **Version** = ((int)0x1F02), **Extensions** = ((int)0x1F03) }
- enum **TextureEnvMode** { **Add** = ((int)0x0104), **Modulate** = ((int)0x2100), **Decal** = ((int)0x2101) }
- enum **TextureEnvParameter** { **TextureEnvMode** = ((int)0x2200), **TextureEnvColor** = ((int)0x2201) }
- enum **TextureEnvTarget** { **TextureEnv** = ((int)0x2300) }
- enum **TextureMagFilter** { **Nearest** = ((int)0x2600), **Linear** = ((int)0x2601) }
- enum **TextureMinFilter** { **NearestMipmapNearest** = ((int)0x2700), **LinearMipmapNearest** = ((int)0x2701), **NearestMipmapLinear** = ((int)0x2702), **LinearMipmapLinear** = ((int)0x2703) }
- enum **TextureParameterName** { **TextureMagFilter** = ((int)0x2800), **TextureMinFilter** = ((int)0x2801), **TextureWrapS** = ((int)0x2802), **TextureWrapT** = ((int)0x2803) }
- enum **TextureUnit** {
Texture0 = ((int)0x84C0), **Texture1** = ((int)0x84C1), **Texture2** = ((int)0x84C2),
Texture3 = ((int)0x84C3),
Texture4 = ((int)0x84C4), **Texture5** = ((int)0x84C5), **Texture6** = ((int)0x84C6),
Texture7 = ((int)0x84C7),
Texture8 = ((int)0x84C8), **Texture9** = ((int)0x84C9), **Texture10** = ((int)0x84CA),
Texture11 = ((int)0x84CB),
Texture12 = ((int)0x84CC), **Texture13** = ((int)0x84CD), **Texture14** = ((int)0x84CE),
Texture15 = ((int)0x84CF),
Texture16 = ((int)0x84D0), **Texture17** = ((int)0x84D1), **Texture18** = ((int)0x84D2),
Texture19 = ((int)0x84D3),
Texture20 = ((int)0x84D4), **Texture21** = ((int)0x84D5), **Texture22** = ((int)0x84D6),
Texture23 = ((int)0x84D7),
Texture24 = ((int)0x84D8), **Texture25** = ((int)0x84D9), **Texture26** = ((int)0x84DA),
Texture27 = ((int)0x84DB),

```

Texture28 = ((int)0x84DC), Texture29 = ((int)0x84DD), Texture30 =
((int)0x84DE), Texture31 = ((int)0x84DF) }
• enum TextureWrapMode { Repeat = ((int)0x2901), ClampToEdge =
((int)0x812F) }

```

4.6 Package OpenTK.Graphics.ES11

Classes

- class [GL](#)
Provides access to [OpenGL ES 1.1](#) methods.

Enumerations

- enum **All** {
False = ((int)0), **NoError** = ((int)0), **NoneOes** = ((int)0), **Zero** = ((int)0),
Points = ((int)0x0000), **DepthBufferBit** = ((int)0x00000100), **StencilBufferBit**
= ((int)0x00000400), **ColorBufferBit** = ((int)0x00004000),
Lines = ((int)0x0001), **LineLoop** = ((int)0x0002), **LineStrip** = ((int)0x0003),
Triangles = ((int)0x0004),
TriangleStrip = ((int)0x0005), **TriangleFan** = ((int)0x0006), **Add** =
((int)0x0104), **Never** = ((int)0x0200),
Less = ((int)0x0201), **Equal** = ((int)0x0202), **Lequal** = ((int)0x0203), **Greater**
= ((int)0x0204),
Notequal = ((int)0x0205), **Gequal** = ((int)0x0206), **Always** = ((int)0x0207), **SrcColor**
= ((int)0x0300),
OneMinusSrcColor = ((int)0x0301), **SrcAlpha** = ((int)0x0302), **OneMinusSrcAlpha**
= ((int)0x0303), **DstAlpha** = ((int)0x0304),
OneMinusDstAlpha = ((int)0x0305), **DstColor** = ((int)0x0306), **OneMinusDstColor**
= ((int)0x0307), **SrcAlphaSaturate** = ((int)0x0308),
Front = ((int)0x0404), **Back** = ((int)0x0405), **FrontAndBack** = ((int)0x0408),
InvalidEnum = ((int)0x0500),
InvalidValue = ((int)0x0501), **InvalidOperation** = ((int)0x0502), **StackOverflow**
= ((int)0x0503), **StackUnderflow** = ((int)0x0504),
OutOfMemory = ((int)0x0505), **InvalidFramebufferOperationOes** =
((int)0x0506), **Exp** = ((int)0x0800), **Exp2** = ((int)0x0801),
Cw = ((int)0x0900), **Ccw** = ((int)0x0901), **CurrentColor** = ((int)0x0B00), **CurrentNormal**
= ((int)0x0B02),

CurrentTextureCoords = ((int)0x0B03), **PointSmooth** = ((int)0x0B10), **PointSize** = ((int)0x0B11), **SmoothPointSizeRange** = ((int)0x0B12),
LineSmooth = ((int)0x0B20), **LineWidth** = ((int)0x0B21), **SmoothLineWidthRange** = ((int)0x0B22), **CullFace** = ((int)0x0B44),
CullFaceMode = ((int)0x0B45), **FrontFace** = ((int)0x0B46), **Lighting** = ((int)0x0B50), **LightModelTwoSide** = ((int)0x0B52),
LightModelAmbient = ((int)0x0B53), **ShadeModel** = ((int)0x0B54), **ColorMaterial** = ((int)0x0B57), **Fog** = ((int)0x0B60),
FogDensity = ((int)0x0B62), **FogStart** = ((int)0x0B63), **FogEnd** = ((int)0x0B64), **FogMode** = ((int)0x0B65),
FogColor = ((int)0x0B66), **DepthRange** = ((int)0x0B70), **DepthTest** = ((int)0x0B71), **DepthWritemask** = ((int)0x0B72),
DepthClearValue = ((int)0x0B73), **DepthFunc** = ((int)0x0B74), **StencilTest** = ((int)0x0B90), **StencilClearValue** = ((int)0x0B91),
StencilFunc = ((int)0x0B92), **StencilValueMask** = ((int)0x0B93), **StencilFail** = ((int)0x0B94), **StencilPassDepthFail** = ((int)0x0B95),
StencilPassDepthPass = ((int)0x0B96), **StencilRef** = ((int)0x0B97), **StencilWritemask** = ((int)0x0B98), **MatrixMode** = ((int)0x0BA0),
Normalize = ((int)0x0BA1), **Viewport** = ((int)0x0BA2), **ModelviewStackDepth** = ((int)0x0BA3), **ProjectionStackDepth** = ((int)0x0BA4),
TextureStackDepth = ((int)0x0BA5), **ModelviewMatrix** = ((int)0x0BA6), **ProjectionMatrix** = ((int)0x0BA7), **TextureMatrix** = ((int)0x0BA8),
AlphaTest = ((int)0x0BC0), **AlphaTestFunc** = ((int)0x0BC1), **AlphaTestRef** = ((int)0x0BC2), **Dither** = ((int)0x0BD0),
BlendDst = ((int)0x0BE0), **BlendSrc** = ((int)0x0BE1), **Blend** = ((int)0x0BE2), **LogicOpMode** = ((int)0x0BF0),
ColorLogicOp = ((int)0x0BF2), **ScissorBox** = ((int)0x0C10), **ScissorTest** = ((int)0x0C11), **ColorClearValue** = ((int)0x0C22),
ColorWritemask = ((int)0x0C23), **PerspectiveCorrectionHint** = ((int)0x0C50), **PointSmoothHint** = ((int)0x0C51), **LineSmoothHint** = ((int)0x0C52),
FogHint = ((int)0x0C54), **UnpackAlignment** = ((int)0x0CF5), **PackAlignment** = ((int)0x0D05), **AlphaScale** = ((int)0x0D1C),
MaxLights = ((int)0x0D31), **MaxClipPlanes** = ((int)0x0D32), **MaxClipPlanesImg** = ((int)0x0D32), **MaxTextureSize** = ((int)0x0D33),
MaxModelviewStackDepth = ((int)0x0D36), **MaxProjectionStackDepth** = ((int)0x0D38), **MaxTextureStackDepth** = ((int)0x0D39), **MaxViewportDims** = ((int)0x0D3A),
SubpixelBits = ((int)0x0D50), **RedBits** = ((int)0x0D52), **GreenBits** = ((int)0x0D53), **BlueBits** = ((int)0x0D54),

AlphaBits = ((int)0x0D55), **DepthBits** = ((int)0x0D56), **StencilBits** = ((int)0x0D57), **Texture2D** = ((int)0x0DE1),
DontCare = ((int)0x1100), **Fastest** = ((int)0x1101), **Nicest** = ((int)0x1102), **Ambient** = ((int)0x1200),
Diffuse = ((int)0x1201), **Specular** = ((int)0x1202), **Position** = ((int)0x1203), **SpotDirection** = ((int)0x1204),
SpotExponent = ((int)0x1205), **SpotCutoff** = ((int)0x1206), **ConstantAttenuation** = ((int)0x1207), **LinearAttenuation** = ((int)0x1208),
QuadraticAttenuation = ((int)0x1209), **Byte** = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402),
UnsignedShort = ((int)0x1403), **Float** = ((int)0x1406), **Fixed** = ((int)0x140C), **FixedOes** = ((int)0x140C),
Clear = ((int)0x1500), **And** = ((int)0x1501), **AndReverse** = ((int)0x1502), **Copy** = ((int)0x1503),
AndInverted = ((int)0x1504), **Noop** = ((int)0x1505), **Xor** = ((int)0x1506), **Or** = ((int)0x1507),
Nor = ((int)0x1508), **Equiv** = ((int)0x1509), **Invert** = ((int)0x150A), **OrReverse** = ((int)0x150B),
CopyInverted = ((int)0x150C), **OrInverted** = ((int)0x150D), **Nand** = ((int)0x150E), **Set** = ((int)0x150F),
Emission = ((int)0x1600), **Shininess** = ((int)0x1601), **AmbientAndDiffuse** = ((int)0x1602), **Modelview** = ((int)0x1700),
Projection = ((int)0x1701), **Texture** = ((int)0x1702), **Alpha** = ((int)0x1906), **Rgb** = ((int)0x1907),
Rgba = ((int)0x1908), **Luminance** = ((int)0x1909), **LuminanceAlpha** = ((int)0x190A), **Flat** = ((int)0x1D00),
Smooth = ((int)0x1D01), **Keep** = ((int)0x1E00), **Replace** = ((int)0x1E01), **Incr** = ((int)0x1E02),
Decr = ((int)0x1E03), **Vendor** = ((int)0x1F00), **Renderer** = ((int)0x1F01), **Version** = ((int)0x1F02),
Extensions = ((int)0x1F03), **Modulate** = ((int)0x2100), **Decal** = ((int)0x2101), **TextureEnvMode** = ((int)0x2200),
TextureEnvColor = ((int)0x2201), **TextureEnv** = ((int)0x2300), **TextureGenModeOes** = ((int)0x2500), **Nearest** = ((int)0x2600),
Linear = ((int)0x2601), **NearestMipmapNearest** = ((int)0x2700), **LinearMipmapNearest** = ((int)0x2701), **NearestMipmapLinear** = ((int)0x2702),
LinearMipmapLinear = ((int)0x2703), **TextureMagFilter** = ((int)0x2800), **TextureMinFilter** = ((int)0x2801), **TextureWrapS** = ((int)0x2802),
TextureWrapT = ((int)0x2803), **Repeat** = ((int)0x2901), **PolygonOffsetUnits** = ((int)0x2A00), **ClipPlane0** = ((int)0x3000),


```

ClipPlane0Img = ((int)0x3000), ClipPlane1 = ((int)0x3001), ClipPlane1Img
= ((int)0x3001), ClipPlane2 = ((int)0x3002),
ClipPlane2Img = ((int)0x3002), ClipPlane3 = ((int)0x3003), ClipPlane3Img
= ((int)0x3003), ClipPlane4 = ((int)0x3004),
ClipPlane4Img = ((int)0x3004), ClipPlane5 = ((int)0x3005), ClipPlane5Img
= ((int)0x3005), Light0 = ((int)0x4000),
Light1 = ((int)0x4001), Light2 = ((int)0x4002), Light3 = ((int)0x4003), Light4
= ((int)0x4004),
Light5 = ((int)0x4005), Light6 = ((int)0x4006), Light7 = ((int)0x4007), FuncAddOes = ((int)0x8006),
BlendEquationOes = ((int)0x8009), BlendEquationRgbOes = ((int)0x8009),
FuncSubtractOes = ((int)0x800A), FuncReverseSubtractOes =
((int)0x800B),
UnsignedShort4444 = ((int)0x8033), UnsignedShort5551 = ((int)0x8034),
PolygonOffsetFill = ((int)0x8037), PolygonOffsetFactor = ((int)0x8038),
RescaleNormal = ((int)0x803A), Rgb8Oes = ((int)0x8051), Rgba4Oes =
((int)0x8056), Rgb5A1Oes = ((int)0x8057),
Rgba8Oes = ((int)0x8058), TextureBinding2D = ((int)0x8069), VertexArray
= ((int)0x8074), NormalArray = ((int)0x8075),
ColorArray = ((int)0x8076), TextureCoordArray = ((int)0x8078), VertexAr-
raySize = ((int)0x807A), VertexArrayType = ((int)0x807B),
VertexArrayStride = ((int)0x807C), NormalArrayType = ((int)0x807E), Nor-
malArrayStride = ((int)0x807F), ColorArraySize = ((int)0x8081),
ColorArrayType = ((int)0x8082), ColorArrayStride = ((int)0x8083), Texture-
CoordArraySize = ((int)0x8088), TextureCoordArrayType = ((int)0x8089),
TextureCoordArrayStride = ((int)0x808A), VertexArrayPointer =
((int)0x808E), NormalArrayPointer = ((int)0x808F), ColorArrayPointer =
((int)0x8090),
TextureCoordArrayPointer = ((int)0x8092), Multisample = ((int)0x809D),
SampleAlphaToCoverage = ((int)0x809E), SampleAlphaToOne =
((int)0x809F),
SampleCoverage = ((int)0x80A0), SampleBuffers = ((int)0x80A8), Samples
= ((int)0x80A9), SampleCoverageValue = ((int)0x80AA),
SampleCoverageInvert = ((int)0x80AB), BlendDstRgbOes = ((int)0x80C8),
BlendSrcRgbOes = ((int)0x80C9), BlendDstAlphaOes = ((int)0x80CA),
BlendSrcAlphaOes = ((int)0x80CB), Bgra = ((int)0x80E1), PointSizeMin =
((int)0x8126), PointSizeMax = ((int)0x8127),
PointFadeThresholdSize = ((int)0x8128), PointDistanceAttenuation
= ((int)0x8129), ClampToEdge = ((int)0x812F), GenerateMipmap =
((int)0x8191),

```

GenerateMipmapHint = ((int)0x8192), **DepthComponent16Oes** = ((int)0x81A5), **DepthComponent24Oes** = ((int)0x81A6), **DepthComponent32Oes** = ((int)0x81A7),
UnsignedShort565 = ((int)0x8363), **UnsignedShort4444Rev** = ((int)0x8365), **UnsignedShort1555Rev** = ((int)0x8366), **MirroredRepeatOes** = ((int)0x8370),
AliasedPointSizeRange = ((int)0x846D), **AliasedLineWidthRange** = ((int)0x846E), **Texture0** = ((int)0x84C0), **Texture1** = ((int)0x84C1),
Texture2 = ((int)0x84C2), **Texture3** = ((int)0x84C3), **Texture4** = ((int)0x84C4), **Texture5** = ((int)0x84C5),
Texture6 = ((int)0x84C6), **Texture7** = ((int)0x84C7), **Texture8** = ((int)0x84C8), **Texture9** = ((int)0x84C9),
Texture10 = ((int)0x84CA), **Texture11** = ((int)0x84CB), **Texture12** = ((int)0x84CC), **Texture13** = ((int)0x84CD),
Texture14 = ((int)0x84CE), **Texture15** = ((int)0x84CF), **Texture16** = ((int)0x84D0), **Texture17** = ((int)0x84D1),
Texture18 = ((int)0x84D2), **Texture19** = ((int)0x84D3), **Texture20** = ((int)0x84D4), **Texture21** = ((int)0x84D5),
Texture22 = ((int)0x84D6), **Texture23** = ((int)0x84D7), **Texture24** = ((int)0x84D8), **Texture25** = ((int)0x84D9),
Texture26 = ((int)0x84DA), **Texture27** = ((int)0x84DB), **Texture28** = ((int)0x84DC), **Texture29** = ((int)0x84DD),
Texture30 = ((int)0x84DE), **Texture31** = ((int)0x84DF), **ActiveTexture** = ((int)0x84E0), **ClientActiveTexture** = ((int)0x84E1),
MaxTextureUnits = ((int)0x84E2), **Subtract** = ((int)0x84E7), **MaxRenderbufferSizeOes** = ((int)0x84E8), **AllCompletedNv** = ((int)0x84F2),
FenceStatusNv = ((int)0x84F3), **FenceConditionNv** = ((int)0x84F4), **DepthStencilOes** = ((int)0x84F9), **UnsignedInt248Oes** = ((int)0x84FA),
TextureMaxAnisotropyExt = ((int)0x84FE), **MaxTextureMaxAnisotropyExt** = ((int)0x84FF), **IncrWrapOes** = ((int)0x8507), **DecrWrapOes** = ((int)0x8508),
NormalMapOes = ((int)0x8511), **ReflectionMapOes** = ((int)0x8512), **TextureCubeMapOes** = ((int)0x8513), **TextureBindingCubeMapOes** = ((int)0x8514),
TextureCubeMapPositiveXOes = ((int)0x8515), **TextureCubeMapNegativeXOes** = ((int)0x8516), **TextureCubeMapPositiveYOes** = ((int)0x8517), **TextureCubeMapNegativeYOes** = ((int)0x8518),
TextureCubeMapPositiveZOes = ((int)0x8519), **TextureCubeMapNegativeZOes** = ((int)0x851A), **MaxCubeMapTextureSizeOes** = ((int)0x851C), **Combine** = ((int)0x8570),
CombineRgb = ((int)0x8571), **CombineAlpha** = ((int)0x8572), **RgbScale** = ((int)0x8573), **AddSigned** = ((int)0x8574),

```

Interpolate = ((int)0x8575), Constant = ((int)0x8576), PrimaryColor =
((int)0x8577), Previous = ((int)0x8578),
Src0Rgb = ((int)0x8580), Src1Rgb = ((int)0x8581), Src2Rgb = ((int)0x8582),
Src0Alpha = ((int)0x8588),
Src1Alpha = ((int)0x8589), Src2Alpha = ((int)0x858A), Operand0Rgb =
((int)0x8590), Operand1Rgb = ((int)0x8591),
Operand2Rgb = ((int)0x8592), Operand0Alpha = ((int)0x8598),
Operand1Alpha = ((int)0x8599), Operand2Alpha = ((int)0x859A),
NumCompressedTextureFormats = ((int)0x86A2), CompressedTextureFor-
mats = ((int)0x86A3), MaxVertexUnitsOes = ((int)0x86A4), WeightArray-
TypeOes = ((int)0x86A9),
WeightArrayStrideOes = ((int)0x86AA), WeightArraySizeOes =
((int)0x86AB), WeightArrayPointerOes = ((int)0x86AC), WeightArray-
Oes = ((int)0x86AD),
Dot3Rgb = ((int)0x86AE), Dot3Rgba = ((int)0x86AF), Dot3RgbaImg =
((int)0x86AF), BufferSize = ((int)0x8764),
BufferUsage = ((int)0x8765), AtcRgbaInterpolatedAlphaAmd =
((int)0x87EE), GL_3DcXamd = ((int)0x87F9), GL_3DcXyamd =
((int)0x87FA),
BlendEquationAlphaOes = ((int)0x883D), MatrixPaletteOes = ((int)0x8840),
MaxPaletteMatricesOes = ((int)0x8842), CurrentPaletteMatrixOes =
((int)0x8843),
MatrixIndexArrayOes = ((int)0x8844), MatrixIndexArraySizeOes =
((int)0x8846), MatrixIndexArrayTypeOes = ((int)0x8847), MatrixIndexAr-
rayStrideOes = ((int)0x8848),
MatrixIndexArrayPointerOes = ((int)0x8849), PointSpriteOes =
((int)0x8861), CoordReplaceOes = ((int)0x8862), ArrayBuffer =
((int)0x8892),
ElementArrayBuffer = ((int)0x8893), ArrayBufferBinding = ((int)0x8894),
ElementArrayBufferBinding = ((int)0x8895), VertexArrayBufferBinding =
((int)0x8896),
NormalArrayBufferBinding = ((int)0x8897), ColorArrayBufferBind-
ing = ((int)0x8898), TextureCoordArrayBufferBinding = ((int)0x889A),
WeightArrayBufferBindingOes = ((int)0x889E),
WriteOnlyOes = ((int)0x88B9), BufferAccessOes = ((int)0x88BB),
BufferMappedOes = ((int)0x88BC), BufferMapPointerOes = ((int)0x88BD),
StaticDraw = ((int)0x88E4), DynamicDraw = ((int)0x88E8),
Depth24Stencil8Oes = ((int)0x88F0), PointSizeArrayTypeOes =
((int)0x898A),
PointSizeArrayStrideOes = ((int)0x898B), PointSizeArrayPointerOes =
((int)0x898C), ModelviewMatrixFloatAsIntBitsOes = ((int)0x898D), Projec-
tionMatrixFloatAsIntBitsOes = ((int)0x898E),

```

TextureMatrixFloatAsIntBitsOes = ((int)0x898F), **Palette4Rgb8Oes** = ((int)0x8B90), **Palette4Rgba8Oes** = ((int)0x8B91), **Palette4R5G6B5Oes** = ((int)0x8B92),

Palette4Rgba4Oes = ((int)0x8B93), **Palette4Rgb5A1Oes** = ((int)0x8B94), **Palette8Rgb8Oes** = ((int)0x8B95), **Palette8Rgba8Oes** = ((int)0x8B96),

Palette8R5G6B5Oes = ((int)0x8B97), **Palette8Rgba4Oes** = ((int)0x8B98), **Palette8Rgb5A1Oes** = ((int)0x8B99), **ImplementationColorReadTypeOes** = ((int)0x8B9A),

ImplementationColorReadFormatOes = ((int)0x8B9B), **PointSizeArrayOes** = ((int)0x8B9C), **TextureCropRectOes** = ((int)0x8B9D), **MatrixIndexArrayBufferBindingOes** = ((int)0x8B9E),

PointSizeArrayBufferBindingOes = ((int)0x8B9F), **CompressedRgbPvrtc4Bppv1Img** = ((int)0x8C00), **CompressedRgbaPvrtc4Bppv1Img** = ((int)0x8C01), **CompressedRgbaPvrtc2Bppv1Img** = ((int)0x8C02),

CompressedRgbaPvrtc2Bppv1Img = ((int)0x8C03), **ModulateColorImg** = ((int)0x8C04), **RecipAddSignedAlphaImg** = ((int)0x8C05), **TextureAlphaModulateImg** = ((int)0x8C06),

FactorAlphaModulateImg = ((int)0x8C07), **FragmentAlphaModulateImg** = ((int)0x8C08), **AddBlendImg** = ((int)0x8C09), **AtcRgbAmd** = ((int)0x8C92),

AtcRgbaExplicitAlphaAmd = ((int)0x8C93), **FramebufferBindingOes** = ((int)0x8CA6), **RenderbufferBindingOes** = ((int)0x8CA7), **FramebufferAttachmentObjectTypeOes** = ((int)0x8CD0),

FramebufferAttachmentObjectNameOes = ((int)0x8CD1), **FramebufferAttachmentTextureLevelOes** = ((int)0x8CD2), **FramebufferAttachmentTextureCubeMapFaceOes** = ((int)0x8CD3), **FramebufferCompleteOes** = ((int)0x8CD5),

FramebufferIncompleteAttachmentOes = ((int)0x8CD6), **FramebufferIncompleteMissingAttachmentOes** = ((int)0x8CD7), **FramebufferIncompleteDimensionsOes** = ((int)0x8CD9), **FramebufferIncompleteFormatsOes** = ((int)0x8CDA),

FramebufferUnsupportedOes = ((int)0x8CDD), **ColorAttachment0Oes** = ((int)0x8CE0), **DepthAttachmentOes** = ((int)0x8D00), **StencilAttachmentOes** = ((int)0x8D20),

FramebufferOes = ((int)0x8D40), **RenderbufferOes** = ((int)0x8D41), **RenderbufferWidthOes** = ((int)0x8D42), **RenderbufferHeightOes** = ((int)0x8D43),

RenderbufferInternalFormatOes = ((int)0x8D44), **StencilIndex1Oes** = ((int)0x8D46), **StencilIndex4Oes** = ((int)0x8D47), **StencilIndex8Oes** = ((int)0x8D48),

RenderbufferRedSizeOes = ((int)0x8D50), **RenderbufferGreenSizeOes** = ((int)0x8D51), **RenderbufferBlueSizeOes** = ((int)0x8D52), **RenderbufferAlphaSizeOes** = ((int)0x8D53),

RenderbufferDepthSizeOes = ((int)0x8D54), **RenderbufferStencilSizeOes** = ((int)0x8D55), **TextureGenStrOes** = ((int)0x8D60), **Rgb565Oes** = ((int)0x8D62),

Etc1Rgb8Oes = ((int)0x8D64), **PerfmonGlobalModeQcom** = ((int)0x8FA0), **AmdCompressed3DcTexture** = ((int)1), **AmdCompressedAtcTexture** = ((int)1),

ExtTextureFilterAnisotropic = ((int)1), **ExtTextureFormatBgra8888** = ((int)1), **ImgReadFormat** = ((int)1), **ImgTextureCompressionPvrtc** = ((int)1),

ImgTextureEnvEnhancedFixedFunction = ((int)1), **ImgUserClipPlane** = ((int)1), **NvFence** = ((int)1), **OesBlendEquationSeparate** = ((int)1),

OesBlendFuncSeparate = ((int)1), **OesBlendSubtract** = ((int)1), **OesByteCoordinates** = ((int)1), **OesCompressedEtc1Rgb8Texture** = ((int)1),

OesCompressedPalettedTexture = ((int)1), **OesDepth24** = ((int)1), **OesDepth32** = ((int)1), **OesDrawTexture** = ((int)1),

OesEglImage = ((int)1), **OesElementIndexUint** = ((int)1), **OesExtendedMatrixPalette** = ((int)1), **OesFboRenderMipmap** = ((int)1),

OesFixedPoint = ((int)1), **OesFramebufferObject** = ((int)1), **OesMapbuffer** = ((int)1), **OesMatrixGet** = ((int)1),

OesMatrixPalette = ((int)1), **OesPackedDepthStencil** = ((int)1), **OesPointSizeArray** = ((int)1), **OesPointSprite** = ((int)1),

OesQueryMatrix = ((int)1), **OesReadFormat** = ((int)1), **OesRgb8Rgba8** = ((int)1), **OesSinglePrecision** = ((int)1),

OesStencil1 = ((int)1), **OesStencil4** = ((int)1), **OesStencil8** = ((int)1), **OesStencilWrap** = ((int)1),

OesTextureCubeMap = ((int)1), **OesTextureEnvCrossbar** = ((int)1), **OesTextureMirroredRepeat** = ((int)1), **One** = ((int)1),

QcomDriverControl = ((int)1), **QcomPerfmonGlobalMode** = ((int)1), **True** = ((int)1), **VersionEsC110** = ((int)1),

VersionEsC111 = ((int)1), **VersionEsCm10** = ((int)1), **VersionEsCm11** = ((int)1) }

- enum **AlphaFunction** {

Never = ((int)0x0200), **Less** = ((int)0x0201), **Equal** = ((int)0x0202), **Lequal** = ((int)0x0203),

Greater = ((int)0x0204), **Notequal** = ((int)0x0205), **Gequal** = ((int)0x0206), **Always** = ((int)0x0207) }

- enum **Amdcompressed3Dctexture** { **GL_3DcXamd** = ((int)0x87F9), **GL_3DcXyamd** = ((int)0x87FA), **AmdCompressed3DcTexture** = ((int)1) }

- enum **AmdcompressedAtctexture** { **AtcRgbaInterpolatedAlphaamd** = ((int)0x87EE), **AtcRgbamd** = ((int)0x8C92), **AtcRgbaExplicitAlphaamd** = ((int)0x8C93), **AmdCompressedAtcTexture** = ((int)1) }

- enum **BeginMode** {
Points = ((int)0x0000), **Lines** = ((int)0x0001), **LineLoop** = ((int)0x0002),
LineStrip = ((int)0x0003),
Triangles = ((int)0x0004), **TriangleStrip** = ((int)0x0005), **TriangleFan** =
((int)0x0006) }
- enum **BlendingFactorDest** {
Zero = ((int)0), **SrcColor** = ((int)0x0300), **OneMinusSrcColor** = ((int)0x0301),
SrcAlpha = ((int)0x0302),
OneMinusSrcAlpha = ((int)0x0303), **DstAlpha** = ((int)0x0304), **OneMinusD-**
stAlpha = ((int)0x0305), **One** = ((int)1) }
- enum **BlendingFactorSrc** { **DstColor** = ((int)0x0306), **OneMinusDstColor** =
((int)0x0307), **SrcAlphaSaturate** = ((int)0x0308) }
- enum **Boolean** { **False** = ((int)0), **True** = ((int)1) }
- enum **BufferObjects** {
BufferSize = ((int)0x8764), **BufferUsage** = ((int)0x8765), **ArrayBuffer** =
((int)0x8892), **ElementArrayBuffer** = ((int)0x8893),
ArrayBufferBinding = ((int)0x8894), **ElementArrayBufferBinding** =
((int)0x8895), **VertexArrayBufferBinding** = ((int)0x8896), **NormalArray-**
BufferBinding = ((int)0x8897),
ColorArrayBufferBinding = ((int)0x8898), **TextureCoordArrayBufferBind-**
ing = ((int)0x889A), **StaticDraw** = ((int)0x88E4), **DynamicDraw** =
((int)0x88E8) }
- enum **ClearBufferMask** { **DepthBufferBit** = ((int)0x00000100), **Stencil-**
BufferBit = ((int)0x00000400), **ColorBufferBit** = ((int)0x00004000) }
- enum **ClipPlaneName** {
ClipPlane0 = ((int)0x3000), **ClipPlane1** = ((int)0x3001), **ClipPlane2** =
((int)0x3002), **ClipPlane3** = ((int)0x3003),
ClipPlane4 = ((int)0x3004), **ClipPlane5** = ((int)0x3005) }
- enum **CullFaceMode** { **Front** = ((int)0x0404), **Back** = ((int)0x0405),
FrontAndBack = ((int)0x0408) }
- enum **DataType** {
Byte = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402),
UnsignedShort = ((int)0x1403),
Float = ((int)0x1406), **Fixed** = ((int)0x140C) }
- enum **EnableCap** {
PointSmooth = ((int)0x0B10), **LineSmooth** = ((int)0x0B20), **CullFace** =
((int)0x0B44), **Lighting** = ((int)0x0B50),
ColorMaterial = ((int)0x0B57), **Fog** = ((int)0x0B60), **DepthTest** =
((int)0x0B71), **StencilTest** = ((int)0x0B90),
Normalize = ((int)0x0BA1), **AlphaTest** = ((int)0x0BC0), **Dither** =
((int)0x0BD0), **Blend** = ((int)0x0BE2),

```

ColorLogicOp = ((int)0x0BF2), ScissorTest = ((int)0x0C11), Texture2D =
((int)0x0DE1), PolygonOffsetFill = ((int)0x8037),
RescaleNormal = ((int)0x803A), VertexArray = ((int)0x8074), NormalArray
= ((int)0x8075), ColorArray = ((int)0x8076),
TextureCoordArray = ((int)0x8078), Multisample = ((int)0x809D), Sam-
pleAlphaToCoverage = ((int)0x809E), SampleAlphaToOne = ((int)0x809F),
SampleCoverage = ((int)0x80A0) }
• enum ErrorCode {
NoError = ((int)0), InvalidEnum = ((int)0x0500), InvalidValue =
((int)0x0501), InvalidOperation = ((int)0x0502),
StackOverflow = ((int)0x0503), StackUnderflow = ((int)0x0504), Out-
OfMemory = ((int)0x0505) }
• enum ExttextureFilterAnisotropic { TextureMaxAnisotropyExt =
((int)0x84FE), MaxTextureMaxAnisotropyExt = ((int)0x84FF), ExtTex-
tureFilterAnisotropic = ((int)1) }
• enum ExttextureFormatBgra8888 { Bgra = ((int)0x80E1), ExtTextureFor-
matBgra8888 = ((int)1) }
• enum FogMode { Exp = ((int)0x0800), Exp2 = ((int)0x0801) }
• enum FogParameter {
FogDensity = ((int)0x0B62), FogStart = ((int)0x0B63), FogEnd =
((int)0x0B64), FogMode = ((int)0x0B65),
FogColor = ((int)0x0B66) }
• enum FrontFaceDirection { Cw = ((int)0x0900), Ccw = ((int)0x0901) }
• enum GetPName {
CurrentColor = ((int)0x0B00), CurrentNormal = ((int)0x0B02), CurrentTex-
tureCoords = ((int)0x0B03), PointSize = ((int)0x0B11),
SmoothPointSizeRange = ((int)0x0B12), LineWidth = ((int)0x0B21),
SmoothLineWidthRange = ((int)0x0B22), CullFaceMode = ((int)0x0B45),
FrontFace = ((int)0x0B46), ShadeModel = ((int)0x0B54), DepthRange =
((int)0x0B70), DepthWritemask = ((int)0x0B72),
DepthClearValue = ((int)0x0B73), DepthFunc = ((int)0x0B74), StencilClear-
Value = ((int)0x0B91), StencilFunc = ((int)0x0B92),
StencilValueMask = ((int)0x0B93), StencilFail = ((int)0x0B94), StencilPass-
DepthFail = ((int)0x0B95), StencilPassDepthPass = ((int)0x0B96),
StencilRef = ((int)0x0B97), StencilWritemask = ((int)0x0B98), MatrixMode
= ((int)0x0BA0), Viewport = ((int)0x0BA2),
ModelviewStackDepth = ((int)0x0BA3), ProjectionStackDepth =
((int)0x0BA4), TextureStackDepth = ((int)0x0BA5), ModelviewMatrix
= ((int)0x0BA6),
ProjectionMatrix = ((int)0x0BA7), TextureMatrix = ((int)0x0BA8), AlphaT-
estFunc = ((int)0x0BC1), AlphaTestRef = ((int)0x0BC2),

```

```

BlendDst = ((int)0x0BE0), BlendSrc = ((int)0x0BE1), LogicOpMode =
((int)0x0BF0), ScissorBox = ((int)0x0C10),

ScissorTest = ((int)0x0C11), ColorClearValue = ((int)0x0C22), Color-
Writemask = ((int)0x0C23), UnpackAlignment = ((int)0x0CF5),

PackAlignment = ((int)0x0D05), MaxLights = ((int)0x0D31), MaxClipPlanes
= ((int)0x0D32), MaxTextureSize = ((int)0x0D33),

MaxModelviewStackDepth = ((int)0x0D36), MaxProjectionStackDepth =
((int)0x0D38), MaxTextureStackDepth = ((int)0x0D39), MaxViewportDims
= ((int)0x0D3A),

SubpixelBits = ((int)0x0D50), RedBits = ((int)0x0D52), GreenBits =
((int)0x0D53), BlueBits = ((int)0x0D54),

AlphaBits = ((int)0x0D55), DepthBits = ((int)0x0D56), StencilBits =
((int)0x0D57), PolygonOffsetUnits = ((int)0x2A00),

PolygonOffsetFill = ((int)0x8037), PolygonOffsetFactor = ((int)0x8038), Tex-
tureBinding2D = ((int)0x8069), VertexArraySize = ((int)0x807A),

VertexArrayType = ((int)0x807B), VertexArrayStride = ((int)0x807C), Nor-
malArrayType = ((int)0x807E), NormalArrayStride = ((int)0x807F),

ColorArraySize = ((int)0x8081), ColorArrayType = ((int)0x8082), ColorAr-
rayStride = ((int)0x8083), TextureCoordArraySize = ((int)0x8088),

TextureCoordArrayType = ((int)0x8089), TextureCoordArrayStride =
((int)0x808A), VertexArrayPointer = ((int)0x808E), NormalArrayPointer =
((int)0x808F),

ColorArrayPointer = ((int)0x8090), TextureCoordArrayPointer =
((int)0x8092), SampleBuffers = ((int)0x80A8), Samples = ((int)0x80A9),

SampleCoverageValue = ((int)0x80AA), SampleCoverageInvert =
((int)0x80AB), PointSizeMin = ((int)0x8126), PointSizeMax = ((int)0x8127),

PointFadeThresholdSize = ((int)0x8128), PointDistanceAttenuation
= ((int)0x8129), AliasedPointSizeRange = ((int)0x846D), Aliased-
LineWidthRange = ((int)0x846E),

MaxTextureUnits = ((int)0x84E2) }
• enum GetTextureParameter { NumCompressedTextureFormats =
((int)0x86A2), CompressedTextureFormats = ((int)0x86A3) }
• enum HintMode { DontCare = ((int)0x1100), Fastest = ((int)0x1101), Nicest
= ((int)0x1102) }
• enum HintTarget {

PerspectiveCorrectionHint = ((int)0x0C50), PointSmoothHint =
((int)0x0C51), LineSmoothHint = ((int)0x0C52), FogHint = ((int)0x0C54),

GenerateMipmapHint = ((int)0x8192) }
• enum ImgreadFormat { Bgra = ((int)0x80E1), UnsignedShort4444Rev =
((int)0x8365), UnsignedShort1555Rev = ((int)0x8366), ImgReadFormat =
((int)1) }

```


- enum **ImgtextureCompressionPvrtc** {
CompressedRgbPvrtc4Bppv1Img = ((int)0x8C00), **CompressedRgbPvrtc2Bppv1Img** = ((int)0x8C01), **CompressedRgbaPvrtc4Bppv1Img** = ((int)0x8C02), **CompressedRgbaPvrtc2Bppv1Img** = ((int)0x8C03),
ImgTextureCompressionPvrtc = ((int)1) }
- enum **ImgtextureEnvEnhancedFixedFunction** {
Dot3RgbaImg = ((int)0x86AF), **ModulateColorImg** = ((int)0x8C04), **RecipeAddSignedAlphaImg** = ((int)0x8C05), **TextureAlphaModulateImg** = ((int)0x8C06),
FactorAlphaModulateImg = ((int)0x8C07), **FragmentAlphaModulateImg** = ((int)0x8C08), **AddBlendImg** = ((int)0x8C09), **ImgTextureEnvEnhancedFixedFunction** = ((int)1) }
- enum **ImguserClipPlane** {
MaxClipPlanesImg = ((int)0x0D32), **ClipPlane0Img** = ((int)0x3000), **ClipPlane1Img** = ((int)0x3001), **ClipPlane2Img** = ((int)0x3002),
ClipPlane3Img = ((int)0x3003), **ClipPlane4Img** = ((int)0x3004), **ClipPlane5Img** = ((int)0x3005), **ImgUserClipPlane** = ((int)1) }
- enum **LightModelParameter** { **LightModelTwoSide** = ((int)0x0B52), **LightModelAmbient** = ((int)0x0B53) }
- enum **LightName** {
Light0 = ((int)0x4000), **Light1** = ((int)0x4001), **Light2** = ((int)0x4002), **Light3** = ((int)0x4003),
Light4 = ((int)0x4004), **Light5** = ((int)0x4005), **Light6** = ((int)0x4006), **Light7** = ((int)0x4007) }
- enum **LightParameter** {
Ambient = ((int)0x1200), **Diffuse** = ((int)0x1201), **Specular** = ((int)0x1202), **Position** = ((int)0x1203),
SpotDirection = ((int)0x1204), **SpotExponent** = ((int)0x1205), **SpotCutoff** = ((int)0x1206), **ConstantAttenuation** = ((int)0x1207),
LinearAttenuation = ((int)0x1208), **QuadraticAttenuation** = ((int)0x1209) }
- enum **LogicOp** {
Clear = ((int)0x1500), **And** = ((int)0x1501), **AndReverse** = ((int)0x1502), **Copy** = ((int)0x1503),
AndInverted = ((int)0x1504), **Noop** = ((int)0x1505), **Xor** = ((int)0x1506), **Or** = ((int)0x1507),
Nor = ((int)0x1508), **Equiv** = ((int)0x1509), **Invert** = ((int)0x150A), **OrReverse** = ((int)0x150B),
CopyInverted = ((int)0x150C), **OrInverted** = ((int)0x150D), **Nand** = ((int)0x150E), **Set** = ((int)0x150F) }
- enum **MaterialParameter** { **Emission** = ((int)0x1600), **Shininess** = ((int)0x1601), **AmbientAndDiffuse** = ((int)0x1602) }

- enum **MatrixMode** { **Modelview** = ((int)0x1700), **Projection** = ((int)0x1701), **Texture** = ((int)0x1702) }
- enum **Nvfence** { **AllCompletedNv** = ((int)0x84F2), **FenceStatusNv** = ((int)0x84F3), **FenceConditionNv** = ((int)0x84F4), **NvFence** = ((int)1) }
- enum **OesblendEquationSeparate** { **BlendEquationRgbOes** = ((int)0x8009), **BlendEquationAlphaOes** = ((int)0x883D), **OesBlendEquationSeparate** = ((int)1) }
- enum **OesblendFuncSeparate** {
BlendDstRgbOes = ((int)0x80C8), **BlendSrcRgbOes** = ((int)0x80C9),
BlendDstAlphaOes = ((int)0x80CA), **BlendSrcAlphaOes** = ((int)0x80CB),
OesBlendFuncSeparate = ((int)1) }
- enum **OesblendSubtract** {
FuncAddOes = ((int)0x8006), **BlendEquationOes** = ((int)0x8009), **FuncSubtractOes** = ((int)0x800A), **FuncReverseSubtractOes** = ((int)0x800B),
OesBlendSubtract = ((int)1) }
- enum **OesbyteCoordinates** { **OesByteCoordinates** = ((int)1) }
- enum **OescompressedEtc1Rgb8Texture** { **Etc1Rgb8Oes** = ((int)0x8D64), **OesCompressedEtc1Rgb8Texture** = ((int)1) }
- enum **OescompressedPalettedTexture** {
Palette4Rgb8Oes = ((int)0x8B90), **Palette4Rgba8Oes** = ((int)0x8B91),
Palette4R5G6B5Oes = ((int)0x8B92), **Palette4Rgba4Oes** = ((int)0x8B93),
Palette4Rgb5A1Oes = ((int)0x8B94), **Palette8Rgb8Oes** = ((int)0x8B95),
Palette8Rgba8Oes = ((int)0x8B96), **Palette8R5G6B5Oes** = ((int)0x8B97),
Palette8Rgba4Oes = ((int)0x8B98), **Palette8Rgb5A1Oes** = ((int)0x8B99),
OesCompressedPalettedTexture = ((int)1) }
- enum **Oesdepth24** { **DepthComponent24Oes** = ((int)0x81A6), **OesDepth24** = ((int)1) }
- enum **Oesdepth32** { **DepthComponent32Oes** = ((int)0x81A7), **OesDepth32** = ((int)1) }
- enum **OesdrawTexture** { **TextureCropRectOes** = ((int)0x8B9D), **OesDrawTexture** = ((int)1) }
- enum **Oeseglimage** { **OesEglImage** = ((int)1) }
- enum **OeselementIndexUint** { **OesElementIndexUint** = ((int)1) }
- enum **OesextendedMatrixPalette** { **OesExtendedMatrixPalette** = ((int)1) }
- enum **OesfboRenderMipmap** { **OesFboRenderMipmap** = ((int)1) }
- enum **OesfixedPoint** { **FixedOes** = ((int)0x140C), **OesFixedPoint** = ((int)1) }
- enum **OesframebufferObject** {
NoneOes = ((int)0), **InvalidFramebufferOperationOes** = ((int)0x0506),
Rgba4Oes = ((int)0x8056), **Rgb5A1Oes** = ((int)0x8057),
DepthComponent16Oes = ((int)0x81A5), **MaxRenderbufferSizeOes** = ((int)0x84E8),
FramebufferBindingOes = ((int)0x8CA6), **RenderbufferBindingOes** = ((int)0x8CA7),

```

FramebufferAttachmentObjectTypeOes = ((int)0x8CD0), FramebufferAttachmentObjectNameOes = ((int)0x8CD1), FramebufferAttachmentTextureLevelOes = ((int)0x8CD2), FramebufferAttachmentTextureCubeMapFaceOes = ((int)0x8CD3),

FramebufferCompleteOes = ((int)0x8CD5), FramebufferIncompleteAttachmentOes = ((int)0x8CD6), FramebufferIncompleteMissingAttachmentOes = ((int)0x8CD7), FramebufferIncompleteDimensionsOes = ((int)0x8CD9),

FramebufferIncompleteFormatsOes = ((int)0x8CDA), FramebufferUnsupportedOes = ((int)0x8CDD), ColorAttachment0Oes = ((int)0x8CE0), DepthAttachmentOes = ((int)0x8D00),

StencilAttachmentOes = ((int)0x8D20), FramebufferOes = ((int)0x8D40), RenderbufferOes = ((int)0x8D41), RenderbufferWidthOes = ((int)0x8D42),

RenderbufferHeightOes = ((int)0x8D43), RenderbufferInternalFormatOes = ((int)0x8D44), RenderbufferRedSizeOes = ((int)0x8D50), RenderbufferGreenSizeOes = ((int)0x8D51),

RenderbufferBlueSizeOes = ((int)0x8D52), RenderbufferAlphaSizeOes = ((int)0x8D53), RenderbufferDepthSizeOes = ((int)0x8D54), RenderbufferStencilSizeOes = ((int)0x8D55),

Rgb565Oes = ((int)0x8D62), OesFramebufferObject = ((int)1) }

• enum Oesmapbuffer {
    WriteOnlyOes = ((int)0x88B9), BufferAccessOes = ((int)0x88BB),
    BufferMappedOes = ((int)0x88BC), BufferMapPointerOes = ((int)0x88BD),
    OesMapbuffer = ((int)1) }

• enum OesmatrixGet { ModelviewMatrixFloatAsIntBitsOes = ((int)0x898D),
    ProjectionMatrixFloatAsIntBitsOes = ((int)0x898E), TextureMatrixFloatAsIntBitsOes = ((int)0x898F), OesMatrixGet = ((int)1) }

• enum OesmatrixPalette {
    MaxVertexUnitsOes = ((int)0x86A4), WeightArrayTypeOes = ((int)0x86A9),
    WeightArrayStrideOes = ((int)0x86AA), WeightArraySizeOes = ((int)0x86AB),

    WeightArrayPointerOes = ((int)0x86AC), WeightArrayOes = ((int)0x86AD),
    MatrixPaletteOes = ((int)0x8840), MaxPaletteMatricesOes = ((int)0x8842),

    CurrentPaletteMatrixOes = ((int)0x8843), MatrixIndexArrayOes = ((int)0x8844), MatrixIndexArraySizeOes = ((int)0x8846), MatrixIndexArrayTypeOes = ((int)0x8847),

    MatrixIndexArrayStrideOes = ((int)0x8848), MatrixIndexArrayPointerOes = ((int)0x8849), WeightArrayBufferBindingOes = ((int)0x889E), MatrixIndexArrayBufferBindingOes = ((int)0x8B9E),

    OesMatrixPalette = ((int)1) }

• enum OespackedDepthStencil { DepthStencilOes = ((int)0x84F9), UnsignedInt248Oes = ((int)0x84FA), Depth24Stencil8Oes = ((int)0x88F0), OesPackedDepthStencil = ((int)1) }

```

- enum **OespointSizeArray** {
PointSizeArrayTypeOes = ((int)0x898A), **PointSizeArrayStrideOes** = ((int)0x898B), **PointSizeArrayPointerOes** = ((int)0x898C), **PointSizeArrayOes** = ((int)0x8B9C),
PointSizeArrayBufferBindingOes = ((int)0x8B9F), **OesPointSizeArray** = ((int)1) }
- enum **OespointSprite** { **PointSpriteOes** = ((int)0x8861), **CoordReplaceOes** = ((int)0x8862), **OesPointSprite** = ((int)1) }
- enum **OesqueryMatrix** { **OesQueryMatrix** = ((int)1) }
- enum **OesreadFormat** { **ImplementationColorReadTypeOes** = ((int)0x8B9A), **ImplementationColorReadFormatOes** = ((int)0x8B9B), **OesReadFormat** = ((int)1) }
- enum **Oesrgb8Rgba8** { **Rgb8Oes** = ((int)0x8051), **Rgba8Oes** = ((int)0x8058), **OesRgba8Rgba8** = ((int)1) }
- enum **OessinglePrecision** { **OesSinglePrecision** = ((int)1) }
- enum **Oesstencil1** { **StencilIndex1Oes** = ((int)0x8D46), **OesStencil1** = ((int)1) }
- enum **Oesstencil4** { **StencilIndex4Oes** = ((int)0x8D47), **OesStencil4** = ((int)1) }
- enum **Oesstencil8** { **StencilIndex8Oes** = ((int)0x8D48), **OesStencil8** = ((int)1) }
- enum **OesstencilWrap** { **IncrWrapOes** = ((int)0x8507), **DecrWrapOes** = ((int)0x8508), **OesStencilWrap** = ((int)1) }
- enum **OestextureCubeMap** {
TextureGenModeOes = ((int)0x2500), **NormalMapOes** = ((int)0x8511), **ReflectionMapOes** = ((int)0x8512), **TextureCubeMapOes** = ((int)0x8513),
TextureBindingCubeMapOes = ((int)0x8514), **TextureCubeMapPositiveXOes** = ((int)0x8515), **TextureCubeMapNegativeXOes** = ((int)0x8516), **TextureCubeMapPositiveYOes** = ((int)0x8517),
TextureCubeMapNegativeYOes = ((int)0x8518), **TextureCubeMapPositiveZOes** = ((int)0x8519), **TextureCubeMapNegativeZOes** = ((int)0x851A), **MaxCubeMapTextureSizeOes** = ((int)0x851C),
TextureGenStrOes = ((int)0x8D60), **OesTextureCubeMap** = ((int)1) }
- enum **OestextureEnvCrossbar** { **OesTextureEnvCrossbar** = ((int)1) }
- enum **OestextureMirroredRepeat** { **MirroredRepeatOes** = ((int)0x8370), **OesTextureMirroredRepeat** = ((int)1) }
- enum **OpenGlescoreVersions** { **VersionEsCl10** = ((int)1), **VersionEsCl11** = ((int)1), **VersionEsCm10** = ((int)1), **VersionEsCm11** = ((int)1) }
- enum **PixelFormat** {
Alpha = ((int)0x1906), **Rgb** = ((int)0x1907), **Rgba** = ((int)0x1908), **Luminance** = ((int)0x1909),
LuminanceAlpha = ((int)0x190A) }

- enum **PixelStoreParameter** { **UnpackAlignment** = ((int)0x0CF5), **PackAlignment** = ((int)0x0D05) }
- enum **PixelType** { **UnsignedShort4444** = ((int)0x8033), **UnsignedShort5551** = ((int)0x8034), **UnsignedShort565** = ((int)0x8363) }
- enum **QcomdriverControl** { **QcomDriverControl** = ((int)1) }
- enum **QcomperfmonGlobalMode** { **PerfmonGlobalModeQcom** = ((int)0x8FA0), **QcomPerfmonGlobalMode** = ((int)1) }
- enum **ShadingModel** { **Flat** = ((int)0x1D00), **Smooth** = ((int)0x1D01) }
- enum **StencilOp** { **Keep** = ((int)0x1E00), **Replace** = ((int)0x1E01), **Incr** = ((int)0x1E02), **Decr** = ((int)0x1E03) }
- enum **StringName** { **Vendor** = ((int)0x1F00), **Renderer** = ((int)0x1F01), **Version** = ((int)0x1F02), **Extensions** = ((int)0x1F03) }
- enum **TextureCombineDot3** {

AlphaScale = ((int)0x0D1C), **Subtract** = ((int)0x84E7), **Combine** = ((int)0x8570), **CombineRgb** = ((int)0x8571),

CombineAlpha = ((int)0x8572), **RgbScale** = ((int)0x8573), **AddSigned** = ((int)0x8574), **Interpolate** = ((int)0x8575),

Constant = ((int)0x8576), **PrimaryColor** = ((int)0x8577), **Previous** = ((int)0x8578), **Src0Rgb** = ((int)0x8580),

Src1Rgb = ((int)0x8581), **Src2Rgb** = ((int)0x8582), **Src0Alpha** = ((int)0x8588), **Src1Alpha** = ((int)0x8589),

Src2Alpha = ((int)0x858A), **Operand0Rgb** = ((int)0x8590), **Operand1Rgb** = ((int)0x8591), **Operand2Rgb** = ((int)0x8592),

Operand0Alpha = ((int)0x8598), **Operand1Alpha** = ((int)0x8599), **Operand2Alpha** = ((int)0x859A), **Dot3Rgb** = ((int)0x86AE),

Dot3Rgba = ((int)0x86AF) }
- enum **TextureEnvMode** { **Add** = ((int)0x0104), **Modulate** = ((int)0x2100), **Decal** = ((int)0x2101) }
- enum **TextureEnvParameter** { **TextureEnvMode** = ((int)0x2200), **TextureEnvColor** = ((int)0x2201) }
- enum **TextureEnvTarget** { **TextureEnv** = ((int)0x2300) }
- enum **TextureMagFilter** { **Nearest** = ((int)0x2600), **Linear** = ((int)0x2601) }
- enum **TextureMinFilter** { **NearestMipmapNearest** = ((int)0x2700), **LinearMipmapNearest** = ((int)0x2701), **NearestMipmapLinear** = ((int)0x2702), **LinearMipmapLinear** = ((int)0x2703) }
- enum **TextureParameterName** {

TextureMagFilter = ((int)0x2800), **TextureMinFilter** = ((int)0x2801), **TextureWrapS** = ((int)0x2802), **TextureWrapT** = ((int)0x2803),

GenerateMipmap = ((int)0x8191) }

- enum **TextureUnit** {
 - Texture0** = ((int)0x84C0), **Texture1** = ((int)0x84C1), **Texture2** = ((int)0x84C2), **Texture3** = ((int)0x84C3),
 - Texture4** = ((int)0x84C4), **Texture5** = ((int)0x84C5), **Texture6** = ((int)0x84C6), **Texture7** = ((int)0x84C7),
 - Texture8** = ((int)0x84C8), **Texture9** = ((int)0x84C9), **Texture10** = ((int)0x84CA), **Texture11** = ((int)0x84CB),
 - Texture12** = ((int)0x84CC), **Texture13** = ((int)0x84CD), **Texture14** = ((int)0x84CE), **Texture15** = ((int)0x84CF),
 - Texture16** = ((int)0x84D0), **Texture17** = ((int)0x84D1), **Texture18** = ((int)0x84D2), **Texture19** = ((int)0x84D3),
 - Texture20** = ((int)0x84D4), **Texture21** = ((int)0x84D5), **Texture22** = ((int)0x84D6), **Texture23** = ((int)0x84D7),
 - Texture24** = ((int)0x84D8), **Texture25** = ((int)0x84D9), **Texture26** = ((int)0x84DA), **Texture27** = ((int)0x84DB),
 - Texture28** = ((int)0x84DC), **Texture29** = ((int)0x84DD), **Texture30** = ((int)0x84DE), **Texture31** = ((int)0x84DF),
 - ActiveTexture** = ((int)0x84E0), **ClientActiveTexture** = ((int)0x84E1) }
- enum **TextureWrapMode** { **Repeat** = ((int)0x2901), **ClampToEdge** = ((int)0x812F) }

4.7 Package OpenTK.Graphics.ES20

Classes

- class [GL](#)
 - Provides access to [OpenGL ES 2.0](#) methods.*

Enumerations

- enum **ActiveAttribType** {
 - Float** = ((int)0X1406), **FloatVec2** = ((int)0X8b50), **FloatVec3** = ((int)0X8b51), **FloatVec4** = ((int)0X8b52),
 - FloatMat2** = ((int)0X8b5a), **FloatMat3** = ((int)0X8b5b), **FloatMat4** = ((int)0X8b5c) }
- enum **ActiveUniformType** {
 - Int** = ((int)0X1404), **Float** = ((int)0X1406), **FloatVec2** = ((int)0X8b50), **FloatVec3** = ((int)0X8b51),

```

FloatVec4 = ((int)0X8b52), IntVec2 = ((int)0X8b53), IntVec3 = ((int)0X8b54),
IntVec4 = ((int)0X8b55),

Bool = ((int)0X8b56), BoolVec2 = ((int)0X8b57), BoolVec3 = ((int)0X8b58),
BoolVec4 = ((int)0X8b59),

FloatMat2 = ((int)0X8b5a), FloatMat3 = ((int)0X8b5b), FloatMat4 =
((int)0X8b5c), Sampler2D = ((int)0X8b5e),

SamplerCube = ((int)0X8b60) }
• enum All {
False = ((int)0), NoError = ((int)0), None = ((int)0), Zero = ((int)0),

Points = ((int)0x0000), DepthBufferBit = ((int)0x00000100), StencilBufferBit
= ((int)0x00000400), ColorBufferBit = ((int)0x00004000),

Lines = ((int)0x0001), LineLoop = ((int)0x0002), LineStrip = ((int)0x0003),
Triangles = ((int)0x0004),

TriangleStrip = ((int)0x0005), TriangleFan = ((int)0x0006), Never =
((int)0x0200), Less = ((int)0x0201),

Equal = ((int)0x0202), Lequal = ((int)0x0203), Greater = ((int)0x0204), Note-
qual = ((int)0x0205),

Gequal = ((int)0x0206), Always = ((int)0x0207), SrcColor = ((int)0x0300),
OneMinusSrcColor = ((int)0x0301),

SrcAlpha = ((int)0x0302), OneMinusSrcAlpha = ((int)0x0303), DstAlpha =
((int)0x0304), OneMinusDstAlpha = ((int)0x0305),

DstColor = ((int)0x0306), OneMinusDstColor = ((int)0x0307), SrcAlphaSat-
urate = ((int)0x0308), Front = ((int)0x0404),

Back = ((int)0x0405), FrontAndBack = ((int)0x0408), InvalidEnum =
((int)0x0500), InvalidValue = ((int)0x0501),

InvalidOperation = ((int)0x0502), OutOfMemory = ((int)0x0505), Invalid-
FramebufferOperation = ((int)0x0506), Cw = ((int)0x0900),

Ccw = ((int)0x0901), LineWidth = ((int)0x0B21), CullFace = ((int)0x0B44),
CullFaceMode = ((int)0x0B45),

FrontFace = ((int)0x0B46), DepthRange = ((int)0x0B70), DepthTest =
((int)0x0B71), DepthWritemask = ((int)0x0B72),

DepthClearValue = ((int)0x0B73), DepthFunc = ((int)0x0B74), StencilTest =
((int)0x0B90), StencilClearValue = ((int)0x0B91),

StencilFunc = ((int)0x0B92), StencilValueMask = ((int)0x0B93), StencilFail
= ((int)0x0B94), StencilPassDepthFail = ((int)0x0B95),

StencilPassDepthPass = ((int)0x0B96), StencilRef = ((int)0x0B97), Stencil-
Writemask = ((int)0x0B98), Viewport = ((int)0x0BA2),

Dither = ((int)0x0BD0), Blend = ((int)0x0BE2), ScissorBox = ((int)0x0C10),
ScissorTest = ((int)0x0C11),

```

ColorClearValue = ((int)0x0C22), **ColorWritemask** = ((int)0x0C23), **UnpackAlignment** = ((int)0x0CF5), **PackAlignment** = ((int)0x0D05),
MaxTextureSize = ((int)0x0D33), **MaxViewportDims** = ((int)0x0D3A), **SubpixelBits** = ((int)0x0D50), **RedBits** = ((int)0x0D52),
GreenBits = ((int)0x0D53), **BlueBits** = ((int)0x0D54), **AlphaBits** = ((int)0x0D55), **DepthBits** = ((int)0x0D56),
StencilBits = ((int)0x0D57), **Texture2D** = ((int)0x0DE1), **DontCare** = ((int)0x1100), **Fastest** = ((int)0x1101),
Nicest = ((int)0x1102), **Byte** = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402),
UnsignedShort = ((int)0x1403), **Int** = ((int)0x1404), **UnsignedInt** = ((int)0x1405), **Float** = ((int)0x1406),
Fixed = ((int)0x140C), **Invert** = ((int)0x150A), **Texture** = ((int)0x1702), **StencilIndex** = ((int)0x1901),
DepthComponent = ((int)0x1902), **Alpha** = ((int)0x1906), **Rgb** = ((int)0x1907), **Rgba** = ((int)0x1908),
Luminance = ((int)0x1909), **LuminanceAlpha** = ((int)0x190A), **Keep** = ((int)0x1E00), **Replace** = ((int)0x1E01),
Incr = ((int)0x1E02), **Decr** = ((int)0x1E03), **Vendor** = ((int)0x1F00), **Renderer** = ((int)0x1F01),
Version = ((int)0x1F02), **Extensions** = ((int)0x1F03), **Nearest** = ((int)0x2600), **Linear** = ((int)0x2601),
NearestMipmapNearest = ((int)0x2700), **LinearMipmapNearest** = ((int)0x2701), **NearestMipmapLinear** = ((int)0x2702), **LinearMipmapLinear** = ((int)0x2703),
TextureMagFilter = ((int)0x2800), **TextureMinFilter** = ((int)0x2801), **TextureWrapS** = ((int)0x2802), **TextureWrapT** = ((int)0x2803),
Repeat = ((int)0x2901), **PolygonOffsetUnits** = ((int)0x2A00), **ConstantColor** = ((int)0x8001), **OneMinusConstantColor** = ((int)0x8002),
ConstantAlpha = ((int)0x8003), **OneMinusConstantAlpha** = ((int)0x8004), **BlendColor** = ((int)0x8005), **FuncAdd** = ((int)0x8006),
BlendEquation = ((int)0x8009), **BlendEquationRgb** = ((int)0x8009), **FuncSubtract** = ((int)0x800A), **FuncReverseSubtract** = ((int)0x800B),
UnsignedShort4444 = ((int)0x8033), **UnsignedShort5551** = ((int)0x8034), **PolygonOffsetFill** = ((int)0x8037), **PolygonOffsetFactor** = ((int)0x8038),
Rgb8Oes = ((int)0x8051), **Rgba4** = ((int)0x8056), **Rgb5A1** = ((int)0x8057), **Rgba8Oes** = ((int)0x8058),
TextureBinding2D = ((int)0x8069), **TextureBinding3DOes** = ((int)0x806A), **Texture3DOes** = ((int)0x806F), **TextureWrapROes** = ((int)0x8072),


```

Max3DTextureSizeOes = ((int)0x8073), SampleAlphaToCoverage =
((int)0x809E), SampleCoverage = ((int)0x80A0), SampleBuffers =
((int)0x80A8),

Samples = ((int)0x80A9), SampleCoverageValue = ((int)0x80AA), Sample-
CoverageInvert = ((int)0x80AB), BlendDstRgb = ((int)0x80C8),

BlendSrcRgb = ((int)0x80C9), BlendDstAlpha = ((int)0x80CA), BlendSrcAl-
pha = ((int)0x80CB), Bgra = ((int)0x80E1),

ClampToEdge = ((int)0x812F), GenerateMipmapHint = ((int)0x8192),
DepthComponent16 = ((int)0x81A5), DepthComponent24Oes =
((int)0x81A6),

DepthComponent32Oes = ((int)0x81A7), UnsignedShort565 = ((int)0x8363),
UnsignedShort4444Rev = ((int)0x8365), UnsignedShort1555Rev =
((int)0x8366),

UnsignedInt2101010RevExt = ((int)0x8368), MirroredRepeat =
((int)0x8370), AliasedPointSizeRange = ((int)0x846D), Aliased-
LineWidthRange = ((int)0x846E),

Texture0 = ((int)0x84C0), Texture1 = ((int)0x84C1), Texture2 = ((int)0x84C2),
Texture3 = ((int)0x84C3),

Texture4 = ((int)0x84C4), Texture5 = ((int)0x84C5), Texture6 = ((int)0x84C6),
Texture7 = ((int)0x84C7),

Texture8 = ((int)0x84C8), Texture9 = ((int)0x84C9), Texture10 =
((int)0x84CA), Texture11 = ((int)0x84CB),

Texture12 = ((int)0x84CC), Texture13 = ((int)0x84CD), Texture14 =
((int)0x84CE), Texture15 = ((int)0x84CF),

Texture16 = ((int)0x84D0), Texture17 = ((int)0x84D1), Texture18 =
((int)0x84D2), Texture19 = ((int)0x84D3),

Texture20 = ((int)0x84D4), Texture21 = ((int)0x84D5), Texture22 =
((int)0x84D6), Texture23 = ((int)0x84D7),

Texture24 = ((int)0x84D8), Texture25 = ((int)0x84D9), Texture26 =
((int)0x84DA), Texture27 = ((int)0x84DB),

Texture28 = ((int)0x84DC), Texture29 = ((int)0x84DD), Texture30 =
((int)0x84DE), Texture31 = ((int)0x84DF),

ActiveTexture = ((int)0x84E0), MaxRenderbufferSize = ((int)0x84E8), All-
CompletedNv = ((int)0x84F2), FenceStatusNv = ((int)0x84F3),

FenceConditionNv = ((int)0x84F4), DepthStencilOes = ((int)0x84F9),
UnsignedInt248Oes = ((int)0x84FA), TextureMaxAnisotropyExt =
((int)0x84FE),

MaxTextureMaxAnisotropyExt = ((int)0x84FF), IncrWrap = ((int)0x8507),
DecrWrap = ((int)0x8508), TextureCubeMap = ((int)0x8513),

```

TextureBindingCubeMap = ((int)0x8514), **TextureCubeMapPositiveX** = ((int)0x8515), **TextureCubeMapNegativeX** = ((int)0x8516), **TextureCubeMapPositiveY** = ((int)0x8517),
TextureCubeMapNegativeY = ((int)0x8518), **TextureCubeMapPositiveZ** = ((int)0x8519), **TextureCubeMapNegativeZ** = ((int)0x851A), **MaxCubeMapTextureSize** = ((int)0x851C),
VertexAttribArrayEnabled = ((int)0x8622), **VertexAttribArraySize** = ((int)0x8623), **VertexAttribArrayStride** = ((int)0x8624), **VertexAttribArrayType** = ((int)0x8625),
CurrentVertexAttrib = ((int)0x8626), **VertexAttribArrayPointer** = ((int)0x8645), **NumCompressedTextureFormats** = ((int)0x86A2), **CompressedTextureFormats** = ((int)0x86A3),
Z400BinaryAmd = ((int)0x8740), **ProgramBinaryLengthOes** = ((int)0x8741), **BufferSize** = ((int)0x8764), **BufferUsage** = ((int)0x8765),
AtcRgbaInterpolatedAlphaAmd = ((int)0x87EE), **GL_3DcXAmd** = ((int)0x87F9), **GL_3DcXyAmd** = ((int)0x87FA), **NumProgramBinaryFormatsOes** = ((int)0x87FE),
ProgramBinaryFormatsOes = ((int)0x87FF), **StencilBackFunc** = ((int)0x8800), **StencilBackFail** = ((int)0x8801), **StencilBackPassDepthFail** = ((int)0x8802),
StencilBackPassDepthPass = ((int)0x8803), **BlendEquationAlpha** = ((int)0x883D), **MaxVertexAttribs** = ((int)0x8869), **VertexAttribArrayNormalized** = ((int)0x886A),
MaxTextureImageUnits = ((int)0x8872), **ArrayBuffer** = ((int)0x8892), **ElementArrayBuffer** = ((int)0x8893), **ArrayBufferBinding** = ((int)0x8894),
ElementArrayBufferBinding = ((int)0x8895), **VertexAttribArrayBufferBinding** = ((int)0x889F), **WriteOnlyOes** = ((int)0x88B9), **BufferAccessOes** = ((int)0x88BB),
BufferMappedOes = ((int)0x88BC), **BufferMapPointerOes** = ((int)0x88BD), **StreamDraw** = ((int)0x88E0), **StaticDraw** = ((int)0x88E4),
DynamicDraw = ((int)0x88E8), **Depth24Stencil8Oes** = ((int)0x88F0), **FragmentShader** = ((int)0x8B30), **VertexShader** = ((int)0x8B31),
MaxVertexTextureImageUnits = ((int)0x8B4C), **MaxCombinedTextureImageUnits** = ((int)0x8B4D), **ShaderType** = ((int)0x8B4F), **FloatVec2** = ((int)0x8B50),
FloatVec3 = ((int)0x8B51), **FloatVec4** = ((int)0x8B52), **IntVec2** = ((int)0x8B53), **IntVec3** = ((int)0x8B54),
IntVec4 = ((int)0x8B55), **Bool** = ((int)0x8B56), **BoolVec2** = ((int)0x8B57), **BoolVec3** = ((int)0x8B58),
BoolVec4 = ((int)0x8B59), **FloatMat2** = ((int)0x8B5A), **FloatMat3** = ((int)0x8B5B), **FloatMat4** = ((int)0x8B5C),

Sampler2D = ((int)0x8B5E), **Sampler3DOes** = ((int)0x8B5F), **SamplerCube** = ((int)0x8B60), **DeleteStatus** = ((int)0x8B80),
CompileStatus = ((int)0x8B81), **LinkStatus** = ((int)0x8B82), **ValidateStatus** = ((int)0x8B83), **InfoLogLength** = ((int)0x8B84),
AttachedShaders = ((int)0x8B85), **ActiveUniforms** = ((int)0x8B86), **ActiveUniformMaxLength** = ((int)0x8B87), **ShaderSourceLength** = ((int)0x8B88),
ActiveAttributes = ((int)0x8B89), **ActiveAttributeMaxLength** = ((int)0x8B8A), **FragmentShaderDerivativeHintOes** = ((int)0x8B8B), **ShadingLanguageVersion** = ((int)0x8B8C),
CurrentProgram = ((int)0x8B8D), **Palette4Rgb8Oes** = ((int)0x8B90), **Palette4Rgba8Oes** = ((int)0x8B91), **Palette4R5G6B5Oes** = ((int)0x8B92),
Palette4Rgba4Oes = ((int)0x8B93), **Palette4Rgb5A1Oes** = ((int)0x8B94), **Palette8Rgb8Oes** = ((int)0x8B95), **Palette8Rgba8Oes** = ((int)0x8B96),
Palette8R5G6B5Oes = ((int)0x8B97), **Palette8Rgba4Oes** = ((int)0x8B98), **Palette8Rgb5A1Oes** = ((int)0x8B99), **ImplementationColorReadType** = ((int)0x8B9A),
ImplementationColorReadFormat = ((int)0x8B9B), **CounterTypeAmd** = ((int)0x8BC0), **CounterRangeAmd** = ((int)0x8BC1), **UnsignedInt64Amd** = ((int)0x8BC2),
PercentageAmd = ((int)0x8BC3), **PerfmonResultAvailableAmd** = ((int)0x8BC4), **PerfmonResultSizeAmd** = ((int)0x8BC5), **PerfmonResultAmd** = ((int)0x8BC6),
CompressedRgbPvrtc4Bppv1Img = ((int)0x8C00), **CompressedR5G6B5Pvrtc4Bppv1Img** = ((int)0x8C01), **CompressedRgbaPvrtc4Bppv1Img** = ((int)0x8C02), **CompressedR5G6B5Pvrtc2Bppv1Img** = ((int)0x8C03),
AtcRgbAmd = ((int)0x8C92), **AtcRgbaExplicitAlphaAmd** = ((int)0x8C93), **StencilBackRef** = ((int)0x8CA3), **StencilBackValueMask** = ((int)0x8CA4),
StencilBackWritemask = ((int)0x8CA5), **FramebufferBinding** = ((int)0x8CA6), **RenderbufferBinding** = ((int)0x8CA7), **FramebufferAttachmentObjectType** = ((int)0x8CD0),
FramebufferAttachmentObjectName = ((int)0x8CD1), **FramebufferAttachmentTextureLevel** = ((int)0x8CD2), **FramebufferAttachmentTextureCubeMapFace** = ((int)0x8CD3), **FramebufferAttachmentTexture3DZoffsetOes** = ((int)0x8CD4),
FramebufferComplete = ((int)0x8CD5), **FramebufferIncompleteAttachment** = ((int)0x8CD6), **FramebufferIncompleteMissingAttachment** = ((int)0x8CD7), **FramebufferIncompleteDimensions** = ((int)0x8CD9),
FramebufferUnsupported = ((int)0x8CDD), **ColorAttachment0** = ((int)0x8CE0), **DepthAttachment** = ((int)0x8D00), **StencilAttachment** = ((int)0x8D20),

```

Framebuffer = ((int)0x8D40), Renderbuffer = ((int)0x8D41), Renderbuffer-
Width = ((int)0x8D42), RenderbufferHeight = ((int)0x8D43),
RenderbufferInternalFormat = ((int)0x8D44), StencilIndex1Oes =
((int)0x8D46), StencilIndex4Oes = ((int)0x8D47), StencilIndex8 =
((int)0x8D48),
RenderbufferRedSize = ((int)0x8D50), RenderbufferGreenSize =
((int)0x8D51), RenderbufferBlueSize = ((int)0x8D52), RenderbufferAl-
phaSize = ((int)0x8D53),
RenderbufferDepthSize = ((int)0x8D54), RenderbufferStencilSize =
((int)0x8D55), HalfFloatOes = ((int)0x8D61), Rgb565 = ((int)0x8D62),
Etc1Rgb8Oes = ((int)0x8D64), LowFloat = ((int)0x8DF0), MediumFloat =
((int)0x8DF1), HighFloat = ((int)0x8DF2),
LowInt = ((int)0x8DF3), MediumInt = ((int)0x8DF4), HighInt =
((int)0x8DF5), UnsignedInt1010102Oes = ((int)0x8DF6),
Int1010102Oes = ((int)0x8DF7), ShaderBinaryFormats = ((int)0x8DF8),
NumShaderBinaryFormats = ((int)0x8DF9), ShaderCompiler =
((int)0x8DFA),
MaxVertexUniformVectors = ((int)0x8DFB), MaxVaryingVectors =
((int)0x8DFC), MaxFragmentUniformVectors = ((int)0x8DFD), Perf-
monGlobalModeQcom = ((int)0x8FA0),
AmdCompressed3DcTexture = ((int)1), AmdCompressedAtcTexture =
((int)1), AmdPerformanceMonitor = ((int)1), AmdProgramBinaryZ400 =
((int)1),
EsVersion20 = ((int)1), ExtTextureFilterAnisotropic = ((int)1), ExtTexture-
FormatBgra8888 = ((int)1), ExtTextureType2101010Rev = ((int)1),
ImgReadFormat = ((int)1), ImgTextureCompressionPvrtc = ((int)1),
NvFence = ((int)1), OesCompressedEtc1Rgb8Texture = ((int)1),
OesCompressedPalettedTexture = ((int)1), OesDepth24 = ((int)1), Oes-
Depth32 = ((int)1), OesDepthTexture = ((int)1),
OesEglImage = ((int)1), OesElementIndexUint = ((int)1), OesFboRender-
Mipmap = ((int)1), OesFragmentPrecisionHigh = ((int)1),
OesGetProgramBinary = ((int)1), OesMapbuffer = ((int)1), OesPacked-
DepthStencil = ((int)1), OesRgb8Rgba8 = ((int)1),
OesStandardDerivatives = ((int)1), OesStencil1 = ((int)1), OesStencil4 =
((int)1), OesTexture3D = ((int)1),
OesTextureFloat = ((int)1), OesTextureFloatLinear = ((int)1), OesTexture-
HalfFloat = ((int)1), OesTextureHalfFloatLinear = ((int)1),
OesTextureNpot = ((int)1), OesVertexHalfFloat = ((int)1), OesVertex-
Type1010102 = ((int)1), One = ((int)1),
QcomDriverControl = ((int)1), QcomPerfmonGlobalMode = ((int)1), True =
((int)1) }

```

- enum **Amdcompressed3Dctexture** { **GL_3DcXAmd** = ((int)0x87F9), **GL_3DcXyAmd** = ((int)0x87FA), **AmdCompressed3DcTexture** = ((int)1) }
- enum **AmdcompressedAtctexture** { **AtcRgbaInterpolatedAlphaAmd** = ((int)0x87EE), **AtcRgbAmd** = ((int)0x8C92), **AtcRgbaExplicitAlphaAmd** = ((int)0x8C93), **AmdCompressedAtcTexture** = ((int)1) }
- enum **AmdperformanceMonitor** {
CounterTypeAmd = ((int)0x8BC0), **CounterRangeAmd** = ((int)0x8BC1), **UnsignedInt64Amd** = ((int)0x8BC2), **PercentageAmd** = ((int)0x8BC3),
PerfmonResultAvailableAmd = ((int)0x8BC4), **PerfmonResultSizeAmd** = ((int)0x8BC5), **PerfmonResultAmd** = ((int)0x8BC6), **AmdPerformanceMonitor** = ((int)1) }
- enum **AmdprogramBinaryZ400** { **Z400BinaryAmd** = ((int)0x8740), **AmdProgramBinaryZ400** = ((int)1) }
- enum **BeginMode** {
Points = ((int)0x0000), **Lines** = ((int)0x0001), **LineLoop** = ((int)0x0002), **LineStrip** = ((int)0x0003),
Triangles = ((int)0x0004), **TriangleStrip** = ((int)0x0005), **TriangleFan** = ((int)0x0006) }
- enum **BlendEquationMode** { **FuncAdd** = ((int)0X8006), **FuncSubtract** = ((int)0X800a), **FuncReverseSubtract** = ((int)0X800b) }
- enum **BlendEquationSeparate** { **FuncAdd** = ((int)0x8006), **BlendEquation** = ((int)0x8009), **BlendEquationAlpha** = ((int)0x883D) }
- enum **BlendingFactorDest** {
Zero = ((int)0), **SrcColor** = ((int)0x0300), **OneMinusSrcColor** = ((int)0x0301), **SrcAlpha** = ((int)0x0302),
OneMinusSrcAlpha = ((int)0x0303), **DstAlpha** = ((int)0x0304), **OneMinusDstAlpha** = ((int)0x0305), **DstColor** = ((int)0X0306),
OneMinusDstColor = ((int)0X0307), **SrcAlphaSaturate** = ((int)0X0308), **ConstantColor** = ((int)0X8001), **OneMinusConstantColor** = ((int)0X8002),
ConstantAlpha = ((int)0X8003), **OneMinusConstantAlpha** = ((int)0X8004), **One** = ((int)1) }
- enum **BlendingFactorSrc** {
Zero = ((int)0), **SrcColor** = ((int)0X0300), **OneMinusSrcColor** = ((int)0X0301), **SrcAlpha** = ((int)0X0302),
OneMinusSrcAlpha = ((int)0X0303), **DstAlpha** = ((int)0X0304), **OneMinusDstAlpha** = ((int)0X0305), **DstColor** = ((int)0x0306),
OneMinusDstColor = ((int)0x0307), **SrcAlphaSaturate** = ((int)0x0308), **ConstantColor** = ((int)0X8001), **OneMinusConstantColor** = ((int)0X8002),
ConstantAlpha = ((int)0X8003), **OneMinusConstantAlpha** = ((int)0X8004), **One** = ((int)1) }

- enum **BlendSubtract** { **FuncSubtract** = ((int)0x800A), **FuncReverseSubtract** = ((int)0x800B) }
- enum **Boolean** { **False** = ((int)0), **True** = ((int)1) }
- enum **BufferObjects** {
 - CurrentVertexAttrib** = ((int)0x8626), **BufferSize** = ((int)0x8764), **BufferUsage** = ((int)0x8765), **ArrayBuffer** = ((int)0x8892),
 - ElementArrayBuffer** = ((int)0x8893), **ArrayBufferBinding** = ((int)0x8894), **ElementArrayBufferBinding** = ((int)0x8895), **StreamDraw** = ((int)0x88E0),
 - StaticDraw** = ((int)0x88E4), **DynamicDraw** = ((int)0x88E8) }
- enum **BufferParameterName** { **BufferSize** = ((int)0x8764), **BufferUsage** = ((int)0x8765) }
- enum **BufferTarget** { **ArrayBuffer** = ((int)0x8892), **ElementArrayBuffer** = ((int)0x8893) }
- enum **BufferUsage** { **StreamDraw** = ((int)0x88e0), **StaticDraw** = ((int)0x88e4), **DynamicDraw** = ((int)0x88e8) }
- enum **ClearBufferMask** { **DepthBufferBit** = ((int)0x00000100), **StencilBufferBit** = ((int)0x00000400), **ColorBufferBit** = ((int)0x00004000) }
- enum **CullFaceMode** { **Front** = ((int)0x0404), **Back** = ((int)0x0405), **FrontAndBack** = ((int)0x0408) }
- enum **DataType** {
 - Byte** = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402), **UnsignedShort** = ((int)0x1403),
 - Int** = ((int)0x1404), **UnsignedInt** = ((int)0x1405), **Float** = ((int)0x1406), **Fixed** = ((int)0x140C) }
- enum **DepthFunction** {
 - Never** = ((int)0x0200), **Less** = ((int)0x0201), **Equal** = ((int)0x0202), **Lequal** = ((int)0x0203),
 - Greater** = ((int)0x0204), **Notequal** = ((int)0x0205), **Gequal** = ((int)0x0206), **Always** = ((int)0x0207) }
- enum **DrawElementsType** { **UnsignedByte** = ((int)0x1401), **UnsignedShort** = ((int)0x1403) }
- enum **EnableCap** {
 - CullFace** = ((int)0x0B44), **DepthTest** = ((int)0x0B71), **StencilTest** = ((int)0x0B90), **Dither** = ((int)0x0BD0),
 - Blend** = ((int)0x0BE2), **ScissorTest** = ((int)0x0C11), **Texture2D** = ((int)0x0DE1), **PolygonOffsetFill** = ((int)0x8037),
 - SampleAlphaToCoverage** = ((int)0x809E), **SampleCoverage** = ((int)0x80A0) }
- enum **ErrorCode** {
 - NoError** = ((int)0), **InvalidEnum** = ((int)0x0500), **InvalidValue** = ((int)0x0501), **InvalidOperation** = ((int)0x0502),

```

OutOfMemory = ((int)0x0505), InvalidFramebufferOperation =
((int)0X0506) }
• enum ExttextureFilterAnisotropic { TextureMaxAnisotropyExt =
((int)0x84FE), MaxTextureMaxAnisotropyExt = ((int)0x84FF), ExtTex-
tureFilterAnisotropic = ((int)1) }
• enum ExttextureFormatBgra8888 { Bgra = ((int)0x80E1), ExtTextureFor-
matBgra8888 = ((int)1) }
• enum ExttextureType2101010Rev { UnsignedInt2101010RevExt =
((int)0x8368), ExtTextureType2101010Rev = ((int)1) }
• enum FramebufferErrorCode {
FramebufferComplete = ((int)0X8cd5), FramebufferIncompleteAttach-
ment = ((int)0X8cd6), FramebufferIncompleteMissingAttachment =
((int)0X8cd7), FramebufferIncompleteDimensions = ((int)0X8cd9),
FramebufferUnsupported = ((int)0X8cdd) }
• enum FramebufferObject {
None = ((int)0), InvalidFramebufferOperation = ((int)0x0506), StencilIndex
= ((int)0x1901), Rgba4 = ((int)0x8056),
Rgb5A1 = ((int)0x8057), DepthComponent16 = ((int)0x81A5), MaxRender-
bufferSize = ((int)0x84E8), FramebufferBinding = ((int)0x8CA6),
RenderbufferBinding = ((int)0x8CA7), FramebufferAttachmentObjectType
= ((int)0x8CD0), FramebufferAttachmentObjectName = ((int)0x8CD1),
FramebufferAttachmentTextureLevel = ((int)0x8CD2),
FramebufferAttachmentTextureCubeMapFace = ((int)0x8CD3), Frame-
bufferComplete = ((int)0x8CD5), FramebufferIncompleteAttachment =
((int)0x8CD6), FramebufferIncompleteMissingAttachment = ((int)0x8CD7),
FramebufferIncompleteDimensions = ((int)0x8CD9), FramebufferUnsup-
ported = ((int)0x8CDD), ColorAttachment0 = ((int)0x8CE0), DepthAttach-
ment = ((int)0x8D00),
StencilAttachment = ((int)0x8D20), Framebuffer = ((int)0x8D40), Render-
buffer = ((int)0x8D41), RenderbufferWidth = ((int)0x8D42),
RenderbufferHeight = ((int)0x8D43), RenderbufferInternalFormat =
((int)0x8D44), StencilIndex8 = ((int)0x8D48), RenderbufferRedSize =
((int)0x8D50),
RenderbufferGreenSize = ((int)0x8D51), RenderbufferBlueSize =
((int)0x8D52), RenderbufferAlphaSize = ((int)0x8D53), Renderbuffer-
DepthSize = ((int)0x8D54),
RenderbufferStencilSize = ((int)0x8D55), Rgb565 = ((int)0x8D62) }
• enum FramebufferParameterName { FramebufferAttachmentObjectType
= ((int)0X8cd0), FramebufferAttachmentObjectName = ((int)0X8cd1),
FramebufferAttachmentTextureLevel = ((int)0X8cd2), FramebufferAttach-
mentTextureCubeMapFace = ((int)0X8cd3) }

```

- enum **FramebufferSlot** { **ColorAttachment0** = ((int)0X8ce0), **DepthAttachment** = ((int)0X8d00), **StencilAttachment** = ((int)0X8d20) }
- enum **FramebufferTarget** { **Framebuffer** = ((int)0X8d40) }
- enum **FrontFaceDirection** { **Cw** = ((int)0x0900), **Ccw** = ((int)0x0901) }
- enum **GetPName** {

LineWidth = ((int)0x0B21), **CullFace** = ((int)0X0b44), **CullFaceMode** = ((int)0x0B45), **FrontFace** = ((int)0x0B46),

DepthRange = ((int)0x0B70), **DepthTest** = ((int)0X0b71), **DepthWritemask** = ((int)0x0B72), **DepthClearValue** = ((int)0x0B73),

DepthFunc = ((int)0x0B74), **StencilTest** = ((int)0X0b90), **StencilClearValue** = ((int)0x0B91), **StencilFunc** = ((int)0x0B92),

StencilValueMask = ((int)0x0B93), **StencilFail** = ((int)0x0B94), **StencilPassDepthFail** = ((int)0x0B95), **StencilPassDepthPass** = ((int)0x0B96),

StencilRef = ((int)0x0B97), **StencilWritemask** = ((int)0x0B98), **Viewport** = ((int)0x0BA2), **Dither** = ((int)0X0bd0),

Blend = ((int)0X0be2), **ScissorBox** = ((int)0x0C10), **ScissorTest** = ((int)0X0c11), **ColorClearValue** = ((int)0x0C22),

ColorWritemask = ((int)0x0C23), **UnpackAlignment** = ((int)0x0CF5), **PackAlignment** = ((int)0x0D05), **MaxTextureSize** = ((int)0x0D33),

MaxViewportDims = ((int)0x0D3A), **SubpixelBits** = ((int)0x0D50), **RedBits** = ((int)0x0D52), **GreenBits** = ((int)0x0D53),

BlueBits = ((int)0x0D54), **AlphaBits** = ((int)0x0D55), **DepthBits** = ((int)0x0D56), **StencilBits** = ((int)0x0D57),

Texture2D = ((int)0X0de1), **PolygonOffsetUnits** = ((int)0x2A00), **BlendColor** = ((int)0X8005), **BlendEquation** = ((int)0X8009),

BlendEquationRgb = ((int)0X8009), **PolygonOffsetFill** = ((int)0X8037), **PolygonOffsetFactor** = ((int)0x8038), **TextureBinding2D** = ((int)0x8069),

SampleAlphaToCoverage = ((int)0X809e), **SampleCoverage** = ((int)0X80a0), **SampleBuffers** = ((int)0x80A8), **Samples** = ((int)0x80A9),

SampleCoverageValue = ((int)0x80AA), **SampleCoverageInvert** = ((int)0x80AB), **BlendDstRgb** = ((int)0X80c8), **BlendSrcRgb** = ((int)0X80c9),

BlendDstAlpha = ((int)0X80ca), **BlendSrcAlpha** = ((int)0X80cb), **GenerateMipmapHint** = ((int)0X8192), **AliasedPointSizeRange** = ((int)0x846D),

AliasedLineWidthRange = ((int)0x846E), **ActiveTexture** = ((int)0X84e0), **MaxRenderbufferSize** = ((int)0X84e8), **TextureBindingCubeMap** = ((int)0X8514),

MaxCubeMapTextureSize = ((int)0X851c), **NumCompressedTextureFormats** = ((int)0X86a2), **CompressedTextureFormats** = ((int)0X86a3), **StencilBackFunc** = ((int)0x8800),


```

StencilBackFail = ((int)0x8801), StencilBackPassDepthFail = ((int)0x8802),
StencilBackPassDepthPass = ((int)0x8803), BlendEquationAlpha =
((int)0x883d),

MaxVertexAttribs = ((int)0X8869), MaxTextureImageUnits = ((int)0X8872),
ArrayBufferBinding = ((int)0X8894), ElementArrayBufferBinding =
((int)0X8895),

MaxVertexTextureImageUnits = ((int)0X8b4c), MaxCombinedTex-
tureImageUnits = ((int)0X8b4d), CurrentProgram = ((int)0X8b8d), Im-
plementationColorReadType = ((int)0X8b9a),

ImplementationColorReadFormat = ((int)0X8b9b), StencilBackRef =
((int)0x8CA3), StencilBackValueMask = ((int)0x8CA4), StencilBack-
Writemask = ((int)0x8CA5),

FramebufferBinding = ((int)0X8ca6), RenderbufferBinding = ((int)0X8ca7),
ShaderBinaryFormats = ((int)0X8df8), NumShaderBinaryFormats =
((int)0X8df9),

ShaderCompiler = ((int)0X8dfa), MaxVertexUniformVectors =
((int)0X8dfb), MaxVaryingVectors = ((int)0X8dfc), MaxFragmentUni-
formVectors = ((int)0X8dfd) }

• enum GetTextureParameter {

    TextureMagFilter = ((int)0X2800), TextureMinFilter = ((int)0X2801), Tex-
tureWrapS = ((int)0X2802), TextureWrapT = ((int)0X2803),

    NumCompressedTextureFormats = ((int)0x86A2), CompressedTextureFor-
formats = ((int)0x86A3) }

• enum HintMode { DontCare = ((int)0x1100), Fastest = ((int)0x1101), Nicest
= ((int)0x1102) }

• enum HintTarget { GenerateMipmapHint = ((int)0x8192) }

• enum ImgreadFormat { Bgra = ((int)0x80E1), UnsignedShort4444Rev =
((int)0x8365), UnsignedShort1555Rev = ((int)0x8366), ImgReadFormat =
((int)1) }

• enum ImgtextureCompressionPvrtc {

    CompressedRgbPvrtc4Bppv1Img = ((int)0x8C00), Compresse-
dRgbPvrtc2Bppv1Img = ((int)0x8C01), CompressedRgbaPvrtc4Bppv1Img
= ((int)0x8C02), CompressedRgbaPvrtc2Bppv1Img = ((int)0x8C03),

    ImgTextureCompressionPvrtc = ((int)1) }

• enum Nvfence { AllCompletedNv = ((int)0x84F2), FenceStatusNv =
((int)0x84F3), FenceConditionNv = ((int)0x84F4), NvFence = ((int)1) }

• enum OescompressedEtc1Rgb8Texture { Etc1Rgb8Oes = ((int)0x8D64),
OesCompressedEtc1Rgb8Texture = ((int)1) }

• enum OescompressedPalettedTexture {

    Palette4Rgb8Oes = ((int)0x8B90), Palette4Rgba8Oes = ((int)0x8B91),
Palette4R5G6B5Oes = ((int)0x8B92), Palette4Rgba4Oes = ((int)0x8B93),

```

```

Palette4Rgb5A1Oes = ((int)0x8B94), Palette8Rgb8Oes = ((int)0x8B95),
Palette8Rgba8Oes = ((int)0x8B96), Palette8R5G6B5Oes = ((int)0x8B97),
Palette8Rgba4Oes = ((int)0x8B98), Palette8Rgb5A1Oes = ((int)0x8B99),
OesCompressedPalettedTexture = ((int)1) }
• enum Oesdepth24 { DepthComponent24Oes = ((int)0x81A6), OesDepth24 =
  ((int)1) }
• enum Oesdepth32 { DepthComponent32Oes = ((int)0x81A7), OesDepth32 =
  ((int)1) }
• enum OesdepthTexture { OesDepthTexture = ((int)1) }
• enum Oeseglimage { OesEglImage = ((int)1) }
• enum OeselementIndexUint { OesElementIndexUint = ((int)1) }
• enum OesfboRenderMipmap { OesFboRenderMipmap = ((int)1) }
• enum OesfragmentPrecisionHigh { OesFragmentPrecisionHigh = ((int)1) }
• enum OesgetProgramBinary { ProgramBinaryLengthOes = ((int)0x8741),
  NumProgramBinaryFormatsOes = ((int)0x87FE), ProgramBinaryFormat-
sOes = ((int)0x87FF), OesGetProgramBinary = ((int)1) }
• enum Oesmapbuffer {
  WriteOnlyOes = ((int)0x88B9), BufferAccessOes = ((int)0x88BB),
  BufferMappedOes = ((int)0x88BC), BufferMapPointerOes = ((int)0x88BD),
  OesMapbuffer = ((int)1) }
• enum OespackedDepthStencil { DepthStencilOes = ((int)0x84F9), Un-
signedInt248Oes = ((int)0x84FA), Depth24Stencil8Oes = ((int)0x88F0), Oes-
PackedDepthStencil = ((int)1) }
• enum Oesrgb8Rgba8 { Rgb8Oes = ((int)0x8051), Rgba8Oes = ((int)0x8058),
  OesRgb8Rgba8 = ((int)1) }
• enum OesstandardDerivatives { FragmentShaderDerivativeHintOes =
  ((int)0x8B8B), OesStandardDerivatives = ((int)1) }
• enum Oesstencil1 { StencilIndex1Oes = ((int)0x8D46), OesStencil1 = ((int)1)
  }
• enum Oesstencil4 { StencilIndex4Oes = ((int)0x8D47), OesStencil4 = ((int)1)
  }
• enum Oestexture3D {
  TextureBinding3DOes = ((int)0x806A), Texture3DOes = ((int)0x806F), Tex-
tureWrapROes = ((int)0x8072), Max3DTextureSizeOes = ((int)0x8073),
  Sampler3DOes = ((int)0x8B5F), FramebufferAttachmentTex-
ture3DZoffsetOes = ((int)0x8CD4), OesTexture3D = ((int)1) }
• enum OestextureFloat { OesTextureFloat = ((int)1) }
• enum OestextureFloatLinear { OesTextureFloatLinear = ((int)1) }
• enum OestextureHalfFloat { HalfFloatOes = ((int)0x8D61), OesTextureHalf-
Float = ((int)1) }
• enum OestextureHalfFloatLinear { OesTextureHalfFloatLinear = ((int)1) }
• enum OestextureNpot { OesTextureNpot = ((int)1) }

```

- enum **OesvertexHalfFloat** { **OesVertexHalfFloat** = ((int)1) }
- enum **OesvertexType1010102** { **UnsignedInt1010102Oes** = ((int)0x8DF6), **Int1010102Oes** = ((int)0x8DF7), **OesVertexType1010102** = ((int)1) }
- enum **OpenGlescoreVersions** { **EsVersion20** = ((int)1) }
- enum **PixelFormat** {
DepthComponent = ((int)0x1902), **Alpha** = ((int)0x1906), **Rgb** = ((int)0x1907), **Rgba** = ((int)0x1908),
Luminance = ((int)0x1909), **LuminanceAlpha** = ((int)0x190A) }
- enum **PixelInternalFormat** {
Alpha = ((int)0X1906), **Rgb** = ((int)0X1907), **Rgba** = ((int)0X1908), **Luminance** = ((int)0X1909),
LuminanceAlpha = ((int)0X190a) }
- enum **PixelStoreParameter** { **UnpackAlignment** = ((int)0X0cf5), **PackAlignment** = ((int)0X0d05) }
- enum **PixelType** { **UnsignedByte** = ((int)0X1401), **UnsignedShort4444** = ((int)0x8033), **UnsignedShort5551** = ((int)0x8034), **UnsignedShort565** = ((int)0x8363) }
- enum **ProgramParameter** {
DeleteStatus = ((int)0X8b80), **LinkStatus** = ((int)0X8b82), **ValidateStatus** = ((int)0X8b83), **InfoLogLength** = ((int)0X8b84),
AttachedShaders = ((int)0X8b85), **ActiveUniforms** = ((int)0X8b86), **ActiveUniformMaxLength** = ((int)0X8b87), **ActiveAttributes** = ((int)0X8b89),
ActiveAttributeMaxLength = ((int)0X8b8a) }
- enum **QcomdriverControl** { **QcomDriverControl** = ((int)1) }
- enum **QcomperfmonGlobalMode** { **PerfmonGlobalModeQcom** = ((int)0x8FA0), **QcomPerfmonGlobalMode** = ((int)1) }
- enum **ReadFormat** { **ImplementationColorReadType** = ((int)0x8B9A), **ImplementationColorReadFormat** = ((int)0x8B9B) }
- enum **RenderbufferInternalFormat** {
Rgba4 = ((int)0X8056), **Rgb5A1** = ((int)0X8057), **DepthComponent16** = ((int)0X81a5), **StencilIndex8** = ((int)0X8d48),
Rgb565 = ((int)0X8d62) }
- enum **RenderbufferParameterName** {
RenderbufferWidth = ((int)0X8d42), **RenderbufferHeight** = ((int)0X8d43), **RenderbufferInternalFormat** = ((int)0X8d44), **RenderbufferRedSize** = ((int)0X8d50),
RenderbufferGreenSize = ((int)0X8d51), **RenderbufferBlueSize** = ((int)0X8d52), **RenderbufferAlphaSize** = ((int)0X8d53), **RenderbufferDepthSize** = ((int)0X8d54),
RenderbufferStencilSize = ((int)0X8d55) }
- enum **RenderbufferTarget** { **Renderbuffer** = ((int)0X8d41) }

- enum **SeparateBlendFunctions** {
 - ConstantColor** = ((int)0x8001), **OneMinusConstantColor** = ((int)0x8002),
 - ConstantAlpha** = ((int)0x8003), **OneMinusConstantAlpha** = ((int)0x8004),
 - BlendColor** = ((int)0x8005), **BlendDstRgb** = ((int)0x80C8), **BlendSrcRgb** = ((int)0x80C9), **BlendDstAlpha** = ((int)0x80CA),
 - BlendSrcAlpha** = ((int)0x80CB) }
- enum **ShaderBinary** { **ShaderBinaryFormats** = ((int)0x8DF8), **NumShaderBinaryFormats** = ((int)0x8DF9) }
- enum **ShaderBinaryFormat**
- enum **ShaderParameter** {
 - ShaderType** = ((int)0X8b4f), **DeleteStatus** = ((int)0X8b80), **CompileStatus** = ((int)0X8b81), **InfoLogLength** = ((int)0X8b84),
 - ShaderSourceLength** = ((int)0X8b88) }
- enum **ShaderPrecision** {
 - LowFloat** = ((int)0X8df0), **MediumFloat** = ((int)0X8df1), **HighFloat** = ((int)0X8df2), **LowInt** = ((int)0X8df3),
 - MediumInt** = ((int)0X8df4), **HighInt** = ((int)0X8df5) }
- enum **ShaderPrecisionSpecifiedTypes** {
 - LowFloat** = ((int)0x8DF0), **MediumFloat** = ((int)0x8DF1), **HighFloat** = ((int)0x8DF2), **LowInt** = ((int)0x8DF3),
 - MediumInt** = ((int)0x8DF4), **HighInt** = ((int)0x8DF5) }
- enum **Shaders** {
 - MaxVertexAttribs** = ((int)0x8869), **MaxTextureImageUnits** = ((int)0x8872), **FragmentShader** = ((int)0x8B30), **VertexShader** = ((int)0x8B31),
 - MaxVertexTextureImageUnits** = ((int)0x8B4C), **MaxCombinedTextureImageUnits** = ((int)0x8B4D), **ShaderType** = ((int)0x8B4F), **DeleteStatus** = ((int)0x8B80),
 - LinkStatus** = ((int)0x8B82), **ValidateStatus** = ((int)0x8B83), **AttachedShaders** = ((int)0x8B85), **ActiveUniforms** = ((int)0x8B86),
 - ActiveUniformMaxLength** = ((int)0x8B87), **ActiveAttributes** = ((int)0x8B89), **ActiveAttributeMaxLength** = ((int)0x8B8A), **ShadingLanguageVersion** = ((int)0x8B8C),
 - CurrentProgram** = ((int)0x8B8D), **MaxVertexUniformVectors** = ((int)0x8DFB), **MaxVaryingVectors** = ((int)0x8DFC), **MaxFragmentUniformVectors** = ((int)0x8DFD) }
- enum **ShaderSource** { **CompileStatus** = ((int)0x8B81), **InfoLogLength** = ((int)0x8B84), **ShaderSourceLength** = ((int)0x8B88), **ShaderCompiler** = ((int)0x8DFA) }
- enum **ShaderType** { **FragmentShader** = ((int)0X8b30), **VertexShader** = ((int)0X8b31) }

- enum **StencilFunction** {
Never = ((int)0x0200), **Less** = ((int)0x0201), **Equal** = ((int)0x0202), **Lequal** = ((int)0x0203),
Greater = ((int)0x0204), **Notequal** = ((int)0x0205), **Gequal** = ((int)0x0206), **Always** = ((int)0x0207) }
- enum **StencilOp** {
Zero = ((int)0X0000), **Invert** = ((int)0x150A), **Keep** = ((int)0x1E00), **Replace** = ((int)0x1E01),
Incr = ((int)0x1E02), **Decr** = ((int)0x1E03), **IncrWrap** = ((int)0x8507), **DecrWrap** = ((int)0x8508) }
- enum **StringName** {
Vendor = ((int)0x1F00), **Renderer** = ((int)0x1F01), **Version** = ((int)0x1F02), **Extensions** = ((int)0x1F03),
ShadingLanguageVersion = ((int)0X8b8c) }
- enum **TextureMagFilter** { **Nearest** = ((int)0x2600), **Linear** = ((int)0x2601) }
- enum **TextureMinFilter** {
Nearest = ((int)0X2600), **Linear** = ((int)0X2601), **NearestMipmapNearest** = ((int)0x2700), **LinearMipmapNearest** = ((int)0x2701),
NearestMipmapLinear = ((int)0x2702), **LinearMipmapLinear** = ((int)0x2703) }
- enum **TextureParameterName** { **TextureMagFilter** = ((int)0x2800), **TextureMinFilter** = ((int)0x2801), **TextureWrapS** = ((int)0x2802), **TextureWrapT** = ((int)0x2803) }
- enum **TextureTarget** {
Texture2D = ((int)0X0de1), **Texture** = ((int)0x1702), **TextureCubeMap** = ((int)0x8513), **TextureBindingCubeMap** = ((int)0x8514),
TextureCubeMapPositiveX = ((int)0x8515), **TextureCubeMapNegativeX** = ((int)0x8516), **TextureCubeMapPositiveY** = ((int)0x8517), **TextureCubeMapNegativeY** = ((int)0x8518),
TextureCubeMapPositiveZ = ((int)0x8519), **TextureCubeMapNegativeZ** = ((int)0x851A), **MaxCubeMapTextureSize** = ((int)0x851C) }
- enum **TextureUnit** {
Texture0 = ((int)0x84C0), **Texture1** = ((int)0x84C1), **Texture2** = ((int)0x84C2), **Texture3** = ((int)0x84C3),
Texture4 = ((int)0x84C4), **Texture5** = ((int)0x84C5), **Texture6** = ((int)0x84C6), **Texture7** = ((int)0x84C7),
Texture8 = ((int)0x84C8), **Texture9** = ((int)0x84C9), **Texture10** = ((int)0x84CA), **Texture11** = ((int)0x84CB),
Texture12 = ((int)0x84CC), **Texture13** = ((int)0x84CD), **Texture14** = ((int)0x84CE), **Texture15** = ((int)0x84CF),

```

Texture16 = ((int)0x84D0), Texture17 = ((int)0x84D1), Texture18 =
((int)0x84D2), Texture19 = ((int)0x84D3),

Texture20 = ((int)0x84D4), Texture21 = ((int)0x84D5), Texture22 =
((int)0x84D6), Texture23 = ((int)0x84D7),

Texture24 = ((int)0x84D8), Texture25 = ((int)0x84D9), Texture26 =
((int)0x84DA), Texture27 = ((int)0x84DB),

Texture28 = ((int)0x84DC), Texture29 = ((int)0x84DD), Texture30 =
((int)0x84DE), Texture31 = ((int)0x84DF),

ActiveTexture = ((int)0x84E0) }

• enum TextureWrapMode { Repeat = ((int)0x2901), ClampToEdge =
((int)0x812F), MirroredRepeat = ((int)0x8370) }

• enum UniformTypes {

FloatVec2 = ((int)0x8B50), FloatVec3 = ((int)0x8B51), FloatVec4 =
((int)0x8B52), IntVec2 = ((int)0x8B53),

IntVec3 = ((int)0x8B54), IntVec4 = ((int)0x8B55), Bool = ((int)0x8B56),
BoolVec2 = ((int)0x8B57),

BoolVec3 = ((int)0x8B58), BoolVec4 = ((int)0x8B59), FloatMat2 =
((int)0x8B5A), FloatMat3 = ((int)0x8B5B),

FloatMat4 = ((int)0x8B5C), Sampler2D = ((int)0x8B5E), SamplerCube =
((int)0x8B60) }

• enum VertexArrays {

VertexAttribArrayEnabled = ((int)0x8622), VertexAttribArraySize =
((int)0x8623), VertexAttribArrayStride = ((int)0x8624), VertexAttribArray-
Type = ((int)0x8625),

VertexAttribArrayPointer = ((int)0x8645), VertexAttribArrayNormalized =
((int)0x886A), VertexAttribArrayBufferBinding = ((int)0x889F) }

• enum VertexAttribParameter {

VertexAttribArrayEnabled = ((int)0X8622), VertexAttribArraySize =
((int)0X8623), VertexAttribArrayStride = ((int)0X8624), VertexAttribAr-
rayType = ((int)0X8625),

CurrentVertexAttrib = ((int)0X8626), VertexAttribArrayNormalized =
((int)0X886a), VertexAttribArrayBufferBinding = ((int)0X889f) }

• enum VertexAttribPointerParameter { VertexAttribArrayPointer =
((int)0X8645) }

• enum VertexAttribPointerType {

Byte = ((int)0X1400), UnsignedByte = ((int)0X1401), Short = ((int)0X1402),
UnsignedShort = ((int)0X1403),

Float = ((int)0X1406), Fixed = ((int)0X140c) }

```

4.8 Package OpenTK.Graphics.OpenGL

Classes

- class [GL](#)
OpenGL bindings for .NET, implementing the full [OpenGL](#) API, including extensions.

Enumerations

- enum **AccumOp** {
Accum = ((int)0x0100), **Load** = ((int)0x0101), **Return** = ((int)0x0102), **Mult** = ((int)0x0103),
Add = ((int)0x0104) }
- enum **ActiveAttribType** {
Float = ((int)0x1406), **FloatVec2** = ((int)0x8B50), **FloatVec3** = ((int)0x8B51),
FloatVec4 = ((int)0x8B52),
FloatMat2 = ((int)0x8B5A), **FloatMat3** = ((int)0x8B5B), **FloatMat4** = ((int)0x8B5C) }
- enum **ActiveUniformBlockParameter** {
UniformBlockBinding = ((int)0x8A3F), **UniformBlockDataSize** = ((int)0x8A40), **UniformBlockNameLength** = ((int)0x8A41), **UniformBlockActiveUniforms** = ((int)0x8A42),
UniformBlockActiveUniformIndices = ((int)0x8A43), **UniformBlockReferencedByVertexShader** = ((int)0x8A44), **UniformBlockReferencedByFragmentShader** = ((int)0x8A46) }
- enum **ActiveUniformParameter** {
UniformType = ((int)0x8A37), **UniformSize** = ((int)0x8A38), **UniformNameLength** = ((int)0x8A39), **UniformBlockIndex** = ((int)0x8A3A),
UniformOffset = ((int)0x8A3B), **UniformArrayStride** = ((int)0x8A3C), **UniformMatrixStride** = ((int)0x8A3D), **UniformIsRowMajor** = ((int)0x8A3E) }
- enum **ActiveUniformType** {
Int = ((int)0x1404), **Float** = ((int)0x1406), **FloatVec2** = ((int)0x8B50),
FloatVec3 = ((int)0x8B51),
FloatVec4 = ((int)0x8B52), **IntVec2** = ((int)0x8B53), **IntVec3** = ((int)0x8B54),
IntVec4 = ((int)0x8B55),
Bool = ((int)0x8B56), **BoolVec2** = ((int)0x8B57), **BoolVec3** = ((int)0x8B58),
BoolVec4 = ((int)0x8B59),
FloatMat2 = ((int)0x8B5A), **FloatMat3** = ((int)0x8B5B), **FloatMat4** = ((int)0x8B5C), **Sampler1D** = ((int)0x8B5D),

Sampler2D = ((int)0x8B5E), **Sampler3D** = ((int)0x8B5F), **SamplerCube** = ((int)0x8B60), **Sampler1DShadow** = ((int)0x8B61),

Sampler2DShadow = ((int)0x8B62), **Sampler2DRect** = ((int)0x8B63), **Sampler2DRectShadow** = ((int)0x8B64), **FloatMat2x3** = ((int)0x8B65),

FloatMat2x4 = ((int)0x8B66), **FloatMat3x2** = ((int)0x8B67), **FloatMat3x4** = ((int)0x8B68), **FloatMat4x2** = ((int)0x8B69),

FloatMat4x3 = ((int)0x8B6A), **Sampler1DArray** = ((int)0x8DC0), **Sampler2DArray** = ((int)0x8DC1), **SamplerBuffer** = ((int)0x8DC2),

Sampler1DArrayShadow = ((int)0x8DC3), **Sampler2DArrayShadow** = ((int)0x8DC4), **SamplerCubeShadow** = ((int)0x8DC5), **UnsignedIntVec2** = ((int)0x8DC6),

UnsignedIntVec3 = ((int)0x8DC7), **UnsignedIntVec4** = ((int)0x8DC8), **IntSampler1D** = ((int)0x8DC9), **IntSampler2D** = ((int)0x8DCA),

IntSampler3D = ((int)0x8DCB), **IntSamplerCube** = ((int)0x8DCC), **IntSampler2DRect** = ((int)0x8DCD), **IntSampler1DArray** = ((int)0x8DCE),

IntSampler2DArray = ((int)0x8DCF), **IntSamplerBuffer** = ((int)0x8DD0), **UnsignedIntSampler1D** = ((int)0x8DD1), **UnsignedIntSampler2D** = ((int)0x8DD2),

UnsignedIntSampler3D = ((int)0x8DD3), **UnsignedIntSamplerCube** = ((int)0x8DD4), **UnsignedIntSampler2DRect** = ((int)0x8DD5), **UnsignedIntSampler1DArray** = ((int)0x8DD6),

UnsignedIntSampler2DArray = ((int)0x8DD7), **UnsignedIntSamplerBuffer** = ((int)0x8DD8), **Sampler2DMultisample** = ((int)0x9108), **IntSampler2DMultisample** = ((int)0x9109),

UnsignedIntSampler2DMultisample = ((int)0x910A), **Sampler2DMultisampleArray** = ((int)0x910B), **IntSampler2DMultisampleArray** = ((int)0x910C), **UnsignedIntSampler2DMultisampleArray** = ((int)0x910D)

• enum **All** {

False = ((int)0), **NoError** = ((int)0), **None** = ((int)0), **Zero** = ((int)0),

Points = ((int)0x0000), **ClientPixelStoreBit** = ((int)0x00000001), **ContextCoreProfileBit** = ((int)0x00000001), **CurrentBit** = ((int)0x00000001),

GL2XBitAti = ((int)0x00000001), **RedBitAti** = ((int)0x00000001), **SyncFlushCommandsBit** = ((int)0x00000001), **TextureDeformationBitSgix** = ((int)0x00000001),

ClientVertexArrayBit = ((int)0x00000002), **CompBitAti** = ((int)0x00000002), **ContextCompatibilityProfileBit** = ((int)0x00000002), **GeometryDeformationBitSgix** = ((int)0x00000002),

GL4XBitAti = ((int)0x00000002), **GreenBitAti** = ((int)0x00000002), **PointBit** = ((int)0x00000002), **BlueBitAti** = ((int)0x00000004),


```

GL8XBitAti = ((int)0x00000004), LineBit = ((int)0x00000004), NegateBitAti
= ((int)0x00000004), Vertex23BitPgi = ((int)0x00000004),

BiasBitAti = ((int)0x00000008), HalfBitAti = ((int)0x00000008), PolygonBit
= ((int)0x00000008), Vertex4BitPgi = ((int)0x00000008),

PolygonStippleBit = ((int)0x00000010), QuarterBitAti = ((int)0x00000010),
EighthBitAti = ((int)0x00000020), PixelModeBit = ((int)0x00000020),

LightingBit = ((int)0x00000040), SaturateBitAti = ((int)0x00000040), FogBit
= ((int)0x00000080), DepthBufferBit = ((int)0x00000100),

AccumBufferBit = ((int)0x00000200), StencilBufferBit = ((int)0x00000400),
ViewportBit = ((int)0x00000800), TransformBit = ((int)0x00001000),

EnableBit = ((int)0x00002000), ColorBufferBit = ((int)0x00004000), HintBit
= ((int)0x00008000), ContextFlagForwardCompatibleBit = ((int)0x0001),

Lines = ((int)0x0001), MapReadBit = ((int)0x0001), RestartSun =
((int)0x0001), Color3BitPgi = ((int)0x00010000),

EvalBit = ((int)0x00010000), LineLoop = ((int)0x0002), MapWriteBit =
((int)0x0002), ReplaceMiddleSun = ((int)0x0002),

Color4BitPgi = ((int)0x00020000), ListBit = ((int)0x00020000), LineStrip =
((int)0x0003), ReplaceOldestSun = ((int)0x0003),

MapInvalidateRangeBit = ((int)0x0004), Triangles = ((int)0x0004), Edgeflag-
BitPgi = ((int)0x00040000), TextureBit = ((int)0x00040000),

TriangleStrip = ((int)0x0005), TriangleFan = ((int)0x0006), Quads =
((int)0x0007), MapInvalidateBufferBit = ((int)0x0008),

QuadStrip = ((int)0x0008), IndexBitPgi = ((int)0x00080000), ScissorBit =
((int)0x00080000), Polygon = ((int)0x0009),

LinesAdjacency = ((int)0x000A), LinesAdjacencyArb = ((int)0x000A), Line-
sAdjacencyExt = ((int)0x000A), LineStripAdjacency = ((int)0x000B),

LineStripAdjacencyArb = ((int)0x000B), LineStripAdjacencyExt =
((int)0x000B), TrianglesAdjacency = ((int)0x000C), TrianglesAdjacenc-
yArb = ((int)0x000C),

TrianglesAdjacencyExt = ((int)0x000C), TriangleStripAdjacency =
((int)0x000D), TriangleStripAdjacencyArb = ((int)0x000D), TriangleStri-
pAdjacencyExt = ((int)0x000D),

MapFlushExplicitBit = ((int)0x0010), MatAmbientBitPgi =
((int)0x00100000), MapUnsynchronizedBit = ((int)0x0020), MatAmbi-
entAndDiffuseBitPgi = ((int)0x00200000),

MatDiffuseBitPgi = ((int)0x00400000), MatEmissionBitPgi =
((int)0x00800000), Accum = ((int)0x0100), MatColorIndexesBitPgi =
((int)0x01000000),

Load = ((int)0x0101), Return = ((int)0x0102), Mult = ((int)0x0103), Add =
((int)0x0104),

```

Never = ((int)0x0200), **MatShininessBitPgi** = ((int)0x02000000), **Less** = ((int)0x0201), **Equal** = ((int)0x0202),
Lequal = ((int)0x0203), **Greater** = ((int)0x0204), **Notequal** = ((int)0x0205), **Gequal** = ((int)0x0206),
Always = ((int)0x0207), **SrcColor** = ((int)0x0300), **OneMinusSrcColor** = ((int)0x0301), **SrcAlpha** = ((int)0x0302),
OneMinusSrcAlpha = ((int)0x0303), **DstAlpha** = ((int)0x0304), **OneMinusDstAlpha** = ((int)0x0305), **DstColor** = ((int)0x0306),
OneMinusDstColor = ((int)0x0307), **SrcAlphaSaturate** = ((int)0x0308), **FrontLeft** = ((int)0x0400), **MatSpecularBitPgi** = ((int)0x04000000),
FrontRight = ((int)0x0401), **BackLeft** = ((int)0x0402), **BackRight** = ((int)0x0403), **Front** = ((int)0x0404),
Back = ((int)0x0405), **Left** = ((int)0x0406), **Right** = ((int)0x0407), **FrontAndBack** = ((int)0x0408),
Aux0 = ((int)0x0409), **Aux1** = ((int)0x040A), **Aux2** = ((int)0x040B), **Aux3** = ((int)0x040C),
InvalidEnum = ((int)0x0500), **InvalidValue** = ((int)0x0501), **InvalidOperation** = ((int)0x0502), **StackOverflow** = ((int)0x0503),
StackUnderflow = ((int)0x0504), **OutOfMemory** = ((int)0x0505), **InvalidFramebufferOperation** = ((int)0x0506), **InvalidFramebufferOperationExt** = ((int)0x0506),
GL2D = ((int)0x0600), **GL3D** = ((int)0x0601), **GL3DColor** = ((int)0x0602), **GL3DColorTexture** = ((int)0x0603),
GL4DColorTexture = ((int)0x0604), **PassThroughToken** = ((int)0x0700), **PointToken** = ((int)0x0701), **LineToken** = ((int)0x0702),
PolygonToken = ((int)0x0703), **BitmapToken** = ((int)0x0704), **DrawPixelToken** = ((int)0x0705), **CopyPixelToken** = ((int)0x0706),
LineResetToken = ((int)0x0707), **Exp** = ((int)0x0800), **NormalBitPgi** = ((int)0x08000000), **Exp2** = ((int)0x0801),
Cw = ((int)0x0900), **Ccw** = ((int)0x0901), **Coeff** = ((int)0x0A00), **Order** = ((int)0x0A01),
Domain = ((int)0x0A02), **CurrentColor** = ((int)0x0B00), **CurrentIndex** = ((int)0x0B01), **CurrentNormal** = ((int)0x0B02),
CurrentTextureCoords = ((int)0x0B03), **CurrentRasterColor** = ((int)0x0B04), **CurrentRasterIndex** = ((int)0x0B05), **CurrentRasterTextureCoords** = ((int)0x0B06),
CurrentRasterPosition = ((int)0x0B07), **CurrentRasterPositionValid** = ((int)0x0B08), **CurrentRasterDistance** = ((int)0x0B09), **PointSmooth** = ((int)0x0B10),

PointSize = ((int)0x0B11), **PointSizeRange** = ((int)0x0B12), **SmoothPointSizeRange** = ((int)0x0B12), **PointSizeGranularity** = ((int)0x0B13),
SmoothPointSizeGranularity = ((int)0x0B13), **LineSmooth** = ((int)0x0B20), **LineWidth** = ((int)0x0B21), **LineWidthRange** = ((int)0x0B22),
SmoothLineWidthRange = ((int)0x0B22), **LineWidthGranularity** = ((int)0x0B23), **SmoothLineWidthGranularity** = ((int)0x0B23), **LineStipple** = ((int)0x0B24),
LineStipplePattern = ((int)0x0B25), **LineStippleRepeat** = ((int)0x0B26), **ListMode** = ((int)0x0B30), **MaxListNesting** = ((int)0x0B31),
ListBase = ((int)0x0B32), **ListIndex** = ((int)0x0B33), **PolygonMode** = ((int)0x0B40), **PolygonSmooth** = ((int)0x0B41),
PolygonStipple = ((int)0x0B42), **EdgeFlag** = ((int)0x0B43), **CullFace** = ((int)0x0B44), **CullFaceMode** = ((int)0x0B45),
FrontFace = ((int)0x0B46), **Lighting** = ((int)0x0B50), **LightModelLocalViewer** = ((int)0x0B51), **LightModelTwoSide** = ((int)0x0B52),
LightModelAmbient = ((int)0x0B53), **ShadeModel** = ((int)0x0B54), **ColorMaterialFace** = ((int)0x0B55), **ColorMaterialParameter** = ((int)0x0B56),
ColorMaterial = ((int)0x0B57), **Fog** = ((int)0x0B60), **FogIndex** = ((int)0x0B61), **FogDensity** = ((int)0x0B62),
FogStart = ((int)0x0B63), **FogEnd** = ((int)0x0B64), **FogMode** = ((int)0x0B65), **FogColor** = ((int)0x0B66),
DepthRange = ((int)0x0B70), **DepthTest** = ((int)0x0B71), **DepthWritemask** = ((int)0x0B72), **DepthClearValue** = ((int)0x0B73),
DepthFunc = ((int)0x0B74), **AccumClearValue** = ((int)0x0B80), **StencilTest** = ((int)0x0B90), **StencilClearValue** = ((int)0x0B91),
StencilFunc = ((int)0x0B92), **StencilValueMask** = ((int)0x0B93), **StencilFail** = ((int)0x0B94), **StencilPassDepthFail** = ((int)0x0B95),
StencilPassDepthPass = ((int)0x0B96), **StencilRef** = ((int)0x0B97), **StencilWritemask** = ((int)0x0B98), **MatrixMode** = ((int)0x0BA0),
Normalize = ((int)0x0BA1), **Viewport** = ((int)0x0BA2), **Modelview0StackDepthExt** = ((int)0x0BA3), **ModelviewStackDepth** = ((int)0x0BA3),
ProjectionStackDepth = ((int)0x0BA4), **TextureStackDepth** = ((int)0x0BA5), **Modelview0MatrixExt** = ((int)0x0BA6), **ModelviewMatrix** = ((int)0x0BA6),
ProjectionMatrix = ((int)0x0BA7), **TextureMatrix** = ((int)0x0BA8), **AttribStackDepth** = ((int)0x0BB0), **ClientAttribStackDepth** = ((int)0x0BB1),
AlphaTest = ((int)0x0BC0), **AlphaTestFunc** = ((int)0x0BC1), **AlphaTestRef** = ((int)0x0BC2), **Dither** = ((int)0x0BD0),
BlendDst = ((int)0x0BE0), **BlendSrc** = ((int)0x0BE1), **Blend** = ((int)0x0BE2), **LogicOpMode** = ((int)0x0BF0),

IndexLogicOp = ((int)0x0BF1), **LogicOp** = ((int)0x0BF1), **ColorLogicOp** = ((int)0x0BF2), **AuxBuffers** = ((int)0x0C00),
DrawBuffer = ((int)0x0C01), **ReadBuffer** = ((int)0x0C02), **ScissorBox** = ((int)0x0C10), **ScissorTest** = ((int)0x0C11),
IndexClearValue = ((int)0x0C20), **IndexWritemask** = ((int)0x0C21), **ColorClearValue** = ((int)0x0C22), **ColorWritemask** = ((int)0x0C23),
IndexMode = ((int)0x0C30), **RgbaMode** = ((int)0x0C31), **Doublebuffer** = ((int)0x0C32), **Stereo** = ((int)0x0C33),
RenderMode = ((int)0x0C40), **PerspectiveCorrectionHint** = ((int)0x0C50), **PointSmoothHint** = ((int)0x0C51), **LineSmoothHint** = ((int)0x0C52),
PolygonSmoothHint = ((int)0x0C53), **FogHint** = ((int)0x0C54), **TextureGenS** = ((int)0x0C60), **TextureGenT** = ((int)0x0C61),
TextureGenR = ((int)0x0C62), **TextureGenQ** = ((int)0x0C63), **PixelMapIToI** = ((int)0x0C70), **PixelMapSToS** = ((int)0x0C71),
PixelMapIToR = ((int)0x0C72), **PixelMapIToG** = ((int)0x0C73), **PixelMapIToB** = ((int)0x0C74), **PixelMapIToA** = ((int)0x0C75),
PixelMapRToR = ((int)0x0C76), **PixelMapGToG** = ((int)0x0C77), **PixelMapBToB** = ((int)0x0C78), **PixelMapAToA** = ((int)0x0C79),
PixelMapIToISize = ((int)0x0CB0), **PixelMapSToSSize** = ((int)0x0CB1), **PixelMapIToRSize** = ((int)0x0CB2), **PixelMapIToGSize** = ((int)0x0CB3),
PixelMapIToBSize = ((int)0x0CB4), **PixelMapIToASize** = ((int)0x0CB5), **PixelMapRToRSize** = ((int)0x0CB6), **PixelMapGToGSize** = ((int)0x0CB7),
PixelMapBToBSize = ((int)0x0CB8), **PixelMapAToASize** = ((int)0x0CB9), **UnpackSwapBytes** = ((int)0x0CF0), **UnpackLsbFirst** = ((int)0x0CF1),
UnpackRowLength = ((int)0x0CF2), **UnpackSkipRows** = ((int)0x0CF3), **UnpackSkipPixels** = ((int)0x0CF4), **UnpackAlignment** = ((int)0x0CF5),
PackSwapBytes = ((int)0x0D00), **PackLsbFirst** = ((int)0x0D01), **PackRowLength** = ((int)0x0D02), **PackSkipRows** = ((int)0x0D03),
PackSkipPixels = ((int)0x0D04), **PackAlignment** = ((int)0x0D05), **MapColor** = ((int)0x0D10), **MapStencil** = ((int)0x0D11),
IndexShift = ((int)0x0D12), **IndexOffset** = ((int)0x0D13), **RedScale** = ((int)0x0D14), **RedBias** = ((int)0x0D15),
ZoomX = ((int)0x0D16), **ZoomY** = ((int)0x0D17), **GreenScale** = ((int)0x0D18), **GreenBias** = ((int)0x0D19),
BlueScale = ((int)0x0D1A), **BlueBias** = ((int)0x0D1B), **AlphaScale** = ((int)0x0D1C), **AlphaBias** = ((int)0x0D1D),
DepthScale = ((int)0x0D1E), **DepthBias** = ((int)0x0D1F), **MaxEvalOrder** = ((int)0x0D30), **MaxLights** = ((int)0x0D31),
MaxClipDistances = ((int)0x0D32), **MaxClipPlanes** = ((int)0x0D32), **MaxTextureSize** = ((int)0x0D33), **MaxPixelMapTable** = ((int)0x0D34),

```

MaxAttribStackDepth = ((int)0x0D35), MaxModelviewStackDepth =
((int)0x0D36), MaxNameStackDepth = ((int)0x0D37), MaxProjectionStack-
Depth = ((int)0x0D38),
MaxTextureStackDepth = ((int)0x0D39), MaxViewportDims =
((int)0x0D3A), MaxClientAttribStackDepth = ((int)0x0D3B), Subpixel-
Bits = ((int)0x0D50),
IndexBits = ((int)0x0D51), RedBits = ((int)0x0D52), GreenBits =
((int)0x0D53), BlueBits = ((int)0x0D54),
AlphaBits = ((int)0x0D55), DepthBits = ((int)0x0D56), StencilBits =
((int)0x0D57), AccumRedBits = ((int)0x0D58),
AccumGreenBits = ((int)0x0D59), AccumBlueBits = ((int)0x0D5A), Ac-
cumAlphaBits = ((int)0x0D5B), NameStackDepth = ((int)0x0D70),
AutoNormal = ((int)0x0D80), Map1Color4 = ((int)0x0D90), Map1Index =
((int)0x0D91), Map1Normal = ((int)0x0D92),
Map1TextureCoord1 = ((int)0x0D93), Map1TextureCoord2 = ((int)0x0D94),
Map1TextureCoord3 = ((int)0x0D95), Map1TextureCoord4 = ((int)0x0D96),
Map1Vertex3 = ((int)0x0D97), Map1Vertex4 = ((int)0x0D98), Map2Color4 =
((int)0x0DB0), Map2Index = ((int)0x0DB1),
Map2Normal = ((int)0x0DB2), Map2TextureCoord1 = ((int)0x0DB3),
Map2TextureCoord2 = ((int)0x0DB4), Map2TextureCoord3 =
((int)0x0DB5),
Map2TextureCoord4 = ((int)0x0DB6), Map2Vertex3 = ((int)0x0DB7),
Map2Vertex4 = ((int)0x0DB8), Map1GridDomain = ((int)0x0DD0),
Map1GridSegments = ((int)0x0DD1), Map2GridDomain = ((int)0x0DD2),
Map2GridSegments = ((int)0x0DD3), Texture1D = ((int)0x0DE0),
Texture2D = ((int)0x0DE1), FeedbackBufferPointer = ((int)0x0DF0), Feed-
backBufferSize = ((int)0x0DF1), FeedbackBufferType = ((int)0x0DF2),
SelectionBufferPointer = ((int)0x0DF3), SelectionBufferSize =
((int)0x0DF4), TextureWidth = ((int)0x1000), Texcoord1BitPgi =
((int)0x10000000),
TextureHeight = ((int)0x1001), TextureComponents = ((int)0x1003), Tex-
tureInternalFormat = ((int)0x1003), TextureBorderColor = ((int)0x1004),
TextureBorder = ((int)0x1005), DontCare = ((int)0x1100), Fastest =
((int)0x1101), Nicest = ((int)0x1102),
Ambient = ((int)0x1200), Diffuse = ((int)0x1201), Specular = ((int)0x1202),
Position = ((int)0x1203),
SpotDirection = ((int)0x1204), SpotExponent = ((int)0x1205), SpotCutoff =
((int)0x1206), ConstantAttenuation = ((int)0x1207),
LinearAttenuation = ((int)0x1208), QuadraticAttenuation = ((int)0x1209),
Compile = ((int)0x1300), CompileAndExecute = ((int)0x1301),

```

Byte = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402),
UnsignedShort = ((int)0x1403),
Int = ((int)0x1404), **UnsignedInt** = ((int)0x1405), **Float** = ((int)0x1406),
Gl2Bytes = ((int)0x1407),
Gl3Bytes = ((int)0x1408), **Gl4Bytes** = ((int)0x1409), **Double** = ((int)0x140A),
DoubleExt = ((int)0x140A),
HalfApple = ((int)0x140B), **HalfFloat** = ((int)0x140B), **HalfFloatArb** =
((int)0x140B), **HalfFloatNv** = ((int)0x140B),
Clear = ((int)0x1500), **And** = ((int)0x1501), **AndReverse** = ((int)0x1502),
Copy = ((int)0x1503),
AndInverted = ((int)0x1504), **Noop** = ((int)0x1505), **Xor** = ((int)0x1506), **Or**
= ((int)0x1507),
Nor = ((int)0x1508), **Equiv** = ((int)0x1509), **Invert** = ((int)0x150A), **OrReverse**
= ((int)0x150B),
CopyInverted = ((int)0x150C), **OrInverted** = ((int)0x150D), **Nand** =
((int)0x150E), **Set** = ((int)0x150F),
Emission = ((int)0x1600), **Shininess** = ((int)0x1601), **AmbientAndDiffuse** =
((int)0x1602), **ColorIndexes** = ((int)0x1603),
Modelview = ((int)0x1700), **Modelview0Arb** = ((int)0x1700), **Modelview0Ext**
= ((int)0x1700), **Projection** = ((int)0x1701),
Texture = ((int)0x1702), **Color** = ((int)0x1800), **Depth** = ((int)0x1801), **Stencil**
= ((int)0x1802),
ColorIndex = ((int)0x1900), **StencilIndex** = ((int)0x1901), **DepthComponent**
= ((int)0x1902), **Red** = ((int)0x1903),
Green = ((int)0x1904), **Blue** = ((int)0x1905), **Alpha** = ((int)0x1906), **Rgb** =
((int)0x1907),
Rgba = ((int)0x1908), **Luminance** = ((int)0x1909), **LuminanceAlpha** =
((int)0x190A), **RasterPositionUnclippedIbm** = ((int)0x19262),
Bitmap = ((int)0x1A00), **PreferDoublebufferHintPgi** = ((int)0x1A1F8),
ConserveMemoryHintPgi = ((int)0x1A1FD), **ReclaimMemoryHintPgi** =
((int)0x1A1FE),
NativeGraphicsHandlePgi = ((int)0x1A202), **NativeGraphicsBeginHintPgi**
= ((int)0x1A203), **NativeGraphicsEndHintPgi** = ((int)0x1A204), **Al-**
waysFastHintPgi = ((int)0x1A20C),
AlwaysSoftHintPgi = ((int)0x1A20D), **AllowDrawObjHintPgi** =
((int)0x1A20E), **AllowDrawWinHintPgi** = ((int)0x1A20F), **AllowDrawFrgH-**
intPgi = ((int)0x1A210),
AllowDrawMemHintPgi = ((int)0x1A211), **StrictDepthfuncHintPgi**
= ((int)0x1A216), **StrictLightingHintPgi** = ((int)0x1A217), **StrictScis-**
sorHintPgi = ((int)0x1A218),

```

FullStippleHintPgi = ((int)0x1A219), ClipNearHintPgi = ((int)0x1A220),
ClipFarHintPgi = ((int)0x1A221), WideLineHintPgi = ((int)0x1A222),

BackNormalsHintPgi = ((int)0x1A223), VertexDataHintPgi =
((int)0x1A22A), VertexConsistentHintPgi = ((int)0x1A22B), Material-
SideHintPgi = ((int)0x1A22C),

MaxVertexHintPgi = ((int)0x1A22D), Point = ((int)0x1B00), Line =
((int)0x1B01), Fill = ((int)0x1B02),

Render = ((int)0x1C00), Feedback = ((int)0x1C01), Select = ((int)0x1C02),
Flat = ((int)0x1D00),

Smooth = ((int)0x1D01), Keep = ((int)0x1E00), Replace = ((int)0x1E01), Incr
= ((int)0x1E02),

Decr = ((int)0x1E03), Vendor = ((int)0x1F00), Renderer = ((int)0x1F01), Ver-
sion = ((int)0x1F02),

Extensions = ((int)0x1F03), S = ((int)0x2000), MultisampleBit =
((int)0x20000000), MultisampleBit3Dfx = ((int)0x20000000),

MultisampleBitArb = ((int)0x20000000), MultisampleBitExt =
((int)0x20000000), Texcoord2BitPgi = ((int)0x20000000), T = ((int)0x2001),

R = ((int)0x2002), Q = ((int)0x2003), Modulate = ((int)0x2100), Decal =
((int)0x2101),

TextureEnvMode = ((int)0x2200), TextureEnvColor = ((int)0x2201), Tex-
tureEnv = ((int)0x2300), EyeLinear = ((int)0x2400),

ObjectLinear = ((int)0x2401), SphereMap = ((int)0x2402), TextureGenMode
= ((int)0x2500), ObjectPlane = ((int)0x2501),

EyePlane = ((int)0x2502), Nearest = ((int)0x2600), Linear = ((int)0x2601),
NearestMipmapNearest = ((int)0x2700),

LinearMipmapNearest = ((int)0x2701), NearestMipmapLinear =
((int)0x2702), LinearMipmapLinear = ((int)0x2703), TextureMagFilter
= ((int)0x2800),

TextureMinFilter = ((int)0x2801), TextureWrapS = ((int)0x2802), Tex-
tureWrapT = ((int)0x2803), Clamp = ((int)0x2900),

Repeat = ((int)0x2901), PolygonOffsetUnits = ((int)0x2A00), PolygonOffset-
Point = ((int)0x2A01), PolygonOffsetLine = ((int)0x2A02),

R3G3B2 = ((int)0x2A10), V2f = ((int)0x2A20), V3f = ((int)0x2A21), C4ubV2f
= ((int)0x2A22),

C4ubV3f = ((int)0x2A23), C3fV3f = ((int)0x2A24), N3fV3f = ((int)0x2A25),
C4fN3fV3f = ((int)0x2A26),

T2fV3f = ((int)0x2A27), T4fV4f = ((int)0x2A28), T2fC4ubV3f =
((int)0x2A29), T2fC3fV3f = ((int)0x2A2A),

T2fN3fV3f = ((int)0x2A2B), T2fC4fN3fV3f = ((int)0x2A2C), T4fC4fN3fV4f
= ((int)0x2A2D), ClipDistance0 = ((int)0x3000),

```

ClipPlane0 = ((int)0x3000), **ClipDistance1** = ((int)0x3001), **ClipPlane1** = ((int)0x3001), **ClipDistance2** = ((int)0x3002),
ClipPlane2 = ((int)0x3002), **ClipDistance3** = ((int)0x3003), **ClipPlane3** = ((int)0x3003), **ClipDistance4** = ((int)0x3004),
ClipPlane4 = ((int)0x3004), **ClipDistance5** = ((int)0x3005), **ClipPlane5** = ((int)0x3005), **ClipDistance6** = ((int)0x3006),
ClipDistance7 = ((int)0x3007), **Light0** = ((int)0x4000), **Texcoord3BitPgi** = ((int)0x40000000), **Light1** = ((int)0x4001),
Light2 = ((int)0x4002), **Light3** = ((int)0x4003), **Light4** = ((int)0x4004), **Light5** = ((int)0x4005),
Light6 = ((int)0x4006), **Light7** = ((int)0x4007), **AbgrExt** = ((int)0x8000), **Texcoord4BitPgi** = unchecked((int)0x80000000),
ConstantColor = ((int)0x8001), **ConstantColorExt** = ((int)0x8001), **OneMinusConstantColor** = ((int)0x8002), **OneMinusConstantColorExt** = ((int)0x8002),
ConstantAlpha = ((int)0x8003), **ConstantAlphaExt** = ((int)0x8003), **OneMinusConstantAlpha** = ((int)0x8004), **OneMinusConstantAlphaExt** = ((int)0x8004),
BlendColor = ((int)0x8005), **BlendColorExt** = ((int)0x8005), **FuncAdd** = ((int)0x8006), **FuncAddExt** = ((int)0x8006),
Min = ((int)0x8007), **MinExt** = ((int)0x8007), **Max** = ((int)0x8008), **MaxExt** = ((int)0x8008),
BlendEquation = ((int)0x8009), **BlendEquationExt** = ((int)0x8009), **BlendEquationRgb** = ((int)0x8009), **BlendEquationRgbExt** = ((int)0x8009),
FuncSubtract = ((int)0x800A), **FuncSubtractExt** = ((int)0x800A), **FuncReverseSubtract** = ((int)0x800B), **FuncReverseSubtractExt** = ((int)0x800B),
CmykExt = ((int)0x800C), **CmykaExt** = ((int)0x800D), **PackCmykHintExt** = ((int)0x800E), **UnpackCmykHintExt** = ((int)0x800F),
Convolution1D = ((int)0x8010), **Convolution1DExt** = ((int)0x8010), **Convolution2D** = ((int)0x8011), **Convolution2DExt** = ((int)0x8011),
Separable2D = ((int)0x8012), **Separable2DExt** = ((int)0x8012), **ConvolutionBorderMode** = ((int)0x8013), **ConvolutionBorderModeExt** = ((int)0x8013),
ConvolutionFilterScale = ((int)0x8014), **ConvolutionFilterScaleExt** = ((int)0x8014), **ConvolutionFilterBias** = ((int)0x8015), **ConvolutionFilterBiasExt** = ((int)0x8015),
Reduce = ((int)0x8016), **ReduceExt** = ((int)0x8016), **ConvolutionFormat** = ((int)0x8017), **ConvolutionFormatExt** = ((int)0x8017),
ConvolutionWidth = ((int)0x8018), **ConvolutionWidthExt** = ((int)0x8018), **ConvolutionHeight** = ((int)0x8019), **ConvolutionHeightExt** = ((int)0x8019),

MaxConvolutionWidth = ((int)0x801A), **MaxConvolutionWidthExt** = ((int)0x801A), **MaxConvolutionHeight** = ((int)0x801B), **MaxConvolutionHeightExt** = ((int)0x801B),
PostConvolutionRedScale = ((int)0x801C), **PostConvolutionRedScaleExt** = ((int)0x801C), **PostConvolutionGreenScale** = ((int)0x801D), **PostConvolutionGreenScaleExt** = ((int)0x801D),
PostConvolutionBlueScale = ((int)0x801E), **PostConvolutionBlueScaleExt** = ((int)0x801E), **PostConvolutionAlphaScale** = ((int)0x801F), **PostConvolutionAlphaScaleExt** = ((int)0x801F),
PostConvolutionRedBias = ((int)0x8020), **PostConvolutionRedBiasExt** = ((int)0x8020), **PostConvolutionGreenBias** = ((int)0x8021), **PostConvolutionGreenBiasExt** = ((int)0x8021),
PostConvolutionBlueBias = ((int)0x8022), **PostConvolutionBlueBiasExt** = ((int)0x8022), **PostConvolutionAlphaBias** = ((int)0x8023), **PostConvolutionAlphaBiasExt** = ((int)0x8023),
Histogram = ((int)0x8024), **HistogramExt** = ((int)0x8024), **ProxyHistogram** = ((int)0x8025), **ProxyHistogramExt** = ((int)0x8025),
HistogramWidth = ((int)0x8026), **HistogramWidthExt** = ((int)0x8026), **HistogramFormat** = ((int)0x8027), **HistogramFormatExt** = ((int)0x8027),
HistogramRedSize = ((int)0x8028), **HistogramRedSizeExt** = ((int)0x8028), **HistogramGreenSize** = ((int)0x8029), **HistogramGreenSizeExt** = ((int)0x8029),
HistogramBlueSize = ((int)0x802A), **HistogramBlueSizeExt** = ((int)0x802A), **HistogramAlphaSize** = ((int)0x802B), **HistogramAlphaSizeExt** = ((int)0x802B),
HistogramLuminanceSize = ((int)0x802C), **HistogramLuminanceSizeExt** = ((int)0x802C), **HistogramSink** = ((int)0x802D), **HistogramSinkExt** = ((int)0x802D),
Minmax = ((int)0x802E), **MinmaxExt** = ((int)0x802E), **MinmaxFormat** = ((int)0x802F), **MinmaxFormatExt** = ((int)0x802F),
MinmaxSink = ((int)0x8030), **MinmaxSinkExt** = ((int)0x8030), **TableTooLarge** = ((int)0x8031), **TableTooLargeExt** = ((int)0x8031),
UnsignedByte332 = ((int)0x8032), **UnsignedByte332Ext** = ((int)0x8032), **UnsignedShort4444** = ((int)0x8033), **UnsignedShort4444Ext** = ((int)0x8033),
UnsignedShort5551 = ((int)0x8034), **UnsignedShort5551Ext** = ((int)0x8034), **UnsignedInt8888** = ((int)0x8035), **UnsignedInt8888Ext** = ((int)0x8035),
UnsignedInt1010102 = ((int)0x8036), **UnsignedInt1010102Ext** = ((int)0x8036), **PolygonOffsetExt** = ((int)0x8037), **PolygonOffsetFill** = ((int)0x8037),
PolygonOffsetFactor = ((int)0x8038), **PolygonOffsetFactorExt** = ((int)0x8038), **PolygonOffsetBiasExt** = ((int)0x8039), **RescaleNormal** = ((int)0x803A),

RescaleNormalExt = ((int)0x803A), **Alpha4** = ((int)0x803B), **Alpha4Ext** = ((int)0x803B), **Alpha8** = ((int)0x803C),
Alpha8Ext = ((int)0x803C), **Alpha12** = ((int)0x803D), **Alpha12Ext** = ((int)0x803D), **Alpha16** = ((int)0x803E),
Alpha16Ext = ((int)0x803E), **Luminance4** = ((int)0x803F), **Luminance4Ext** = ((int)0x803F), **Luminance8** = ((int)0x8040),
Luminance8Ext = ((int)0x8040), **Luminance12** = ((int)0x8041), **Luminance12Ext** = ((int)0x8041), **Luminance16** = ((int)0x8042),
Luminance16Ext = ((int)0x8042), **Luminance4Alpha4** = ((int)0x8043), **Luminance4Alpha4Ext** = ((int)0x8043), **Luminance6Alpha2** = ((int)0x8044),
Luminance6Alpha2Ext = ((int)0x8044), **Luminance8Alpha8** = ((int)0x8045), **Luminance8Alpha8Ext** = ((int)0x8045), **Luminance12Alpha4** = ((int)0x8046),
Luminance12Alpha4Ext = ((int)0x8046), **Luminance12Alpha12** = ((int)0x8047), **Luminance12Alpha12Ext** = ((int)0x8047), **Luminance16Alpha16** = ((int)0x8048),
Luminance16Alpha16Ext = ((int)0x8048), **Intensity** = ((int)0x8049), **IntensityExt** = ((int)0x8049), **Intensity4** = ((int)0x804A),
Intensity4Ext = ((int)0x804A), **Intensity8** = ((int)0x804B), **Intensity8Ext** = ((int)0x804B), **Intensity12** = ((int)0x804C),
Intensity12Ext = ((int)0x804C), **Intensity16** = ((int)0x804D), **Intensity16Ext** = ((int)0x804D), **Rgb2Ext** = ((int)0x804E),
Rgb4 = ((int)0x804F), **Rgb4Ext** = ((int)0x804F), **Rgb5** = ((int)0x8050), **Rgb5Ext** = ((int)0x8050),
Rgb8 = ((int)0x8051), **Rgb8Ext** = ((int)0x8051), **Rgb10** = ((int)0x8052), **Rgb10Ext** = ((int)0x8052),
Rgb12 = ((int)0x8053), **Rgb12Ext** = ((int)0x8053), **Rgb16** = ((int)0x8054), **Rgb16Ext** = ((int)0x8054),
Rgba2 = ((int)0x8055), **Rgba2Ext** = ((int)0x8055), **Rgba4** = ((int)0x8056), **Rgba4Ext** = ((int)0x8056),
Rgb5A1 = ((int)0x8057), **Rgb5A1Ext** = ((int)0x8057), **Rgba8** = ((int)0x8058), **Rgba8Ext** = ((int)0x8058),
Rgb10A2 = ((int)0x8059), **Rgb10A2Ext** = ((int)0x8059), **Rgba12** = ((int)0x805A), **Rgba12Ext** = ((int)0x805A),
Rgba16 = ((int)0x805B), **Rgba16Ext** = ((int)0x805B), **TextureRedSize** = ((int)0x805C), **TextureRedSizeExt** = ((int)0x805C),
TextureGreenSize = ((int)0x805D), **TextureGreenSizeExt** = ((int)0x805D), **TextureBlueSize** = ((int)0x805E), **TextureBlueSizeExt** = ((int)0x805E),
TextureAlphaSize = ((int)0x805F), **TextureAlphaSizeExt** = ((int)0x805F), **TextureLuminanceSize** = ((int)0x8060), **TextureLuminanceSizeExt** = ((int)0x8060),

```

TextureIntensitySize = ((int)0x8061), TextureIntensitySizeExt =
((int)0x8061), ReplaceExt = ((int)0x8062), ProxyTexture1D = ((int)0x8063),
ProxyTexture1DExt = ((int)0x8063), ProxyTexture2D = ((int)0x8064), ProxyTexture2DExt = ((int)0x8064), TextureTooLargeExt = ((int)0x8065),
TexturePriority = ((int)0x8066), TexturePriorityExt = ((int)0x8066), TextureResident = ((int)0x8067), TextureResidentExt = ((int)0x8067),
Texture1DBindingExt = ((int)0x8068), TextureBinding1D = ((int)0x8068),
Texture2DBindingExt = ((int)0x8069), TextureBinding2D = ((int)0x8069),
Texture3DBindingExt = ((int)0x806A), TextureBinding3D = ((int)0x806A),
PackSkipImages = ((int)0x806B), PackSkipImagesExt = ((int)0x806B),
PackImageHeight = ((int)0x806C), PackImageHeightExt = ((int)0x806C),
UnpackSkipImages = ((int)0x806D), UnpackSkipImagesExt = ((int)0x806D),
UnpackImageHeight = ((int)0x806E), UnpackImageHeightExt =
((int)0x806E), Texture3D = ((int)0x806F), Texture3DExt = ((int)0x806F),
ProxyTexture3D = ((int)0x8070), ProxyTexture3DExt = ((int)0x8070), TextureDepth = ((int)0x8071), TextureDepthExt = ((int)0x8071),
TextureWrapR = ((int)0x8072), TextureWrapRExt = ((int)0x8072),
Max3DTextureSize = ((int)0x8073), Max3DTextureSizeExt = ((int)0x8073),
VertexArray = ((int)0x8074), VertexArrayExt = ((int)0x8074), NormalArray =
((int)0x8075), NormalArrayExt = ((int)0x8075),
ColorArray = ((int)0x8076), ColorArrayExt = ((int)0x8076), IndexArray =
((int)0x8077), IndexArrayExt = ((int)0x8077),
TextureCoordArray = ((int)0x8078), TextureCoordArrayExt =
((int)0x8078), EdgeFlagArray = ((int)0x8079), EdgeFlagArrayExt =
((int)0x8079),
VertexArraySize = ((int)0x807A), VertexArraySizeExt = ((int)0x807A), VertexArrayType = ((int)0x807B), VertexArrayTypeExt = ((int)0x807B),
VertexArrayStride = ((int)0x807C), VertexArrayStrideExt = ((int)0x807C),
VertexArrayCountExt = ((int)0x807D), NormalArrayType = ((int)0x807E),
NormalArrayTypeExt = ((int)0x807E), NormalArrayStride = ((int)0x807F),
NormalArrayStrideExt = ((int)0x807F), NormalArrayCountExt =
((int)0x8080),
ColorArraySize = ((int)0x8081), ColorArraySizeExt = ((int)0x8081), ColorArrayType = ((int)0x8082), ColorArrayTypeExt = ((int)0x8082),
ColorArrayStride = ((int)0x8083), ColorArrayStrideExt = ((int)0x8083),
ColorArrayCountExt = ((int)0x8084), IndexArrayType = ((int)0x8085),
IndexArrayTypeExt = ((int)0x8085), IndexArrayStride = ((int)0x8086), IndexArrayStrideExt = ((int)0x8086), IndexArrayCountExt = ((int)0x8087),
TextureCoordArraySize = ((int)0x8088), TextureCoordArraySizeExt =
((int)0x8088), TextureCoordArrayType = ((int)0x8089), TextureCoordArrayTypeExt = ((int)0x8089),

```

TextureCoordArrayStride = ((int)0x808A), **TextureCoordArrayStrideExt** = ((int)0x808A), **TextureCoordArrayCountExt** = ((int)0x808B), **EdgeFlagArrayStride** = ((int)0x808C),

EdgeFlagArrayStrideExt = ((int)0x808C), **EdgeFlagArrayCountExt** = ((int)0x808D), **VertexArrayPointer** = ((int)0x808E), **VertexArrayPointerExt** = ((int)0x808E),

NormalArrayPointer = ((int)0x808F), **NormalArrayPointerExt** = ((int)0x808F), **ColorArrayPointer** = ((int)0x8090), **ColorArrayPointerExt** = ((int)0x8090),

IndexArrayPointer = ((int)0x8091), **IndexArrayPointerExt** = ((int)0x8091), **TextureCoordArrayPointer** = ((int)0x8092), **TextureCoordArrayPointerExt** = ((int)0x8092),

EdgeFlagArrayPointer = ((int)0x8093), **EdgeFlagArrayPointerExt** = ((int)0x8093), **InterlaceSgix** = ((int)0x8094), **DetailTexture2DSgis** = ((int)0x8095),

DetailTexture2DBindingSgis = ((int)0x8096), **LinearDetailSgis** = ((int)0x8097), **LinearDetailAlphaSgis** = ((int)0x8098), **LinearDetailColorSgis** = ((int)0x8099),

DetailTextureLevelSgis = ((int)0x809A), **DetailTextureModeSgis** = ((int)0x809B), **DetailTextureFuncPointsSgis** = ((int)0x809C), **Multisample** = ((int)0x809D),

MultisampleArb = ((int)0x809D), **MultisampleExt** = ((int)0x809D), **Multi-sampleSgis** = ((int)0x809D), **SampleAlphaToCoverage** = ((int)0x809E),

SampleAlphaToCoverageArb = ((int)0x809E), **SampleAlphaToMaskExt** = ((int)0x809E), **SampleAlphaToMaskSgis** = ((int)0x809E), **SampleAlphaToOne** = ((int)0x809F),

SampleAlphaToOneArb = ((int)0x809F), **SampleAlphaToOneExt** = ((int)0x809F), **SampleAlphaToOneSgis** = ((int)0x809F), **SampleCoverage** = ((int)0x80A0),

SampleCoverageArb = ((int)0x80A0), **SampleMaskExt** = ((int)0x80A0), **SampleMaskSgis** = ((int)0x80A0), **GL1PassExt** = ((int)0x80A1),

GL1PassSgis = ((int)0x80A1), **GL2Pass0Ext** = ((int)0x80A2), **GL2Pass0Sgis** = ((int)0x80A2), **GL2Pass1Ext** = ((int)0x80A3),

GL2Pass1Sgis = ((int)0x80A3), **GL4Pass0Ext** = ((int)0x80A4), **GL4Pass0Sgis** = ((int)0x80A4), **GL4Pass1Ext** = ((int)0x80A5),

GL4Pass1Sgis = ((int)0x80A5), **GL4Pass2Ext** = ((int)0x80A6), **GL4Pass2Sgis** = ((int)0x80A6), **GL4Pass3Ext** = ((int)0x80A7),

GL4Pass3Sgis = ((int)0x80A7), **SampleBuffers** = ((int)0x80A8), **SampleBuffersArb** = ((int)0x80A8), **SampleBuffersExt** = ((int)0x80A8),

SampleBuffersSgis = ((int)0x80A8), **Samples** = ((int)0x80A9), **SamplesArb** = ((int)0x80A9), **SamplesExt** = ((int)0x80A9),

```
SamplesSgis = ((int)0x80A9), SampleCoverageValue = ((int)0x80AA),  
SampleCoverageValueArb = ((int)0x80AA), SampleMaskValueExt =  
((int)0x80AA),  
SampleMaskValueSgis = ((int)0x80AA), SampleCoverageInvert =  
((int)0x80AB), SampleCoverageInvertArb = ((int)0x80AB), Sample-  
MaskInvertExt = ((int)0x80AB),  
SampleMaskInvertSgis = ((int)0x80AB), SamplePatternExt = ((int)0x80AC),  
SamplePatternSgis = ((int)0x80AC), LinearSharpenSgis = ((int)0x80AD),  
LinearSharpenAlphaSgis = ((int)0x80AE), LinearSharpenColorSgis =  
((int)0x80AF), SharpenTextureFuncPointsSgis = ((int)0x80B0), ColorMatrix  
= ((int)0x80B1),  
ColorMatrixSgi = ((int)0x80B1), ColorMatrixStackDepth = ((int)0x80B2),  
ColorMatrixStackDepthSgi = ((int)0x80B2), MaxColorMatrixStackDepth =  
((int)0x80B3),  
MaxColorMatrixStackDepthSgi = ((int)0x80B3), PostColorMatrixRedScale  
= ((int)0x80B4), PostColorMatrixRedScaleSgi = ((int)0x80B4), PostColor-  
MatrixGreenScale = ((int)0x80B5),  
PostColorMatrixGreenScaleSgi = ((int)0x80B5), PostColorMatrixBlueScale  
= ((int)0x80B6), PostColorMatrixBlueScaleSgi = ((int)0x80B6), PostColor-  
MatrixAlphaScale = ((int)0x80B7),  
PostColorMatrixAlphaScaleSgi = ((int)0x80B7), PostColorMatrixRedBias =  
((int)0x80B8), PostColorMatrixRedBiasSgi = ((int)0x80B8), PostColorMa-  
trixGreenBias = ((int)0x80B9),  
PostColorMatrixGreenBiasSgi = ((int)0x80B9), PostColorMatrixBlueBias =  
((int)0x80BA), PostColorMatrixBlueBiasSgi = ((int)0x80BA), PostColorMa-  
trixAlphaBias = ((int)0x80BB),  
PostColorMatrixAlphaBiasSgi = ((int)0x80BB), TextureColorTableSgi =  
((int)0x80BC), ProxyTextureColorTableSgi = ((int)0x80BD), TextureEnvBi-  
asSgix = ((int)0x80BE),  
ShadowAmbientSgix = ((int)0x80BF), TextureCompareFailValue =  
((int)0x80BF), TextureCompareFailValueArb = ((int)0x80BF), BlendDstRgb  
= ((int)0x80C8),  
BlendDstRgbExt = ((int)0x80C8), BlendSrcRgb = ((int)0x80C9), BlendSrc-  
cRgbExt = ((int)0x80C9), BlendDstAlpha = ((int)0x80CA),  
BlendDstAlphaExt = ((int)0x80CA), BlendSrcAlpha = ((int)0x80CB), Blend-  
SrcAlphaExt = ((int)0x80CB), GL422Ext = ((int)0x80CC),  
GL422RevExt = ((int)0x80CD), GL422AverageExt = ((int)0x80CE),  
GL422RevAverageExt = ((int)0x80CF), ColorTable = ((int)0x80D0),  
ColorTableSgi = ((int)0x80D0), PostConvolutionColorTable = ((int)0x80D1),  
PostConvolutionColorTableSgi = ((int)0x80D1), PostColorMatrixCol-  
orTable = ((int)0x80D2),
```

PostColorMatrixColorTableSgi = ((int)0x80D2), **ProxyColorTable** = ((int)0x80D3), **ProxyColorTableSgi** = ((int)0x80D3), **ProxyPostConvolutionColorTable** = ((int)0x80D4),

ProxyPostConvolutionColorTableSgi = ((int)0x80D4), **ProxyPostColorMatrixColorTable** = ((int)0x80D5), **ProxyPostColorMatrixColorTableSgi** = ((int)0x80D5), **ColorTableScale** = ((int)0x80D6),

ColorTableScaleSgi = ((int)0x80D6), **ColorTableBias** = ((int)0x80D7), **ColorTableBiasSgi** = ((int)0x80D7), **ColorTableFormat** = ((int)0x80D8),

ColorTableFormatSgi = ((int)0x80D8), **ColorTableWidth** = ((int)0x80D9), **ColorTableWidthSgi** = ((int)0x80D9), **ColorTableRedSize** = ((int)0x80DA),

ColorTableRedSizeSgi = ((int)0x80DA), **ColorTableGreenSize** = ((int)0x80DB), **ColorTableGreenSizeSgi** = ((int)0x80DB), **ColorTableBlueSize** = ((int)0x80DC),

ColorTableBlueSizeSgi = ((int)0x80DC), **ColorTableAlphaSize** = ((int)0x80DD), **ColorTableAlphaSizeSgi** = ((int)0x80DD), **ColorTableLuminanceSize** = ((int)0x80DE),

ColorTableLuminanceSizeSgi = ((int)0x80DE), **ColorTableIntensitySize** = ((int)0x80DF), **ColorTableIntensitySizeSgi** = ((int)0x80DF), **Bgr** = ((int)0x80E0),

BgrExt = ((int)0x80E0), **Bgra** = ((int)0x80E1), **BgraExt** = ((int)0x80E1), **ColorIndex1Ext** = ((int)0x80E2),

ColorIndex2Ext = ((int)0x80E3), **ColorIndex4Ext** = ((int)0x80E4), **ColorIndex8Ext** = ((int)0x80E5), **ColorIndex12Ext** = ((int)0x80E6),

ColorIndex16Ext = ((int)0x80E7), **MaxElementsVertices** = ((int)0x80E8), **MaxElementsVerticesExt** = ((int)0x80E8), **MaxElementsIndices** = ((int)0x80E9),

MaxElementsIndicesExt = ((int)0x80E9), **PhongWin** = ((int)0x80EA), **PhongHintWin** = ((int)0x80EB), **FogSpecularTextureWin** = ((int)0x80EC),

TextureIndexSizeExt = ((int)0x80ED), **ClipVolumeClippingHintExt** = ((int)0x80F0), **DualAlpha4Sgis** = ((int)0x8110), **DualAlpha8Sgis** = ((int)0x8111),

DualAlpha12Sgis = ((int)0x8112), **DualAlpha16Sgis** = ((int)0x8113), **DualLuminance4Sgis** = ((int)0x8114), **DualLuminance8Sgis** = ((int)0x8115),

DualLuminance12Sgis = ((int)0x8116), **DualLuminance16Sgis** = ((int)0x8117), **DualIntensity4Sgis** = ((int)0x8118), **DualIntensity8Sgis** = ((int)0x8119),

DualIntensity12Sgis = ((int)0x811A), **DualIntensity16Sgis** = ((int)0x811B), **DualLuminanceAlpha4Sgis** = ((int)0x811C), **DualLuminanceAlpha8Sgis** = ((int)0x811D),

QuadAlpha4Sgis = ((int)0x811E), **QuadAlpha8Sgis** = ((int)0x811F), **QuadLuminance4Sgis** = ((int)0x8120), **QuadLuminance8Sgis** = ((int)0x8121),

QuadIntensity4Sgis = ((int)0x8122), **QuadIntensity8Sgis** = ((int)0x8123), **DualTextureSelectSgis** = ((int)0x8124), **QuadTextureSelectSgis** = ((int)0x8125),
PointSizeMin = ((int)0x8126), **PointSizeMinArb** = ((int)0x8126), **PointSizeMinExt** = ((int)0x8126), **PointSizeMinSgis** = ((int)0x8126),
PointSizeMax = ((int)0x8127), **PointSizeMaxArb** = ((int)0x8127), **PointSizeMaxExt** = ((int)0x8127), **PointSizeMaxSgis** = ((int)0x8127),
PointFadeThresholdSize = ((int)0x8128), **PointFadeThresholdSizeArb** = ((int)0x8128), **PointFadeThresholdSizeExt** = ((int)0x8128), **PointFadeThresholdSizeSgis** = ((int)0x8128),
DistanceAttenuationExt = ((int)0x8129), **DistanceAttenuationSgis** = ((int)0x8129), **PointDistanceAttenuation** = ((int)0x8129), **PointDistanceAttenuationArb** = ((int)0x8129),
FogFuncSgis = ((int)0x812A), **FogFuncPointsSgis** = ((int)0x812B), **MaxFogFuncPointsSgis** = ((int)0x812C), **ClampToBorder** = ((int)0x812D),
ClampToBorderArb = ((int)0x812D), **ClampToBorderSgis** = ((int)0x812D), **TextureMultiBufferHintSgis** = ((int)0x812E), **ClampToEdge** = ((int)0x812F),
ClampToEdgeSgis = ((int)0x812F), **PackSkipVolumesSgis** = ((int)0x8130), **PackImageDepthSgis** = ((int)0x8131), **UnpackSkipVolumesSgis** = ((int)0x8132),
UnpackImageDepthSgis = ((int)0x8133), **Texture4DSgis** = ((int)0x8134), **ProxyTexture4DSgis** = ((int)0x8135), **Texture4DsizeSgis** = ((int)0x8136),
TextureWrapQSgis = ((int)0x8137), **Max4DTextureSizeSgis** = ((int)0x8138), **PixelTexGenSgis** = ((int)0x8139), **TextureMinLod** = ((int)0x813A),
TextureMinLodSgis = ((int)0x813A), **TextureMaxLod** = ((int)0x813B), **TextureMaxLodSgis** = ((int)0x813B), **TextureBaseLevel** = ((int)0x813C),
TextureBaseLevelSgis = ((int)0x813C), **TextureMaxLevel** = ((int)0x813D), **TextureMaxLevelSgis** = ((int)0x813D), **PixelTileBestAlignmentSgix** = ((int)0x813E),
PixelTileCacheIncrementSgix = ((int)0x813F), **PixelTileWidthSgix** = ((int)0x8140), **PixelTileHeightSgix** = ((int)0x8141), **PixelTileGridWidthSgix** = ((int)0x8142),
PixelTileGridHeightSgix = ((int)0x8143), **PixelTileGridDepthSgix** = ((int)0x8144), **PixelTileCacheSizeSgix** = ((int)0x8145), **Filter4Sgis** = ((int)0x8146),
TextureFilter4SizeSgis = ((int)0x8147), **SpriteSgix** = ((int)0x8148), **SpriteModeSgix** = ((int)0x8149), **SpriteAxisSgix** = ((int)0x814A),
SpriteTranslationSgix = ((int)0x814B), **SpriteAxialSgix** = ((int)0x814C), **SpriteObjectAlignedSgix** = ((int)0x814D), **SpriteEyeAlignedSgix** = ((int)0x814E),
Texture4DBindingSgis = ((int)0x814F), **IgnoreBorderHp** = ((int)0x8150), **ConstantBorder** = ((int)0x8151), **ConstantBorderHp** = ((int)0x8151),

ReplicateBorder = ((int)0x8153), **ReplicateBorderHp** = ((int)0x8153),
ConvolutionBorderColor = ((int)0x8154), **ConvolutionBorderColorHp** = ((int)0x8154),

ImageScaleXHp = ((int)0x8155), **ImageScaleYHp** = ((int)0x8156), **ImageTranslateXHp** = ((int)0x8157), **ImageTranslateYHp** = ((int)0x8158),

ImageRotateAngleHp = ((int)0x8159), **ImageRotateOriginXHp** = ((int)0x815A), **ImageRotateOriginYHp** = ((int)0x815B), **ImageMagFilterHp** = ((int)0x815C),

ImageMinFilterHp = ((int)0x815D), **ImageCubicWeightHp** = ((int)0x815E), **CubicHp** = ((int)0x815F), **AverageHp** = ((int)0x8160),

ImageTransform2DHp = ((int)0x8161), **PostImageTransformColorTableHp** = ((int)0x8162), **ProxyPostImageTransformColorTableHp** = ((int)0x8163), **OcclusionTestHp** = ((int)0x8165),

OcclusionTestResultHp = ((int)0x8166), **TextureLightingModeHp** = ((int)0x8167), **TexturePostSpecularHp** = ((int)0x8168), **TexturePreSpecularHp** = ((int)0x8169),

LinearClipmapLinearSgix = ((int)0x8170), **TextureClipmapCenterSgix** = ((int)0x8171), **TextureClipmapFrameSgix** = ((int)0x8172), **TextureClipmapOffsetSgix** = ((int)0x8173),

TextureClipmapVirtualDepthSgix = ((int)0x8174), **TextureClipmapLodOffsetSgix** = ((int)0x8175), **TextureClipmapDepthSgix** = ((int)0x8176), **MaxClipmapDepthSgix** = ((int)0x8177),

MaxClipmapVirtualDepthSgix = ((int)0x8178), **PostTextureFilterBiasSgix** = ((int)0x8179), **PostTextureFilterScaleSgix** = ((int)0x817A), **PostTextureFilterBiasRangeSgix** = ((int)0x817B),

PostTextureFilterScaleRangeSgix = ((int)0x817C), **ReferencePlaneSgix** = ((int)0x817D), **ReferencePlaneEquationSgix** = ((int)0x817E), **IrInstrument1Sgix** = ((int)0x817F),

InstrumentBufferPointerSgix = ((int)0x8180), **InstrumentMeasurementsSgix** = ((int)0x8181), **ListPrioritySgix** = ((int)0x8182), **CalligraphicFragmentSgix** = ((int)0x8183),

PixelTexGenQCeilingSgix = ((int)0x8184), **PixelTexGenQRoundSgix** = ((int)0x8185), **PixelTexGenQFloorSgix** = ((int)0x8186), **PixelTexGenAlphaReplaceSgix** = ((int)0x8187),

PixelTexGenAlphaNoReplaceSgix = ((int)0x8188), **PixelTexGenAlphaLsSgix** = ((int)0x8189), **PixelTexGenAlphaMsSgix** = ((int)0x818A), **FramezoomSgix** = ((int)0x818B),

FramezoomFactorSgix = ((int)0x818C), **MaxFramezoomFactorSgix** = ((int)0x818D), **TextureLodBiasSSgix** = ((int)0x818E), **TextureLodBiasTSgix** = ((int)0x818F),


```

TextureLodBiasRSgix = ((int)0x8190), GenerateMipmap = ((int)0x8191),
GenerateMipmapSgix = ((int)0x8191), GenerateMipmapHint =
((int)0x8192),
GenerateMipmapHintSgix = ((int)0x8192), GeometryDeformationSgix =
((int)0x8194), TextureDeformationSgix = ((int)0x8195), Deformations-
MaskSgix = ((int)0x8196),
MaxDeformationOrderSgix = ((int)0x8197), FogOffsetSgix = ((int)0x8198),
FogOffsetValueSgix = ((int)0x8199), TextureCompareSgix = ((int)0x819A),
TextureCompareOperatorSgix = ((int)0x819B), TextureLequalRSgix =
((int)0x819C), TextureGequalRSgix = ((int)0x819D), DepthComponent16 =
((int)0x81A5),
DepthComponent16Arb = ((int)0x81A5), DepthComponent16Sgix =
((int)0x81A5), DepthComponent24 = ((int)0x81A6), DepthCompo-
nent24Arb = ((int)0x81A6),
DepthComponent24Sgix = ((int)0x81A6), DepthComponent32 =
((int)0x81A7), DepthComponent32Arb = ((int)0x81A7), DepthCompo-
nent32Sgix = ((int)0x81A7),
ArrayElementLockFirstExt = ((int)0x81A8), ArrayElementLockCountExt =
((int)0x81A9), CullVertexExt = ((int)0x81AA), CullVertexEyePositionExt =
((int)0x81AB),
CullVertexObjectPositionExt = ((int)0x81AC), IuiV2fExt = ((int)0x81AD),
IuiV3fExt = ((int)0x81AE), IuiN3fV2fExt = ((int)0x81AF),
IuiN3fV3fExt = ((int)0x81B0), T2fIuiV2fExt = ((int)0x81B1), T2fIuiV3fExt =
((int)0x81B2), T2fIuiN3fV2fExt = ((int)0x81B3),
T2fIuiN3fV3fExt = ((int)0x81B4), IndexTestExt = ((int)0x81B5), IndexTest-
FuncExt = ((int)0x81B6), IndexTestRefExt = ((int)0x81B7),
IndexMaterialExt = ((int)0x81B8), IndexMaterialParameterExt =
((int)0x81B9), IndexMaterialFaceExt = ((int)0x81BA), YcrCb422Sgix =
((int)0x81BB),
YcrCb444Sgix = ((int)0x81BC), WrapBorderSun = ((int)0x81D4), Un-
packConstantDataSunx = ((int)0x81D5), TextureConstantDataSunx =
((int)0x81D6),
TriangleListSun = ((int)0x81D7), ReplacementCodeSun = ((int)0x81D8),
GlobalAlphaSun = ((int)0x81D9), GlobalAlphaFactorSun = ((int)0x81DA),
TextureColorWritemaskSgix = ((int)0x81EF), EyeDistanceToPointSgix =
((int)0x81F0), ObjectDistanceToPointSgix = ((int)0x81F1), EyeDistanceToLi-
neSgix = ((int)0x81F2),
ObjectDistanceToLineSgix = ((int)0x81F3), EyePointSgix = ((int)0x81F4),
ObjectPointSgix = ((int)0x81F5), EyeLineSgix = ((int)0x81F6),
ObjectLineSgix = ((int)0x81F7), LightModelColorControl = ((int)0x81F8),
LightModelColorControlExt = ((int)0x81F8), SingleColor = ((int)0x81F9),

```

SingleColorExt = ((int)0x81F9), **SeparateSpecularColor** = ((int)0x81FA), **SeparateSpecularColorExt** = ((int)0x81FA), **SharedTexturePaletteExt** = ((int)0x81FB),

FogScaleSgix = ((int)0x81FC), **FogScaleValueSgix** = ((int)0x81FD), **TextFragmentShaderAti** = ((int)0x8200), **FramebufferAttachmentColorEncoding** = ((int)0x8210),

FramebufferAttachmentComponentType = ((int)0x8211), **FramebufferAttachmentRedSize** = ((int)0x8212), **FramebufferAttachmentGreenSize** = ((int)0x8213), **FramebufferAttachmentBlueSize** = ((int)0x8214),

FramebufferAttachmentAlphaSize = ((int)0x8215), **FramebufferAttachmentDepthSize** = ((int)0x8216), **FramebufferAttachmentStencilSize** = ((int)0x8217), **FramebufferDefault** = ((int)0x8218),

FramebufferUndefined = ((int)0x8219), **DepthStencilAttachment** = ((int)0x821A), **MajorVersion** = ((int)0x821B), **MinorVersion** = ((int)0x821C),

NumExtensions = ((int)0x821D), **ContextFlags** = ((int)0x821E), **Index** = ((int)0x8222), **DepthBuffer** = ((int)0x8223),

StencilBuffer = ((int)0x8224), **CompressedRed** = ((int)0x8225), **CompressedRg** = ((int)0x8226), **Rg** = ((int)0x8227),

RgInteger = ((int)0x8228), **R8** = ((int)0x8229), **R16** = ((int)0x822A), **Rg8** = ((int)0x822B),

Rg16 = ((int)0x822C), **R16f** = ((int)0x822D), **R32f** = ((int)0x822E), **Rg16f** = ((int)0x822F),

Rg32f = ((int)0x8230), **R8i** = ((int)0x8231), **R8ui** = ((int)0x8232), **R16i** = ((int)0x8233),

R16ui = ((int)0x8234), **R32i** = ((int)0x8235), **R32ui** = ((int)0x8236), **Rg8i** = ((int)0x8237),

Rg8ui = ((int)0x8238), **Rg16i** = ((int)0x8239), **Rg16ui** = ((int)0x823A), **Rg32i** = ((int)0x823B),

Rg32ui = ((int)0x823C), **DepthPassInstrumentSgix** = ((int)0x8310), **DepthPassInstrumentCountersSgix** = ((int)0x8311), **DepthPassInstrumentMaxSgix** = ((int)0x8312),

ConvolutionHintSgix = ((int)0x8316), **YcrcbSgix** = ((int)0x8318), **YcrcbaSgix** = ((int)0x8319), **AlphaMinSgix** = ((int)0x8320),

AlphaMaxSgix = ((int)0x8321), **ScalebiasHintSgix** = ((int)0x8322), **AsyncMarkerSgix** = ((int)0x8329), **PixelTexGenModeSgix** = ((int)0x832B),

AsyncHistogramSgix = ((int)0x832C), **MaxAsyncHistogramSgix** = ((int)0x832D), **PixelTransform2DExt** = ((int)0x8330), **PixelMagFilterExt** = ((int)0x8331),

PixelMinFilterExt = ((int)0x8332), **PixelCubicWeightExt** = ((int)0x8333), **CubicExt** = ((int)0x8334), **AverageExt** = ((int)0x8335),

PixelTransform2DStackDepthExt = ((int)0x8336), **MaxPixelTransform2DStackDepthExt** = ((int)0x8337), **PixelTransform2DMatrixExt** = ((int)0x8338), **FragmentMaterialExt** = ((int)0x8349),
FragmentNormalExt = ((int)0x834A), **FragmentColorExt** = ((int)0x834C), **AttenuationExt** = ((int)0x834D), **ShadowAttenuationExt** = ((int)0x834E),
TextureApplicationModeExt = ((int)0x834F), **TextureLightExt** = ((int)0x8350), **TextureMaterialFaceExt** = ((int)0x8351), **TextureMaterialParameterExt** = ((int)0x8352),
PixelTextureSgis = ((int)0x8353), **PixelFragmentRgbSourceSgis** = ((int)0x8354), **PixelFragmentAlphaSourceSgis** = ((int)0x8355), **PixelGroupColorSgis** = ((int)0x8356),
AsyncTexImageSgix = ((int)0x835C), **AsyncDrawPixelsSgix** = ((int)0x835D), **AsyncReadPixelsSgix** = ((int)0x835E), **MaxAsyncTexImageSgix** = ((int)0x835F),
MaxAsyncDrawPixelsSgix = ((int)0x8360), **MaxAsyncReadPixelsSgix** = ((int)0x8361), **UnsignedByte233Rev** = ((int)0x8362), **UnsignedByte233Reversed** = ((int)0x8362),
UnsignedByte233RevExt = ((int)0x8362), **UnsignedShort565** = ((int)0x8363), **UnsignedShort565Ext** = ((int)0x8363), **UnsignedShort565Rev** = ((int)0x8364),
UnsignedShort565Reversed = ((int)0x8364), **UnsignedShort565RevExt** = ((int)0x8364), **UnsignedShort4444Rev** = ((int)0x8365), **UnsignedShort4444Reversed** = ((int)0x8365),
UnsignedShort4444RevExt = ((int)0x8365), **UnsignedShort1555Rev** = ((int)0x8366), **UnsignedShort1555Reversed** = ((int)0x8366), **UnsignedShort1555RevExt** = ((int)0x8366),
UnsignedInt8888Rev = ((int)0x8367), **UnsignedInt8888Reversed** = ((int)0x8367), **UnsignedInt8888RevExt** = ((int)0x8367), **UnsignedInt2101010Rev** = ((int)0x8368),
UnsignedInt2101010Reversed = ((int)0x8368), **UnsignedInt2101010RevExt** = ((int)0x8368), **TextureMaxClampSSgix** = ((int)0x8369), **TextureMaxClampTSgix** = ((int)0x836A),
TextureMaxClampRSgix = ((int)0x836B), **FogFactorToAlphaSgix** = ((int)0x836F), **MirroredRepeat** = ((int)0x8370), **MirroredRepeatArb** = ((int)0x8370),
MirroredRepeatIbm = ((int)0x8370), **RgbS3tc** = ((int)0x83A0), **Rgb4S3tc** = ((int)0x83A1), **RgbaS3tc** = ((int)0x83A2),
Rgba4S3tc = ((int)0x83A3), **VertexPreclipSgix** = ((int)0x83EE), **VertexPreclipHintSgix** = ((int)0x83EF), **CompressedRgbS3tcDxt1Ext** = ((int)0x83F0),
CompressedRgbaS3tcDxt1Ext = ((int)0x83F1), **CompressedRgbaS3tcDxt3Ext** = ((int)0x83F2), **CompressedRgbaS3tcDxt5Ext** = ((int)0x83F3), **ParallelArraysIntel** = ((int)0x83F4),

VertexArrayParallelPointersIntel = ((int)0x83F5), **NormalArrayParallelPointersIntel** = ((int)0x83F6), **ColorArrayParallelPointersIntel** = ((int)0x83F7), **TextureCoordArrayParallelPointersIntel** = ((int)0x83F8),
FragmentLightingSgix = ((int)0x8400), **FragmentColorMaterialSgix** = ((int)0x8401), **FragmentColorMaterialFaceSgix** = ((int)0x8402), **FragmentColorMaterialParameterSgix** = ((int)0x8403),
MaxFragmentLightsSgix = ((int)0x8404), **MaxActiveLightsSgix** = ((int)0x8405), **CurrentRasterNormalSgix** = ((int)0x8406), **LightEnvModeSgix** = ((int)0x8407),
FragmentLightModelLocalViewerSgix = ((int)0x8408), **FragmentLightModelTwoSideSgix** = ((int)0x8409), **FragmentLightModelAmbientSgix** = ((int)0x840A), **FragmentLightModelNormalInterpolationSgix** = ((int)0x840B),
FragmentLight0Sgix = ((int)0x840C), **FragmentLight1Sgix** = ((int)0x840D), **FragmentLight2Sgix** = ((int)0x840E), **FragmentLight3Sgix** = ((int)0x840F),
FragmentLight4Sgix = ((int)0x8410), **FragmentLight5Sgix** = ((int)0x8411), **FragmentLight6Sgix** = ((int)0x8412), **FragmentLight7Sgix** = ((int)0x8413),
PackResampleSgix = ((int)0x842C), **UnpackResampleSgix** = ((int)0x842D), **ResampleReplicateSgix** = ((int)0x842E), **ResampleZeroFillSgix** = ((int)0x842F),
ResampleDecimateSgix = ((int)0x8430), **TangentArrayExt** = ((int)0x8439), **BinormalArrayExt** = ((int)0x843A), **CurrentTangentExt** = ((int)0x843B),
CurrentBinormalExt = ((int)0x843C), **TangentArrayTypeExt** = ((int)0x843E), **TangentArrayStrideExt** = ((int)0x843F), **BinormalArrayTypeExt** = ((int)0x8440),
BinormalArrayStrideExt = ((int)0x8441), **TangentArrayPointerExt** = ((int)0x8442), **BinormalArrayPointerExt** = ((int)0x8443), **Map1TangentExt** = ((int)0x8444),
Map2TangentExt = ((int)0x8445), **Map1BinormalExt** = ((int)0x8446), **Map2BinormalExt** = ((int)0x8447), **NearestClipmapNearestSgix** = ((int)0x844D),
NearestClipmapLinearSgix = ((int)0x844E), **LinearClipmapNearestSgix** = ((int)0x844F), **FogCoordinateSource** = ((int)0x8450), **FogCoordinateSourceExt** = ((int)0x8450),
FogCoordSrc = ((int)0x8450), **FogCoord** = ((int)0x8451), **FogCoordinate** = ((int)0x8451), **FogCoordinateExt** = ((int)0x8451),
FragmentDepth = ((int)0x8452), **FragmentDepthExt** = ((int)0x8452), **CurrentFogCoord** = ((int)0x8453), **CurrentFogCoordinate** = ((int)0x8453),
CurrentFogCoordinateExt = ((int)0x8453), **FogCoordArrayType** = ((int)0x8454), **FogCoordinateArrayType** = ((int)0x8454), **FogCoordinateArrayTypeExt** = ((int)0x8454),

FogCoordArrayStride = ((int)0x8455), **FogCoordinateArrayStride** = ((int)0x8455), **FogCoordinateArrayStrideExt** = ((int)0x8455), **FogCoordArrayPointer** = ((int)0x8456),

FogCoordinateArrayPointer = ((int)0x8456), **FogCoordinateArrayPointerExt** = ((int)0x8456), **FogCoordArray** = ((int)0x8457), **FogCoordinateArray** = ((int)0x8457),

FogCoordinateArrayExt = ((int)0x8457), **ColorSum** = ((int)0x8458), **ColorSumArb** = ((int)0x8458), **ColorSumExt** = ((int)0x8458),

CurrentSecondaryColor = ((int)0x8459), **CurrentSecondaryColorExt** = ((int)0x8459), **SecondaryColorArraySize** = ((int)0x845A), **SecondaryColorArraySizeExt** = ((int)0x845A),

SecondaryColorArrayType = ((int)0x845B), **SecondaryColorArrayTypeExt** = ((int)0x845B), **SecondaryColorArrayStride** = ((int)0x845C), **SecondaryColorArrayStrideExt** = ((int)0x845C),

SecondaryColorArrayPointer = ((int)0x845D), **SecondaryColorArrayPointerExt** = ((int)0x845D), **SecondaryColorArray** = ((int)0x845E), **SecondaryColorArrayExt** = ((int)0x845E),

CurrentRasterSecondaryColor = ((int)0x845F), **AliasedPointSizeRange** = ((int)0x846D), **AliasedLineWidthRange** = ((int)0x846E), **ScreenCoordinatesRend** = ((int)0x8490),

InvertedScreenWRend = ((int)0x8491), **Texture0** = ((int)0x84C0), **Texture0Arb** = ((int)0x84C0), **Texture1** = ((int)0x84C1),

Texture1Arb = ((int)0x84C1), **Texture2** = ((int)0x84C2), **Texture2Arb** = ((int)0x84C2), **Texture3** = ((int)0x84C3),

Texture3Arb = ((int)0x84C3), **Texture4** = ((int)0x84C4), **Texture4Arb** = ((int)0x84C4), **Texture5** = ((int)0x84C5),

Texture5Arb = ((int)0x84C5), **Texture6** = ((int)0x84C6), **Texture6Arb** = ((int)0x84C6), **Texture7** = ((int)0x84C7),

Texture7Arb = ((int)0x84C7), **Texture8** = ((int)0x84C8), **Texture8Arb** = ((int)0x84C8), **Texture9** = ((int)0x84C9),

Texture9Arb = ((int)0x84C9), **Texture10** = ((int)0x84CA), **Texture10Arb** = ((int)0x84CA), **Texture11** = ((int)0x84CB),

Texture11Arb = ((int)0x84CB), **Texture12** = ((int)0x84CC), **Texture12Arb** = ((int)0x84CC), **Texture13** = ((int)0x84CD),

Texture13Arb = ((int)0x84CD), **Texture14** = ((int)0x84CE), **Texture14Arb** = ((int)0x84CE), **Texture15** = ((int)0x84CF),

Texture15Arb = ((int)0x84CF), **Texture16** = ((int)0x84D0), **Texture16Arb** = ((int)0x84D0), **Texture17** = ((int)0x84D1),

Texture17Arb = ((int)0x84D1), **Texture18** = ((int)0x84D2), **Texture18Arb** = ((int)0x84D2), **Texture19** = ((int)0x84D3),

Texture19Arb = ((int)0x84D3), **Texture20** = ((int)0x84D4), **Texture20Arb** = ((int)0x84D4), **Texture21** = ((int)0x84D5),
Texture21Arb = ((int)0x84D5), **Texture22** = ((int)0x84D6), **Texture22Arb** = ((int)0x84D6), **Texture23** = ((int)0x84D7),
Texture23Arb = ((int)0x84D7), **Texture24** = ((int)0x84D8), **Texture24Arb** = ((int)0x84D8), **Texture25** = ((int)0x84D9),
Texture25Arb = ((int)0x84D9), **Texture26** = ((int)0x84DA), **Texture26Arb** = ((int)0x84DA), **Texture27** = ((int)0x84DB),
Texture27Arb = ((int)0x84DB), **Texture28** = ((int)0x84DC), **Texture28Arb** = ((int)0x84DC), **Texture29** = ((int)0x84DD),
Texture29Arb = ((int)0x84DD), **Texture30** = ((int)0x84DE), **Texture30Arb** = ((int)0x84DE), **Texture31** = ((int)0x84DF),
Texture31Arb = ((int)0x84DF), **ActiveTexture** = ((int)0x84E0), **ActiveTextureArb** = ((int)0x84E0), **ClientActiveTexture** = ((int)0x84E1),
ClientActiveTextureArb = ((int)0x84E1), **MaxTextureUnits** = ((int)0x84E2), **MaxTextureUnitsArb** = ((int)0x84E2), **TransposeModelviewMatrix** = ((int)0x84E3),
TransposeModelviewMatrixArb = ((int)0x84E3), **TransposeProjectionMatrix** = ((int)0x84E4), **TransposeProjectionMatrixArb** = ((int)0x84E4), **TransposeTextureMatrix** = ((int)0x84E5),
TransposeTextureMatrixArb = ((int)0x84E5), **TransposeColorMatrix** = ((int)0x84E6), **TransposeColorMatrixArb** = ((int)0x84E6), **Subtract** = ((int)0x84E7),
SubtractArb = ((int)0x84E7), **MaxRenderbufferSize** = ((int)0x84E8), **MaxRenderbufferSizeExt** = ((int)0x84E8), **CompressedAlpha** = ((int)0x84E9),
CompressedAlphaArb = ((int)0x84E9), **CompressedLuminance** = ((int)0x84EA), **CompressedLuminanceArb** = ((int)0x84EA), **CompressedLuminanceAlpha** = ((int)0x84EB),
CompressedLuminanceAlphaArb = ((int)0x84EB), **CompressedIntensity** = ((int)0x84EC), **CompressedIntensityArb** = ((int)0x84EC), **CompressedRgb** = ((int)0x84ED),
CompressedRgbArb = ((int)0x84ED), **CompressedRgba** = ((int)0x84EE), **CompressedRgbaArb** = ((int)0x84EE), **TextureCompressionHint** = ((int)0x84EF),
TextureCompressionHintArb = ((int)0x84EF), **AllCompletedNv** = ((int)0x84F2), **FenceStatusNv** = ((int)0x84F3), **FenceConditionNv** = ((int)0x84F4),
TextureRectangle = ((int)0x84F5), **TextureRectangleArb** = ((int)0x84F5), **TextureRectangleNv** = ((int)0x84F5), **TextureBindingRectangle** = ((int)0x84F6),

TextureBindingRectangleArb = ((int)0x84F6), **TextureBindingRectangleNv** = ((int)0x84F6), **ProxyTextureRectangle** = ((int)0x84F7), **ProxyTextureRectangleArb** = ((int)0x84F7),

ProxyTextureRectangleNv = ((int)0x84F7), **MaxRectangleTextureSize** = ((int)0x84F8), **MaxRectangleTextureSizeArb** = ((int)0x84F8), **MaxRectangleTextureSizeNv** = ((int)0x84F8),

DepthStencil = ((int)0x84F9), **DepthStencilExt** = ((int)0x84F9), **DepthStencilNv** = ((int)0x84F9), **UnsignedInt248** = ((int)0x84FA),

UnsignedInt248Ext = ((int)0x84FA), **UnsignedInt248Nv** = ((int)0x84FA), **MaxTextureLodBias** = ((int)0x84FD), **MaxTextureLodBiasExt** = ((int)0x84FD),

TextureMaxAnisotropyExt = ((int)0x84FE), **TextureMaxAnisotropyNv** = ((int)0x84FF), **TextureFilterControl** = ((int)0x8500), **TextureFilterControlExt** = ((int)0x8500),

TextureLodBias = ((int)0x8501), **TextureLodBiasExt** = ((int)0x8501), **Modelview1StackDepthExt** = ((int)0x8502), **Combine4Nv** = ((int)0x8503),

MaxShininessNv = ((int)0x8504), **MaxSpotExponentNv** = ((int)0x8505), **Modelview1MatrixExt** = ((int)0x8506), **IncrWrap** = ((int)0x8507),

IncrWrapExt = ((int)0x8507), **DecrWrap** = ((int)0x8508), **DecrWrapExt** = ((int)0x8508), **VertexWeightingExt** = ((int)0x8509),

Modelview1Arb = ((int)0x850A), **Modelview1Ext** = ((int)0x850A), **CurrentVertexWeightExt** = ((int)0x850B), **VertexWeightArrayExt** = ((int)0x850C),

VertexWeightArraySizeExt = ((int)0x850D), **VertexWeightArrayTypeExt** = ((int)0x850E), **VertexWeightArrayStrideExt** = ((int)0x850F), **VertexWeightArrayPointerExt** = ((int)0x8510),

NormalMap = ((int)0x8511), **NormalMapArb** = ((int)0x8511), **NormalMapExt** = ((int)0x8511), **NormalMapNv** = ((int)0x8511),

ReflectionMap = ((int)0x8512), **ReflectionMapArb** = ((int)0x8512), **ReflectionMapExt** = ((int)0x8512), **ReflectionMapNv** = ((int)0x8512),

TextureCubeMap = ((int)0x8513), **TextureCubeMapArb** = ((int)0x8513), **TextureCubeMapExt** = ((int)0x8513), **TextureBindingCubeMap** = ((int)0x8514),

TextureBindingCubeMapArb = ((int)0x8514), **TextureBindingCubeMapExt** = ((int)0x8514), **TextureCubeMapPositiveX** = ((int)0x8515), **TextureCubeMapPositiveXArb** = ((int)0x8515),

TextureCubeMapPositiveXExt = ((int)0x8515), **TextureCubeMapNegativeX** = ((int)0x8516), **TextureCubeMapNegativeXArb** = ((int)0x8516), **TextureCubeMapNegativeXExt** = ((int)0x8516),

TextureCubeMapPositiveY = ((int)0x8517), **TextureCubeMapPositiveYArb** = ((int)0x8517), **TextureCubeMapPositiveYExt** = ((int)0x8517), **TextureCubeMapNegativeY** = ((int)0x8518),

TextureCubeMapNegativeYArb = ((int)0x8518), **TextureCubeMapNegativeYExt** = ((int)0x8518), **TextureCubeMapPositiveZ** = ((int)0x8519), **TextureCubeMapPositiveZArb** = ((int)0x8519),

TextureCubeMapPositiveZExt = ((int)0x8519), **TextureCubeMapNegativeZ** = ((int)0x851A), **TextureCubeMapNegativeZArb** = ((int)0x851A), **TextureCubeMapNegativeZExt** = ((int)0x851A),

ProxyTextureCubeMap = ((int)0x851B), **ProxyTextureCubeMapArb** = ((int)0x851B), **ProxyTextureCubeMapExt** = ((int)0x851B), **MaxCubeMapTextureSize** = ((int)0x851C),

MaxCubeMapTextureSizeArb = ((int)0x851C), **MaxCubeMapTextureSizeExt** = ((int)0x851C), **VertexArrayRangeApple** = ((int)0x851D), **VertexArrayRangeNv** = ((int)0x851D),

VertexArrayRangeLengthApple = ((int)0x851E), **VertexArrayRangeLengthNv** = ((int)0x851E), **VertexArrayRangeValidNv** = ((int)0x851F), **VertexArrayStorageHintApple** = ((int)0x851F),

MaxVertexArrayRangeElementNv = ((int)0x8520), **VertexArrayRangePointerApple** = ((int)0x8521), **VertexArrayRangePointerNv** = ((int)0x8521), **RegisterCombinersNv** = ((int)0x8522),

VariableANv = ((int)0x8523), **VariableBNv** = ((int)0x8524), **VariableCNv** = ((int)0x8525), **VariableDNv** = ((int)0x8526),

VariableENv = ((int)0x8527), **VariableFNv** = ((int)0x8528), **VariableGNv** = ((int)0x8529), **ConstantColor0Nv** = ((int)0x852A),

ConstantColor1Nv = ((int)0x852B), **PrimaryColorNv** = ((int)0x852C), **SecondaryColorNv** = ((int)0x852D), **Spare0Nv** = ((int)0x852E),

Spare1Nv = ((int)0x852F), **DiscardNv** = ((int)0x8530), **ETimesFNv** = ((int)0x8531), **Spare0PlusSecondaryColorNv** = ((int)0x8532),

VertexArrayRangeWithoutFlushNv = ((int)0x8533), **MultisampleFilterHintNv** = ((int)0x8534), **PerStageConstantsNv** = ((int)0x8535), **UnsignedIdentityNv** = ((int)0x8536),

UnsignedInvertNv = ((int)0x8537), **ExpandNormalNv** = ((int)0x8538), **ExpandNegateNv** = ((int)0x8539), **HalfBiasNormalNv** = ((int)0x853A),

HalfBiasNegateNv = ((int)0x853B), **SignedIdentityNv** = ((int)0x853C), **SignedNegateNv** = ((int)0x853D), **ScaleByTwoNv** = ((int)0x853E),

ScaleByFourNv = ((int)0x853F), **ScaleByOneHalfNv** = ((int)0x8540), **BiasByNegativeOneHalfNv** = ((int)0x8541), **CombinerInputNv** = ((int)0x8542),

CombinerMappingNv = ((int)0x8543), **CombinerComponentUsageNv** = ((int)0x8544), **CombinerAbDotProductNv** = ((int)0x8545), **CombinerCdDotProductNv** = ((int)0x8546),

CombinerMuxSumNv = ((int)0x8547), **CombinerScaleNv** = ((int)0x8548), **CombinerBiasNv** = ((int)0x8549), **CombinerAbOutputNv** = ((int)0x854A),

CombinerCdOutputNv = ((int)0x854B), **CombinerSumOutputNv** = ((int)0x854C), **MaxGeneralCombinersNv** = ((int)0x854D), **NumGeneralCombinersNv** = ((int)0x854E),
ColorSumClampNv = ((int)0x854F), **Combiner0Nv** = ((int)0x8550), **Combiner1Nv** = ((int)0x8551), **Combiner2Nv** = ((int)0x8552),
Combiner3Nv = ((int)0x8553), **Combiner4Nv** = ((int)0x8554), **Combiner5Nv** = ((int)0x8555), **Combiner6Nv** = ((int)0x8556),
Combiner7Nv = ((int)0x8557), **PrimitiveRestartNv** = ((int)0x8558), **PrimitiveRestartIndexNv** = ((int)0x8559), **FogDistanceModeNv** = ((int)0x855A),
EyeRadialNv = ((int)0x855B), **EyePlaneAbsoluteNv** = ((int)0x855C), **EmbossLightNv** = ((int)0x855D), **EmbossConstantNv** = ((int)0x855E),
EmbossMapNv = ((int)0x855F), **RedMinClampIngr** = ((int)0x8560), **GreenMinClampIngr** = ((int)0x8561), **BlueMinClampIngr** = ((int)0x8562),
AlphaMinClampIngr = ((int)0x8563), **RedMaxClampIngr** = ((int)0x8564), **GreenMaxClampIngr** = ((int)0x8565), **BlueMaxClampIngr** = ((int)0x8566),
AlphaMaxClampIngr = ((int)0x8567), **InterlaceReadIngr** = ((int)0x8568), **Combine** = ((int)0x8570), **CombineArb** = ((int)0x8570),
CombineExt = ((int)0x8570), **CombineRgb** = ((int)0x8571), **CombineRgbArb** = ((int)0x8571), **CombineRgbExt** = ((int)0x8571),
CombineAlpha = ((int)0x8572), **CombineAlphaArb** = ((int)0x8572), **CombineAlphaExt** = ((int)0x8572), **RgbScale** = ((int)0x8573),
RgbScaleArb = ((int)0x8573), **RgbScaleExt** = ((int)0x8573), **AddSigned** = ((int)0x8574), **AddSignedArb** = ((int)0x8574),
AddSignedExt = ((int)0x8574), **Interpolate** = ((int)0x8575), **InterpolateArb** = ((int)0x8575), **InterpolateExt** = ((int)0x8575),
Constant = ((int)0x8576), **ConstantArb** = ((int)0x8576), **ConstantExt** = ((int)0x8576), **PrimaryColor** = ((int)0x8577),
PrimaryColorArb = ((int)0x8577), **PrimaryColorExt** = ((int)0x8577), **Previous** = ((int)0x8578), **PreviousArb** = ((int)0x8578),
PreviousExt = ((int)0x8578), **Source0Rgb** = ((int)0x8580), **Source0RgbArb** = ((int)0x8580), **Source0RgbExt** = ((int)0x8580),
Src0Rgb = ((int)0x8580), **Source1Rgb** = ((int)0x8581), **Source1RgbArb** = ((int)0x8581), **Source1RgbExt** = ((int)0x8581),
Src1Rgb = ((int)0x8581), **Source2Rgb** = ((int)0x8582), **Source2RgbArb** = ((int)0x8582), **Source2RgbExt** = ((int)0x8582),
Src2Rgb = ((int)0x8582), **Source3RgbNv** = ((int)0x8583), **Source0Alpha** = ((int)0x8588), **Source0AlphaArb** = ((int)0x8588),
Source0AlphaExt = ((int)0x8588), **Src0Alpha** = ((int)0x8588), **Source1Alpha** = ((int)0x8589), **Source1AlphaArb** = ((int)0x8589),

```

Source1AlphaExt = ((int)0x8589), Src1Alpha = ((int)0x8589), Source2Alpha
= ((int)0x858A), Source2AlphaArb = ((int)0x858A),

Source2AlphaExt = ((int)0x858A), Src2Alpha = ((int)0x858A),
Source3AlphaNv = ((int)0x858B), Operand0Rgb = ((int)0x8590),

Operand0RgbArb = ((int)0x8590), Operand0RgbExt = ((int)0x8590),
Operand1Rgb = ((int)0x8591), Operand1RgbArb = ((int)0x8591),

Operand1RgbExt = ((int)0x8591), Operand2Rgb = ((int)0x8592),
Operand2RgbArb = ((int)0x8592), Operand2RgbExt = ((int)0x8592),

Operand3RgbNv = ((int)0x8593), Operand0Alpha = ((int)0x8598),
Operand0AlphaArb = ((int)0x8598), Operand0AlphaExt = ((int)0x8598),

Operand1Alpha = ((int)0x8599), Operand1AlphaArb = ((int)0x8599),
Operand1AlphaExt = ((int)0x8599), Operand2Alpha = ((int)0x859A),

Operand2AlphaArb = ((int)0x859A), Operand2AlphaExt = ((int)0x859A),
Operand3AlphaNv = ((int)0x859B), PackSubsampleRateSgix =
((int)0x85A0),

UnpackSubsampleRateSgix = ((int)0x85A1), PixelSubsample444Sgix =
((int)0x85A2), PixelSubsample2424Sgix = ((int)0x85A3), PixelSubsam-
ple4242Sgix = ((int)0x85A4),

PerturbExt = ((int)0x85AE), TextureNormalExt = ((int)0x85AF), Light-
ModelSpecularVectorApple = ((int)0x85B0), TransformHintApple =
((int)0x85B1),

UnpackClientStorageApple = ((int)0x85B2), BufferObjectApple =
((int)0x85B3), VertexArrayBinding = ((int)0x85B5), VertexArrayBindin-
gApple = ((int)0x85B5),

TextureRangeLengthApple = ((int)0x85B7), TextureRangePointerApple =
((int)0x85B8), Ycbcr422Apple = ((int)0x85B9), UnsignedShort88Apple =
((int)0x85BA),

UnsignedShort88Mesa = ((int)0x85BA), UnsignedShort88RevApple =
((int)0x85BB), UnsignedShort88RevMesa = ((int)0x85BB), TextureStorage-
HintApple = ((int)0x85BC),

StoragePrivateApple = ((int)0x85BD), StorageCachedApple = ((int)0x85BE),
StorageSharedApple = ((int)0x85BF), ReplacementCodeArraySun =
((int)0x85C0),

ReplacementCodeArrayTypeSun = ((int)0x85C1), ReplacementCodeAr-
rayStrideSun = ((int)0x85C2), ReplacementCodeArrayPointerSun =
((int)0x85C3), R1uiV3fSun = ((int)0x85C4),

R1uiC4ubV3fSun = ((int)0x85C5), R1uiC3fV3fSun = ((int)0x85C6),
R1uiN3fV3fSun = ((int)0x85C7), R1uiC4fN3fV3fSun = ((int)0x85C8),

R1uiT2fV3fSun = ((int)0x85C9), R1uiT2fN3fV3fSun = ((int)0x85CA),
R1uiT2fC4fN3fV3fSun = ((int)0x85CB), SliceAccumSun = ((int)0x85CC),

```

QuadMeshSun = ((int)0x8614), **TriangleMeshSun** = ((int)0x8615), **VertexProgram** = ((int)0x8620), **VertexProgramArb** = ((int)0x8620),
VertexProgramNv = ((int)0x8620), **VertexStateProgramNv** = ((int)0x8621),
ArrayEnabled = ((int)0x8622), **VertexAttribPointerEnabled** = ((int)0x8622),
VertexAttribPointerEnabledArb = ((int)0x8622), **ArraySize** = ((int)0x8623),
AttribArraySizeNv = ((int)0x8623), **VertexAttribPointerSize** = ((int)0x8623),
VertexAttribPointerSizeArb = ((int)0x8623), **ArrayStride** = ((int)0x8624),
AttribArrayStrideNv = ((int)0x8624), **VertexAttribPointerStride** = ((int)0x8624),
VertexAttribPointerStrideArb = ((int)0x8624), **ArrayType** = ((int)0x8625),
AttribArrayTypeNv = ((int)0x8625), **VertexAttribPointerType** = ((int)0x8625),
VertexAttribPointerTypeArb = ((int)0x8625), **CurrentAttribNv** = ((int)0x8626),
CurrentVertexAttrib = ((int)0x8626), **CurrentVertexAttribArb** = ((int)0x8626),
ProgramLength = ((int)0x8627), **ProgramLengthArb** = ((int)0x8627), **ProgramLengthNv** = ((int)0x8627),
ProgramString = ((int)0x8628),
ProgramStringArb = ((int)0x8628), **ProgramStringNv** = ((int)0x8628), **ModelviewProjectionNv** = ((int)0x8629),
IdentityNv = ((int)0x862A),
InverseNv = ((int)0x862B), **TransposeNv** = ((int)0x862C), **InverseTransposeNv** = ((int)0x862D),
MaxProgramMatrixStackDepthArb = ((int)0x862E),
MaxTrackMatrixStackDepthNv = ((int)0x862E), **MaxProgramMatricesArb** = ((int)0x862F),
MaxTrackMatricesNv = ((int)0x862F), **Matrix0Nv** = ((int)0x8630),
Matrix1Nv = ((int)0x8631), **Matrix2Nv** = ((int)0x8632), **Matrix3Nv** = ((int)0x8633),
Matrix4Nv = ((int)0x8634),
Matrix5Nv = ((int)0x8635), **Matrix6Nv** = ((int)0x8636), **Matrix7Nv** = ((int)0x8637),
CurrentMatrixStackDepthArb = ((int)0x8640),
CurrentMatrixStackDepthNv = ((int)0x8640), **CurrentMatrixArb** = ((int)0x8641),
CurrentMatrixNv = ((int)0x8641), **ProgramPointSize** = ((int)0x8642),
ProgramPointSizeArb = ((int)0x8642), **ProgramPointSizeExt** = ((int)0x8642),
VertexProgramPointSize = ((int)0x8642), **VertexProgramPointSizeArb** = ((int)0x8642),
VertexProgramPointSizeNv = ((int)0x8642), **VertexProgramTwoSide** = ((int)0x8643),
VertexProgramTwoSideArb = ((int)0x8643), **VertexProgramTwoSideNv** = ((int)0x8643),
ProgramParameterNv = ((int)0x8644), **ArrayPointer** = ((int)0x8645), **AttribArrayPointerNv** = ((int)0x8645),
VertexAttribPointer = ((int)0x8645), **VertexAttribPointerNv** = ((int)0x8645),

VertexAttribPointerArb = ((int)0x8645), **ProgramTargetNv** = ((int)0x8646), **ProgramResidentNv** = ((int)0x8647), **TrackMatrixNv** = ((int)0x8648),

TrackMatrixTransformNv = ((int)0x8649), **VertexProgramBindingNv** = ((int)0x864A), **ProgramErrorPositionArb** = ((int)0x864B), **ProgramErrorPositionNv** = ((int)0x864B),

OffsetTextureRectangleNv = ((int)0x864C), **OffsetTextureRectangleScaleNv** = ((int)0x864D), **DotProductTextureRectangleNv** = ((int)0x864E), **DepthClamp** = ((int)0x864F),

DepthClampNv = ((int)0x864F), **VertexAttribPointer0Nv** = ((int)0x8650), **VertexAttribPointer1Nv** = ((int)0x8651), **VertexAttribPointer2Nv** = ((int)0x8652),

VertexAttribPointer3Nv = ((int)0x8653), **VertexAttribPointer4Nv** = ((int)0x8654), **VertexAttribPointer5Nv** = ((int)0x8655), **VertexAttribPointer6Nv** = ((int)0x8656),

VertexAttribPointer7Nv = ((int)0x8657), **VertexAttribPointer8Nv** = ((int)0x8658), **VertexAttribPointer9Nv** = ((int)0x8659), **VertexAttribPointer10Nv** = ((int)0x865A),

VertexAttribPointer11Nv = ((int)0x865B), **VertexAttribPointer12Nv** = ((int)0x865C), **VertexAttribPointer13Nv** = ((int)0x865D), **VertexAttribPointer14Nv** = ((int)0x865E),

VertexAttribPointer15Nv = ((int)0x865F), **Map1VertexAttrib04Nv** = ((int)0x8660), **Map1VertexAttrib14Nv** = ((int)0x8661), **Map1VertexAttrib24Nv** = ((int)0x8662),

Map1VertexAttrib34Nv = ((int)0x8663), **Map1VertexAttrib44Nv** = ((int)0x8664), **Map1VertexAttrib54Nv** = ((int)0x8665), **Map1VertexAttrib64Nv** = ((int)0x8666),

Map1VertexAttrib74Nv = ((int)0x8667), **Map1VertexAttrib84Nv** = ((int)0x8668), **Map1VertexAttrib94Nv** = ((int)0x8669), **Map1VertexAttrib104Nv** = ((int)0x866A),

Map1VertexAttrib114Nv = ((int)0x866B), **Map1VertexAttrib124Nv** = ((int)0x866C), **Map1VertexAttrib134Nv** = ((int)0x866D), **Map1VertexAttrib144Nv** = ((int)0x866E),

Map1VertexAttrib154Nv = ((int)0x866F), **Map2VertexAttrib04Nv** = ((int)0x8670), **Map2VertexAttrib14Nv** = ((int)0x8671), **Map2VertexAttrib24Nv** = ((int)0x8672),

Map2VertexAttrib34Nv = ((int)0x8673), **Map2VertexAttrib44Nv** = ((int)0x8674), **Map2VertexAttrib54Nv** = ((int)0x8675), **Map2VertexAttrib64Nv** = ((int)0x8676),

Map2VertexAttrib74Nv = ((int)0x8677), **ProgramBinding** = ((int)0x8677), **ProgramBindingArb** = ((int)0x8677), **Map2VertexAttrib84Nv** = ((int)0x8678),

```

Map2VertexAttrib94Nv = ((int)0x8679), Map2VertexAttrib104Nv
= ((int)0x867A), Map2VertexAttrib114Nv = ((int)0x867B),
Map2VertexAttrib124Nv = ((int)0x867C),
Map2VertexAttrib134Nv = ((int)0x867D), Map2VertexAttrib144Nv =
((int)0x867E), Map2VertexAttrib154Nv = ((int)0x867F), TextureCom-
pressedImageSize = ((int)0x86A0),
TextureCompressedImageSizeArb = ((int)0x86A0), TextureCompressed =
((int)0x86A1), TextureCompressedArb = ((int)0x86A1), NumCompressed-
TextureFormats = ((int)0x86A2),
NumCompressedTextureFormatsArb = ((int)0x86A2), CompressedTexture-
Formats = ((int)0x86A3), CompressedTextureFormatsArb = ((int)0x86A3),
MaxVertexUnitsArb = ((int)0x86A4),
ActiveVertexUnitsArb = ((int)0x86A5), WeightSumUnityArb =
((int)0x86A6), VertexBlendArb = ((int)0x86A7), CurrentWeightArb =
((int)0x86A8),
WeightArrayTypeArb = ((int)0x86A9), WeightArrayStrideArb =
((int)0x86AA), WeightArraySizeArb = ((int)0x86AB), WeightArray-
PointerArb = ((int)0x86AC),
WeightArrayArb = ((int)0x86AD), Dot3Rgb = ((int)0x86AE), Dot3RgbArb
= ((int)0x86AE), Dot3Rgba = ((int)0x86AF),
Dot3RgbaArb = ((int)0x86AF), CompressedRgbFxt13Dfx = ((int)0x86B0),
CompressedRgbaFxt13Dfx = ((int)0x86B1), Multisample3Dfx =
((int)0x86B2),
SampleBuffers3Dfx = ((int)0x86B3), Samples3Dfx = ((int)0x86B4),
Eval2DNv = ((int)0x86C0), EvalTriangular2DNv = ((int)0x86C1),
MapTessellationNv = ((int)0x86C2), MapAttribUOrderNv = ((int)0x86C3),
MapAttribVOrderNv = ((int)0x86C4), EvalFractionalTessellationNv =
((int)0x86C5),
EvalVertexAttrib0Nv = ((int)0x86C6), EvalVertexAttrib1Nv = ((int)0x86C7),
EvalVertexAttrib2Nv = ((int)0x86C8), EvalVertexAttrib3Nv = ((int)0x86C9),
EvalVertexAttrib4Nv = ((int)0x86CA), EvalVertexAttrib5Nv =
((int)0x86CB), EvalVertexAttrib6Nv = ((int)0x86CC), EvalVertexAttrib7Nv
= ((int)0x86CD),
EvalVertexAttrib8Nv = ((int)0x86CE), EvalVertexAttrib9Nv =
((int)0x86CF), EvalVertexAttrib10Nv = ((int)0x86D0), EvalVertexAt-
trib11Nv = ((int)0x86D1),
EvalVertexAttrib12Nv = ((int)0x86D2), EvalVertexAttrib13Nv =
((int)0x86D3), EvalVertexAttrib14Nv = ((int)0x86D4), EvalVertexAt-
trib15Nv = ((int)0x86D5),
MaxMapTessellationNv = ((int)0x86D6), MaxRationalEvalOrderNv =
((int)0x86D7), RgbaUnsignedDotProductMappingNv = ((int)0x86D9), Un-
signedIntS8S888Nv = ((int)0x86DA),

```

UnsignedInt8S8S8RevNv = ((int)0x86DB), **DsdtMagIntensityNv** = ((int)0x86DC), **ShaderConsistentNv** = ((int)0x86DD), **TextureShaderNv** = ((int)0x86DE),

ShaderOperationNv = ((int)0x86DF), **CullModesNv** = ((int)0x86E0), **OffsetTexture2DMatrixNv** = ((int)0x86E1), **OffsetTextureMatrixNv** = ((int)0x86E1),

OffsetTexture2DScaleNv = ((int)0x86E2), **OffsetTextureScaleNv** = ((int)0x86E2), **OffsetTexture2DBiasNv** = ((int)0x86E3), **OffsetTextureBiasNv** = ((int)0x86E3),

PreviousTextureInputNv = ((int)0x86E4), **ConstEyeNv** = ((int)0x86E5), **PassThroughNv** = ((int)0x86E6), **CullFragmentNv** = ((int)0x86E7),

OffsetTexture2DNv = ((int)0x86E8), **DependentArTexture2DNv** = ((int)0x86E9), **DependentGbTexture2DNv** = ((int)0x86EA), **DotProductNv** = ((int)0x86EC),

DotProductDepthReplaceNv = ((int)0x86ED), **DotProductTexture2DNv** = ((int)0x86EE), **DotProductTexture3DNv** = ((int)0x86EF), **DotProductTextureCubeMapNv** = ((int)0x86F0),

DotProductDiffuseCubeMapNv = ((int)0x86F1), **DotProductReflectCubeMapNv** = ((int)0x86F2), **DotProductConstEyeReflectCubeMapNv** = ((int)0x86F3), **HiloNv** = ((int)0x86F4),

DsdtNv = ((int)0x86F5), **DsdtMagNv** = ((int)0x86F6), **DsdtMagVibNv** = ((int)0x86F7), **Hilo16Nv** = ((int)0x86F8),

SignedHiloNv = ((int)0x86F9), **SignedHilo16Nv** = ((int)0x86FA), **SignedRgbaNv** = ((int)0x86FB), **SignedRgba8Nv** = ((int)0x86FC),

SignedRgbNv = ((int)0x86FE), **SignedRgb8Nv** = ((int)0x86FF), **SignedLuminanceNv** = ((int)0x8701), **SignedLuminance8Nv** = ((int)0x8702),

SignedLuminanceAlphaNv = ((int)0x8703), **SignedLuminance8Alpha8Nv** = ((int)0x8704), **SignedAlphaNv** = ((int)0x8705), **SignedAlpha8Nv** = ((int)0x8706),

SignedIntensityNv = ((int)0x8707), **SignedIntensity8Nv** = ((int)0x8708), **Dsdt8Nv** = ((int)0x8709), **Dsdt8Mag8Nv** = ((int)0x870A),

Dsdt8Mag8Intensity8Nv = ((int)0x870B), **SignedRgbUnsignedAlphaNv** = ((int)0x870C), **SignedRgb8UnsignedAlpha8Nv** = ((int)0x870D), **HiScaleNv** = ((int)0x870E),

LoScaleNv = ((int)0x870F), **DsScaleNv** = ((int)0x8710), **DtScaleNv** = ((int)0x8711), **MagnitudeScaleNv** = ((int)0x8712),

VibranceScaleNv = ((int)0x8713), **HiBiasNv** = ((int)0x8714), **LoBiasNv** = ((int)0x8715), **DsBiasNv** = ((int)0x8716),

DtBiasNv = ((int)0x8717), **MagnitudeBiasNv** = ((int)0x8718), **VibranceBiasNv** = ((int)0x8719), **TextureBorderValuesNv** = ((int)0x871A),

TextureHiSizeNv = ((int)0x871B), **TextureLoSizeNv** = ((int)0x871C), **TextureDsSizeNv** = ((int)0x871D), **TextureDtSizeNv** = ((int)0x871E),
TextureMagSizeNv = ((int)0x871F), **Modelview2Arb** = ((int)0x8722), **Modelview3Arb** = ((int)0x8723), **Modelview4Arb** = ((int)0x8724),
Modelview5Arb = ((int)0x8725), **Modelview6Arb** = ((int)0x8726), **Modelview7Arb** = ((int)0x8727), **Modelview8Arb** = ((int)0x8728),
Modelview9Arb = ((int)0x8729), **Modelview10Arb** = ((int)0x872A), **Modelview11Arb** = ((int)0x872B), **Modelview12Arb** = ((int)0x872C),
Modelview13Arb = ((int)0x872D), **Modelview14Arb** = ((int)0x872E), **Modelview15Arb** = ((int)0x872F), **Modelview16Arb** = ((int)0x8730),
Modelview17Arb = ((int)0x8731), **Modelview18Arb** = ((int)0x8732), **Modelview19Arb** = ((int)0x8733), **Modelview20Arb** = ((int)0x8734),
Modelview21Arb = ((int)0x8735), **Modelview22Arb** = ((int)0x8736), **Modelview23Arb** = ((int)0x8737), **Modelview24Arb** = ((int)0x8738),
Modelview25Arb = ((int)0x8739), **Modelview26Arb** = ((int)0x873A), **Modelview27Arb** = ((int)0x873B), **Modelview28Arb** = ((int)0x873C),
Modelview29Arb = ((int)0x873D), **Modelview30Arb** = ((int)0x873E), **Modelview31Arb** = ((int)0x873F), **Dot3RgbExt** = ((int)0x8740),
Dot3RgbaExt = ((int)0x8741), **MirrorClampAti** = ((int)0x8742), **MirrorClampExt** = ((int)0x8742), **MirrorClampToEdgeAti** = ((int)0x8743),
MirrorClampToEdgeExt = ((int)0x8743), **ModulateAddAti** = ((int)0x8744), **ModulateSignedAddAti** = ((int)0x8745), **ModulateSubtractAti** = ((int)0x8746),
YcbcrMesa = ((int)0x8757), **PackInvertMesa** = ((int)0x8758), **Texture1DStackMesax** = ((int)0x8759), **Texture2DStackMesax** = ((int)0x875A),
ProxyTexture1DStackMesax = ((int)0x875B), **ProxyTexture2DStackMesax** = ((int)0x875C), **Texture1DStackBindingMesax** = ((int)0x875D), **Texture2DStackBindingMesax** = ((int)0x875E),
StaticAti = ((int)0x8760), **DynamicAti** = ((int)0x8761), **PreserveAti** = ((int)0x8762), **DiscardAti** = ((int)0x8763),
BufferSize = ((int)0x8764), **BufferSizeArb** = ((int)0x8764), **ObjectBufferSizeAti** = ((int)0x8764), **BufferUsage** = ((int)0x8765),
BufferUsageArb = ((int)0x8765), **ObjectBufferUsageAti** = ((int)0x8765), **ArrayObjectBufferAti** = ((int)0x8766), **ArrayObjectOffsetAti** = ((int)0x8767),
ElementArrayApple = ((int)0x8768), **ElementArrayAti** = ((int)0x8768), **ElementArrayTypeApple** = ((int)0x8769), **ElementArrayTypeAti** = ((int)0x8769),
ElementArrayPointerApple = ((int)0x876A), **ElementArrayPointerAti** = ((int)0x876A), **MaxVertexStreamsAti** = ((int)0x876B), **VertexStream0Ati** = ((int)0x876C),

VertexStream1Ati = ((int)0x876D), **VertexStream2Ati** = ((int)0x876E), **VertexStream3Ati** = ((int)0x876F), **VertexStream4Ati** = ((int)0x8770),
VertexStream5Ati = ((int)0x8771), **VertexStream6Ati** = ((int)0x8772), **VertexStream7Ati** = ((int)0x8773), **VertexSourceAti** = ((int)0x8774),
BumpRotMatrixAti = ((int)0x8775), **BumpRotMatrixSizeAti** = ((int)0x8776), **BumpNumTexUnitsAti** = ((int)0x8777), **BumpTexUnitsAti** = ((int)0x8778),
DudvAti = ((int)0x8779), **Du8dv8Ati** = ((int)0x877A), **BumpEnvmapAti** = ((int)0x877B), **BumpTargetAti** = ((int)0x877C),
VertexShaderExt = ((int)0x8780), **VertexShaderBindingExt** = ((int)0x8781), **OpIndexExt** = ((int)0x8782), **OpNegateExt** = ((int)0x8783),
OpDot3Ext = ((int)0x8784), **OpDot4Ext** = ((int)0x8785), **OpMulExt** = ((int)0x8786), **OpAddExt** = ((int)0x8787),
OpMaddExt = ((int)0x8788), **OpFracExt** = ((int)0x8789), **OpMaxExt** = ((int)0x878A), **OpMinExt** = ((int)0x878B),
OpSetGeExt = ((int)0x878C), **OpSetLtExt** = ((int)0x878D), **OpClampExt** = ((int)0x878E), **OpFloorExt** = ((int)0x878F),
OpRoundExt = ((int)0x8790), **OpExpBase2Ext** = ((int)0x8791), **OpLogBase2Ext** = ((int)0x8792), **OpPowerExt** = ((int)0x8793),
OpRecipExt = ((int)0x8794), **OpRecipSqrtExt** = ((int)0x8795), **OpSubExt** = ((int)0x8796), **OpCrossProductExt** = ((int)0x8797),
OpMultiplyMatrixExt = ((int)0x8798), **OpMovExt** = ((int)0x8799), **OutputVertexExt** = ((int)0x879A), **OutputColor0Ext** = ((int)0x879B),
OutputColor1Ext = ((int)0x879C), **OutputTextureCoord0Ext** = ((int)0x879D), **OutputTextureCoord1Ext** = ((int)0x879E), **OutputTextureCoord2Ext** = ((int)0x879F),
OutputTextureCoord3Ext = ((int)0x87A0), **OutputTextureCoord4Ext** = ((int)0x87A1), **OutputTextureCoord5Ext** = ((int)0x87A2), **OutputTextureCoord6Ext** = ((int)0x87A3),
OutputTextureCoord7Ext = ((int)0x87A4), **OutputTextureCoord8Ext** = ((int)0x87A5), **OutputTextureCoord9Ext** = ((int)0x87A6), **OutputTextureCoord10Ext** = ((int)0x87A7),
OutputTextureCoord11Ext = ((int)0x87A8), **OutputTextureCoord12Ext** = ((int)0x87A9), **OutputTextureCoord13Ext** = ((int)0x87AA), **OutputTextureCoord14Ext** = ((int)0x87AB),
OutputTextureCoord15Ext = ((int)0x87AC), **OutputTextureCoord16Ext** = ((int)0x87AD), **OutputTextureCoord17Ext** = ((int)0x87AE), **OutputTextureCoord18Ext** = ((int)0x87AF),
OutputTextureCoord19Ext = ((int)0x87B0), **OutputTextureCoord20Ext** = ((int)0x87B1), **OutputTextureCoord21Ext** = ((int)0x87B2), **OutputTextureCoord22Ext** = ((int)0x87B3),


```

OutputTextureCoord23Ext = ((int)0x87B4), OutputTextureCoord24Ext =
((int)0x87B5), OutputTextureCoord25Ext = ((int)0x87B6), OutputTexture-
Coord26Ext = ((int)0x87B7),
OutputTextureCoord27Ext = ((int)0x87B8), OutputTextureCoord28Ext =
((int)0x87B9), OutputTextureCoord29Ext = ((int)0x87BA), OutputTexture-
Coord30Ext = ((int)0x87BB),
OutputTextureCoord31Ext = ((int)0x87BC), OutputFogExt = ((int)0x87BD),
ScalarExt = ((int)0x87BE), VectorExt = ((int)0x87BF),
MatrixExt = ((int)0x87C0), VariantExt = ((int)0x87C1), InvariantExt =
((int)0x87C2), LocalConstantExt = ((int)0x87C3),
LocalExt = ((int)0x87C4), MaxVertexShaderInstructionsExt =
((int)0x87C5), MaxVertexShaderVariantsExt = ((int)0x87C6), MaxVer-
texShaderInvariantsExt = ((int)0x87C7),
MaxVertexShaderLocalConstantsExt = ((int)0x87C8), MaxVertexShader-
LocalsExt = ((int)0x87C9), MaxOptimizedVertexShaderInstructionsExt =
((int)0x87CA), MaxOptimizedVertexShaderVariantsExt = ((int)0x87CB),
MaxOptimizedVertexShaderLocalConstantsExt = ((int)0x87CC), Max-
OptimizedVertexShaderInvariantsExt = ((int)0x87CD), MaxOptimized-
VertexShaderLocalsExt = ((int)0x87CE), VertexShaderInstructionsExt =
((int)0x87CF),
VertexShaderVariantsExt = ((int)0x87D0), VertexShaderInvariantsExt =
((int)0x87D1), VertexShaderLocalConstantsExt = ((int)0x87D2), Ver-
texShaderLocalsExt = ((int)0x87D3),
VertexShaderOptimizedExt = ((int)0x87D4), XExt = ((int)0x87D5), YExt =
((int)0x87D6), ZExt = ((int)0x87D7),
WExt = ((int)0x87D8), NegativeXExt = ((int)0x87D9), NegativeYExt =
((int)0x87DA), NegativeZExt = ((int)0x87DB),
NegativeWExt = ((int)0x87DC), ZeroExt = ((int)0x87DD), OneExt =
((int)0x87DE), NegativeOneExt = ((int)0x87DF),
NormalizedRangeExt = ((int)0x87E0), FullRangeExt = ((int)0x87E1), Cur-
rentVertexExt = ((int)0x87E2), MvpMatrixExt = ((int)0x87E3),
VariantValueExt = ((int)0x87E4), VariantDatatypeExt = ((int)0x87E5), Vari-
antArrayStrideExt = ((int)0x87E6), VariantArrayTypeExt = ((int)0x87E7),
VariantArrayExt = ((int)0x87E8), VariantArrayPointerExt = ((int)0x87E9),
InvariantValueExt = ((int)0x87EA), InvariantDatatypeExt = ((int)0x87EB),
LocalConstantValueExt = ((int)0x87EC), LocalConstantDatatypeExt =
((int)0x87ED), PnTrianglesAti = ((int)0x87F0), MaxPnTrianglesTessellation-
LevelAti = ((int)0x87F1),
PnTrianglesPointModeAti = ((int)0x87F2), PnTrianglesNormalModeAti =
((int)0x87F3), PnTrianglesTessellationLevelAti = ((int)0x87F4), PnTriangle-
sPointModeLinearAti = ((int)0x87F5),

```

PnTrianglesPointModeCubicAti = ((int)0x87F6), **PnTrianglesNormalModeLinearAti** = ((int)0x87F7), **PnTrianglesNormalModeQuadraticAti** = ((int)0x87F8), **VboFreeMemoryAti** = ((int)0x87FB),
TextureFreeMemoryAti = ((int)0x87FC), **RenderbufferFreeMemoryAti** = ((int)0x87FD), **StencilBackFunc** = ((int)0x8800), **StencilBackFuncAti** = ((int)0x8800),
StencilBackFail = ((int)0x8801), **StencilBackFailAti** = ((int)0x8801), **StencilBackPassDepthFail** = ((int)0x8802), **StencilBackPassDepthFailAti** = ((int)0x8802),
StencilBackPassDepthPass = ((int)0x8803), **StencilBackPassDepthPassAti** = ((int)0x8803), **FragmentProgram** = ((int)0x8804), **FragmentProgramArb** = ((int)0x8804),
ProgramAluInstructionsArb = ((int)0x8805), **ProgramTexInstructionsArb** = ((int)0x8806), **ProgramTexIndirectionsArb** = ((int)0x8807), **ProgramNativeAluInstructionsArb** = ((int)0x8808),
ProgramNativeTexInstructionsArb = ((int)0x8809), **ProgramNativeTexIndirectionsArb** = ((int)0x880A), **MaxProgramAluInstructionsArb** = ((int)0x880B), **MaxProgramTexInstructionsArb** = ((int)0x880C),
MaxProgramTexIndirectionsArb = ((int)0x880D), **MaxProgramNativeAluInstructionsArb** = ((int)0x880E), **MaxProgramNativeTexInstructionsArb** = ((int)0x880F), **MaxProgramNativeTexIndirectionsArb** = ((int)0x8810),
Rgba32f = ((int)0x8814), **Rgba32fArb** = ((int)0x8814), **RgbaFloat32Apple** = ((int)0x8814), **RgbaFloat32Ati** = ((int)0x8814),
Rgb32f = ((int)0x8815), **Rgb32fArb** = ((int)0x8815), **RgbFloat32Apple** = ((int)0x8815), **RgbFloat32Ati** = ((int)0x8815),
Alpha32fArb = ((int)0x8816), **AlphaFloat32Apple** = ((int)0x8816), **AlphaFloat32Ati** = ((int)0x8816), **Intensity32fArb** = ((int)0x8817),
IntensityFloat32Apple = ((int)0x8817), **IntensityFloat32Ati** = ((int)0x8817), **Luminance32fArb** = ((int)0x8818), **LuminanceFloat32Apple** = ((int)0x8818),
LuminanceFloat32Ati = ((int)0x8818), **LuminanceAlpha32fArb** = ((int)0x8819), **LuminanceAlphaFloat32Apple** = ((int)0x8819), **LuminanceAlphaFloat32Ati** = ((int)0x8819),
Rgba16f = ((int)0x881A), **Rgba16fArb** = ((int)0x881A), **RgbaFloat16Apple** = ((int)0x881A), **RgbaFloat16Ati** = ((int)0x881A),
Rgb16f = ((int)0x881B), **Rgb16fArb** = ((int)0x881B), **RgbFloat16Apple** = ((int)0x881B), **RgbFloat16Ati** = ((int)0x881B),
Alpha16fArb = ((int)0x881C), **AlphaFloat16Apple** = ((int)0x881C), **AlphaFloat16Ati** = ((int)0x881C), **Intensity16fArb** = ((int)0x881D),
IntensityFloat16Apple = ((int)0x881D), **IntensityFloat16Ati** = ((int)0x881D), **Luminance16fArb** = ((int)0x881E), **LuminanceFloat16Apple** = ((int)0x881E),

```
LuminanceFloat16Ati = ((int)0x881E), LuminanceAlpha16fArb =  
((int)0x881F), LuminanceAlphaFloat16Apple = ((int)0x881F), Lumi-  
nanceAlphaFloat16Ati = ((int)0x881F),  
RgbaFloatMode = ((int)0x8820), RgbaFloatModeArb = ((int)0x8820), Type-  
RgbaFloatAti = ((int)0x8820), MaxDrawBuffers = ((int)0x8824),  
MaxDrawBuffersArb = ((int)0x8824), MaxDrawBuffersAti = ((int)0x8824),  
DrawBuffer0 = ((int)0x8825), DrawBuffer0Arb = ((int)0x8825),  
DrawBuffer0Ati = ((int)0x8825), DrawBuffer1 = ((int)0x8826), Draw-  
Buffer1Arb = ((int)0x8826), DrawBuffer1Ati = ((int)0x8826),  
DrawBuffer2 = ((int)0x8827), DrawBuffer2Arb = ((int)0x8827), Draw-  
Buffer2Ati = ((int)0x8827), DrawBuffer3 = ((int)0x8828),  
DrawBuffer3Arb = ((int)0x8828), DrawBuffer3Ati = ((int)0x8828), Draw-  
Buffer4 = ((int)0x8829), DrawBuffer4Arb = ((int)0x8829),  
DrawBuffer4Ati = ((int)0x8829), DrawBuffer5 = ((int)0x882A), Draw-  
Buffer5Arb = ((int)0x882A), DrawBuffer5Ati = ((int)0x882A),  
DrawBuffer6 = ((int)0x882B), DrawBuffer6Arb = ((int)0x882B), Draw-  
Buffer6Ati = ((int)0x882B), DrawBuffer7 = ((int)0x882C),  
DrawBuffer7Arb = ((int)0x882C), DrawBuffer7Ati = ((int)0x882C), Draw-  
Buffer8 = ((int)0x882D), DrawBuffer8Arb = ((int)0x882D),  
DrawBuffer8Ati = ((int)0x882D), DrawBuffer9 = ((int)0x882E), Draw-  
Buffer9Arb = ((int)0x882E), DrawBuffer9Ati = ((int)0x882E),  
DrawBuffer10 = ((int)0x882F), DrawBuffer10Arb = ((int)0x882F), Draw-  
Buffer10Ati = ((int)0x882F), DrawBuffer11 = ((int)0x8830),  
DrawBuffer11Arb = ((int)0x8830), DrawBuffer11Ati = ((int)0x8830), Draw-  
Buffer12 = ((int)0x8831), DrawBuffer12Arb = ((int)0x8831),  
DrawBuffer12Ati = ((int)0x8831), DrawBuffer13 = ((int)0x8832), Draw-  
Buffer13Arb = ((int)0x8832), DrawBuffer13Ati = ((int)0x8832),  
DrawBuffer14 = ((int)0x8833), DrawBuffer14Arb = ((int)0x8833), Draw-  
Buffer14Ati = ((int)0x8833), DrawBuffer15 = ((int)0x8834),  
DrawBuffer15Arb = ((int)0x8834), DrawBuffer15Ati = ((int)0x8834),  
ColorClearUnclampedValueAti = ((int)0x8835), BlendEquationAlpha =  
((int)0x883D),  
BlendEquationAlphaExt = ((int)0x883D), MatrixPaletteArb = ((int)0x8840),  
MaxMatrixPaletteStackDepthArb = ((int)0x8841), MaxPaletteMatricesArb  
= ((int)0x8842),  
CurrentPaletteMatrixArb = ((int)0x8843), MatrixIndexArrayArb =  
((int)0x8844), CurrentMatrixIndexArb = ((int)0x8845), MatrixIndexArray-  
SizeArb = ((int)0x8846),  
MatrixIndexArrayTypeArb = ((int)0x8847), MatrixIndexArrayStrideArb =  
((int)0x8848), MatrixIndexArrayPointerArb = ((int)0x8849), TextureDepth-  
Size = ((int)0x884A),
```

TextureDepthSizeArb = ((int)0x884A), **DepthTextureMode** = ((int)0x884B),
DepthTextureModeArb = ((int)0x884B), **TextureCompareMode** = ((int)0x884C),
TextureCompareModeArb = ((int)0x884C), **TextureCompareFunc** = ((int)0x884D),
TextureCompareFuncArb = ((int)0x884D), **CompareRefDepthToTextureExt** = ((int)0x884E),
CompareRefToTexture = ((int)0x884E), **CompareRToTexture** = ((int)0x884E),
CompareRToTextureArb = ((int)0x884E), **TextureCubeMapSeamless** = ((int)0x884F),
OffsetProjectiveTexture2DNv = ((int)0x8850), **OffsetProjectiveTexture2DScaleNv** = ((int)0x8851),
OffsetProjectiveTextureRectangleNv = ((int)0x8852), **OffsetProjectiveTextureRectangleScaleNv** = ((int)0x8853),
OffsetHiloTexture2DNv = ((int)0x8854), **OffsetHiloTextureRectangleNv** = ((int)0x8855),
OffsetHiloProjectiveTexture2DNv = ((int)0x8856), **OffsetHiloProjectiveTextureRectangleNv** = ((int)0x8857),
DependentHiloTexture2DNv = ((int)0x8858), **DependentRgbTexture3DNv** = ((int)0x8859),
DependentRgbTextureCubeMapNv = ((int)0x885A), **DotProductPassThroughNv** = ((int)0x885B),
DotProductTexture1DNv = ((int)0x885C), **DotProductAffineDepthReplaceNv** = ((int)0x885D),
Hilo8Nv = ((int)0x885E), **SignedHilo8Nv** = ((int)0x885F),
ForceBlueToOneNv = ((int)0x8860), **PointSprite** = ((int)0x8861), **PointSpriteArb** = ((int)0x8861),
PointSpriteNv = ((int)0x8861), **CoordReplace** = ((int)0x8862), **CoordReplaceArb** = ((int)0x8862),
CoordReplaceNv = ((int)0x8862), **PointSpriteRModeNv** = ((int)0x8863),
PixelCounterBitsNv = ((int)0x8864), **QueryCounterBits** = ((int)0x8864), **QueryCounterBitsArb** = ((int)0x8864),
CurrentOcclusionQueryIdNv = ((int)0x8865), **CurrentQuery** = ((int)0x8865), **CurrentQueryArb** = ((int)0x8865),
PixelCountNv = ((int)0x8866), **QueryResult** = ((int)0x8866), **QueryResultArb** = ((int)0x8866),
PixelCountAvailableNv = ((int)0x8867), **QueryResultAvailable** = ((int)0x8867),
QueryResultAvailableArb = ((int)0x8867), **MaxFragmentProgramLocalParametersNv** = ((int)0x8868),
MaxVertexAttribs = ((int)0x8869), **MaxVertexAttribsArb** = ((int)0x8869), **ArrayNormalized** = ((int)0x886A),
VertexAttribArrayNormalized = ((int)0x886A), **VertexAttribArrayNormalizedArb** = ((int)0x886A),
DepthStencilToRgbaNv = ((int)0x886E), **DepthStencilToBgraNv** = ((int)0x886F),
FragmentProgramNv = ((int)0x8870), **MaxTextureCoords** = ((int)0x8871), **MaxTextureCoordsArb** = ((int)0x8871),
MaxTextureCoordsNv = ((int)0x8871),

MaxTextureImageUnits = ((int)0x8872), **MaxTextureImageUnitsArb** = ((int)0x8872), **MaxTextureImageUnitsNv** = ((int)0x8872), **FragmentProgramBindingNv** = ((int)0x8873),
ProgramErrorStringArb = ((int)0x8874), **ProgramErrorStringNv** = ((int)0x8874), **ProgramFormatAsciiArb** = ((int)0x8875), **ProgramFormat** = ((int)0x8876),
ProgramFormatArb = ((int)0x8876), **WritePixelDataRangeNv** = ((int)0x8878), **ReadPixelDataRangeNv** = ((int)0x8879), **WritePixelDataRangeLengthNv** = ((int)0x887A),
ReadPixelDataRangeLengthNv = ((int)0x887B), **WritePixelDataRangePointerNv** = ((int)0x887C), **ReadPixelDataRangePointerNv** = ((int)0x887D), **FloatRNv** = ((int)0x8880),
FloatRgNv = ((int)0x8881), **FloatRgbNv** = ((int)0x8882), **FloatRgbaNv** = ((int)0x8883), **FloatR16Nv** = ((int)0x8884),
FloatR32Nv = ((int)0x8885), **FloatRg16Nv** = ((int)0x8886), **FloatRg32Nv** = ((int)0x8887), **FloatRgb16Nv** = ((int)0x8888),
FloatRgb32Nv = ((int)0x8889), **FloatRgba16Nv** = ((int)0x888A), **FloatRgba32Nv** = ((int)0x888B), **TextureFloatComponentsNv** = ((int)0x888C),
FloatClearColorValueNv = ((int)0x888D), **FloatRgbaModeNv** = ((int)0x888E), **TextureUnsignedRemapModeNv** = ((int)0x888F), **DepthBoundsTestExt** = ((int)0x8890),
DepthBoundsExt = ((int)0x8891), **ArrayBuffer** = ((int)0x8892), **ArrayBufferArb** = ((int)0x8892), **ElementArrayBuffer** = ((int)0x8893),
ElementArrayBufferArb = ((int)0x8893), **ArrayBufferBinding** = ((int)0x8894), **ArrayBufferBindingArb** = ((int)0x8894), **ElementArrayBufferBinding** = ((int)0x8895),
ElementArrayBufferBindingArb = ((int)0x8895), **VertexArrayBufferBinding** = ((int)0x8896), **VertexArrayBufferBindingArb** = ((int)0x8896), **NormalArrayBufferBinding** = ((int)0x8897),
NormalArrayBufferBindingArb = ((int)0x8897), **ColorArrayBufferBinding** = ((int)0x8898), **ColorArrayBufferBindingArb** = ((int)0x8898), **IndexArrayBufferBinding** = ((int)0x8899),
IndexArrayBufferBindingArb = ((int)0x8899), **TextureCoordArrayBufferBinding** = ((int)0x889A), **TextureCoordArrayBufferBindingArb** = ((int)0x889A), **EdgeFlagArrayBufferBinding** = ((int)0x889B),
EdgeFlagArrayBufferBindingArb = ((int)0x889B), **SecondaryColorArrayBufferBinding** = ((int)0x889C), **SecondaryColorArrayBufferBindingArb** = ((int)0x889C), **FogCoordArrayBufferBinding** = ((int)0x889D),
FogCoordinateArrayBufferBinding = ((int)0x889D), **FogCoordinateArrayBufferBindingArb** = ((int)0x889D), **WeightArrayBufferBinding** = ((int)0x889E), **WeightArrayBufferBindingArb** = ((int)0x889E),

VertexAttribArrayBufferBinding = ((int)0x889F), **VertexAttribArrayBufferBindingArb** = ((int)0x889F), **ProgramInstruction** = ((int)0x88A0), **ProgramInstructionsArb** = ((int)0x88A0),
MaxProgramInstructions = ((int)0x88A1), **MaxProgramInstructionsArb** = ((int)0x88A1), **ProgramNativeInstructions** = ((int)0x88A2), **ProgramNativeInstructionsArb** = ((int)0x88A2),
MaxProgramNativeInstructions = ((int)0x88A3), **MaxProgramNativeInstructionsArb** = ((int)0x88A3), **ProgramTemporaries** = ((int)0x88A4), **ProgramTemporariesArb** = ((int)0x88A4),
MaxProgramTemporaries = ((int)0x88A5), **MaxProgramTemporariesArb** = ((int)0x88A5), **ProgramNativeTemporaries** = ((int)0x88A6), **ProgramNativeTemporariesArb** = ((int)0x88A6),
MaxProgramNativeTemporaries = ((int)0x88A7), **MaxProgramNativeTemporariesArb** = ((int)0x88A7), **ProgramParameters** = ((int)0x88A8), **ProgramParametersArb** = ((int)0x88A8),
MaxProgramParameters = ((int)0x88A9), **MaxProgramParametersArb** = ((int)0x88A9), **ProgramNativeParameters** = ((int)0x88AA), **ProgramNativeParametersArb** = ((int)0x88AA),
MaxProgramNativeParameters = ((int)0x88AB), **MaxProgramNativeParametersArb** = ((int)0x88AB), **ProgramAttribs** = ((int)0x88AC), **ProgramAttribsArb** = ((int)0x88AC),
MaxProgramAttribs = ((int)0x88AD), **MaxProgramAttribsArb** = ((int)0x88AD), **ProgramNativeAttribs** = ((int)0x88AE), **ProgramNativeAttribsArb** = ((int)0x88AE),
MaxProgramNativeAttribs = ((int)0x88AF), **MaxProgramNativeAttribsArb** = ((int)0x88AF), **ProgramAddressRegisters** = ((int)0x88B0), **ProgramAddressRegistersArb** = ((int)0x88B0),
MaxProgramAddressRegisters = ((int)0x88B1), **MaxProgramAddressRegistersArb** = ((int)0x88B1), **ProgramNativeAddressRegisters** = ((int)0x88B2), **ProgramNativeAddressRegistersArb** = ((int)0x88B2),
MaxProgramNativeAddressRegisters = ((int)0x88B3), **MaxProgramNativeAddressRegistersArb** = ((int)0x88B3), **MaxProgramLocalParameters** = ((int)0x88B4), **MaxProgramLocalParametersArb** = ((int)0x88B4),
MaxProgramEnvParameters = ((int)0x88B5), **MaxProgramEnvParametersArb** = ((int)0x88B5), **ProgramUnderNativeLimits** = ((int)0x88B6), **ProgramUnderNativeLimitsArb** = ((int)0x88B6),
TransposeCurrentMatrixArb = ((int)0x88B7), **ReadOnly** = ((int)0x88B8), **ReadOnlyArb** = ((int)0x88B8), **WriteOnly** = ((int)0x88B9),
WriteOnlyArb = ((int)0x88B9), **ReadWrite** = ((int)0x88BA), **ReadWriteArb** = ((int)0x88BA), **BufferAccess** = ((int)0x88BB),
BufferAccessArb = ((int)0x88BB), **BufferMapped** = ((int)0x88BC), **BufferMappedArb** = ((int)0x88BC), **BufferMapPointer** = ((int)0x88BD),

```

BufferMapPointerArb = ((int)0x88BD), TimeElapsedExt = ((int)0x88BF),
Matrix0 = ((int)0x88C0), Matrix0Arb = ((int)0x88C0),
Matrix1 = ((int)0x88C1), Matrix1Arb = ((int)0x88C1), Matrix2 =
((int)0x88C2), Matrix2Arb = ((int)0x88C2),
Matrix3 = ((int)0x88C3), Matrix3Arb = ((int)0x88C3), Matrix4 =
((int)0x88C4), Matrix4Arb = ((int)0x88C4),
Matrix5 = ((int)0x88C5), Matrix5Arb = ((int)0x88C5), Matrix6 =
((int)0x88C6), Matrix6Arb = ((int)0x88C6),
Matrix7 = ((int)0x88C7), Matrix7Arb = ((int)0x88C7), Matrix8 =
((int)0x88C8), Matrix8Arb = ((int)0x88C8),
Matrix9 = ((int)0x88C9), Matrix9Arb = ((int)0x88C9), Matrix10 =
((int)0x88CA), Matrix10Arb = ((int)0x88CA),
Matrix11 = ((int)0x88CB), Matrix11Arb = ((int)0x88CB), Matrix12 =
((int)0x88CC), Matrix12Arb = ((int)0x88CC),
Matrix13 = ((int)0x88CD), Matrix13Arb = ((int)0x88CD), Matrix14 =
((int)0x88CE), Matrix14Arb = ((int)0x88CE),
Matrix15 = ((int)0x88CF), Matrix15Arb = ((int)0x88CF), Matrix16 =
((int)0x88D0), Matrix16Arb = ((int)0x88D0),
Matrix17 = ((int)0x88D1), Matrix17Arb = ((int)0x88D1), Matrix18 =
((int)0x88D2), Matrix18Arb = ((int)0x88D2),
Matrix19 = ((int)0x88D3), Matrix19Arb = ((int)0x88D3), Matrix20 =
((int)0x88D4), Matrix20Arb = ((int)0x88D4),
Matrix21 = ((int)0x88D5), Matrix21Arb = ((int)0x88D5), Matrix22 =
((int)0x88D6), Matrix22Arb = ((int)0x88D6),
Matrix23 = ((int)0x88D7), Matrix23Arb = ((int)0x88D7), Matrix24 =
((int)0x88D8), Matrix24Arb = ((int)0x88D8),
Matrix25 = ((int)0x88D9), Matrix25Arb = ((int)0x88D9), Matrix26 =
((int)0x88DA), Matrix26Arb = ((int)0x88DA),
Matrix27 = ((int)0x88DB), Matrix27Arb = ((int)0x88DB), Matrix28 =
((int)0x88DC), Matrix28Arb = ((int)0x88DC),
Matrix29 = ((int)0x88DD), Matrix29Arb = ((int)0x88DD), Matrix30 =
((int)0x88DE), Matrix30Arb = ((int)0x88DE),
Matrix31 = ((int)0x88DF), Matrix31Arb = ((int)0x88DF), StreamDraw =
((int)0x88E0), StreamDrawArb = ((int)0x88E0),
StreamRead = ((int)0x88E1), StreamReadArb = ((int)0x88E1), StreamCopy =
((int)0x88E2), StreamCopyArb = ((int)0x88E2),
StaticDraw = ((int)0x88E4), StaticDrawArb = ((int)0x88E4), StaticRead =
((int)0x88E5), StaticReadArb = ((int)0x88E5),
StaticCopy = ((int)0x88E6), StaticCopyArb = ((int)0x88E6), DynamicDraw =
((int)0x88E8), DynamicDrawArb = ((int)0x88E8),

```

DynamicRead = ((int)0x88E9), **DynamicReadArb** = ((int)0x88E9), **DynamicCopy** = ((int)0x88EA), **DynamicCopyArb** = ((int)0x88EA),

PixelPackBuffer = ((int)0x88EB), **PixelPackBufferArb** = ((int)0x88EB), **PixelPackBufferExt** = ((int)0x88EB), **PixelUnpackBuffer** = ((int)0x88EC),

PixelUnpackBufferArb = ((int)0x88EC), **PixelUnpackBufferExt** = ((int)0x88EC), **PixelPackBufferBinding** = ((int)0x88ED), **PixelPackBufferBindingArb** = ((int)0x88ED),

PixelPackBufferBindingExt = ((int)0x88ED), **PixelUnpackBufferBinding** = ((int)0x88EF), **PixelUnpackBufferBindingArb** = ((int)0x88EF), **PixelUnpackBufferBindingExt** = ((int)0x88EF),

Depth24Stencil8 = ((int)0x88F0), **Depth24Stencil8Ext** = ((int)0x88F0), **TextureStencilSize** = ((int)0x88F1), **TextureStencilSizeExt** = ((int)0x88F1),

StencilTagBitsExt = ((int)0x88F2), **StencilClearTagValueExt** = ((int)0x88F3), **MaxProgramExecInstructionsNv** = ((int)0x88F4), **MaxProgramCallDepthNv** = ((int)0x88F5),

MaxProgramIfDepthNv = ((int)0x88F6), **MaxProgramLoopDepthNv** = ((int)0x88F7), **MaxProgramLoopCountNv** = ((int)0x88F8), **VertexAttribArrayInteger** = ((int)0x88FD),

VertexAttribArrayIntegerNv = ((int)0x88FD), **ArrayDivisor** = ((int)0x88FE), **VertexAttribArrayDivisorArb** = ((int)0x88FE), **MaxArrayTextureLayers** = ((int)0x88FF),

MaxArrayTextureLayersExt = ((int)0x88FF), **MinProgramTexelOffset** = ((int)0x8904), **MinProgramTexelOffsetNv** = ((int)0x8904), **MaxProgramTexelOffset** = ((int)0x8905),

MaxProgramTexelOffsetNv = ((int)0x8905), **ProgramAttribComponentsNv** = ((int)0x8906), **ProgramResultComponentsNv** = ((int)0x8907), **MaxProgramAttribComponentsNv** = ((int)0x8908),

MaxProgramResultComponentsNv = ((int)0x8909), **StencilTestTwoSideExt** = ((int)0x8910), **ActiveStencilFaceExt** = ((int)0x8911), **MirrorClampToBorderExt** = ((int)0x8912),

SamplesPassed = ((int)0x8914), **SamplesPassedArb** = ((int)0x8914), **GeometryVerticesOut** = ((int)0x8916), **GeometryInputType** = ((int)0x8917),

GeometryOutputType = ((int)0x8918), **ClampVertexColor** = ((int)0x891A), **ClampVertexColorArb** = ((int)0x891A), **ClampFragmentColor** = ((int)0x891B),

ClampFragmentColorArb = ((int)0x891B), **ClampReadColor** = ((int)0x891C), **ClampReadColorArb** = ((int)0x891C), **FixedOnly** = ((int)0x891D),

FixedOnlyArb = ((int)0x891D), **FragmentShaderAti** = ((int)0x8920), **Reg0Ati** = ((int)0x8921), **Reg1Ati** = ((int)0x8922),

Reg2Ati = ((int)0x8923), **Reg3Ati** = ((int)0x8924), **Reg4Ati** = ((int)0x8925),
Reg5Ati = ((int)0x8926),
Reg6Ati = ((int)0x8927), **Reg7Ati** = ((int)0x8928), **Reg8Ati** = ((int)0x8929),
Reg9Ati = ((int)0x892A),
Reg10Ati = ((int)0x892B), **Reg11Ati** = ((int)0x892C), **Reg12Ati** =
((int)0x892D), **Reg13Ati** = ((int)0x892E),
Reg14Ati = ((int)0x892F), **Reg15Ati** = ((int)0x8930), **Reg16Ati** =
((int)0x8931), **Reg17Ati** = ((int)0x8932),
Reg18Ati = ((int)0x8933), **Reg19Ati** = ((int)0x8934), **Reg20Ati** =
((int)0x8935), **Reg21Ati** = ((int)0x8936),
Reg22Ati = ((int)0x8937), **Reg23Ati** = ((int)0x8938), **Reg24Ati** =
((int)0x8939), **Reg25Ati** = ((int)0x893A),
Reg26Ati = ((int)0x893B), **Reg27Ati** = ((int)0x893C), **Reg28Ati** =
((int)0x893D), **Reg29Ati** = ((int)0x893E),
Reg30Ati = ((int)0x893F), **Reg31Ati** = ((int)0x8940), **Con0Ati** = ((int)0x8941),
Con1Ati = ((int)0x8942),
Con2Ati = ((int)0x8943), **Con3Ati** = ((int)0x8944), **Con4Ati** = ((int)0x8945),
Con5Ati = ((int)0x8946),
Con6Ati = ((int)0x8947), **Con7Ati** = ((int)0x8948), **Con8Ati** = ((int)0x8949),
Con9Ati = ((int)0x894A),
Con10Ati = ((int)0x894B), **Con11Ati** = ((int)0x894C), **Con12Ati** =
((int)0x894D), **Con13Ati** = ((int)0x894E),
Con14Ati = ((int)0x894F), **Con15Ati** = ((int)0x8950), **Con16Ati** =
((int)0x8951), **Con17Ati** = ((int)0x8952),
Con18Ati = ((int)0x8953), **Con19Ati** = ((int)0x8954), **Con20Ati** =
((int)0x8955), **Con21Ati** = ((int)0x8956),
Con22Ati = ((int)0x8957), **Con23Ati** = ((int)0x8958), **Con24Ati** =
((int)0x8959), **Con25Ati** = ((int)0x895A),
Con26Ati = ((int)0x895B), **Con27Ati** = ((int)0x895C), **Con28Ati** =
((int)0x895D), **Con29Ati** = ((int)0x895E),
Con30Ati = ((int)0x895F), **Con31Ati** = ((int)0x8960), **MovAti** = ((int)0x8961),
AddAti = ((int)0x8963),
MulAti = ((int)0x8964), **SubAti** = ((int)0x8965), **Dot3Ati** = ((int)0x8966),
Dot4Ati = ((int)0x8967),
MadAti = ((int)0x8968), **LerpAti** = ((int)0x8969), **CndAti** = ((int)0x896A),
Cnd0Ati = ((int)0x896B),
Dot2AddAti = ((int)0x896C), **SecondaryInterpolatorAti** = ((int)0x896D),
NumFragmentRegistersAti = ((int)0x896E), **NumFragmentConstantsAti** =
((int)0x896F),

NumPassesAti = ((int)0x8970), **NumInstructionsPerPassAti** = ((int)0x8971),
NumInstructionsTotalAti = ((int)0x8972), **NumInputInterpolatorComponentsAti** = ((int)0x8973),
NumLoopbackComponentsAti = ((int)0x8974), **ColorAlphaPairingAti** = ((int)0x8975), **SwizzleStrAti** = ((int)0x8976), **SwizzleStqAti** = ((int)0x8977),
SwizzleStrDrAti = ((int)0x8978), **SwizzleStqDqAti** = ((int)0x8979), **SwizzleStrqAti** = ((int)0x897A), **SwizzleStrqDqAti** = ((int)0x897B),
InterlaceOml = ((int)0x8980), **InterlaceReadOml** = ((int)0x8981), **FormatSubsample2424Oml** = ((int)0x8982), **FormatSubsample244244Oml** = ((int)0x8983),
PackResampleOml = ((int)0x8984), **UnpackResampleOml** = ((int)0x8985), **ResampleReplicateOml** = ((int)0x8986), **ResampleZeroFillOml** = ((int)0x8987),
ResampleAverageOml = ((int)0x8988), **ResampleDecimateOml** = ((int)0x8989), **VertexAttribMap1Apple** = ((int)0x8A00), **VertexAttribMap2Apple** = ((int)0x8A01),
VertexAttribMap1SizeApple = ((int)0x8A02), **VertexAttribMap1CoeffApple** = ((int)0x8A03), **VertexAttribMap1OrderApple** = ((int)0x8A04), **VertexAttribMap1DomainApple** = ((int)0x8A05),
VertexAttribMap2SizeApple = ((int)0x8A06), **VertexAttribMap2CoeffApple** = ((int)0x8A07), **VertexAttribMap2OrderApple** = ((int)0x8A08), **VertexAttribMap2DomainApple** = ((int)0x8A09),
DrawPixelsApple = ((int)0x8A0A), **FenceApple** = ((int)0x8A0B), **ColorFloatApple** = ((int)0x8A0F), **UniformBuffer** = ((int)0x8A11),
BufferSerializedModifyApple = ((int)0x8A12), **BufferFlushingUnmapApple** = ((int)0x8A13), **AuxDepthStencilApple** = ((int)0x8A14), **PackRowBytesApple** = ((int)0x8A15),
UnpackRowBytesApple = ((int)0x8A16), **ReleasedApple** = ((int)0x8A19), **VolatileApple** = ((int)0x8A1A), **RetainedApple** = ((int)0x8A1B),
UndefinedApple = ((int)0x8A1C), **PurgeableApple** = ((int)0x8A1D), **UniformBufferBinding** = ((int)0x8A28), **UniformBufferStart** = ((int)0x8A29),
UniformBufferSize = ((int)0x8A2A), **MaxVertexUniformBlocks** = ((int)0x8A2B), **MaxGeometryUniformBlocks** = ((int)0x8A2C), **MaxFragmentUniformBlocks** = ((int)0x8A2D),
MaxCombinedUniformBlocks = ((int)0x8A2E), **MaxUniformBufferBindings** = ((int)0x8A2F), **MaxUniformBlockSize** = ((int)0x8A30), **MaxCombinedVertexUniformComponents** = ((int)0x8A31),
MaxCombinedGeometryUniformComponents = ((int)0x8A32), **MaxCombinedFragmentUniformComponents** = ((int)0x8A33), **UniformBufferOffsetAlignment** = ((int)0x8A34), **ActiveUniformBlockMaxNameLength** = ((int)0x8A35),

ActiveUniformBlocks = ((int)0x8A36), **UniformType** = ((int)0x8A37), **UniformSize** = ((int)0x8A38), **UniformNameLength** = ((int)0x8A39),
UniformBlockIndex = ((int)0x8A3A), **UniformOffset** = ((int)0x8A3B), **UniformArrayStride** = ((int)0x8A3C), **UniformMatrixStride** = ((int)0x8A3D),
UniformIsRowMajor = ((int)0x8A3E), **UniformBlockBinding** = ((int)0x8A3F), **UniformBlockDataSize** = ((int)0x8A40), **UniformBlockNameLength** = ((int)0x8A41),
UniformBlockActiveUniforms = ((int)0x8A42), **UniformBlockActiveUniformIndices** = ((int)0x8A43), **UniformBlockReferencedByVertexShader** = ((int)0x8A44), **UniformBlockReferencedByGeometryShader** = ((int)0x8A45),
UniformBlockReferencedByFragmentShader = ((int)0x8A46), **FragmentShader** = ((int)0x8B30), **FragmentShaderArb** = ((int)0x8B30), **VertexShader** = ((int)0x8B31),
VertexShaderArb = ((int)0x8B31), **ProgramObjectArb** = ((int)0x8B40), **ShaderObjectArb** = ((int)0x8B48), **MaxFragmentUniformComponents** = ((int)0x8B49),
MaxFragmentUniformComponentsArb = ((int)0x8B49), **MaxVertexUniformComponents** = ((int)0x8B4A), **MaxVertexUniformComponentsArb** = ((int)0x8B4A), **MaxVaryingComponents** = ((int)0x8B4B),
MaxVaryingComponentsExt = ((int)0x8B4B), **MaxVaryingFloats** = ((int)0x8B4B), **MaxVaryingFloatsArb** = ((int)0x8B4B), **MaxVertexTextureImageUnits** = ((int)0x8B4C),
MaxVertexTextureImageUnitsArb = ((int)0x8B4C), **MaxCombinedTextureImageUnits** = ((int)0x8B4D), **MaxCombinedTextureImageUnitsArb** = ((int)0x8B4D), **ObjectTypeArb** = ((int)0x8B4E),
ObjectSubtypeArb = ((int)0x8B4F), **ShaderType** = ((int)0x8B4F), **FloatVec2** = ((int)0x8B50), **FloatVec2Arb** = ((int)0x8B50),
FloatVec3 = ((int)0x8B51), **FloatVec3Arb** = ((int)0x8B51), **FloatVec4** = ((int)0x8B52), **FloatVec4Arb** = ((int)0x8B52),
IntVec2 = ((int)0x8B53), **IntVec2Arb** = ((int)0x8B53), **IntVec3** = ((int)0x8B54), **IntVec3Arb** = ((int)0x8B54),
IntVec4 = ((int)0x8B55), **IntVec4Arb** = ((int)0x8B55), **Bool** = ((int)0x8B56), **BoolArb** = ((int)0x8B56),
BoolVec2 = ((int)0x8B57), **BoolVec2Arb** = ((int)0x8B57), **BoolVec3** = ((int)0x8B58), **BoolVec3Arb** = ((int)0x8B58),
BoolVec4 = ((int)0x8B59), **BoolVec4Arb** = ((int)0x8B59), **FloatMat2** = ((int)0x8B5A), **FloatMat2Arb** = ((int)0x8B5A),
FloatMat3 = ((int)0x8B5B), **FloatMat3Arb** = ((int)0x8B5B), **FloatMat4** = ((int)0x8B5C), **FloatMat4Arb** = ((int)0x8B5C),

Sampler1D = ((int)0x8B5D), **Sampler1DArb** = ((int)0x8B5D), **Sampler2D** = ((int)0x8B5E), **Sampler2DArb** = ((int)0x8B5E),

Sampler3D = ((int)0x8B5F), **Sampler3DArb** = ((int)0x8B5F), **SamplerCube** = ((int)0x8B60), **SamplerCubeArb** = ((int)0x8B60),

Sampler1DShadow = ((int)0x8B61), **Sampler1DShadowArb** = ((int)0x8B61), **Sampler2DShadow** = ((int)0x8B62), **Sampler2DShadowArb** = ((int)0x8B62),

Sampler2DRect = ((int)0x8B63), **Sampler2DRectArb** = ((int)0x8B63), **Sampler2DRectShadow** = ((int)0x8B64), **Sampler2DRectShadowArb** = ((int)0x8B64),

FloatMat2x3 = ((int)0x8B65), **FloatMat2x4** = ((int)0x8B66), **FloatMat3x2** = ((int)0x8B67), **FloatMat3x4** = ((int)0x8B68),

FloatMat4x2 = ((int)0x8B69), **FloatMat4x3** = ((int)0x8B6A), **DeleteStatus** = ((int)0x8B80), **ObjectDeleteStatusArb** = ((int)0x8B80),

CompileStatus = ((int)0x8B81), **ObjectCompileStatusArb** = ((int)0x8B81), **LinkStatus** = ((int)0x8B82), **ObjectLinkStatusArb** = ((int)0x8B82),

ObjectValidateStatusArb = ((int)0x8B83), **ValidateStatus** = ((int)0x8B83), **InfoLogLength** = ((int)0x8B84), **ObjectInfoLogLengthArb** = ((int)0x8B84),

AttachedShaders = ((int)0x8B85), **ObjectAttachedObjectsArb** = ((int)0x8B85), **ActiveUniforms** = ((int)0x8B86), **ObjectActiveUniformsArb** = ((int)0x8B86),

ActiveUniformMaxLength = ((int)0x8B87), **ObjectActiveUniformMaxLengthArb** = ((int)0x8B87), **ObjectShaderSourceLengthArb** = ((int)0x8B88), **ShaderSourceLength** = ((int)0x8B88),

ActiveAttributes = ((int)0x8B89), **ObjectActiveAttributesArb** = ((int)0x8B89), **ActiveAttributeMaxLength** = ((int)0x8B8A), **ObjectActiveAttributeMaxLengthArb** = ((int)0x8B8A),

FragmentShaderDerivativeHint = ((int)0x8B8B), **FragmentShaderDerivativeHintArb** = ((int)0x8B8B), **ShadingLanguageVersion** = ((int)0x8B8C), **ShadingLanguageVersionArb** = ((int)0x8B8C),

CurrentProgram = ((int)0x8B8D), **ImplementationColorReadTypeOes** = ((int)0x8B9A), **ImplementationColorReadFormatOes** = ((int)0x8B9B), **CounterTypeAmd** = ((int)0x8BC0),

CounterRangeAmd = ((int)0x8BC1), **UnsignedInt64Amd** = ((int)0x8BC2), **PercentageAmd** = ((int)0x8BC3), **PerfmonResultAvailableAmd** = ((int)0x8BC4),

PerfmonResultSizeAmd = ((int)0x8BC5), **PerfmonResultAmd** = ((int)0x8BC6), **TextureRedType** = ((int)0x8C10), **TextureRedTypeArb** = ((int)0x8C10),

TextureGreenType = ((int)0x8C11), **TextureGreenTypeArb** = ((int)0x8C11), **TextureBlueType** = ((int)0x8C12), **TextureBlueTypeArb** = ((int)0x8C12),

```
TextureAlphaType = ((int)0x8C13), TextureAlphaTypeArb = ((int)0x8C13),
TextureLuminanceType = ((int)0x8C14), TextureLuminanceTypeArb =
((int)0x8C14),
TextureIntensityType = ((int)0x8C15), TextureIntensityTypeArb =
((int)0x8C15), TextureDepthType = ((int)0x8C16), TextureDepthTypeArb =
((int)0x8C16),
UnsignedNormalized = ((int)0x8C17), UnsignedNormalizedArb =
((int)0x8C17), Texture1DArray = ((int)0x8C18), Texture1DArrayExt =
((int)0x8C18),
ProxyTexture1DArray = ((int)0x8C19), ProxyTexture1DArrayExt =
((int)0x8C19), Texture2DArray = ((int)0x8C1A), Texture2DArrayExt =
((int)0x8C1A),
ProxyTexture2DArray = ((int)0x8C1B), ProxyTexture2DArrayExt =
((int)0x8C1B), TextureBinding1DArray = ((int)0x8C1C), TextureBind-
ing1DArrayExt = ((int)0x8C1C),
TextureBinding2DArray = ((int)0x8C1D), TextureBinding2DArrayExt =
((int)0x8C1D), GeometryProgramNv = ((int)0x8C26), MaxProgramOut-
putVerticesNv = ((int)0x8C27),
MaxProgramTotalOutputComponentsNv = ((int)0x8C28), MaxGeometry-
TextureImageUnits = ((int)0x8C29), MaxGeometryTextureImageUnitsArb =
((int)0x8C29), MaxGeometryTextureImageUnitsExt = ((int)0x8C29),
TextureBuffer = ((int)0x8C2A), TextureBufferArb = ((int)0x8C2A), Texture-
BufferExt = ((int)0x8C2A), MaxTextureBufferSize = ((int)0x8C2B),
MaxTextureBufferSizeArb = ((int)0x8C2B), MaxTextureBufferSizeExt =
((int)0x8C2B), TextureBindingBuffer = ((int)0x8C2C), TextureBinding-
BufferArb = ((int)0x8C2C),
TextureBindingBufferExt = ((int)0x8C2C), TextureBufferDataStoreBinding
= ((int)0x8C2D), TextureBufferDataStoreBindingArb = ((int)0x8C2D), Tex-
tureBufferDataStoreBindingExt = ((int)0x8C2D),
TextureBufferFormat = ((int)0x8C2E), TextureBufferFormatArb =
((int)0x8C2E), TextureBufferFormatExt = ((int)0x8C2E), SampleShad-
ing = ((int)0x8C36),
MinSampleShadingValue = ((int)0x8C37), R11fG11fB10f = ((int)0x8C3A),
R11fG11fB10fExt = ((int)0x8C3A), UnsignedInt10F11F11FRev =
((int)0x8C3B),
UnsignedInt10F11F11FRevExt = ((int)0x8C3B), RgbaSignedComponent-
sExt = ((int)0x8C3C), Rgb9E5 = ((int)0x8C3D), Rgb9E5Ext = ((int)0x8C3D),
UnsignedInt5999Rev = ((int)0x8C3E), UnsignedInt5999RevExt =
((int)0x8C3E), TextureSharedSize = ((int)0x8C3F), TextureSharedSize-
Ext = ((int)0x8C3F),
Srgb = ((int)0x8C40), SrgbExt = ((int)0x8C40), Srgb8 = ((int)0x8C41),
Srgb8Ext = ((int)0x8C41),
```

SrgbAlpha = ((int)0x8C42), **SrgbAlphaExt** = ((int)0x8C42), **Srgb8Alpha8** = ((int)0x8C43), **Srgb8Alpha8Ext** = ((int)0x8C43),

SluminanceAlpha = ((int)0x8C44), **SluminanceAlphaExt** = ((int)0x8C44), **Sluminance8Alpha8** = ((int)0x8C45), **Sluminance8Alpha8Ext** = ((int)0x8C45),

Sluminance = ((int)0x8C46), **SluminanceExt** = ((int)0x8C46), **Sluminance8** = ((int)0x8C47), **Sluminance8Ext** = ((int)0x8C47),

CompressedSrgb = ((int)0x8C48), **CompressedSrgbExt** = ((int)0x8C48), **CompressedSrgbAlpha** = ((int)0x8C49), **CompressedSrgbAlphaExt** = ((int)0x8C49),

CompressedSluminance = ((int)0x8C4A), **CompressedSluminanceExt** = ((int)0x8C4A), **CompressedSluminanceAlpha** = ((int)0x8C4B), **CompressedSluminanceAlphaExt** = ((int)0x8C4B),

CompressedSrgbS3tcDxt1Ext = ((int)0x8C4C), **CompressedSrgbAlphaS3tcDxt1Ext** = ((int)0x8C4D), **CompressedSrgbAlphaS3tcDxt3Ext** = ((int)0x8C4E), **CompressedSrgbAlphaS3tcDxt5Ext** = ((int)0x8C4F),

CompressedLuminanceLatc1Ext = ((int)0x8C70), **CompressedSignedLuminanceLatc1Ext** = ((int)0x8C71), **CompressedLuminanceAlphaLatc2Ext** = ((int)0x8C72), **CompressedSignedLuminanceAlphaLatc2Ext** = ((int)0x8C73),

TransformFeedbackVaryingMaxLength = ((int)0x8C76), **TransformFeedbackVaryingMaxLengthExt** = ((int)0x8C76), **BackPrimaryColorNv** = ((int)0x8C77), **BackSecondaryColorNv** = ((int)0x8C78),

TextureCoordNv = ((int)0x8C79), **ClipDistanceNv** = ((int)0x8C7A), **VertexIdNv** = ((int)0x8C7B), **PrimitiveIdNv** = ((int)0x8C7C),

GenericAttribNv = ((int)0x8C7D), **TransformFeedbackAttribsNv** = ((int)0x8C7E), **TransformFeedbackBufferMode** = ((int)0x8C7F), **TransformFeedbackBufferModeExt** = ((int)0x8C7F),

TransformFeedbackBufferModeNv = ((int)0x8C7F), **MaxTransformFeedbackSeparateComponents** = ((int)0x8C80), **MaxTransformFeedbackSeparateComponentsExt** = ((int)0x8C80), **MaxTransformFeedbackSeparateComponentsNv** = ((int)0x8C80),

ActiveVaryingsNv = ((int)0x8C81), **ActiveVaryingMaxLengthNv** = ((int)0x8C82), **TransformFeedbackVaryings** = ((int)0x8C83), **TransformFeedbackVaryingsExt** = ((int)0x8C83),

TransformFeedbackVaryingsNv = ((int)0x8C83), **TransformFeedbackBufferStart** = ((int)0x8C84), **TransformFeedbackBufferStartExt** = ((int)0x8C84), **TransformFeedbackBufferStartNv** = ((int)0x8C84),

TransformFeedbackBufferSize = ((int)0x8C85), **TransformFeedbackBufferSizeExt** = ((int)0x8C85), **TransformFeedbackBufferSizeNv** = ((int)0x8C85), **TransformFeedbackRecordNv** = ((int)0x8C86),

PrimitivesGenerated = ((int)0x8C87), **PrimitivesGeneratedExt** = ((int)0x8C87), **PrimitivesGeneratedNv** = ((int)0x8C87), **TransformFeedbackPrimitivesWritten** = ((int)0x8C88),

TransformFeedbackPrimitivesWrittenExt = ((int)0x8C88), **TransformFeedbackPrimitivesWrittenNv** = ((int)0x8C88), **RasterizerDiscard** = ((int)0x8C89), **RasterizerDiscardExt** = ((int)0x8C89),

RasterizerDiscardNv = ((int)0x8C89), **MaxTransformFeedbackInterleavedAttribsNv** = ((int)0x8C8A), **MaxTransformFeedbackInterleavedComponents** = ((int)0x8C8A), **MaxTransformFeedbackInterleavedComponentsExt** = ((int)0x8C8A),

MaxTransformFeedbackSeparateAttribs = ((int)0x8C8B), **MaxTransformFeedbackSeparateAttribsExt** = ((int)0x8C8B), **MaxTransformFeedbackSeparateAttribsNv** = ((int)0x8C8B), **InterleavedAttribs** = ((int)0x8C8C),

InterleavedAttribsExt = ((int)0x8C8C), **InterleavedAttribsNv** = ((int)0x8C8C), **SeparateAttribs** = ((int)0x8C8D), **SeparateAttribsExt** = ((int)0x8C8D),

SeparateAttribsNv = ((int)0x8C8D), **TransformFeedbackBuffer** = ((int)0x8C8E), **TransformFeedbackBufferExt** = ((int)0x8C8E), **TransformFeedbackBufferNv** = ((int)0x8C8E),

TransformFeedbackBufferBinding = ((int)0x8C8F), **TransformFeedbackBufferBindingExt** = ((int)0x8C8F), **TransformFeedbackBufferBindingNv** = ((int)0x8C8F), **PointSpriteCoordOrigin** = ((int)0x8CA0),

LowerLeft = ((int)0x8CA1), **UpperLeft** = ((int)0x8CA2), **StencilBackRef** = ((int)0x8CA3), **StencilBackValueMask** = ((int)0x8CA4),

StencilBackWritemask = ((int)0x8CA5), **DrawFramebufferBinding** = ((int)0x8CA6), **DrawFramebufferBindingExt** = ((int)0x8CA6), **FramebufferBinding** = ((int)0x8CA6),

FramebufferBindingExt = ((int)0x8CA6), **RenderbufferBinding** = ((int)0x8CA7), **RenderbufferBindingExt** = ((int)0x8CA7), **ReadFramebuffer** = ((int)0x8CA8),

ReadFramebufferExt = ((int)0x8CA8), **DrawFramebuffer** = ((int)0x8CA9), **DrawFramebufferExt** = ((int)0x8CA9), **ReadFramebufferBinding** = ((int)0x8CAA),

ReadFramebufferBindingExt = ((int)0x8CAA), **RenderbufferCoverageSamplesNv** = ((int)0x8CAB), **RenderbufferSamples** = ((int)0x8CAB), **RenderbufferSamplesExt** = ((int)0x8CAB),

DepthComponent32f = ((int)0x8CAC), **Depth32fStencil8** = ((int)0x8CAD), **FramebufferAttachmentObjectType** = ((int)0x8CD0), **FramebufferAttachmentObjectTypeExt** = ((int)0x8CD0),

FramebufferAttachmentObjectName = ((int)0x8CD1), **FramebufferAttachmentObjectNameExt** = ((int)0x8CD1), **FramebufferAttachmentTex-**

tureLevel = ((int)0x8CD2), **FramebufferAttachmentTextureLevelExt** = ((int)0x8CD2),

FramebufferAttachmentTextureCubeMapFace = ((int)0x8CD3), **FramebufferAttachmentTextureCubeMapFaceExt** = ((int)0x8CD3), **FramebufferAttachmentTexture3DZoffsetExt** = ((int)0x8CD4), **FramebufferAttachmentTextureLayer** = ((int)0x8CD4),

FramebufferAttachmentTextureLayerExt = ((int)0x8CD4), **FramebufferComplete** = ((int)0x8CD5), **FramebufferCompleteExt** = ((int)0x8CD5), **FramebufferIncompleteAttachment** = ((int)0x8CD6),

FramebufferIncompleteAttachmentExt = ((int)0x8CD6), **FramebufferIncompleteMissingAttachment** = ((int)0x8CD7), **FramebufferIncompleteMissingAttachmentExt** = ((int)0x8CD7), **FramebufferIncompleteDimensionsExt** = ((int)0x8CD9),

FramebufferIncompleteFormatsExt = ((int)0x8CDA), **FramebufferIncompleteDrawBuffer** = ((int)0x8CDB), **FramebufferIncompleteDrawBufferExt** = ((int)0x8CDB), **FramebufferIncompleteReadBuffer** = ((int)0x8CDC),

FramebufferIncompleteReadBufferExt = ((int)0x8CDC), **FramebufferUnsupported** = ((int)0x8CDD), **FramebufferUnsupportedExt** = ((int)0x8CDD), **MaxColorAttachments** = ((int)0x8CDF),

MaxColorAttachmentsExt = ((int)0x8CDF), **ColorAttachment0** = ((int)0x8CE0), **ColorAttachment0Ext** = ((int)0x8CE0), **ColorAttachment1** = ((int)0x8CE1),

ColorAttachment1Ext = ((int)0x8CE1), **ColorAttachment2** = ((int)0x8CE2), **ColorAttachment2Ext** = ((int)0x8CE2), **ColorAttachment3** = ((int)0x8CE3),

ColorAttachment3Ext = ((int)0x8CE3), **ColorAttachment4** = ((int)0x8CE4), **ColorAttachment4Ext** = ((int)0x8CE4), **ColorAttachment5** = ((int)0x8CE5),

ColorAttachment5Ext = ((int)0x8CE5), **ColorAttachment6** = ((int)0x8CE6), **ColorAttachment6Ext** = ((int)0x8CE6), **ColorAttachment7** = ((int)0x8CE7),

ColorAttachment7Ext = ((int)0x8CE7), **ColorAttachment8** = ((int)0x8CE8), **ColorAttachment8Ext** = ((int)0x8CE8), **ColorAttachment9** = ((int)0x8CE9),

ColorAttachment9Ext = ((int)0x8CE9), **ColorAttachment10** = ((int)0x8CEA), **ColorAttachment10Ext** = ((int)0x8CEA), **ColorAttachment11** = ((int)0x8CEB),

ColorAttachment11Ext = ((int)0x8CEB), **ColorAttachment12** = ((int)0x8CEC), **ColorAttachment12Ext** = ((int)0x8CEC), **ColorAttachment13** = ((int)0x8CED),

ColorAttachment13Ext = ((int)0x8CED), **ColorAttachment14** = ((int)0x8CEE), **ColorAttachment14Ext** = ((int)0x8CEE), **ColorAttachment15** = ((int)0x8CEF),

ColorAttachment15Ext = ((int)0x8CEF), **DepthAttachment** = ((int)0x8D00), **DepthAttachmentExt** = ((int)0x8D00), **StencilAttachment** = ((int)0x8D20),

StencilAttachmentExt = ((int)0x8D20), **Framebuffer** = ((int)0x8D40),
FramebufferExt = ((int)0x8D40), **Renderbuffer** = ((int)0x8D41),
RenderbufferExt = ((int)0x8D41), **RenderbufferWidth** = ((int)0x8D42), **RenderbufferWidthExt** = ((int)0x8D42), **RenderbufferHeight** = ((int)0x8D43),
RenderbufferHeightExt = ((int)0x8D43), **RenderbufferInternalFormat** = ((int)0x8D44), **RenderbufferInternalFormatExt** = ((int)0x8D44), **StencilIndex1** = ((int)0x8D46),
StencilIndex1Ext = ((int)0x8D46), **StencilIndex4** = ((int)0x8D47), **StencilIndex4Ext** = ((int)0x8D47), **StencilIndex8** = ((int)0x8D48),
StencilIndex8Ext = ((int)0x8D48), **StencilIndex16** = ((int)0x8D49), **StencilIndex16Ext** = ((int)0x8D49), **RenderbufferRedSize** = ((int)0x8D50),
RenderbufferRedSizeExt = ((int)0x8D50), **RenderbufferGreenSize** = ((int)0x8D51), **RenderbufferGreenSizeExt** = ((int)0x8D51), **RenderbufferBlueSize** = ((int)0x8D52),
RenderbufferBlueSizeExt = ((int)0x8D52), **RenderbufferAlphaSize** = ((int)0x8D53), **RenderbufferAlphaSizeExt** = ((int)0x8D53), **RenderbufferDepthSize** = ((int)0x8D54),
RenderbufferDepthSizeExt = ((int)0x8D54), **RenderbufferStencilSize** = ((int)0x8D55), **RenderbufferStencilSizeExt** = ((int)0x8D55), **FramebufferIncompleteMultisample** = ((int)0x8D56),
FramebufferIncompleteMultisampleExt = ((int)0x8D56), **MaxSamples** = ((int)0x8D57), **MaxSamplesExt** = ((int)0x8D57), **Rgba32ui** = ((int)0x8D70),
Rgba32uiExt = ((int)0x8D70), **Rgb32ui** = ((int)0x8D71), **Rgb32uiExt** = ((int)0x8D71), **Alpha32uiExt** = ((int)0x8D72),
Intensity32uiExt = ((int)0x8D73), **Luminance32uiExt** = ((int)0x8D74), **LuminanceAlpha32uiExt** = ((int)0x8D75), **Rgba16ui** = ((int)0x8D76),
Rgba16uiExt = ((int)0x8D76), **Rgb16ui** = ((int)0x8D77), **Rgb16uiExt** = ((int)0x8D77), **Alpha16uiExt** = ((int)0x8D78),
Intensity16uiExt = ((int)0x8D79), **Luminance16uiExt** = ((int)0x8D7A), **LuminanceAlpha16uiExt** = ((int)0x8D7B), **Rgba8ui** = ((int)0x8D7C),
Rgba8uiExt = ((int)0x8D7C), **Rgb8ui** = ((int)0x8D7D), **Rgb8uiExt** = ((int)0x8D7D), **Alpha8uiExt** = ((int)0x8D7E),
Intensity8uiExt = ((int)0x8D7F), **Luminance8uiExt** = ((int)0x8D80), **LuminanceAlpha8uiExt** = ((int)0x8D81), **Rgba32i** = ((int)0x8D82),
Rgba32iExt = ((int)0x8D82), **Rgb32i** = ((int)0x8D83), **Rgb32iExt** = ((int)0x8D83), **Alpha32iExt** = ((int)0x8D84),
Intensity32iExt = ((int)0x8D85), **Luminance32iExt** = ((int)0x8D86), **LuminanceAlpha32iExt** = ((int)0x8D87), **Rgba16i** = ((int)0x8D88),
Rgba16iExt = ((int)0x8D88), **Rgb16i** = ((int)0x8D89), **Rgb16iExt** = ((int)0x8D89), **Alpha16iExt** = ((int)0x8D8A),

Intensity16iExt = ((int)0x8D8B), **Luminance16iExt** = ((int)0x8D8C), **LuminanceAlpha16iExt** = ((int)0x8D8D), **Rgba8i** = ((int)0x8D8E),
Rgba8iExt = ((int)0x8D8E), **Rgb8i** = ((int)0x8D8F), **Rgb8iExt** = ((int)0x8D8F), **Alpha8iExt** = ((int)0x8D90),
Intensity8iExt = ((int)0x8D91), **Luminance8iExt** = ((int)0x8D92), **LuminanceAlpha8iExt** = ((int)0x8D93), **RedInteger** = ((int)0x8D94),
RedIntegerExt = ((int)0x8D94), **GreenInteger** = ((int)0x8D95), **GreenIntegerExt** = ((int)0x8D95), **BlueInteger** = ((int)0x8D96),
BlueIntegerExt = ((int)0x8D96), **AlphaInteger** = ((int)0x8D97), **AlphaIntegerExt** = ((int)0x8D97), **RgbInteger** = ((int)0x8D98),
RgbIntegerExt = ((int)0x8D98), **RgbaInteger** = ((int)0x8D99), **RgbaIntegerExt** = ((int)0x8D99), **BgrInteger** = ((int)0x8D9A),
BgrIntegerExt = ((int)0x8D9A), **BgraInteger** = ((int)0x8D9B), **BgraIntegerExt** = ((int)0x8D9B), **LuminanceIntegerExt** = ((int)0x8D9C),
LuminanceAlphaIntegerExt = ((int)0x8D9D), **RgbaIntegerModeExt** = ((int)0x8D9E), **MaxProgramParameterBufferBindingsNv** = ((int)0x8DA0),
MaxProgramParameterBufferSizeNv = ((int)0x8DA1),
VertexProgramParameterBufferNv = ((int)0x8DA2), **GeometryProgramParameterBufferNv** = ((int)0x8DA3), **FragmentProgramParameterBufferNv** = ((int)0x8DA4), **MaxProgramGenericAttribsNv** = ((int)0x8DA5),
MaxProgramGenericResultsNv = ((int)0x8DA6), **FramebufferAttachmentLayered** = ((int)0x8DA7), **FramebufferAttachmentLayeredArb** = ((int)0x8DA7), **FramebufferAttachmentLayeredExt** = ((int)0x8DA7),
FramebufferIncompleteLayerTargets = ((int)0x8DA8), **FramebufferIncompleteLayerTargetsArb** = ((int)0x8DA8), **FramebufferIncompleteLayerTargetsExt** = ((int)0x8DA8), **FramebufferIncompleteLayerCount** = ((int)0x8DA9),
FramebufferIncompleteLayerCountArb = ((int)0x8DA9), **FramebufferIncompleteLayerCountExt** = ((int)0x8DA9), **DepthComponent32fNv** = ((int)0x8DAB), **Depth32fStencil8Nv** = ((int)0x8DAC),
Float32UnsignedInt248Rev = ((int)0x8DAD), **Float32UnsignedInt248RevNv** = ((int)0x8DAD), **DepthBufferFloatModeNv** = ((int)0x8DAF), **FramebufferSrgb** = ((int)0x8DB9),
FramebufferSrgbExt = ((int)0x8DB9), **FramebufferSrgbCapableExt** = ((int)0x8DBA), **CompressedRedRgtc1** = ((int)0x8DBB), **CompressedRedRgtc1Ext** = ((int)0x8DBB),
CompressedSignedRedRgtc1 = ((int)0x8DBC), **CompressedSignedRedRgtc1Ext** = ((int)0x8DBC), **CompressedRedGreenRgtc2Ext** = ((int)0x8DBD), **CompressedRgRgtc2** = ((int)0x8DBD),
CompressedSignedRedGreenRgtc2Ext = ((int)0x8DBE), **CompressedSignedRgRgtc2** = ((int)0x8DBE), **Sampler1DArray** = ((int)0x8DC0), **Sampler1DArrayExt** = ((int)0x8DC0),

```

Sampler2DArray = ((int)0x8DC1), Sampler2DArrayExt = ((int)0x8DC1),
SamplerBuffer = ((int)0x8DC2), SamplerBufferExt = ((int)0x8DC2),

Sampler1DArrayShadow = ((int)0x8DC3), Sampler1DArrayShadowExt
= ((int)0x8DC3), Sampler2DArrayShadow = ((int)0x8DC4), Sam-
pler2DArrayShadowExt = ((int)0x8DC4),

SamplerCubeShadow = ((int)0x8DC5), SamplerCubeShadowExt =
((int)0x8DC5), UnsignedIntVec2 = ((int)0x8DC6), UnsignedIntVec2Ext
= ((int)0x8DC6),

UnsignedIntVec3 = ((int)0x8DC7), UnsignedIntVec3Ext = ((int)0x8DC7),
UnsignedIntVec4 = ((int)0x8DC8), UnsignedIntVec4Ext = ((int)0x8DC8),

IntSampler1D = ((int)0x8DC9), IntSampler1DExt = ((int)0x8DC9), IntSam-
pler2D = ((int)0x8DCA), IntSampler2DExt = ((int)0x8DCA),

IntSampler3D = ((int)0x8DCB), IntSampler3DExt = ((int)0x8DCB), IntSam-
plerCube = ((int)0x8DCC), IntSamplerCubeExt = ((int)0x8DCC),

IntSampler2DRect = ((int)0x8DCD), IntSampler2DRectExt =
((int)0x8DCD), IntSampler1DArray = ((int)0x8DCE), IntSam-
pler1DArrayExt = ((int)0x8DCE),

IntSampler2DArray = ((int)0x8DCF), IntSampler2DArrayExt =
((int)0x8DCF), IntSamplerBuffer = ((int)0x8DD0), IntSamplerBuffer-
Ext = ((int)0x8DD0),

UnsignedIntSampler1D = ((int)0x8DD1), UnsignedIntSampler1DExt =
((int)0x8DD1), UnsignedIntSampler2D = ((int)0x8DD2), UnsignedIntSam-
pler2DExt = ((int)0x8DD2),

UnsignedIntSampler3D = ((int)0x8DD3), UnsignedIntSampler3DExt =
((int)0x8DD3), UnsignedIntSamplerCube = ((int)0x8DD4), Unsigned-
IntSamplerCubeExt = ((int)0x8DD4),

UnsignedIntSampler2DRect = ((int)0x8DD5), UnsignedIntSam-
pler2DRectExt = ((int)0x8DD5), UnsignedIntSampler1DArray =
((int)0x8DD6), UnsignedIntSampler1DArrayExt = ((int)0x8DD6),

UnsignedIntSampler2DArray = ((int)0x8DD7), UnsignedIntSam-
pler2DArrayExt = ((int)0x8DD7), UnsignedIntSamplerBuffer =
((int)0x8DD8), UnsignedIntSamplerBufferExt = ((int)0x8DD8),

GeometryShader = ((int)0x8DD9), GeometryShaderArb = ((int)0x8DD9),
GeometryShaderExt = ((int)0x8DD9), GeometryVerticesOutArb =
((int)0x8DDA),

GeometryVerticesOutExt = ((int)0x8DDA), GeometryInputTypeArb =
((int)0x8ddb), GeometryInputTypeExt = ((int)0x8ddb), GeometryOutput-
TypeArb = ((int)0x8DDC),

GeometryOutputTypeExt = ((int)0x8DDC), MaxGeometryVaryingCom-
ponents = ((int)0x8DDD), MaxGeometryVaryingComponentsArb =
((int)0x8DDD), MaxGeometryVaryingComponentsExt = ((int)0x8DDD),

```

MaxVertexVaryingComponents = ((int)0x8DDE), **MaxVertexVaryingComponentsArb** = ((int)0x8DDE), **MaxVertexVaryingComponentsExt** = ((int)0x8DDE), **MaxGeometryUniformComponents** = ((int)0x8DDF),
MaxGeometryUniformComponentsArb = ((int)0x8DDF), **MaxGeometryUniformComponentsExt** = ((int)0x8DDF), **MaxGeometryOutputVertices** = ((int)0x8DE0), **MaxGeometryOutputVerticesArb** = ((int)0x8DE0),
MaxGeometryOutputVerticesExt = ((int)0x8DE0), **MaxGeometryTotalOutputComponents** = ((int)0x8DE1), **MaxGeometryTotalOutputComponentsArb** = ((int)0x8DE1), **MaxGeometryTotalOutputComponentsExt** = ((int)0x8DE1),
MaxVertexBindableUniformsExt = ((int)0x8DE2), **MaxFragmentBindableUniformsExt** = ((int)0x8DE3), **MaxGeometryBindableUniformsExt** = ((int)0x8DE4), **MaxBindableUniformSizeExt** = ((int)0x8DED),
UniformBufferExt = ((int)0x8DEE), **UniformBufferBindingExt** = ((int)0x8DEF), **RenderbufferColorSamplesNv** = ((int)0x8E10), **MaxMultisampleCoverageModesNv** = ((int)0x8E11),
MultisampleCoverageModesNv = ((int)0x8E12), **QueryWait** = ((int)0x8E13), **QueryWaitNv** = ((int)0x8E13), **QueryNoWait** = ((int)0x8E14),
QueryNoWaitNv = ((int)0x8E14), **QueryByRegionWait** = ((int)0x8E15), **QueryByRegionWaitNv** = ((int)0x8E15), **QueryByRegionNoWait** = ((int)0x8E16),
QueryByRegionNoWaitNv = ((int)0x8E16), **TransformFeedbackNv** = ((int)0x8E22), **TransformFeedbackBufferPausedNv** = ((int)0x8E23), **TransformFeedbackBufferActiveNv** = ((int)0x8E24),
TransformFeedbackBindingNv = ((int)0x8E25), **FrameNv** = ((int)0x8E26), **FieldsNv** = ((int)0x8E27), **CurrentTimeNv** = ((int)0x8E28),
NumFillStreamsNv = ((int)0x8E29), **PresentTimeNv** = ((int)0x8E2A), **PresentDurationNv** = ((int)0x8E2B), **ProgramMatrixExt** = ((int)0x8E2D),
TransposeProgramMatrixExt = ((int)0x8E2E), **ProgramMatrixStackDepthExt** = ((int)0x8E2F), **TextureSwizzleRExt** = ((int)0x8E42), **TextureSwizzleGExt** = ((int)0x8E43),
TextureSwizzleBExt = ((int)0x8E44), **TextureSwizzleAExt** = ((int)0x8E45), **TextureSwizzleRgbaExt** = ((int)0x8E46), **QuadsFollowProvokingVertexConvention** = ((int)0x8E4C),
QuadsFollowProvokingVertexConventionExt = ((int)0x8E4C), **FirstVertexConvention** = ((int)0x8E4D), **FirstVertexConventionExt** = ((int)0x8E4D), **LastVertexConvention** = ((int)0x8E4E),
LastVertexConventionExt = ((int)0x8E4E), **ProvokingVertex** = ((int)0x8E4F), **ProvokingVertexExt** = ((int)0x8E4F), **SamplePosition** = ((int)0x8E50),
SamplePositionNv = ((int)0x8E50), **SampleMask** = ((int)0x8E51), **SampleMaskNv** = ((int)0x8E51), **SampleMaskValue** = ((int)0x8E52),

```
SampleMaskValueNv = ((int)0x8E52), TextureBindingRenderbufferNv =
((int)0x8E53), TextureRenderbufferDataStoreBindingNv = ((int)0x8E54),
TextureRenderbufferNv = ((int)0x8E55),
SamplerRenderbufferNv = ((int)0x8E56), IntSamplerRenderbufferNv
= ((int)0x8E57), UnsignedIntSamplerRenderbufferNv = ((int)0x8E58),
MaxSampleMaskWords = ((int)0x8E59),
MaxSampleMaskWordsNv = ((int)0x8E59), MinProgramTextureGath-
erOffset = ((int)0x8E5E), MaxProgramTextureGatherOffset = ((int)0x8E5F),
CopyReadBuffer = ((int)0x8F36),
CopyWriteBuffer = ((int)0x8F37), RedSnorm = ((int)0x8F90), RgSnorm =
((int)0x8F91), RgbSnorm = ((int)0x8F92),
RgbaSnorm = ((int)0x8F93), R8Snorm = ((int)0x8F94), Rg8Snorm =
((int)0x8F95), Rgb8Snorm = ((int)0x8F96),
Rgba8Snorm = ((int)0x8F97), R16Snorm = ((int)0x8F98), Rg16Snorm =
((int)0x8F99), Rgb16Snorm = ((int)0x8F9A),
Rgba16Snorm = ((int)0x8F9B), SignedNormalized = ((int)0x8F9C), Primi-
tiveRestart = ((int)0x8F9D), PrimitiveRestartIndex = ((int)0x8F9E),
MaxProgramTextureGatherComponents = ((int)0x8F9F), SamplerBuffer-
Amd = ((int)0x9001), IntSamplerBufferAmd = ((int)0x9002), Unsigned-
IntSamplerBufferAmd = ((int)0x9003),
TessellationModeAmd = ((int)0x9004), TessellationFactorAmd =
((int)0x9005), DiscreteAmd = ((int)0x9006), ContinuousAmd = ((int)0x9007),
TextureCubeMapArray = ((int)0x9009), TextureBindingCubeMapArray =
((int)0x900A), ProxyTextureCubeMapArray = ((int)0x900B), SamplerCube-
MapArray = ((int)0x900C),
SamplerCubeMapArrayShadow = ((int)0x900D), IntSamplerCube-
MapArray = ((int)0x900E), UnsignedIntSamplerCubeMapArray =
((int)0x900F), AlphaSnorm = ((int)0x9010),
LuminanceSnorm = ((int)0x9011), LuminanceAlphaSnorm = ((int)0x9012),
IntensitySnorm = ((int)0x9013), Alpha8Snorm = ((int)0x9014),
Luminance8Snorm = ((int)0x9015), Luminance8Alpha8Snorm =
((int)0x9016), Intensity8Snorm = ((int)0x9017), Alpha16Snorm =
((int)0x9018),
Luminance16Snorm = ((int)0x9019), Luminance16Alpha16Snorm =
((int)0x901A), Intensity16Snorm = ((int)0x901B), Texture2DMultisample =
((int)0x9100),
ProxyTexture2DMultisample = ((int)0x9101), Texture2DMultisampleArray
= ((int)0x9102), ProxyTexture2DMultisampleArray = ((int)0x9103), Tex-
tureBinding2DMultisample = ((int)0x9104),
TextureBinding2DMultisampleArray = ((int)0x9105), TextureSamples
= ((int)0x9106), TextureFixedSampleLocations = ((int)0x9107), Sam-
pler2DMultisample = ((int)0x9108),
```

IntSampler2DMultisample = ((int)0x9109), **UnsignedIntSampler2DMultisample** = ((int)0x910A), **Sampler2DMultisampleArray** = ((int)0x910B), **IntSampler2DMultisampleArray** = ((int)0x910C),

UnsignedIntSampler2DMultisampleArray = ((int)0x910D), **MaxColorTextureSamples** = ((int)0x910E), **MaxDepthTextureSamples** = ((int)0x910F), **MaxIntegerSamples** = ((int)0x9110),

MaxServerWaitTimeout = ((int)0x9111), **ObjectType** = ((int)0x9112), **SyncCondition** = ((int)0x9113), **SyncStatus** = ((int)0x9114),

SyncFlags = ((int)0x9115), **SyncFence** = ((int)0x9116), **SyncGpuCommandsComplete** = ((int)0x9117), **Unsignaled** = ((int)0x9118),

Signaled = ((int)0x9119), **AlreadySignaled** = ((int)0x911A), **TimeoutExpired** = ((int)0x911B), **ConditionSatisfied** = ((int)0x911C),

WaitFailed = ((int)0x911D), **BufferAccessFlags** = ((int)0x911F), **BufferMapLength** = ((int)0x9120), **BufferMapOffset** = ((int)0x9121),

MaxVertexOutputComponents = ((int)0x9122), **MaxGeometryInputComponents** = ((int)0x9123), **MaxGeometryOutputComponents** = ((int)0x9124), **MaxFragmentInputComponents** = ((int)0x9125),

ContextProfileMask = ((int)0x9126), **AllAttribBits** = unchecked((int)0xFFFFFFFF), **ClientAllAttribBits** = unchecked((int)0xFFFFFFFF), **InvalidIndex** = unchecked((int)0xFFFFFFFF),

TimeoutIgnored = unchecked((int)0xFFFFFFFFFFFFFFFF), **One** = ((int)1), **True** = ((int)1), **CullVertexIbm** = ((int)103050),

VertexArrayListIbm = ((int)103070), **NormalArrayIbm** = ((int)103071), **ColorArrayIbm** = ((int)103072), **IndexArrayIbm** = ((int)103073),

TextureCoordArrayIbm = ((int)103074), **EdgeFlagArrayIbm** = ((int)103075), **FogCoordinateArrayIbm** = ((int)103076), **SecondaryColorArrayIbm** = ((int)103077),

VertexArrayListStrideIbm = ((int)103080), **NormalArrayListStrideIbm** = ((int)103081), **ColorArrayListStrideIbm** = ((int)103082), **IndexArrayListStrideIbm** = ((int)103083),

TextureCoordArrayListStrideIbm = ((int)103084), **EdgeFlagArrayListStrideIbm** = ((int)103085), **FogCoordinateArrayListStrideIbm** = ((int)103086), **SecondaryColorArrayListStrideIbm** = ((int)103087),

Two = ((int)2), **Three** = ((int)3), **Four** = ((int)4) }

- enum **AlphaFunction** {

Never = ((int)0x0200), **Less** = ((int)0x0201), **Equal** = ((int)0x0202), **Lequal** = ((int)0x0203),

Greater = ((int)0x0204), **Notequal** = ((int)0x0205), **Gequal** = ((int)0x0206), **Always** = ((int)0x0207) }

- enum **AmdDrawBuffersBlend**

- enum **AmdPerformanceMonitor** {
CounterTypeAmd = ((int)0x8BC0), **CounterRangeAmd** = ((int)0x8BC1), **UnsignedInt64Amd** = ((int)0x8BC2), **PercentageAmd** = ((int)0x8BC3),
PerfmonResultAvailableAmd = ((int)0x8BC4), **PerfmonResultSizeAmd** = ((int)0x8BC5), **PerfmonResultAmd** = ((int)0x8BC6) }
- enum **AmdTextureTexture4**
- enum **AmdVertexShaderTessellator** {
SamplerBufferAmd = ((int)0x9001), **IntSamplerBufferAmd** = ((int)0x9002), **UnsignedIntSamplerBufferAmd** = ((int)0x9003), **TessellationModeAmd** = ((int)0x9004),
TessellationFactorAmd = ((int)0x9005), **DiscreteAmd** = ((int)0x9006), **ContinuousAmd** = ((int)0x9007) }
- enum **AppleAuxDepthStencil** { **AuxDepthStencilApple** = ((int)0x8A14) }
- enum **AppleClientStorage** { **UnpackClientStorageApple** = ((int)0x85B2) }
- enum **AppleElementArray** { **ElementArrayApple** = ((int)0x8768), **ElementArrayTypeApple** = ((int)0x8769), **ElementArrayPointerApple** = ((int)0x876A) }
- enum **AppleFence** { **DrawPixelsApple** = ((int)0x8A0A), **FenceApple** = ((int)0x8A0B) }
- enum **AppleFloatPixels** {
HalfApple = ((int)0x140B), **RgbaFloat32Apple** = ((int)0x8814), **RgbFloat32Apple** = ((int)0x8815), **AlphaFloat32Apple** = ((int)0x8816),
IntensityFloat32Apple = ((int)0x8817), **LuminanceFloat32Apple** = ((int)0x8818), **LuminanceAlphaFloat32Apple** = ((int)0x8819), **RgbaFloat16Apple** = ((int)0x881A),
RgbFloat16Apple = ((int)0x881B), **AlphaFloat16Apple** = ((int)0x881C), **IntensityFloat16Apple** = ((int)0x881D), **LuminanceFloat16Apple** = ((int)0x881E),
LuminanceAlphaFloat16Apple = ((int)0x881F), **ColorFloatApple** = ((int)0x8A0F) }
- enum **AppleFlushBufferRange** { **BufferSerializedModifyApple** = ((int)0x8A12), **BufferFlushingUnmapApple** = ((int)0x8A13) }
- enum **AppleObjectPurgeable** {
BufferObjectApple = ((int)0x85B3), **ReleasedApple** = ((int)0x8A19), **VolatileApple** = ((int)0x8A1A), **RetainedApple** = ((int)0x8A1B),
UndefinedApple = ((int)0x8A1C), **PurgeableApple** = ((int)0x8A1D) }
- enum **AppleRowBytes** { **PackRowBytesApple** = ((int)0x8A15), **UnpackRowBytesApple** = ((int)0x8A16) }
- enum **AppleSpecularVector** { **LightModelSpecularVectorApple** = ((int)0x85B0) }

- enum **AppleTextureRange** {
TextureRangeLengthApple = ((int)0x85B7), **TextureRangePointerApple** = ((int)0x85B8), **TextureStorageHintApple** = ((int)0x85BC), **StoragePrivateApple** = ((int)0x85BD),
StorageCachedApple = ((int)0x85BE), **StorageSharedApple** = ((int)0x85BF)
}
- enum **AppleTransformHint** { **TransformHintApple** = ((int)0x85B1) }
- enum **AppleVertexArrayObject** { **VertexArrayBindingApple** = ((int)0x85B5) }
- enum **AppleVertexArrayRange** {
VertexArrayRangeApple = ((int)0x851D), **VertexArrayRangeLengthApple** = ((int)0x851E), **VertexArrayStorageHintApple** = ((int)0x851F), **VertexArrayRangePointerApple** = ((int)0x8521),
StorageCachedApple = ((int)0x85BE), **StorageSharedApple** = ((int)0x85BF)
}
- enum **AppleVertexProgramEvaluators** {
VertexAttribMap1Apple = ((int)0x8A00), **VertexAttribMap2Apple** = ((int)0x8A01), **VertexAttribMap1SizeApple** = ((int)0x8A02), **VertexAttribMap1CoeffApple** = ((int)0x8A03),
VertexAttribMap1OrderApple = ((int)0x8A04), **VertexAttribMap1DomainApple** = ((int)0x8A05), **VertexAttribMap2SizeApple** = ((int)0x8A06), **VertexAttribMap2CoeffApple** = ((int)0x8A07),
VertexAttribMap2OrderApple = ((int)0x8A08), **VertexAttribMap2DomainApple** = ((int)0x8A09) }
- enum **AppleYcbcr422** { **Ycbcr422Apple** = ((int)0x85B9), **UnsignedShort88Apple** = ((int)0x85BA), **UnsignedShort88RevApple** = ((int)0x85BB) }
- enum **ArbColorBufferFloat** {
RgbaFloatModeArb = ((int)0x8820), **ClampVertexColorArb** = ((int)0x891A), **ClampFragmentColorArb** = ((int)0x891B), **ClampReadColorArb** = ((int)0x891C),
FixedOnlyArb = ((int)0x891D) }
- enum **ArbCompatibility**
- enum **ArbCopyBuffer** { **CopyReadBuffer** = ((int)0x8F36), **CopyWriteBuffer** = ((int)0x8F37) }
- enum **ArbDepthBufferFloat** { **DepthComponent32f** = ((int)0x8CAC), **Depth32fStencil8** = ((int)0x8CAD), **Float32UnsignedInt248Rev** = ((int)0x8DAD) }
- enum **ArbDepthClamp** { **DepthClamp** = ((int)0x864F) }
- enum **ArbDepthTexture** {


```

DepthComponent16Arb = ((int)0x81A5), DepthComponent24Arb =
((int)0x81A6), DepthComponent32Arb = ((int)0x81A7), TextureDepth-
SizeArb = ((int)0x884A),
DepthTextureModeArb = ((int)0x884B) }
• enum ArbDrawBuffers {
MaxDrawBuffersArb = ((int)0x8824), DrawBuffer0Arb = ((int)0x8825),
DrawBuffer1Arb = ((int)0x8826), DrawBuffer2Arb = ((int)0x8827),
DrawBuffer3Arb = ((int)0x8828), DrawBuffer4Arb = ((int)0x8829), Draw-
Buffer5Arb = ((int)0x882A), DrawBuffer6Arb = ((int)0x882B),
DrawBuffer7Arb = ((int)0x882C), DrawBuffer8Arb = ((int)0x882D), Draw-
Buffer9Arb = ((int)0x882E), DrawBuffer10Arb = ((int)0x882F),
DrawBuffer11Arb = ((int)0x8830), DrawBuffer12Arb = ((int)0x8831),
DrawBuffer13Arb = ((int)0x8832), DrawBuffer14Arb = ((int)0x8833),
DrawBuffer15Arb = ((int)0x8834) }
• enum ArbDrawBuffersBlend
• enum ArbDrawElementsBaseVertex
• enum ArbDrawInstanced
• enum ArbFragmentCoordConventions
• enum ArbFragmentProgram {
FragmentProgramArb = ((int)0x8804), ProgramAluInstructionsArb =
((int)0x8805), ProgramTexInstructionsArb = ((int)0x8806), Program-
TexIndirectionsArb = ((int)0x8807),
ProgramNativeAluInstructionsArb = ((int)0x8808), ProgramNativeTex-
InstructionsArb = ((int)0x8809), ProgramNativeTexIndirectionsArb =
((int)0x880A), MaxProgramAluInstructionsArb = ((int)0x880B),
MaxProgramTexInstructionsArb = ((int)0x880C), MaxProgramTexIndi-
rectionsArb = ((int)0x880D), MaxProgramNativeAluInstructionsArb =
((int)0x880E), MaxProgramNativeTexInstructionsArb = ((int)0x880F),
MaxProgramNativeTexIndirectionsArb = ((int)0x8810), MaxTextureCoord-
sArb = ((int)0x8871), MaxTextureImageUnitsArb = ((int)0x8872) }
• enum ArbFragmentProgramShadow
• enum ArbFragmentShader { FragmentShaderArb = ((int)0x8B30),
MaxFragmentUniformComponentsArb = ((int)0x8B49), Frag-
mentShaderDerivativeHintArb = ((int)0x8B8B) }
• enum ArbFramebufferObject {
InvalidFramebufferOperation = ((int)0x0506), FramebufferAttachment-
ColorEncoding = ((int)0x8210), FramebufferAttachmentComponentType =
((int)0x8211), FramebufferAttachmentRedSize = ((int)0x8212),
FramebufferAttachmentGreenSize = ((int)0x8213), FramebufferAt-
achmentBlueSize = ((int)0x8214), FramebufferAttachmentAlphaSize =
((int)0x8215), FramebufferAttachmentDepthSize = ((int)0x8216),

```

FramebufferAttachmentStencilSize = ((int)0x8217), **FramebufferDefault** = ((int)0x8218), **FramebufferUndefined** = ((int)0x8219), **DepthStencilAttachment** = ((int)0x821A),

MaxRenderbufferSize = ((int)0x84E8), **DepthStencil** = ((int)0x84F9), **UnsignedInt248** = ((int)0x84FA), **Depth24Stencil8** = ((int)0x88F0),

TextureStencilSize = ((int)0x88F1), **TextureRedType** = ((int)0x8C10), **TextureGreenType** = ((int)0x8C11), **TextureBlueType** = ((int)0x8C12),

TextureAlphaType = ((int)0x8C13), **TextureDepthType** = ((int)0x8C16), **UnsignedNormalized** = ((int)0x8C17), **DrawFramebufferBinding** = ((int)0x8CA6),

FramebufferBinding = ((int)0x8CA6), **RenderbufferBinding** = ((int)0x8CA7), **ReadFramebuffer** = ((int)0x8CA8), **DrawFramebuffer** = ((int)0x8CA9),

ReadFramebufferBinding = ((int)0x8CAA), **RenderbufferSamples** = ((int)0x8CAB), **FramebufferAttachmentObjectType** = ((int)0x8CD0), **FramebufferAttachmentObjectName** = ((int)0x8CD1),

FramebufferAttachmentTextureLevel = ((int)0x8CD2), **FramebufferAttachmentTextureCubeMapFace** = ((int)0x8CD3), **FramebufferAttachmentTextureLayer** = ((int)0x8CD4), **FramebufferComplete** = ((int)0x8CD5),

FramebufferIncompleteAttachment = ((int)0x8CD6), **FramebufferIncompleteMissingAttachment** = ((int)0x8CD7), **FramebufferIncompleteDrawBuffer** = ((int)0x8CDB), **FramebufferIncompleteReadBuffer** = ((int)0x8CDC),

FramebufferUnsupported = ((int)0x8CDD), **MaxColorAttachments** = ((int)0x8CDF), **ColorAttachment0** = ((int)0x8CE0), **ColorAttachment1** = ((int)0x8CE1),

ColorAttachment2 = ((int)0x8CE2), **ColorAttachment3** = ((int)0x8CE3), **ColorAttachment4** = ((int)0x8CE4), **ColorAttachment5** = ((int)0x8CE5),

ColorAttachment6 = ((int)0x8CE6), **ColorAttachment7** = ((int)0x8CE7), **ColorAttachment8** = ((int)0x8CE8), **ColorAttachment9** = ((int)0x8CE9),

ColorAttachment10 = ((int)0x8CEA), **ColorAttachment11** = ((int)0x8CEB), **ColorAttachment12** = ((int)0x8CEC), **ColorAttachment13** = ((int)0x8CED),

ColorAttachment14 = ((int)0x8CEE), **ColorAttachment15** = ((int)0x8CEF), **DepthAttachment** = ((int)0x8D00), **StencilAttachment** = ((int)0x8D20),

Framebuffer = ((int)0x8D40), **Renderbuffer** = ((int)0x8D41), **RenderbufferWidth** = ((int)0x8D42), **RenderbufferHeight** = ((int)0x8D43),

RenderbufferInternalFormat = ((int)0x8D44), **StencilIndex1** = ((int)0x8D46), **StencilIndex4** = ((int)0x8D47), **StencilIndex8** = ((int)0x8D48),

StencilIndex16 = ((int)0x8D49), **RenderbufferRedSize** = ((int)0x8D50), **RenderbufferGreenSize** = ((int)0x8D51), **RenderbufferBlueSize** = ((int)0x8D52),

```

RenderbufferAlphaSize = ((int)0x8D53), RenderbufferDepthSize =
((int)0x8D54), RenderbufferStencilSize = ((int)0x8D55), Framebuffer-
IncompleteMultisample = ((int)0x8D56),
MaxSamples = ((int)0x8D57) }
• enum ArbFramebufferObjectDeprecated { Index = ((int)0x8222), Texture-
LuminanceType = ((int)0x8C14), TextureIntensityType = ((int)0x8C15) }
• enum ArbFramebufferSrgb { FramebufferSrgb = ((int)0x8DB9) }
• enum ArbGeometryShader4 {
LinesAdjacencyArb = ((int)0x000A), LineStripAdjacencyArb =
((int)0x000B), TrianglesAdjacencyArb = ((int)0x000C), TriangleStri-
pAdjacencyArb = ((int)0x000D),
ProgramPointSizeArb = ((int)0x8642), MaxVaryingComponents =
((int)0x8B4B), MaxGeometryTextureImageUnitsArb = ((int)0x8C29),
FramebufferAttachmentTextureLayer = ((int)0x8CD4),
FramebufferAttachmentLayeredArb = ((int)0x8DA7), FramebufferIncom-
pleteLayerTargetsArb = ((int)0x8DA8), FramebufferIncompleteLayer-
CountArb = ((int)0x8DA9), GeometryShaderArb = ((int)0x8DD9),
GeometryVerticesOutArb = ((int)0x8DDA), GeometryInputTypeArb =
((int)0x8DDB), GeometryOutputTypeArb = ((int)0x8DDC), MaxGeometry-
VaryingComponentsArb = ((int)0x8DDD),
MaxVertexVaryingComponentsArb = ((int)0x8DDE), MaxGeometryUni-
formComponentsArb = ((int)0x8DDF), MaxGeometryOutputVerticesArb =
((int)0x8DE0), MaxGeometryTotalOutputComponentsArb = ((int)0x8DE1)
}
• enum ArbHalfFloatPixel { HalfFloatArb = ((int)0x140B) }
• enum ArbHalfFloatVertex { HalfFloat = ((int)0x140B) }
• enum ArbImaging {
ConstantColor = ((int)0x8001), OneMinusConstantColor = ((int)0x8002),
ConstantAlpha = ((int)0x8003), OneMinusConstantAlpha = ((int)0x8004),
BlendColor = ((int)0x8005), FuncAdd = ((int)0x8006), Min = ((int)0x8007),
Max = ((int)0x8008),
BlendEquation = ((int)0x8009), FuncSubtract = ((int)0x800A), FuncRevers-
eSubtract = ((int)0x800B) }
• enum ArbImagingDeprecated {
Convolution1D = ((int)0x8010), Convolution2D = ((int)0x8011), Separable2D
= ((int)0x8012), ConvolutionBorderMode = ((int)0x8013),
ConvolutionFilterScale = ((int)0x8014), ConvolutionFilterBias =
((int)0x8015), Reduce = ((int)0x8016), ConvolutionFormat = ((int)0x8017),
ConvolutionWidth = ((int)0x8018), ConvolutionHeight = ((int)0x8019), Max-
ConvolutionWidth = ((int)0x801A), MaxConvolutionHeight = ((int)0x801B),

```

PostConvolutionRedScale = ((int)0x801C), **PostConvolutionGreenScale** = ((int)0x801D), **PostConvolutionBlueScale** = ((int)0x801E), **PostConvolutionAlphaScale** = ((int)0x801F),

PostConvolutionRedBias = ((int)0x8020), **PostConvolutionGreenBias** = ((int)0x8021), **PostConvolutionBlueBias** = ((int)0x8022), **PostConvolutionAlphaBias** = ((int)0x8023),

Histogram = ((int)0x8024), **ProxyHistogram** = ((int)0x8025), **HistogramWidth** = ((int)0x8026), **HistogramFormat** = ((int)0x8027),

HistogramRedSize = ((int)0x8028), **HistogramGreenSize** = ((int)0x8029), **HistogramBlueSize** = ((int)0x802A), **HistogramAlphaSize** = ((int)0x802B),

HistogramLuminanceSize = ((int)0x802C), **HistogramSink** = ((int)0x802D), **Minmax** = ((int)0x802E), **MinmaxFormat** = ((int)0x802F),

MinmaxSink = ((int)0x8030), **TableTooLarge** = ((int)0x8031), **ColorMatrix** = ((int)0x80B1), **ColorMatrixStackDepth** = ((int)0x80B2),

MaxColorMatrixStackDepth = ((int)0x80B3), **PostColorMatrixRedScale** = ((int)0x80B4), **PostColorMatrixGreenScale** = ((int)0x80B5), **PostColorMatrixBlueScale** = ((int)0x80B6),

PostColorMatrixAlphaScale = ((int)0x80B7), **PostColorMatrixRedBias** = ((int)0x80B8), **PostColorMatrixGreenBias** = ((int)0x80B9), **PostColorMatrixBlueBias** = ((int)0x80BA),

PostColorMatrixAlphaBias = ((int)0x80BB), **ColorTable** = ((int)0x80D0), **PostConvolutionColorTable** = ((int)0x80D1), **PostColorMatrixColorTable** = ((int)0x80D2),

ProxyColorTable = ((int)0x80D3), **ProxyPostConvolutionColorTable** = ((int)0x80D4), **ProxyPostColorMatrixColorTable** = ((int)0x80D5), **ColorTableScale** = ((int)0x80D6),

ColorTableBias = ((int)0x80D7), **ColorTableFormat** = ((int)0x80D8), **ColorTableWidth** = ((int)0x80D9), **ColorTableRedSize** = ((int)0x80DA),

ColorTableGreenSize = ((int)0x80DB), **ColorTableBlueSize** = ((int)0x80DC), **ColorTableAlphaSize** = ((int)0x80DD), **ColorTableLuminanceSize** = ((int)0x80DE),

ColorTableIntensitySize = ((int)0x80DF), **ConstantBorder** = ((int)0x8151), **ReplicateBorder** = ((int)0x8153), **ConvolutionBorderColor** = ((int)0x8154) }

- enum **ArbInstancedArrays** { **VertexAttribArrayDivisorArb** = ((int)0x88FE) }

- enum **ArbMapBufferRange** {

MapReadBit = ((int)0x0001), **MapWriteBit** = ((int)0x0002), **MapInvalidat-eRangeBit** = ((int)0x0004), **MapInvalidateBufferBit** = ((int)0x0008),

MapFlushExplicitBit = ((int)0x0010), **MapUnsynchronizedBit** = ((int)0x0020) }

- enum **ArbMatrixPalette** {
 - MatrixPaletteArb** = ((int)0x8840), **MaxMatrixPaletteStackDepthArb** = ((int)0x8841), **MaxPaletteMatricesArb** = ((int)0x8842), **CurrentPaletteMatrixArb** = ((int)0x8843),
 - MatrixIndexArrayArb** = ((int)0x8844), **CurrentMatrixIndexArb** = ((int)0x8845), **MatrixIndexArraySizeArb** = ((int)0x8846), **MatrixIndexArrayTypeArb** = ((int)0x8847),
 - MatrixIndexArrayStrideArb** = ((int)0x8848), **MatrixIndexArrayPointerArb** = ((int)0x8849) }
- enum **ArbMultisample** {
 - MultisampleBitArb** = ((int)0x20000000), **MultisampleArb** = ((int)0x809D), **SampleAlphaToCoverageArb** = ((int)0x809E), **SampleAlphaToOneArb** = ((int)0x809F),
 - SampleCoverageArb** = ((int)0x80A0), **SampleBuffersArb** = ((int)0x80A8), **SamplesArb** = ((int)0x80A9), **SampleCoverageValueArb** = ((int)0x80AA),
 - SampleCoverageInvertArb** = ((int)0x80AB) }
- enum **ArbMultitexture** {
 - Texture0Arb** = ((int)0x84C0), **Texture1Arb** = ((int)0x84C1), **Texture2Arb** = ((int)0x84C2), **Texture3Arb** = ((int)0x84C3),
 - Texture4Arb** = ((int)0x84C4), **Texture5Arb** = ((int)0x84C5), **Texture6Arb** = ((int)0x84C6), **Texture7Arb** = ((int)0x84C7),
 - Texture8Arb** = ((int)0x84C8), **Texture9Arb** = ((int)0x84C9), **Texture10Arb** = ((int)0x84CA), **Texture11Arb** = ((int)0x84CB),
 - Texture12Arb** = ((int)0x84CC), **Texture13Arb** = ((int)0x84CD), **Texture14Arb** = ((int)0x84CE), **Texture15Arb** = ((int)0x84CF),
 - Texture16Arb** = ((int)0x84D0), **Texture17Arb** = ((int)0x84D1), **Texture18Arb** = ((int)0x84D2), **Texture19Arb** = ((int)0x84D3),
 - Texture20Arb** = ((int)0x84D4), **Texture21Arb** = ((int)0x84D5), **Texture22Arb** = ((int)0x84D6), **Texture23Arb** = ((int)0x84D7),
 - Texture24Arb** = ((int)0x84D8), **Texture25Arb** = ((int)0x84D9), **Texture26Arb** = ((int)0x84DA), **Texture27Arb** = ((int)0x84DB),
 - Texture28Arb** = ((int)0x84DC), **Texture29Arb** = ((int)0x84DD), **Texture30Arb** = ((int)0x84DE), **Texture31Arb** = ((int)0x84DF),
 - ActiveTextureArb** = ((int)0x84E0), **ClientActiveTextureArb** = ((int)0x84E1), **MaxTextureUnitsArb** = ((int)0x84E2) }
- enum **ArbOcclusionQuery** {
 - QueryCounterBitsArb** = ((int)0x8864), **CurrentQueryArb** = ((int)0x8865), **QueryResultArb** = ((int)0x8866), **QueryResultAvailableArb** = ((int)0x8867),
 - SamplesPassedArb** = ((int)0x8914) }

- enum **ArbPixelBufferObject** { **PixelPackBufferArb** = ((int)0x88EB), **PixelUnpackBufferArb** = ((int)0x88EC), **PixelPackBufferBindingArb** = ((int)0x88ED), **PixelUnpackBufferBindingArb** = ((int)0x88EF) }
- enum **ArbPointParameters** { **PointSizeMinArb** = ((int)0x8126), **PointSizeMaxArb** = ((int)0x8127), **PointFadeThresholdSizeArb** = ((int)0x8128), **PointDistanceAttenuationArb** = ((int)0x8129) }
- enum **ArbPointSprite** { **PointSpriteArb** = ((int)0x8861), **CoordReplaceArb** = ((int)0x8862) }
- enum **ArbProvokingVertex** { **QuadsFollowProvokingVertexConvention** = ((int)0x8E4C), **FirstVertexConvention** = ((int)0x8E4D), **LastVertexConvention** = ((int)0x8E4E), **ProvokingVertex** = ((int)0x8E4F) }
- enum **ArbSampleShading** { **SampleShading** = ((int)0x8C36), **MinSampleShadingValue** = ((int)0x8C37) }
- enum **ArbSeamlessCubeMap** { **TextureCubeMapSeamless** = ((int)0x884F) }
- enum **ArbShaderObjects** {
ProgramObjectArb = ((int)0x8B40), **ShaderObjectArb** = ((int)0x8B48), **ObjectTypeArb** = ((int)0x8B4E), **ObjectSubtypeArb** = ((int)0x8B4F),
FloatVec2Arb = ((int)0x8B50), **FloatVec3Arb** = ((int)0x8B51), **FloatVec4Arb** = ((int)0x8B52), **IntVec2Arb** = ((int)0x8B53),
IntVec3Arb = ((int)0x8B54), **IntVec4Arb** = ((int)0x8B55), **BoolArb** = ((int)0x8B56), **BoolVec2Arb** = ((int)0x8B57),
BoolVec3Arb = ((int)0x8B58), **BoolVec4Arb** = ((int)0x8B59), **FloatMat2Arb** = ((int)0x8B5A), **FloatMat3Arb** = ((int)0x8B5B),
FloatMat4Arb = ((int)0x8B5C), **Sampler1DArb** = ((int)0x8B5D), **Sampler2DArb** = ((int)0x8B5E), **Sampler3DArb** = ((int)0x8B5F),
SamplerCubeArb = ((int)0x8B60), **Sampler1DShadowArb** = ((int)0x8B61), **Sampler2DShadowArb** = ((int)0x8B62), **Sampler2DRectArb** = ((int)0x8B63),
Sampler2DRectShadowArb = ((int)0x8B64), **ObjectDeleteStatusArb** = ((int)0x8B80), **ObjectCompileStatusArb** = ((int)0x8B81), **ObjectLinkStatusArb** = ((int)0x8B82),
ObjectValidateStatusArb = ((int)0x8B83), **ObjectInfoLogLengthArb** = ((int)0x8B84), **ObjectAttachedObjectsArb** = ((int)0x8B85), **ObjectActiveUniformsArb** = ((int)0x8B86),
ObjectActiveUniformMaxLengthArb = ((int)0x8B87), **ObjectShaderSourceLengthArb** = ((int)0x8B88) }
- enum **ArbShaderTextureLod**
- enum **ArbShadingLanguage100** { **ShadingLanguageVersionArb** = ((int)0x8B8C) }
- enum **ArbShadow** { **TextureCompareModeArb** = ((int)0x884C), **TextureCompareFuncArb** = ((int)0x884D), **CompareRToTextureArb** = ((int)0x884E) }

- enum **ArbShadowAmbient** { **TextureCompareFailValueArb** = ((int)0x80BF) }
- enum **ArbSync** {
 - SyncFlushCommandsBit** = ((int)0x00000001), **MaxServerWaitTimeout** = ((int)0x9111), **ObjectType** = ((int)0x9112), **SyncCondition** = ((int)0x9113),
 - SyncStatus** = ((int)0x9114), **SyncFlags** = ((int)0x9115), **SyncFence** = ((int)0x9116), **SyncGpuCommandsComplete** = ((int)0x9117),
 - Unsignaled** = ((int)0x9118), **Signaled** = ((int)0x9119), **AlreadySignaled** = ((int)0x911A), **TimeoutExpired** = ((int)0x911B),
 - ConditionSatisfied** = ((int)0x911C), **WaitFailed** = ((int)0x911D), **TimeoutIgnored** = unchecked((int)0xFFFFFFFFFFFFFFFF) }
- enum **ArbTextureBorderClamp** { **ClampToBorderArb** = ((int)0x812D) }
- enum **ArbTextureBufferObject** {
 - TextureBufferArb** = ((int)0x8C2A), **MaxTextureBufferSizeArb** = ((int)0x8C2B), **TextureBindingBufferArb** = ((int)0x8C2C), **TextureBufferDataStoreBindingArb** = ((int)0x8C2D),
 - TextureBufferFormatArb** = ((int)0x8C2E) }
- enum **ArbTextureCompression** {
 - CompressedAlphaArb** = ((int)0x84E9), **CompressedLuminanceArb** = ((int)0x84EA), **CompressedLuminanceAlphaArb** = ((int)0x84EB), **CompressedIntensityArb** = ((int)0x84EC),
 - CompressedRgbArb** = ((int)0x84ED), **CompressedRgbaArb** = ((int)0x84EE), **TextureCompressionHintArb** = ((int)0x84EF), **TextureCompressedImageSizeArb** = ((int)0x86A0),
 - TextureCompressedArb** = ((int)0x86A1), **NumCompressedTextureFormatsArb** = ((int)0x86A2), **CompressedTextureFormatsArb** = ((int)0x86A3) }
- enum **ArbTextureCompressionRgtc** { **CompressedRedRgtc1** = ((int)0x8DBB), **CompressedSignedRedRgtc1** = ((int)0x8DBC), **CompressedRgRgtc2** = ((int)0x8DBD), **CompressedSignedRgRgtc2** = ((int)0x8DBE) }
- enum **ArbTextureCubeMap** {
 - NormalMapArb** = ((int)0x8511), **ReflectionMapArb** = ((int)0x8512), **TextureCubeMapArb** = ((int)0x8513), **TextureBindingCubeMapArb** = ((int)0x8514),
 - TextureCubeMapPositiveXArb** = ((int)0x8515), **TextureCubeMapNegativeXArb** = ((int)0x8516), **TextureCubeMapPositiveYArb** = ((int)0x8517), **TextureCubeMapNegativeYArb** = ((int)0x8518),
 - TextureCubeMapPositiveZArb** = ((int)0x8519), **TextureCubeMapNegativeZArb** = ((int)0x851A), **ProxyTextureCubeMapArb** = ((int)0x851B), **MaxCubeMapTextureSizeArb** = ((int)0x851C) }

- enum **ArbTextureCubeMapArray** {
TextureCubeMapArray = ((int)0x9009), **TextureBindingCubeMapArray** = ((int)0x900A), **ProxyTextureCubeMapArray** = ((int)0x900B), **SamplerCubeMapArray** = ((int)0x900C),
SamplerCubeMapArrayShadow = ((int)0x900D), **IntSamplerCubeMapArray** = ((int)0x900E), **UnsignedIntSamplerCubeMapArray** = ((int)0x900F) }
- enum **ArbTextureEnvAdd**
- enum **ArbTextureEnvCombine** {
SubtractArb = ((int)0x84E7), **CombineArb** = ((int)0x8570), **CombineRgbaArb** = ((int)0x8571), **CombineAlphaArb** = ((int)0x8572),
RgbScaleArb = ((int)0x8573), **AddSignedArb** = ((int)0x8574), **InterpolateArb** = ((int)0x8575), **ConstantArb** = ((int)0x8576),
PrimaryColorArb = ((int)0x8577), **PreviousArb** = ((int)0x8578), **Source0RgbArb** = ((int)0x8580), **Source1RgbArb** = ((int)0x8581),
Source2RgbArb = ((int)0x8582), **Source0AlphaArb** = ((int)0x8588), **Source1AlphaArb** = ((int)0x8589), **Source2AlphaArb** = ((int)0x858A),
Operand0RgbArb = ((int)0x8590), **Operand1RgbArb** = ((int)0x8591), **Operand2RgbArb** = ((int)0x8592), **Operand0AlphaArb** = ((int)0x8598),
Operand1AlphaArb = ((int)0x8599), **Operand2AlphaArb** = ((int)0x859A) }
- enum **ArbTextureEnvCrossbar**
- enum **ArbTextureEnvDot3** { **Dot3RgbArb** = ((int)0x86AE), **Dot3RgbaArb** = ((int)0x86AF) }
- enum **ArbTextureFloat** {
Rgba32fArb = ((int)0x8814), **Rgb32fArb** = ((int)0x8815), **Alpha32fArb** = ((int)0x8816), **Intensity32fArb** = ((int)0x8817),
Luminance32fArb = ((int)0x8818), **LuminanceAlpha32fArb** = ((int)0x8819), **Rgba16fArb** = ((int)0x881A), **Rgb16fArb** = ((int)0x881B),
Alpha16fArb = ((int)0x881C), **Intensity16fArb** = ((int)0x881D), **Luminance16fArb** = ((int)0x881E), **LuminanceAlpha16fArb** = ((int)0x881F),
TextureRedTypeArb = ((int)0x8C10), **TextureGreenTypeArb** = ((int)0x8C11), **TextureBlueTypeArb** = ((int)0x8C12), **TextureAlphaTypeArb** = ((int)0x8C13),
TextureLuminanceTypeArb = ((int)0x8C14), **TextureIntensityTypeArb** = ((int)0x8C15), **TextureDepthTypeArb** = ((int)0x8C16), **UnsignedNormalizedArb** = ((int)0x8C17) }
- enum **ArbTextureGather** { **MinProgramTextureGatherOffset** = ((int)0x8E5E), **MaxProgramTextureGatherOffset** = ((int)0x8E5F), **MaxProgramTextureGatherComponents** = ((int)0x8F9F) }
- enum **ArbTextureMirroredRepeat** { **MirroredRepeatArb** = ((int)0x8370) }

- enum **ArbTextureMultisample** {
 - SamplePosition** = ((int)0x8E50), **SampleMask** = ((int)0x8E51), **SampleMaskValue** = ((int)0x8E52), **MaxSampleMaskWords** = ((int)0x8E59),
 - Texture2DMultisample** = ((int)0x9100), **ProxyTexture2DMultisample** = ((int)0x9101), **Texture2DMultisampleArray** = ((int)0x9102), **ProxyTexture2DMultisampleArray** = ((int)0x9103),
 - TextureBinding2DMultisample** = ((int)0x9104), **TextureBinding2DMultisampleArray** = ((int)0x9105), **TextureSamples** = ((int)0x9106), **TextureFixedSampleLocations** = ((int)0x9107),
 - Sampler2DMultisample** = ((int)0x9108), **IntSampler2DMultisample** = ((int)0x9109), **UnsignedIntSampler2DMultisample** = ((int)0x910A), **Sampler2DMultisampleArray** = ((int)0x910B),
 - IntSampler2DMultisampleArray** = ((int)0x910C), **UnsignedIntSampler2DMultisampleArray** = ((int)0x910D), **MaxColorTextureSamples** = ((int)0x910E), **MaxDepthTextureSamples** = ((int)0x910F),
 - MaxIntegerSamples** = ((int)0x9110) }
- enum **ArbTextureNonPowerOfTwo**
- enum **ArbTextureQueryLod**
- enum **ArbTextureRectangle** { **TextureRectangleArb** = ((int)0x84F5), **TextureBindingRectangleArb** = ((int)0x84F6), **ProxyTextureRectangleArb** = ((int)0x84F7), **MaxRectangleTextureSizeArb** = ((int)0x84F8) }
- enum **ArbTextureRg** {
 - Rg** = ((int)0x8227), **RgInteger** = ((int)0x8228), **R8** = ((int)0x8229), **R16** = ((int)0x822A),
 - Rg8** = ((int)0x822B), **Rg16** = ((int)0x822C), **R16f** = ((int)0x822D), **R32f** = ((int)0x822E),
 - Rg16f** = ((int)0x822F), **Rg32f** = ((int)0x8230), **R8i** = ((int)0x8231), **R8ui** = ((int)0x8232),
 - R16i** = ((int)0x8233), **R16ui** = ((int)0x8234), **R32i** = ((int)0x8235), **R32ui** = ((int)0x8236),
 - Rg8i** = ((int)0x8237), **Rg8ui** = ((int)0x8238), **Rg16i** = ((int)0x8239), **Rg16ui** = ((int)0x823A),
 - Rg32i** = ((int)0x823B), **Rg32ui** = ((int)0x823C) }
- enum **ArbTransposeMatrix** { **TransposeModelviewMatrixArb** = ((int)0x84E3), **TransposeProjectionMatrixArb** = ((int)0x84E4), **TransposeTextureMatrixArb** = ((int)0x84E5), **TransposeColorMatrixArb** = ((int)0x84E6) }
- enum **ArbUniformBufferObject** {
 - UniformBuffer** = ((int)0x8A11), **UniformBufferBinding** = ((int)0x8A28), **UniformBufferStart** = ((int)0x8A29), **UniformBufferSize** = ((int)0x8A2A),

MaxVertexUniformBlocks = ((int)0x8A2B), **MaxGeometryUniformBlocks** = ((int)0x8A2C), **MaxFragmentUniformBlocks** = ((int)0x8A2D), **MaxCombinedUniformBlocks** = ((int)0x8A2E),

MaxUniformBufferBindings = ((int)0x8A2F), **MaxUniformBlockSize** = ((int)0x8A30), **MaxCombinedVertexUniformComponents** = ((int)0x8A31), **MaxCombinedGeometryUniformComponents** = ((int)0x8A32),

MaxCombinedFragmentUniformComponents = ((int)0x8A33), **UniformBufferOffsetAlignment** = ((int)0x8A34), **ActiveUniformBlockMaxNameLength** = ((int)0x8A35), **ActiveUniformBlocks** = ((int)0x8A36),

UniformType = ((int)0x8A37), **UniformSize** = ((int)0x8A38), **UniformNameLength** = ((int)0x8A39), **UniformBlockIndex** = ((int)0x8A3A),

UniformOffset = ((int)0x8A3B), **UniformArrayStride** = ((int)0x8A3C), **UniformMatrixStride** = ((int)0x8A3D), **UniformIsRowMajor** = ((int)0x8A3E),

UniformBlockBinding = ((int)0x8A3F), **UniformBlockDataSize** = ((int)0x8A40), **UniformBlockNameLength** = ((int)0x8A41), **UniformBlockActiveUniforms** = ((int)0x8A42),

UniformBlockActiveUniformIndices = ((int)0x8A43), **UniformBlockReferencedByVertexShader** = ((int)0x8A44), **UniformBlockReferencedByGeometryShader** = ((int)0x8A45), **UniformBlockReferencedByFragmentShader** = ((int)0x8A46),

InvalidIndex = unchecked((int)0xFFFFFFFF) }

- enum **ArbVertexArrayBgra** { **Bgra** = ((int)0x80E1) }
- enum **ArbVertexArrayObject** { **VertexArrayBinding** = ((int)0x85B5) }
- enum **ArbVertexBlend** {

Modelview0Arb = ((int)0x1700), **Modelview1Arb** = ((int)0x850A), **MaxVertexUnitsArb** = ((int)0x86A4), **ActiveVertexUnitsArb** = ((int)0x86A5),

WeightSumUnityArb = ((int)0x86A6), **VertexBlendArb** = ((int)0x86A7), **CurrentWeightArb** = ((int)0x86A8), **WeightArrayTypeArb** = ((int)0x86A9),

WeightArrayStrideArb = ((int)0x86AA), **WeightArraySizeArb** = ((int)0x86AB), **WeightArrayPointerArb** = ((int)0x86AC), **WeightArrayArb** = ((int)0x86AD),

Modelview2Arb = ((int)0x8722), **Modelview3Arb** = ((int)0x8723), **Modelview4Arb** = ((int)0x8724), **Modelview5Arb** = ((int)0x8725),

Modelview6Arb = ((int)0x8726), **Modelview7Arb** = ((int)0x8727), **Modelview8Arb** = ((int)0x8728), **Modelview9Arb** = ((int)0x8729),

Modelview10Arb = ((int)0x872A), **Modelview11Arb** = ((int)0x872B), **Modelview12Arb** = ((int)0x872C), **Modelview13Arb** = ((int)0x872D),

Modelview14Arb = ((int)0x872E), **Modelview15Arb** = ((int)0x872F), **Modelview16Arb** = ((int)0x8730), **Modelview17Arb** = ((int)0x8731),

Modelview18Arb = ((int)0x8732), **Modelview19Arb** = ((int)0x8733), **Modelview20Arb** = ((int)0x8734), **Modelview21Arb** = ((int)0x8735),

```

Modelview22Arb = ((int)0x8736), Modelview23Arb = ((int)0x8737), Modelview24Arb = ((int)0x8738), Modelview25Arb = ((int)0x8739),
Modelview26Arb = ((int)0x873A), Modelview27Arb = ((int)0x873B), Modelview28Arb = ((int)0x873C), Modelview29Arb = ((int)0x873D),
Modelview30Arb = ((int)0x873E), Modelview31Arb = ((int)0x873F) }
• enum ArbVertexBufferObject {
    BufferSizeArb = ((int)0x8764), BufferUsageArb = ((int)0x8765), ArrayBufferArb = ((int)0x8892), ElementArrayBufferArb = ((int)0x8893),
    ArrayBufferBindingArb = ((int)0x8894), ElementArrayBufferBindingArb = ((int)0x8895), VertexArrayBufferBindingArb = ((int)0x8896), NormalArrayBufferBindingArb = ((int)0x8897),
    ColorArrayBufferBindingArb = ((int)0x8898), IndexArrayBufferBindingArb = ((int)0x8899), TextureCoordArrayBufferBindingArb = ((int)0x889A), EdgeFlagArrayBufferBindingArb = ((int)0x889B),
    SecondaryColorArrayBufferBindingArb = ((int)0x889C), FogCoordinateArrayBufferBindingArb = ((int)0x889D), WeightArrayBufferBindingArb = ((int)0x889E), VertexAttribArrayBufferBindingArb = ((int)0x889F),
    ReadOnlyArb = ((int)0x88B8), WriteOnlyArb = ((int)0x88B9), ReadWriteArb = ((int)0x88BA), BufferAccessArb = ((int)0x88BB),
    BufferMappedArb = ((int)0x88BC), BufferMapPointerArb = ((int)0x88BD), StreamDrawArb = ((int)0x88E0), StreamReadArb = ((int)0x88E1),
    StreamCopyArb = ((int)0x88E2), StaticDrawArb = ((int)0x88E4), StaticReadArb = ((int)0x88E5), StaticCopyArb = ((int)0x88E6),
    DynamicDrawArb = ((int)0x88E8), DynamicReadArb = ((int)0x88E9), DynamicCopyArb = ((int)0x88EA) }
• enum ArbVertexProgram {
    ColorSumArb = ((int)0x8458), VertexProgramArb = ((int)0x8620), VertexAttribArrayEnabledArb = ((int)0x8622), VertexAttribArraySizeArb = ((int)0x8623),
    VertexAttribArrayStrideArb = ((int)0x8624), VertexAttribArrayTypeArb = ((int)0x8625), CurrentVertexAttribArb = ((int)0x8626), ProgramLengthArb = ((int)0x8627),
    ProgramStringArb = ((int)0x8628), MaxProgramMatrixStackDepthArb = ((int)0x862E), MaxProgramMatricesArb = ((int)0x862F), CurrentMatrixStackDepthArb = ((int)0x8640),
    CurrentMatrixArb = ((int)0x8641), VertexProgramPointSizeArb = ((int)0x8642), VertexProgramTwoSideArb = ((int)0x8643), VertexAttribArrayPointerArb = ((int)0x8645),
    ProgramErrorPositionArb = ((int)0x864B), ProgramBindingArb = ((int)0x8677), MaxVertexAttribsArb = ((int)0x8869), VertexAttribArrayNormalizedArb = ((int)0x886A),

```

ProgramErrorStringArb = ((int)0x8874), **ProgramFormatAsciiArb** = ((int)0x8875), **ProgramFormatArb** = ((int)0x8876), **ProgramInstructionsArb** = ((int)0x88A0),

MaxProgramInstructionsArb = ((int)0x88A1), **ProgramNativeInstructionsArb** = ((int)0x88A2), **MaxProgramNativeInstructionsArb** = ((int)0x88A3), **ProgramTemporariesArb** = ((int)0x88A4),

MaxProgramTemporariesArb = ((int)0x88A5), **ProgramNativeTemporariesArb** = ((int)0x88A6), **MaxProgramNativeTemporariesArb** = ((int)0x88A7), **ProgramParametersArb** = ((int)0x88A8),

MaxProgramParametersArb = ((int)0x88A9), **ProgramNativeParametersArb** = ((int)0x88AA), **MaxProgramNativeParametersArb** = ((int)0x88AB), **ProgramAttribsArb** = ((int)0x88AC),

MaxProgramAttribsArb = ((int)0x88AD), **ProgramNativeAttribsArb** = ((int)0x88AE), **MaxProgramNativeAttribsArb** = ((int)0x88AF), **ProgramAddressRegistersArb** = ((int)0x88B0),

MaxProgramAddressRegistersArb = ((int)0x88B1), **ProgramNativeAddressRegistersArb** = ((int)0x88B2), **MaxProgramNativeAddressRegistersArb** = ((int)0x88B3), **MaxProgramLocalParametersArb** = ((int)0x88B4),

MaxProgramEnvParametersArb = ((int)0x88B5), **ProgramUnderNativeLimitsArb** = ((int)0x88B6), **TransposeCurrentMatrixArb** = ((int)0x88B7), **Matrix0Arb** = ((int)0x88C0),

Matrix1Arb = ((int)0x88C1), **Matrix2Arb** = ((int)0x88C2), **Matrix3Arb** = ((int)0x88C3), **Matrix4Arb** = ((int)0x88C4),

Matrix5Arb = ((int)0x88C5), **Matrix6Arb** = ((int)0x88C6), **Matrix7Arb** = ((int)0x88C7), **Matrix8Arb** = ((int)0x88C8),

Matrix9Arb = ((int)0x88C9), **Matrix10Arb** = ((int)0x88CA), **Matrix11Arb** = ((int)0x88CB), **Matrix12Arb** = ((int)0x88CC),

Matrix13Arb = ((int)0x88CD), **Matrix14Arb** = ((int)0x88CE), **Matrix15Arb** = ((int)0x88CF), **Matrix16Arb** = ((int)0x88D0),

Matrix17Arb = ((int)0x88D1), **Matrix18Arb** = ((int)0x88D2), **Matrix19Arb** = ((int)0x88D3), **Matrix20Arb** = ((int)0x88D4),

Matrix21Arb = ((int)0x88D5), **Matrix22Arb** = ((int)0x88D6), **Matrix23Arb** = ((int)0x88D7), **Matrix24Arb** = ((int)0x88D8),

Matrix25Arb = ((int)0x88D9), **Matrix26Arb** = ((int)0x88DA), **Matrix27Arb** = ((int)0x88DB), **Matrix28Arb** = ((int)0x88DC),

Matrix29Arb = ((int)0x88DD), **Matrix30Arb** = ((int)0x88DE), **Matrix31Arb** = ((int)0x88DF) }

- enum **ArbVertexShader** {

VertexShaderArb = ((int)0x8B31), **MaxVertexUniformComponentsArb** = ((int)0x8B4A), **MaxVaryingFloatsArb** = ((int)0x8B4B), **MaxVertexTextureImageUnitsArb** = ((int)0x8B4C),

```

MaxCombinedTextureImageUnitsArb = ((int)0x8B4D), ObjectActiveAt-
tributesArb = ((int)0x8B89), ObjectActiveAttributeMaxLengthArb =
((int)0x8B8A) }
• enum ArbWindowPos
• enum ArrayCap {
VertexArray = ((int)0x8074), NormalArray = ((int)0x8075), ColorArray =
((int)0x8076), IndexArray = ((int)0x8077),
TextureCoordArray = ((int)0x8078), EdgeFlagArray = ((int)0x8079), Fog-
CoordArray = ((int)0x8457), SecondaryColorArray = ((int)0x845E) }
• enum AssemblyProgramFormatArb { ProgramFormatAsciiArb =
((int)0x8875) }
• enum AssemblyProgramParameterArb {
ProgramLength = ((int)0x8627), ProgramBinding = ((int)0x8677), Pro-
gramAluInstructionsArb = ((int)0x8805), ProgramTexInstructionsArb =
((int)0x8806),
ProgramTexIndirectionsArb = ((int)0x8807), ProgramNativeAluIn-
structionsArb = ((int)0x8808), ProgramNativeTexInstructionsArb =
((int)0x8809), ProgramNativeTexIndirectionsArb = ((int)0x880A),
MaxProgramAluInstructionsArb = ((int)0x880B), MaxProgramTexInstruc-
tionsArb = ((int)0x880C), MaxProgramTexIndirectionsArb = ((int)0x880D),
MaxProgramNativeAluInstructionsArb = ((int)0x880E),
MaxProgramNativeTexInstructionsArb = ((int)0x880F), MaxProgramNa-
tiveTexIndirectionsArb = ((int)0x8810), ProgramFormat = ((int)0x8876),
ProgramInstruction = ((int)0x88A0),
MaxProgramInstructions = ((int)0x88A1), ProgramNativeInstructions
= ((int)0x88A2), MaxProgramNativeInstructions = ((int)0x88A3), Pro-
gramTemporaries = ((int)0x88A4),
MaxProgramTemporaries = ((int)0x88A5), ProgramNativeTemporaries =
((int)0x88A6), MaxProgramNativeTemporaries = ((int)0x88A7), Program-
Parameters = ((int)0x88A8),
MaxProgramParameters = ((int)0x88A9), ProgramNativeParameters =
((int)0x88AA), MaxProgramNativeParameters = ((int)0x88AB), Progra-
mAttribs = ((int)0x88AC),
MaxProgramAttribs = ((int)0x88AD), ProgramNativeAttribs =
((int)0x88AE), MaxProgramNativeAttribs = ((int)0x88AF), ProgramAd-
dressRegisters = ((int)0x88B0),
MaxProgramAddressRegisters = ((int)0x88B1), ProgramNativeAd-
dressRegisters = ((int)0x88B2), MaxProgramNativeAddressRegisters =
((int)0x88B3), MaxProgramLocalParameters = ((int)0x88B4),
MaxProgramEnvParameters = ((int)0x88B5), ProgramUnderNativeLimits
= ((int)0x88B6) }

```

- enum **AssemblyProgramStringParameterArb** { **ProgramString** = ((int)0x8628) }
- enum **AssemblyProgramTargetArb** { **VertexProgram** = ((int)0x8620), **FragmentProgram** = ((int)0x8804), **GeometryProgramNv** = ((int)0x8C26) }
- enum **AtiDrawBuffers** {
MaxDrawBuffersAti = ((int)0x8824), **DrawBuffer0Ati** = ((int)0x8825),
DrawBuffer1Ati = ((int)0x8826), **DrawBuffer2Ati** = ((int)0x8827),
DrawBuffer3Ati = ((int)0x8828), **DrawBuffer4Ati** = ((int)0x8829), **DrawBuffer5Ati** = ((int)0x882A), **DrawBuffer6Ati** = ((int)0x882B),
DrawBuffer7Ati = ((int)0x882C), **DrawBuffer8Ati** = ((int)0x882D), **DrawBuffer9Ati** = ((int)0x882E), **DrawBuffer10Ati** = ((int)0x882F),
DrawBuffer11Ati = ((int)0x8830), **DrawBuffer12Ati** = ((int)0x8831), **DrawBuffer13Ati** = ((int)0x8832), **DrawBuffer14Ati** = ((int)0x8833),
DrawBuffer15Ati = ((int)0x8834) }
- enum **AtiElementArray** { **ElementArrayAti** = ((int)0x8768), **ElementArrayTypeAti** = ((int)0x8769), **ElementArrayPointerAti** = ((int)0x876A) }
- enum **AtiEnvmapBumpmap** {
BumpRotMatrixAti = ((int)0x8775), **BumpRotMatrixSizeAti** = ((int)0x8776), **BumpNumTexUnitsAti** = ((int)0x8777), **BumpTexUnitsAti** = ((int)0x8778),
DudvAti = ((int)0x8779), **Du8dv8Ati** = ((int)0x877A), **BumpEnvmapAti** = ((int)0x877B), **BumpTargetAti** = ((int)0x877C) }
- enum **AtiFragmentShader** {
GL2XBitAti = ((int)0x00000001), **RedBitAti** = ((int)0x00000001), **CompBitAti** = ((int)0x00000002), **GL4XBitAti** = ((int)0x00000002),
GreenBitAti = ((int)0x00000002), **BlueBitAti** = ((int)0x00000004), **GL8XBitAti** = ((int)0x00000004), **NegateBitAti** = ((int)0x00000004),
BiasBitAti = ((int)0x00000008), **HalfBitAti** = ((int)0x00000008), **QuarterBitAti** = ((int)0x00000010), **EighthBitAti** = ((int)0x00000020),
SaturateBitAti = ((int)0x00000040), **FragmentShaderAti** = ((int)0x8920), **Reg0Ati** = ((int)0x8921), **Reg1Ati** = ((int)0x8922),
Reg2Ati = ((int)0x8923), **Reg3Ati** = ((int)0x8924), **Reg4Ati** = ((int)0x8925), **Reg5Ati** = ((int)0x8926),
Reg6Ati = ((int)0x8927), **Reg7Ati** = ((int)0x8928), **Reg8Ati** = ((int)0x8929), **Reg9Ati** = ((int)0x892A),
Reg10Ati = ((int)0x892B), **Reg11Ati** = ((int)0x892C), **Reg12Ati** = ((int)0x892D), **Reg13Ati** = ((int)0x892E),
Reg14Ati = ((int)0x892F), **Reg15Ati** = ((int)0x8930), **Reg16Ati** = ((int)0x8931), **Reg17Ati** = ((int)0x8932),
Reg18Ati = ((int)0x8933), **Reg19Ati** = ((int)0x8934), **Reg20Ati** = ((int)0x8935), **Reg21Ati** = ((int)0x8936),

```

Reg22Ati = ((int)0x8937), Reg23Ati = ((int)0x8938), Reg24Ati =
((int)0x8939), Reg25Ati = ((int)0x893A),
Reg26Ati = ((int)0x893B), Reg27Ati = ((int)0x893C), Reg28Ati =
((int)0x893D), Reg29Ati = ((int)0x893E),
Reg30Ati = ((int)0x893F), Reg31Ati = ((int)0x8940), Con0Ati = ((int)0x8941),
Con1Ati = ((int)0x8942),
Con2Ati = ((int)0x8943), Con3Ati = ((int)0x8944), Con4Ati = ((int)0x8945),
Con5Ati = ((int)0x8946),
Con6Ati = ((int)0x8947), Con7Ati = ((int)0x8948), Con8Ati = ((int)0x8949),
Con9Ati = ((int)0x894A),
Con10Ati = ((int)0x894B), Con11Ati = ((int)0x894C), Con12Ati =
((int)0x894D), Con13Ati = ((int)0x894E),
Con14Ati = ((int)0x894F), Con15Ati = ((int)0x8950), Con16Ati =
((int)0x8951), Con17Ati = ((int)0x8952),
Con18Ati = ((int)0x8953), Con19Ati = ((int)0x8954), Con20Ati =
((int)0x8955), Con21Ati = ((int)0x8956),
Con22Ati = ((int)0x8957), Con23Ati = ((int)0x8958), Con24Ati =
((int)0x8959), Con25Ati = ((int)0x895A),
Con26Ati = ((int)0x895B), Con27Ati = ((int)0x895C), Con28Ati =
((int)0x895D), Con29Ati = ((int)0x895E),
Con30Ati = ((int)0x895F), Con31Ati = ((int)0x8960), MovAti = ((int)0x8961),
AddAti = ((int)0x8963),
MulAti = ((int)0x8964), SubAti = ((int)0x8965), Dot3Ati = ((int)0x8966),
Dot4Ati = ((int)0x8967),
MadAti = ((int)0x8968), LerpAti = ((int)0x8969), CndAti = ((int)0x896A),
Cnd0Ati = ((int)0x896B),
Dot2AddAti = ((int)0x896C), SecondaryInterpolatorAti = ((int)0x896D),
NumFragmentRegistersAti = ((int)0x896E), NumFragmentConstantsAti =
((int)0x896F),
NumPassesAti = ((int)0x8970), NumInstructionsPerPassAti = ((int)0x8971),
NumInstructionsTotalAti = ((int)0x8972), NumInputInterpolatorCompo-
ponentsAti = ((int)0x8973),
NumLoopbackComponentsAti = ((int)0x8974), ColorAlphaPairingAti =
((int)0x8975), SwizzleStrAti = ((int)0x8976), SwizzleStqAti = ((int)0x8977),
SwizzleStrDrAti = ((int)0x8978), SwizzleStqDqAti = ((int)0x8979), Swiz-
zleStrqAti = ((int)0x897A), SwizzleStrqDqAti = ((int)0x897B) }
• enum AtiMapObjectBuffer
• enum AtiMeminfo { VboFreeMemoryAti = ((int)0x87FB), TextureFreeMem-
oryAti = ((int)0x87FC), RenderbufferFreeMemoryAti = ((int)0x87FD) }
• enum AtiPixelFormatFloat { TypeRgbaFloatAti = ((int)0x8820), Color-
ClearUnclampedValueAti = ((int)0x8835) }

```

- enum **AtiPnTriangles** {
 - PnTrianglesAti** = ((int)0x87F0), **MaxPnTrianglesTessellationLevelAti** = ((int)0x87F1), **PnTrianglesPointModeAti** = ((int)0x87F2), **PnTrianglesNormalModeAti** = ((int)0x87F3),
 - PnTrianglesTessellationLevelAti** = ((int)0x87F4), **PnTrianglesPointModeLinearAti** = ((int)0x87F5), **PnTrianglesPointModeCubicAti** = ((int)0x87F6), **PnTrianglesNormalModeLinearAti** = ((int)0x87F7),
 - PnTrianglesNormalModeQuadraticAti** = ((int)0x87F8) }
- enum **AtiSeparateStencil** { **StencilBackFuncAti** = ((int)0x8800), **StencilBackFailAti** = ((int)0x8801), **StencilBackPassDepthFailAti** = ((int)0x8802), **StencilBackPassDepthPassAti** = ((int)0x8803) }
- enum **AtiTextFragmentShader** { **TextFragmentShaderAti** = ((int)0x8200) }
- enum **AtiTextureEnvCombine3** { **ModulateAddAti** = ((int)0x8744), **ModulateSignedAddAti** = ((int)0x8745), **ModulateSubtractAti** = ((int)0x8746) }
- enum **AtiTextureFloat** {
 - RgbaFloat32Ati** = ((int)0x8814), **RgbFloat32Ati** = ((int)0x8815), **AlphaFloat32Ati** = ((int)0x8816), **IntensityFloat32Ati** = ((int)0x8817),
 - LuminanceFloat32Ati** = ((int)0x8818), **LuminanceAlphaFloat32Ati** = ((int)0x8819), **RgbFloat16Ati** = ((int)0x881A), **RgbFloat16Ati** = ((int)0x881B),
 - AlphaFloat16Ati** = ((int)0x881C), **IntensityFloat16Ati** = ((int)0x881D), **LuminanceFloat16Ati** = ((int)0x881E), **LuminanceAlphaFloat16Ati** = ((int)0x881F) }
- enum **AtiTextureMirrorOnce** { **MirrorClampAti** = ((int)0x8742), **MirrorClampToEdgeAti** = ((int)0x8743) }
- enum **AtiVertexArrayObject** {
 - StaticAti** = ((int)0x8760), **DynamicAti** = ((int)0x8761), **PreserveAti** = ((int)0x8762), **DiscardAti** = ((int)0x8763),
 - ObjectBufferSizeAti** = ((int)0x8764), **ObjectBufferUsageAti** = ((int)0x8765), **ArrayObjectBufferAti** = ((int)0x8766), **ArrayObjectOffsetAti** = ((int)0x8767) }
- enum **AtiVertexAttribArrayObject**
- enum **AtiVertexStreams** {
 - MaxVertexStreamsAti** = ((int)0x876B), **VertexStream0Ati** = ((int)0x876C), **VertexStream1Ati** = ((int)0x876D), **VertexStream2Ati** = ((int)0x876E),
 - VertexStream3Ati** = ((int)0x876F), **VertexStream4Ati** = ((int)0x8770), **VertexStream5Ati** = ((int)0x8771), **VertexStream6Ati** = ((int)0x8772),
 - VertexStream7Ati** = ((int)0x8773), **VertexSourceAti** = ((int)0x8774) }
- enum **AttribMask** {
 - CurrentBit** = ((int)0x00000001), **PointBit** = ((int)0x00000002), **LineBit** = ((int)0x00000004), **PolygonBit** = ((int)0x00000008),


```

PolygonStippleBit = ((int)0x00000010), PixelModeBit = ((int)0x00000020),
LightingBit = ((int)0x00000040), FogBit = ((int)0x00000080),

DepthBufferBit = ((int)0x00000100), AccumBufferBit = ((int)0x00000200),
StencilBufferBit = ((int)0x00000400), ViewportBit = ((int)0x00000800),

TransformBit = ((int)0x00001000), EnableBit = ((int)0x00002000), Color-
BufferBit = ((int)0x00004000), HintBit = ((int)0x00008000),

EvalBit = ((int)0x00010000), ListBit = ((int)0x00020000), TextureBit =
((int)0x00040000), ScissorBit = ((int)0x00080000),

MultisampleBit = ((int)0x20000000), AllAttribBits =
unchecked((int)0xFFFFFFFF) }
• enum BeginFeedbackMode { Points = ((int)0x0000), Lines = ((int)0x0001),
Triangles = ((int)0x0004) }
• enum BeginMode {
Points = ((int)0x0000), Lines = ((int)0x0001), LineLoop = ((int)0x0002),
LineStrip = ((int)0x0003),

Triangles = ((int)0x0004), TriangleStrip = ((int)0x0005), TriangleFan =
((int)0x0006), Quads = ((int)0x0007),

QuadStrip = ((int)0x0008), Polygon = ((int)0x0009), LinesAdjacency =
((int)0xA), LineStripAdjacency = ((int)0xB),

TrianglesAdjacency = ((int)0xC), TriangleStripAdjacency = ((int)0xD) }
• enum BlendEquationMode {
FuncAdd = ((int)0x8006), Min = ((int)0x8007), Max = ((int)0x8008), Func-
Subtract = ((int)0x800A),

FuncReverseSubtract = ((int)0x800B) }
• enum BlendEquationModeExt {
LogicOp = ((int)0x0BF1), FuncAddExt = ((int)0x8006), MinExt =
((int)0x8007), MaxExt = ((int)0x8008),

FuncSubtractExt = ((int)0x800A), FuncReverseSubtractExt = ((int)0x800B),
AlphaMinSgix = ((int)0x8320), AlphaMaxSgix = ((int)0x8321) }
• enum BlendingFactorDest {
Zero = ((int)0), SrcColor = ((int)0x0300), OneMinusSrcColor = ((int)0x0301),
SrcAlpha = ((int)0x0302),

OneMinusSrcAlpha = ((int)0x0303), DstAlpha = ((int)0x0304), OneMinusD-
stAlpha = ((int)0x0305), DstColor = ((int)0x0306),

OneMinusDstColor = ((int)0x0307), ConstantColor = ((int)0x8001), Con-
stantColorExt = ((int)0x8001), OneMinusConstantColor = ((int)0x8002),

OneMinusConstantColorExt = ((int)0x8002), ConstantAlpha =
((int)0x8003), ConstantAlphaExt = ((int)0x8003), OneMinusConstantAlpha
= ((int)0x8004),

OneMinusConstantAlphaExt = ((int)0x8004), One = ((int)1) }

```

- enum **BlendingFactorSrc** {
Zero = ((int)0), **SrcAlpha** = ((int)0x0302), **OneMinusSrcAlpha** = ((int)0x0303), **DstAlpha** = ((int)0x0304),
OneMinusDstAlpha = ((int)0x0305), **DstColor** = ((int)0x0306), **OneMinusDstColor** = ((int)0x0307), **SrcAlphaSaturate** = ((int)0x0308),
ConstantColor = ((int)0x8001), **ConstantColorExt** = ((int)0x8001), **OneMinusConstantColor** = ((int)0x8002), **OneMinusConstantColorExt** = ((int)0x8002),
ConstantAlpha = ((int)0x8003), **ConstantAlphaExt** = ((int)0x8003), **OneMinusConstantAlpha** = ((int)0x8004), **OneMinusConstantAlphaExt** = ((int)0x8004),
One = ((int)1) }
- enum **BlitFramebufferFilter** { **Nearest** = ((int)0x2600), **Linear** = ((int)0x2601) }
- enum **Boolean** { **False** = ((int)0), **True** = ((int)1) }
- enum **BufferAccess** { **ReadOnly** = ((int)0x88B8), **WriteOnly** = ((int)0x88B9), **ReadWrite** = ((int)0x88BA) }
- enum **BufferAccessArb** { **ReadOnly** = ((int)0x88B8), **WriteOnly** = ((int)0x88B9), **ReadWrite** = ((int)0x88BA) }
- enum **BufferAccessMask** {
MapReadBit = ((int)0x0001), **MapWriteBit** = ((int)0x0002), **MapInvalidat-eRangeBit** = ((int)0x0004), **MapInvalidateBufferBit** = ((int)0x0008),
MapFlushExplicitBit = ((int)0x0010), **MapUnsynchronizedBit** = ((int)0x0020) }
- enum **BufferParameterApple** { **BufferSerializedModifyApple** = ((int)0x8A12), **BufferFlushingUnmapApple** = ((int)0x8A13) }
- enum **BufferParameterName** { **BufferSize** = ((int)0x8764), **BufferUsage** = ((int)0x8765), **BufferAccess** = ((int)0x88BB), **BufferMapped** = ((int)0x88BC) }
- enum **BufferParameterNameArb** { **BufferSize** = ((int)0x8764), **BufferUsage** = ((int)0x8765), **BufferAccess** = ((int)0x88BB), **BufferMapped** = ((int)0x88BC) }
- enum **BufferPointer** { **BufferMapPointer** = ((int)0x88BD) }
- enum **BufferPointerNameArb** { **BufferMapPointer** = ((int)0x88BD) }
- enum **BufferTarget** {
ArrayBuffer = ((int)0x8892), **ElementArrayBuffer** = ((int)0x8893), **Pixel-PackBuffer** = ((int)0x88EB), **PixelUnpackBuffer** = ((int)0x88EC),
UniformBuffer = ((int)0x8A11), **TextureBuffer** = ((int)0x8C2A), **Transform-FeedbackBuffer** = ((int)0x8C8E), **CopyReadBuffer** = ((int)0x8F36),
CopyWriteBuffer = ((int)0x8F37) }
- enum **BufferTargetArb** { **ArrayBuffer** = ((int)0x8892), **ElementArrayBuffer** = ((int)0x8893) }

- enum **BufferUsageArb** {
StreamDraw = ((int)0x88E0), **StreamRead** = ((int)0x88E1), **StreamCopy** = ((int)0x88E2), **StaticDraw** = ((int)0x88E4),
StaticRead = ((int)0x88E5), **StaticCopy** = ((int)0x88E6), **DynamicDraw** = ((int)0x88E8), **DynamicRead** = ((int)0x88E9),
DynamicCopy = ((int)0x88EA) }
- enum **BufferUsageHint** {
StreamDraw = ((int)0x88E0), **StreamRead** = ((int)0x88E1), **StreamCopy** = ((int)0x88E2), **StaticDraw** = ((int)0x88E4),
StaticRead = ((int)0x88E5), **StaticCopy** = ((int)0x88E6), **DynamicDraw** = ((int)0x88E8), **DynamicRead** = ((int)0x88E9),
DynamicCopy = ((int)0x88EA) }
- enum **ClampColorMode** { **False** = ((int)0), **FixedOnly** = ((int)0x891D), **True** = ((int)1) }
- enum **ClampColorTarget** { **ClampVertexColor** = ((int)0x891A), **ClampFragmentColor** = ((int)0x891B), **ClampReadColor** = ((int)0x891C) }
- enum **ClearBuffer** { **Color** = ((int)0x1800), **Depth** = ((int)0x1801), **Stencil** = ((int)0x1802), **DepthStencil** = ((int)0x84F9) }
- enum **ClearBufferMask** { **DepthBufferBit** = ((int)0x00000100), **AccumBufferBit** = ((int)0x00000200), **StencilBufferBit** = ((int)0x00000400), **ColorBufferBit** = ((int)0x00004000) }
- enum **ClientAttribMask** { **ClientPixelStoreBit** = ((int)0x00000001), **ClientVertexArrayBit** = ((int)0x00000002), **ClientAllAttribBits** = unchecked((int)0xFFFFFFFF) }
- enum **ClipPlaneName** {
ClipPlane0 = ((int)0x3000), **ClipPlane1** = ((int)0x3001), **ClipPlane2** = ((int)0x3002), **ClipPlane3** = ((int)0x3003),
ClipPlane4 = ((int)0x3004), **ClipPlane5** = ((int)0x3005) }
- enum **ColorMaterialFace** { **Front** = ((int)0x0404), **Back** = ((int)0x0405), **FrontAndBack** = ((int)0x0408) }
- enum **ColorMaterialParameter** {
Ambient = ((int)0x1200), **Diffuse** = ((int)0x1201), **Specular** = ((int)0x1202), **Emission** = ((int)0x1600),
AmbientAndDiffuse = ((int)0x1602) }
- enum **ColorPointerType** {
Byte = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402), **UnsignedShort** = ((int)0x1403),
Int = ((int)0x1404), **UnsignedInt** = ((int)0x1405), **Float** = ((int)0x1406), **Double** = ((int)0x140A),
HalfFloat = ((int)0x140B) }

- enum **ColorTableParameterPName** { **ColorTableScale** = ((int)0x80D6), **ColorTableBias** = ((int)0x80D7) }
- enum **ColorTableParameterPNameSgi** { **ColorTableScaleSgi** = ((int)0x80D6), **ColorTableBiasSgi** = ((int)0x80D7) }
- enum **ColorTableTarget** {
ColorTable = ((int)0x80D0), **PostConvolutionColorTable** = ((int)0x80D1),
PostColorMatrixColorTable = ((int)0x80D2), **ProxyColorTable** = ((int)0x80D3),
ProxyPostConvolutionColorTable = ((int)0x80D4), **ProxyPostColorMatrixColorTable** = ((int)0x80D5) }
- enum **ColorTableTargetSgi** {
TextureColorTableSgi = ((int)0x80BC), **ProxyTextureColorTableSgi** = ((int)0x80BD), **ColorTableSgi** = ((int)0x80D0), **PostConvolutionColorTableSgi** = ((int)0x80D1),
PostColorMatrixColorTableSgi = ((int)0x80D2), **ProxyColorTableSgi** = ((int)0x80D3), **ProxyPostConvolutionColorTableSgi** = ((int)0x80D4), **ProxyPostColorMatrixColorTableSgi** = ((int)0x80D5) }
- enum **ConditionalRenderType** { **QueryWait** = ((int)0x8E13), **QueryNoWait** = ((int)0x8E14), **QueryByRegionWait** = ((int)0x8E15), **QueryByRegionNoWait** = ((int)0x8E16) }
- enum **ConvolutionBorderModeExt** { **ReduceExt** = ((int)0x8016) }
- enum **ConvolutionParameter** { **ConvolutionBorderMode** = ((int)0x8013), **ConvolutionFilterScale** = ((int)0x8014), **ConvolutionFilterBias** = ((int)0x8015) }
- enum **ConvolutionParameterExt** { **ConvolutionBorderModeExt** = ((int)0x8013), **ConvolutionFilterScaleExt** = ((int)0x8014), **ConvolutionFilterBiasExt** = ((int)0x8015) }
- enum **ConvolutionParameterValue** { **Reduce** = ((int)0x8016), **ConstantBorder** = ((int)0x8151), **ReplicateBorder** = ((int)0x8153) }
- enum **ConvolutionTarget** { **Convolution1D** = ((int)0x8010), **Convolution2D** = ((int)0x8011), **Separable2D** = ((int)0x8012) }
- enum **ConvolutionTargetExt** { **Convolution1DExt** = ((int)0x8010), **Convolution2DExt** = ((int)0x8011) }
- enum **CullFaceMode** { **Front** = ((int)0x0404), **Back** = ((int)0x0405), **FrontAndBack** = ((int)0x0408) }
- enum **DataType** {
Byte = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402), **UnsignedShort** = ((int)0x1403),
Int = ((int)0x1404), **UnsignedInt** = ((int)0x1405), **Float** = ((int)0x1406), **GL2Bytes** = ((int)0x1407),
GL3Bytes = ((int)0x1408), **GL4Bytes** = ((int)0x1409), **Double** = ((int)0x140A), **DoubleExt** = ((int)0x140A) }

- enum **DepthFunction** {
 - Never** = ((int)0x0200), **Less** = ((int)0x0201), **Equal** = ((int)0x0202), **Lequal** = ((int)0x0203),
 - Greater** = ((int)0x0204), **Notequal** = ((int)0x0205), **Gequal** = ((int)0x0206), **Always** = ((int)0x0207) }
- enum **DrawBufferMode** {
 - None** = ((int)0), **FrontLeft** = ((int)0x0400), **FrontRight** = ((int)0x0401), **BackLeft** = ((int)0x0402),
 - BackRight** = ((int)0x0403), **Front** = ((int)0x0404), **Back** = ((int)0x0405), **Left** = ((int)0x0406),
 - Right** = ((int)0x0407), **FrontAndBack** = ((int)0x0408), **Aux0** = ((int)0x0409), **Aux1** = ((int)0x040A),
 - Aux2** = ((int)0x040B), **Aux3** = ((int)0x040C), **ColorAttachment0** = ((int)0x8CE0), **ColorAttachment1** = ((int)0x8CE1),
 - ColorAttachment2** = ((int)0x8CE2), **ColorAttachment3** = ((int)0x8CE3), **ColorAttachment4** = ((int)0x8CE4), **ColorAttachment5** = ((int)0x8CE5),
 - ColorAttachment6** = ((int)0x8CE6), **ColorAttachment7** = ((int)0x8CE7), **ColorAttachment8** = ((int)0x8CE8), **ColorAttachment9** = ((int)0x8CE9),
 - ColorAttachment10** = ((int)0x8CEA), **ColorAttachment11** = ((int)0x8CEB), **ColorAttachment12** = ((int)0x8CEC), **ColorAttachment13** = ((int)0x8CED),
 - ColorAttachment14** = ((int)0x8CEE), **ColorAttachment15** = ((int)0x8CEF) }
- enum **DrawBuffersEnum** {
 - None** = ((int)0), **FrontLeft** = ((int)0x0400), **FrontRight** = ((int)0x0401), **BackLeft** = ((int)0x0402),
 - BackRight** = ((int)0x0403), **Aux0** = ((int)0x0409), **Aux1** = ((int)0x040A), **Aux2** = ((int)0x040B),
 - Aux3** = ((int)0x040C), **ColorAttachment0** = ((int)0x8CE0), **ColorAttachment1** = ((int)0x8CE1), **ColorAttachment2** = ((int)0x8CE2),
 - ColorAttachment3** = ((int)0x8CE3), **ColorAttachment4** = ((int)0x8CE4), **ColorAttachment5** = ((int)0x8CE5), **ColorAttachment6** = ((int)0x8CE6),
 - ColorAttachment7** = ((int)0x8CE7), **ColorAttachment8** = ((int)0x8CE8), **ColorAttachment9** = ((int)0x8CE9), **ColorAttachment10** = ((int)0x8CEA),
 - ColorAttachment11** = ((int)0x8CEB), **ColorAttachment12** = ((int)0x8CEC), **ColorAttachment13** = ((int)0x8CED), **ColorAttachment14** = ((int)0x8CEE),
 - ColorAttachment15** = ((int)0x8CEF) }
- enum **DrawElementsType** { **UnsignedByte** = ((int)0x1401), **UnsignedShort** = ((int)0x1403), **UnsignedInt** = ((int)0x1405) }
- enum **EnableCap** {
 - PointSmooth** = ((int)0x0B10), **LineSmooth** = ((int)0x0B20), **LineStipple** = ((int)0x0B24), **PolygonSmooth** = ((int)0x0B41),

PolygonStipple = ((int)0x0B42), **CullFace** = ((int)0x0B44), **Lighting** = ((int)0x0B50), **ColorMaterial** = ((int)0x0B57),
Fog = ((int)0x0B60), **DepthTest** = ((int)0x0B71), **StencilTest** = ((int)0x0B90), **Normalize** = ((int)0x0BA1),
AlphaTest = ((int)0x0BC0), **Dither** = ((int)0x0BD0), **Blend** = ((int)0x0BE2), **IndexLogicOp** = ((int)0x0BF1),
ColorLogicOp = ((int)0x0BF2), **ScissorTest** = ((int)0x0C11), **TextureGenS** = ((int)0x0C60), **TextureGenT** = ((int)0x0C61),
TextureGenR = ((int)0x0C62), **TextureGenQ** = ((int)0x0C63), **AutoNormal** = ((int)0x0D80), **Map1Color4** = ((int)0x0D90),
Map1Index = ((int)0x0D91), **Map1Normal** = ((int)0x0D92), **Map1TextureCoord1** = ((int)0x0D93), **Map1TextureCoord2** = ((int)0x0D94),
Map1TextureCoord3 = ((int)0x0D95), **Map1TextureCoord4** = ((int)0x0D96), **Map1Vertex3** = ((int)0x0D97), **Map1Vertex4** = ((int)0x0D98),
Map2Color4 = ((int)0x0DB0), **Map2Index** = ((int)0x0DB1), **Map2Normal** = ((int)0x0DB2), **Map2TextureCoord1** = ((int)0x0DB3),
Map2TextureCoord2 = ((int)0x0DB4), **Map2TextureCoord3** = ((int)0x0DB5), **Map2TextureCoord4** = ((int)0x0DB6), **Map2Vertex3** = ((int)0x0DB7),
Map2Vertex4 = ((int)0x0DB8), **Texture1D** = ((int)0x0DE0), **Texture2D** = ((int)0x0DE1), **PolygonOffsetPoint** = ((int)0x2A01),
PolygonOffsetLine = ((int)0x2A02), **ClipPlane0** = ((int)0x3000), **ClipPlane1** = ((int)0x3001), **ClipPlane2** = ((int)0x3002),
ClipPlane3 = ((int)0x3003), **ClipPlane4** = ((int)0x3004), **ClipPlane5** = ((int)0x3005), **Light0** = ((int)0x4000),
Light1 = ((int)0x4001), **Light2** = ((int)0x4002), **Light3** = ((int)0x4003), **Light4** = ((int)0x4004),
Light5 = ((int)0x4005), **Light6** = ((int)0x4006), **Light7** = ((int)0x4007), **Convolution1D** = ((int)0x8010),
Convolution1DExt = ((int)0x8010), **Convolution2D** = ((int)0x8011), **Convolution2DExt** = ((int)0x8011), **Separable2D** = ((int)0x8012),
Separable2DExt = ((int)0x8012), **Histogram** = ((int)0x8024), **HistogramExt** = ((int)0x8024), **MinmaxExt** = ((int)0x802E),
PolygonOffsetFill = ((int)0x8037), **RescaleNormal** = ((int)0x803A), **RescaleNormalExt** = ((int)0x803A), **Texture3DExt** = ((int)0x806F),
VertexArray = ((int)0x8074), **NormalArray** = ((int)0x8075), **ColorArray** = ((int)0x8076), **IndexArray** = ((int)0x8077),
TextureCoordArray = ((int)0x8078), **EdgeFlagArray** = ((int)0x8079), **InterlaceSgix** = ((int)0x8094), **Multisample** = ((int)0x809D),

SampleAlphaToCoverage = ((int)0x809E), **SampleAlphaToMaskSgis** = ((int)0x809E), **SampleAlphaToOne** = ((int)0x809F), **SampleAlphaToOneSgis** = ((int)0x809F),

SampleCoverage = ((int)0x80A0), **SampleMaskSgis** = ((int)0x80A0), **TextureColorTableSgi** = ((int)0x80BC), **ColorTable** = ((int)0x80D0),

ColorTableSgi = ((int)0x80D0), **PostConvolutionColorTable** = ((int)0x80D1), **PostConvolutionColorTableSgi** = ((int)0x80D1), **PostColorMatrixColorTable** = ((int)0x80D2),

PostColorMatrixColorTableSgi = ((int)0x80D2), **Texture4DSgis** = ((int)0x8134), **PixelTexGenSgis** = ((int)0x8139), **SpriteSgis** = ((int)0x8148),

ReferencePlaneSgis = ((int)0x817D), **IrInstrument1Sgis** = ((int)0x817F), **CalligraphicFragmentSgis** = ((int)0x8183), **FramezoomSgis** = ((int)0x818B),

FogOffsetSgis = ((int)0x8198), **SharedTexturePaletteExt** = ((int)0x81FB), **AsyncHistogramSgis** = ((int)0x832C), **PixelTextureSgis** = ((int)0x8353),

AsyncTexImageSgis = ((int)0x835C), **AsyncDrawPixelsSgis** = ((int)0x835D), **AsyncReadPixelsSgis** = ((int)0x835E), **FragmentLightingSgis** = ((int)0x8400),

FragmentColorMaterialSgis = ((int)0x8401), **FragmentLight0Sgis** = ((int)0x840C), **FragmentLight1Sgis** = ((int)0x840D), **FragmentLight2Sgis** = ((int)0x840E),

FragmentLight3Sgis = ((int)0x840F), **FragmentLight4Sgis** = ((int)0x8410), **FragmentLight5Sgis** = ((int)0x8411), **FragmentLight6Sgis** = ((int)0x8412),

FragmentLight7Sgis = ((int)0x8413), **FogCoordArray** = ((int)0x8457), **ColorSum** = ((int)0x8458), **SecondaryColorArray** = ((int)0x845E),

TextureCubeMap = ((int)0x8513), **ProgramPointSize** = ((int)0x8642), **VertexProgramPointSize** = ((int)0x8642), **VertexProgramTwoSide** = ((int)0x8643),

DepthClamp = ((int)0x864F), **TextureCubeMapSeamless** = ((int)0x884F), **PointSprite** = ((int)0x8861), **RasterizerDiscard** = ((int)0x8C89),

FramebufferSrgb = ((int)0x8DB9), **SampleMask** = ((int)0x8E51), **PrimitiveRestart** = ((int)0x8F9D) }

- enum **ErrorCode** {

NoError = ((int)0), **InvalidEnum** = ((int)0x0500), **InvalidValue** = ((int)0x0501), **InvalidOperation** = ((int)0x0502),

StackOverflow = ((int)0x0503), **StackUnderflow** = ((int)0x0504), **OutOfMemory** = ((int)0x0505), **InvalidFramebufferOperation** = ((int)0x0506),

InvalidFramebufferOperationExt = ((int)0x0506), **TableTooLargeExt** = ((int)0x8031), **TextureTooLargeExt** = ((int)0x8065) }

- enum **Ext422Pixels** { **Gl422Ext** = ((int)0x80CC), **Gl422RevExt** = ((int)0x80CD), **Gl422AverageExt** = ((int)0x80CE), **Gl422RevAverageExt** = ((int)0x80CF) }

- enum **ExtAbgr** { **AbgrExt** = ((int)0x8000) }
- enum **ExtBgra** { **BgrExt** = ((int)0x80E0), **BgraExt** = ((int)0x80E1) }
- enum **ExtBindableUniform** {
MaxVertexBindableUniformsExt = ((int)0x8DE2), **MaxFragmentBindableUniformsExt** = ((int)0x8DE3), **MaxGeometryBindableUniformsExt** = ((int)0x8DE4), **MaxBindableUniformSizeExt** = ((int)0x8DED),
UniformBufferExt = ((int)0x8DEE), **UniformBufferBindingExt** = ((int)0x8DEF) }
- enum **ExtBlendColor** {
ConstantColor = ((int)0x8001), **ConstantColorExt** = ((int)0x8001), **OneMinusConstantColor** = ((int)0x8002), **OneMinusConstantColorExt** = ((int)0x8002),
ConstantAlpha = ((int)0x8003), **ConstantAlphaExt** = ((int)0x8003), **OneMinusConstantAlpha** = ((int)0x8004), **OneMinusConstantAlphaExt** = ((int)0x8004),
BlendColor = ((int)0x8005), **BlendColorExt** = ((int)0x8005) }
- enum **ExtBlendEquationSeparate** { **BlendEquationRgbExt** = ((int)0x8009), **BlendEquationAlphaExt** = ((int)0x883D) }
- enum **ExtBlendFuncSeparate** { **BlendDstRgbExt** = ((int)0x80C8), **BlendSrcRgbExt** = ((int)0x80C9), **BlendDstAlphaExt** = ((int)0x80CA), **BlendSrcAlphaExt** = ((int)0x80CB) }
- enum **ExtBlendLogicOp**
- enum **ExtBlendMinmax** {
FuncAdd = ((int)0x8006), **FuncAddExt** = ((int)0x8006), **Min** = ((int)0x8007), **MinExt** = ((int)0x8007),
Max = ((int)0x8008), **MaxExt** = ((int)0x8008), **BlendEquation** = ((int)0x8009), **BlendEquationExt** = ((int)0x8009) }
- enum **ExtBlendSubtract** { **FuncSubtract** = ((int)0x800A), **FuncSubtractExt** = ((int)0x800A), **FuncReverseSubtract** = ((int)0x800B), **FuncReverseSubtractExt** = ((int)0x800B) }
- enum **ExtClipVolumeHint** { **ClipVolumeClippingHintExt** = ((int)0x80F0) }
- enum **ExtCmyka** { **CmykExt** = ((int)0x800C), **CmykaExt** = ((int)0x800D), **PackCmykHintExt** = ((int)0x800E), **UnpackCmykHintExt** = ((int)0x800F) }
- enum **ExtColorSubtable**
- enum **ExtCompiledVertexArray** { **ArrayElementLockFirstExt** = ((int)0x81A8), **ArrayElementLockCountExt** = ((int)0x81A9) }
- enum **ExtConvolution** {
Convolution1DExt = ((int)0x8010), **Convolution2DExt** = ((int)0x8011), **Separable2DExt** = ((int)0x8012), **ConvolutionBorderModeExt** = ((int)0x8013),
ConvolutionFilterScaleExt = ((int)0x8014), **ConvolutionFilterBiasExt** = ((int)0x8015), **ReduceExt** = ((int)0x8016), **ConvolutionFormatExt** = ((int)0x8017),

- ConvolutionWidthExt** = ((int)0x8018), **ConvolutionHeightExt** = ((int)0x8019), **MaxConvolutionWidthExt** = ((int)0x801A), **MaxConvolutionHeightExt** = ((int)0x801B),
- PostConvolutionRedScaleExt** = ((int)0x801C), **PostConvolutionGreenScaleExt** = ((int)0x801D), **PostConvolutionBlueScaleExt** = ((int)0x801E), **PostConvolutionAlphaScaleExt** = ((int)0x801F),
- PostConvolutionRedBiasExt** = ((int)0x8020), **PostConvolutionGreenBiasExt** = ((int)0x8021), **PostConvolutionBlueBiasExt** = ((int)0x8022), **PostConvolutionAlphaBiasExt** = ((int)0x8023) }
- enum **ExtCoordinateFrame** {
 - TangentArrayExt** = ((int)0x8439), **BinormalArrayExt** = ((int)0x843A), **CurrentTangentExt** = ((int)0x843B), **CurrentBinormalExt** = ((int)0x843C),
 - TangentArrayTypeExt** = ((int)0x843E), **TangentArrayStrideExt** = ((int)0x843F), **BinormalArrayTypeExt** = ((int)0x8440), **BinormalArrayStrideExt** = ((int)0x8441),
 - TangentArrayPointerExt** = ((int)0x8442), **BinormalArrayPointerExt** = ((int)0x8443), **Map1TangentExt** = ((int)0x8444), **Map2TangentExt** = ((int)0x8445),
 - Map1BinormalExt** = ((int)0x8446), **Map2BinormalExt** = ((int)0x8447) }
- enum **ExtCopyTexture**
- enum **ExtCullVertex** { **CullVertexExt** = ((int)0x81AA), **CullVertexEyePositionExt** = ((int)0x81AB), **CullVertexObjectPositionExt** = ((int)0x81AC) }
- enum **ExtDepthBoundsTest** { **DepthBoundsTestExt** = ((int)0x8890), **DepthBoundsExt** = ((int)0x8891) }
- enum **ExtDirectStateAccess** { **ProgramMatrixExt** = ((int)0x8E2D), **TransposeProgramMatrixExt** = ((int)0x8E2E), **ProgramMatrixStackDepthExt** = ((int)0x8E2F) }
- enum **ExtDrawBuffers2**
- enum **ExtDrawInstanced**
- enum **ExtDrawRangeElements** { **MaxElementsVerticesExt** = ((int)0x80E8), **MaxElementsIndicesExt** = ((int)0x80E9) }
- enum **ExtFogCoord** {
 - FogCoordinateSourceExt** = ((int)0x8450), **FogCoordinateExt** = ((int)0x8451), **FragmentDepthExt** = ((int)0x8452), **CurrentFogCoordinateExt** = ((int)0x8453),
 - FogCoordinateArrayTypeExt** = ((int)0x8454), **FogCoordinateArrayStrideExt** = ((int)0x8455), **FogCoordinateArrayPointerExt** = ((int)0x8456), **FogCoordinateArrayExt** = ((int)0x8457) }
- enum **ExtFramebufferBlit** { **DrawFramebufferBindingExt** = ((int)0x8CA6), **ReadFramebufferExt** = ((int)0x8CA8), **DrawFramebufferExt** = ((int)0x8CA9), **ReadFramebufferBindingExt** = ((int)0x8CAA) }

- enum **ExtFramebufferMultisample** { **RenderbufferSamplesExt** = ((int)0x8CAB), **FramebufferIncompleteMultisampleExt** = ((int)0x8D56), **MaxSamplesExt** = ((int)0x8D57) }
- enum **ExtFramebufferObject** {
 - InvalidFramebufferOperationExt** = ((int)0x0506), **MaxRenderbufferSizeExt** = ((int)0x84E8), **FramebufferBindingExt** = ((int)0x8CA6), **RenderbufferBindingExt** = ((int)0x8CA7),
 - FramebufferAttachmentObjectTypeExt** = ((int)0x8CD0), **FramebufferAttachmentObjectNameExt** = ((int)0x8CD1), **FramebufferAttachmentTextureLevelExt** = ((int)0x8CD2), **FramebufferAttachmentTextureCubeMapFaceExt** = ((int)0x8CD3),
 - FramebufferAttachmentTexture3DZoffsetExt** = ((int)0x8CD4), **FramebufferCompleteExt** = ((int)0x8CD5), **FramebufferIncompleteAttachmentExt** = ((int)0x8CD6), **FramebufferIncompleteMissingAttachmentExt** = ((int)0x8CD7),
 - FramebufferIncompleteDimensionsExt** = ((int)0x8CD9), **FramebufferIncompleteFormatsExt** = ((int)0x8CDA), **FramebufferIncompleteDrawBufferExt** = ((int)0x8CDB), **FramebufferIncompleteReadBufferExt** = ((int)0x8CDC),
 - FramebufferUnsupportedExt** = ((int)0x8CDD), **MaxColorAttachmentsExt** = ((int)0x8CDF), **ColorAttachment0Ext** = ((int)0x8CE0), **ColorAttachment1Ext** = ((int)0x8CE1),
 - ColorAttachment2Ext** = ((int)0x8CE2), **ColorAttachment3Ext** = ((int)0x8CE3), **ColorAttachment4Ext** = ((int)0x8CE4), **ColorAttachment5Ext** = ((int)0x8CE5),
 - ColorAttachment6Ext** = ((int)0x8CE6), **ColorAttachment7Ext** = ((int)0x8CE7), **ColorAttachment8Ext** = ((int)0x8CE8), **ColorAttachment9Ext** = ((int)0x8CE9),
 - ColorAttachment10Ext** = ((int)0x8CEA), **ColorAttachment11Ext** = ((int)0x8CEB), **ColorAttachment12Ext** = ((int)0x8CEC), **ColorAttachment13Ext** = ((int)0x8CED),
 - ColorAttachment14Ext** = ((int)0x8CEE), **ColorAttachment15Ext** = ((int)0x8CEF), **DepthAttachmentExt** = ((int)0x8D00), **StencilAttachmentExt** = ((int)0x8D20),
 - FramebufferExt** = ((int)0x8D40), **RenderbufferExt** = ((int)0x8D41), **RenderbufferWidthExt** = ((int)0x8D42), **RenderbufferHeightExt** = ((int)0x8D43),
 - RenderbufferInternalFormatExt** = ((int)0x8D44), **StencilIndex1Ext** = ((int)0x8D46), **StencilIndex4Ext** = ((int)0x8D47), **StencilIndex8Ext** = ((int)0x8D48),
 - StencilIndex16Ext** = ((int)0x8D49), **RenderbufferRedSizeExt** = ((int)0x8D50), **RenderbufferGreenSizeExt** = ((int)0x8D51), **RenderbufferBlueSizeExt** = ((int)0x8D52),

```

RenderbufferAlphaSizeExt = ((int)0x8D53), RenderbufferDepthSizeExt =
((int)0x8D54), RenderbufferStencilSizeExt = ((int)0x8D55) }

• enum ExtFramebufferSrgb { FramebufferSrgbExt = ((int)0x8DB9), Frame-
bufferSrgbCapableExt = ((int)0x8DBA) }

• enum ExtGeometryShader4 {

LinesAdjacencyExt = ((int)0x000A), LineStripAdjacencyExt =
((int)0x000B), TrianglesAdjacencyExt = ((int)0x000C), TriangleStri-
pAdjacencyExt = ((int)0x000D),

ProgramPointSizeExt = ((int)0x8642), MaxVaryingComponentsExt =
((int)0x8B4B), MaxGeometryTextureImageUnitsExt = ((int)0x8C29),
FramebufferAttachmentTextureLayerExt = ((int)0x8CD4),

FramebufferAttachmentLayeredExt = ((int)0x8DA7), FramebufferIncom-
pleteLayerTargetsExt = ((int)0x8DA8), FramebufferIncompleteLayerCoun-
tExt = ((int)0x8DA9), GeometryShaderExt = ((int)0x8DD9),

GeometryVerticesOutExt = ((int)0x8DDA), GeometryInputTypeExt =
((int)0x8ddb), GeometryOutputTypeExt = ((int)0x8DDC), MaxGeometry-
VaryingComponentsExt = ((int)0x8DDD),

MaxVertexVaryingComponentsExt = ((int)0x8DDE), MaxGeometryUni-
formComponentsExt = ((int)0x8DDF), MaxGeometryOutputVerticesExt =
((int)0x8DE0), MaxGeometryTotalOutputComponentsExt = ((int)0x8DE1) }

• enum ExtGpuProgramParameters

• enum ExtGpuShader4 {

Sampler1DArrayExt = ((int)0x8DC0), Sampler2DArrayExt = ((int)0x8DC1),
SamplerBufferExt = ((int)0x8DC2), Sampler1DArrayShadowExt =
((int)0x8DC3),

Sampler2DArrayShadowExt = ((int)0x8DC4), SamplerCubeShadowExt =
((int)0x8DC5), UnsignedIntVec2Ext = ((int)0x8DC6), UnsignedIntVec3Ext =
((int)0x8DC7),

UnsignedIntVec4Ext = ((int)0x8DC8), IntSampler1DExt = ((int)0x8DC9),
IntSampler2DExt = ((int)0x8DCA), IntSampler3DExt = ((int)0x8DCB),

IntSamplerCubeExt = ((int)0x8DCC), IntSampler2DRectExt =
((int)0x8DCD), IntSampler1DArrayExt = ((int)0x8DCE), IntSam-
pler2DArrayExt = ((int)0x8DCF),

IntSamplerBufferExt = ((int)0x8DD0), UnsignedIntSampler1DExt =
((int)0x8DD1), UnsignedIntSampler2DExt = ((int)0x8DD2), Unsigned-
IntSampler3DExt = ((int)0x8DD3),

UnsignedIntSamplerCubeExt = ((int)0x8DD4), UnsignedIntSam-
pler2DRectExt = ((int)0x8DD5), UnsignedIntSampler1DArrayExt =
((int)0x8DD6), UnsignedIntSampler2DArrayExt = ((int)0x8DD7),

UnsignedIntSamplerBufferExt = ((int)0x8DD8) }

```

- enum **ExtHistogram** {
 - HistogramExt** = ((int)0x8024), **ProxyHistogramExt** = ((int)0x8025), **HistogramWidthExt** = ((int)0x8026), **HistogramFormatExt** = ((int)0x8027),
 - HistogramRedSizeExt** = ((int)0x8028), **HistogramGreenSizeExt** = ((int)0x8029), **HistogramBlueSizeExt** = ((int)0x802A), **HistogramAlphaSizeExt** = ((int)0x802B),
 - HistogramLuminanceSize** = ((int)0x802C), **HistogramLuminanceSizeExt** = ((int)0x802C), **HistogramSinkExt** = ((int)0x802D), **MinmaxExt** = ((int)0x802E),
 - MinmaxFormatExt** = ((int)0x802F), **MinmaxSinkExt** = ((int)0x8030), **TableTooLargeExt** = ((int)0x8031) }
- enum **ExtIndexArrayFormats** {
 - IuiV2fExt** = ((int)0x81AD), **IuiV3fExt** = ((int)0x81AE), **IuiN3fV2fExt** = ((int)0x81AF), **IuiN3fV3fExt** = ((int)0x81B0),
 - T2fIuiV2fExt** = ((int)0x81B1), **T2fIuiV3fExt** = ((int)0x81B2), **T2fIuiN3fV2fExt** = ((int)0x81B3), **T2fIuiN3fV3fExt** = ((int)0x81B4)
- enum **ExtIndexFunc** { **IndexTestExt** = ((int)0x81B5), **IndexTestFuncExt** = ((int)0x81B6), **IndexTestRefExt** = ((int)0x81B7) }
- enum **ExtIndexMaterial** { **IndexMaterialExt** = ((int)0x81B8), **IndexMaterialParameterExt** = ((int)0x81B9), **IndexMaterialFaceExt** = ((int)0x81BA) }
- enum **ExtIndexTexture**
- enum **ExtLightTexture** {
 - FragmentMaterialExt** = ((int)0x8349), **FragmentNormalExt** = ((int)0x834A), **FragmentColorExt** = ((int)0x834C), **AttenuationExt** = ((int)0x834D),
 - ShadowAttenuationExt** = ((int)0x834E), **TextureApplicationModeExt** = ((int)0x834F), **TextureLightExt** = ((int)0x8350), **TextureMaterialFaceExt** = ((int)0x8351),
 - TextureMaterialParameterExt** = ((int)0x8352), **FragmentDepthExt** = ((int)0x8452) }
- enum **ExtMiscAttribute**
- enum **ExtMultiDrawArrays**
- enum **ExtMultisample** {
 - MultisampleBitExt** = ((int)0x20000000), **MultisampleExt** = ((int)0x809D), **SampleAlphaToMaskExt** = ((int)0x809E), **SampleAlphaToOneExt** = ((int)0x809F),
 - SampleMaskExt** = ((int)0x80A0), **GL1PassExt** = ((int)0x80A1), **GL2Pass0Ext** = ((int)0x80A2), **GL2Pass1Ext** = ((int)0x80A3),
 - GL4Pass0Ext** = ((int)0x80A4), **GL4Pass1Ext** = ((int)0x80A5), **GL4Pass2Ext** = ((int)0x80A6), **GL4Pass3Ext** = ((int)0x80A7),

```

SampleBuffersExt = ((int)0x80A8), SamplesExt = ((int)0x80A9), SampleMaskValueExt = ((int)0x80AA), SampleMaskInvertExt = ((int)0x80AB),
SamplePatternExt = ((int)0x80AC) }
• enum ExtPackedDepthStencil { DepthStencilExt = ((int)0x84F9), UnsignedInt248Ext = ((int)0x84FA), Depth24Stencil8Ext = ((int)0x88F0), TextureStencilSizeExt = ((int)0x88F1) }
• enum ExtPackedFloat { R11fG11fB10fExt = ((int)0x8C3A), UnsignedInt10F11F11FRevExt = ((int)0x8C3B), RgbaSignedComponentsExt = ((int)0x8C3C) }
• enum ExtPackedPixels {
    UnsignedByte332Ext = ((int)0x8032), UnsignedShort4444Ext = ((int)0x8033), UnsignedShort5551Ext = ((int)0x8034), UnsignedInt8888Ext = ((int)0x8035),
    UnsignedInt1010102Ext = ((int)0x8036), UnsignedByte233RevExt = ((int)0x8362), UnsignedShort565Ext = ((int)0x8363), UnsignedShort565RevExt = ((int)0x8364),
    UnsignedShort4444RevExt = ((int)0x8365), UnsignedShort1555RevExt = ((int)0x8366), UnsignedInt8888RevExt = ((int)0x8367), UnsignedInt2101010RevExt = ((int)0x8368) }
• enum ExtPalettedTexture {
    ColorIndex1Ext = ((int)0x80E2), ColorIndex2Ext = ((int)0x80E3), ColorIndex4Ext = ((int)0x80E4), ColorIndex8Ext = ((int)0x80E5),
    ColorIndex12Ext = ((int)0x80E6), ColorIndex16Ext = ((int)0x80E7), TextureIndexSizeExt = ((int)0x80ED) }
• enum ExtPixelBufferObject { PixelPackBufferExt = ((int)0x88EB), PixelUnpackBufferExt = ((int)0x88EC), PixelPackBufferBindingExt = ((int)0x88ED), PixelUnpackBufferBindingExt = ((int)0x88EF) }
• enum ExtPixelTransform {
    PixelTransform2DExt = ((int)0x8330), PixelMagFilterExt = ((int)0x8331), PixelMinFilterExt = ((int)0x8332), PixelCubicWeightExt = ((int)0x8333),
    CubicExt = ((int)0x8334), AverageExt = ((int)0x8335), PixelTransform2DStackDepthExt = ((int)0x8336), MaxPixelTransform2DStackDepthExt = ((int)0x8337),
    PixelTransform2DMatrixExt = ((int)0x8338) }
• enum ExtPixelTransformColorTable
• enum ExtPointParameters { PointSizeMinExt = ((int)0x8126), PointSizeMaxExt = ((int)0x8127), PointFadeThresholdSizeExt = ((int)0x8128), DistanceAttenuationExt = ((int)0x8129) }
• enum ExtPolygonOffset { PolygonOffsetExt = ((int)0x8037), PolygonOffsetFactorExt = ((int)0x8038), PolygonOffsetBiasExt = ((int)0x8039) }
• enum ExtProvokingVertex { QuadsFollowProvokingVertexConventionExt = ((int)0x8E4C), FirstVertexConventionExt = ((int)0x8E4D), LastVertexConventionExt = ((int)0x8E4E), ProvokingVertexExt = ((int)0x8E4F) }

```

- enum **ExtRescaleNormal** { **RescaleNormalExt** = ((int)0x803A) }
- enum **ExtSecondaryColor** {

 ColorSumExt = ((int)0x8458), **CurrentSecondaryColorExt** = ((int)0x8459),

 SecondaryColorArraySizeExt = ((int)0x845A), **SecondaryColorArrayTypeExt** = ((int)0x845B),

 SecondaryColorArrayStrideExt = ((int)0x845C), **SecondaryColorArrayPointerExt** = ((int)0x845D), **SecondaryColorArrayExt** = ((int)0x845E) }
- enum **ExtSeparateSpecularColor** { **LightModelColorControlExt** = ((int)0x81F8), **SingleColorExt** = ((int)0x81F9), **SeparateSpecularColorExt** = ((int)0x81FA) }
- enum **ExtShadowFuncs**
- enum **ExtSharedTexturePalette** { **SharedTexturePaletteExt** = ((int)0x81FB) }
- enum **ExtStencilClearTag** { **StencilTagBitsExt** = ((int)0x88F2), **StencilClearTagValueExt** = ((int)0x88F3) }
- enum **ExtStencilTwoSide** { **StencilTestTwoSideExt** = ((int)0x8910), **ActiveStencilFaceExt** = ((int)0x8911) }
- enum **ExtStencilWrap** { **IncrWrapExt** = ((int)0x8507), **DecrWrapExt** = ((int)0x8508) }
- enum **ExtSubtexture**
- enum **ExtTexture** {

 Alpha4Ext = ((int)0x803B), **Alpha8Ext** = ((int)0x803C), **Alpha12Ext** = ((int)0x803D), **Alpha16Ext** = ((int)0x803E),

 Luminance4Ext = ((int)0x803F), **Luminance8Ext** = ((int)0x8040), **Luminance12Ext** = ((int)0x8041), **Luminance16Ext** = ((int)0x8042),

 Luminance4Alpha4Ext = ((int)0x8043), **Luminance6Alpha2Ext** = ((int)0x8044), **Luminance8Alpha8Ext** = ((int)0x8045), **Luminance12Alpha4Ext** = ((int)0x8046),

 Luminance12Alpha12Ext = ((int)0x8047), **Luminance16Alpha16Ext** = ((int)0x8048), **IntensityExt** = ((int)0x8049), **Intensity4Ext** = ((int)0x804A),

 Intensity8Ext = ((int)0x804B), **Intensity12Ext** = ((int)0x804C), **Intensity16Ext** = ((int)0x804D), **Rgb2Ext** = ((int)0x804E),

 Rgb4Ext = ((int)0x804F), **Rgb5Ext** = ((int)0x8050), **Rgb8Ext** = ((int)0x8051), **Rgb10Ext** = ((int)0x8052),

 Rgb12Ext = ((int)0x8053), **Rgb16Ext** = ((int)0x8054), **Rgba2Ext** = ((int)0x8055), **Rgba4Ext** = ((int)0x8056),

 Rgb5A1Ext = ((int)0x8057), **Rgba8Ext** = ((int)0x8058), **Rgb10A2Ext** = ((int)0x8059), **Rgba12Ext** = ((int)0x805A),

 Rgba16Ext = ((int)0x805B), **TextureRedSizeExt** = ((int)0x805C), **TextureGreenSizeExt** = ((int)0x805D), **TextureBlueSizeExt** = ((int)0x805E),

```

TextureAlphaSizeExt = ((int)0x805F), TextureLuminanceSizeExt =
((int)0x8060), TextureIntensitySizeExt = ((int)0x8061), ReplaceExt =
((int)0x8062),

ProxyTexture1DExt = ((int)0x8063), ProxyTexture2DExt = ((int)0x8064),
TextureTooLargeExt = ((int)0x8065) }

• enum ExtTexture3D {

PackSkipImagesExt = ((int)0x806B), PackImageHeightExt = ((int)0x806C),
UnpackSkipImagesExt = ((int)0x806D), UnpackImageHeightExt =
((int)0x806E),

Texture3DExt = ((int)0x806F), ProxyTexture3DExt = ((int)0x8070), Texture-
DepthExt = ((int)0x8071), TextureWrapRExt = ((int)0x8072),

Max3DTextureSizeExt = ((int)0x8073) }

• enum ExtTextureArray {

CompareRefDepthToTextureExt = ((int)0x884E), MaxArrayTextureLay-
ersExt = ((int)0x88FF), Texture1DArrayExt = ((int)0x8C18), ProxyTex-
ture1DArrayExt = ((int)0x8C19),

Texture2DArrayExt = ((int)0x8C1A), ProxyTexture2DArrayExt =
((int)0x8C1B), TextureBinding1DArrayExt = ((int)0x8C1C), Texture-
Binding2DArrayExt = ((int)0x8C1D),

FramebufferAttachmentTextureLayerExt = ((int)0x8CD4) }

• enum ExtTextureBufferObject {

TextureBufferExt = ((int)0x8C2A), MaxTextureBufferSizeExt =
((int)0x8C2B), TextureBindingBufferExt = ((int)0x8C2C), TextureBuffer-
DataStoreBindingExt = ((int)0x8C2D),

TextureBufferFormatExt = ((int)0x8C2E) }

• enum ExtTextureCompressionLatc { CompressedLuminanceLatc1Ext =
((int)0x8C70), CompressedSignedLuminanceLatc1Ext = ((int)0x8C71),
CompressedLuminanceAlphaLatc2Ext = ((int)0x8C72), Compressed-
SignedLuminanceAlphaLatc2Ext = ((int)0x8C73) }

• enum ExtTextureCompressionRgtc { CompressedRedRgtc1Ext =
((int)0x8DBB), CompressedSignedRedRgtc1Ext = ((int)0x8DBC), Com-
pressedRedGreenRgtc2Ext = ((int)0x8DBD), CompressedSignedRedGreen-
Rgtc2Ext = ((int)0x8DBE) }

• enum ExtTextureCompressionS3tc { CompressedRgbS3tcDxt1Ext =
((int)0x83F0), CompressedRgbaS3tcDxt1Ext = ((int)0x83F1), Compresse-
dRgbaS3tcDxt3Ext = ((int)0x83F2), CompressedRgbaS3tcDxt5Ext =
((int)0x83F3) }

• enum ExtTextureCubeMap {

NormalMapExt = ((int)0x8511), ReflectionMapExt = ((int)0x8512), Texture-
CubeMapExt = ((int)0x8513), TextureBindingCubeMapExt = ((int)0x8514),

```

TextureCubeMapPositiveXExt = ((int)0x8515), **TextureCubeMapNegativeXExt** = ((int)0x8516), **TextureCubeMapPositiveYExt** = ((int)0x8517), **TextureCubeMapNegativeYExt** = ((int)0x8518),

TextureCubeMapPositiveZExt = ((int)0x8519), **TextureCubeMapNegativeZExt** = ((int)0x851A), **ProxyTextureCubeMapExt** = ((int)0x851B), **MaxCubeMapTextureSizeExt** = ((int)0x851C) }

- enum **ExtTextureEnvAdd**

- enum **ExtTextureEnvCombine** {

CombineExt = ((int)0x8570), **CombineRgbExt** = ((int)0x8571), **CombineAlphaExt** = ((int)0x8572), **RgbScaleExt** = ((int)0x8573),

AddSignedExt = ((int)0x8574), **InterpolateExt** = ((int)0x8575), **ConstantExt** = ((int)0x8576), **PrimaryColorExt** = ((int)0x8577),

PreviousExt = ((int)0x8578), **Source0RgbExt** = ((int)0x8580), **Source1RgbExt** = ((int)0x8581), **Source2RgbExt** = ((int)0x8582),

Source0AlphaExt = ((int)0x8588), **Source1AlphaExt** = ((int)0x8589), **Source2AlphaExt** = ((int)0x858A), **Operand0RgbExt** = ((int)0x8590),

Operand1RgbExt = ((int)0x8591), **Operand2RgbExt** = ((int)0x8592), **Operand0AlphaExt** = ((int)0x8598), **Operand1AlphaExt** = ((int)0x8599),

Operand2AlphaExt = ((int)0x859A) }

- enum **ExtTextureEnvDot3** { **Dot3RgbExt** = ((int)0x8740), **Dot3RgbaExt** = ((int)0x8741) }

- enum **ExtTextureFilterAnisotropic** { **TextureMaxAnisotropyExt** = ((int)0x84FE), **MaxTextureMaxAnisotropyExt** = ((int)0x84FF) }

- enum **ExtTextureInteger** {

Rgba32uiExt = ((int)0x8D70), **Rgb32uiExt** = ((int)0x8D71), **Alpha32uiExt** = ((int)0x8D72), **Intensity32uiExt** = ((int)0x8D73),

Luminance32uiExt = ((int)0x8D74), **LuminanceAlpha32uiExt** = ((int)0x8D75), **Rgba16uiExt** = ((int)0x8D76), **Rgb16uiExt** = ((int)0x8D77),

Alpha16uiExt = ((int)0x8D78), **Intensity16uiExt** = ((int)0x8D79), **Luminance16uiExt** = ((int)0x8D7A), **LuminanceAlpha16uiExt** = ((int)0x8D7B),

Rgba8uiExt = ((int)0x8D7C), **Rgb8uiExt** = ((int)0x8D7D), **Alpha8uiExt** = ((int)0x8D7E), **Intensity8uiExt** = ((int)0x8D7F),

Luminance8uiExt = ((int)0x8D80), **LuminanceAlpha8uiExt** = ((int)0x8D81), **Rgba32iExt** = ((int)0x8D82), **Rgb32iExt** = ((int)0x8D83),

Alpha32iExt = ((int)0x8D84), **Intensity32iExt** = ((int)0x8D85), **Luminance32iExt** = ((int)0x8D86), **LuminanceAlpha32iExt** = ((int)0x8D87),

Rgba16iExt = ((int)0x8D88), **Rgb16iExt** = ((int)0x8D89), **Alpha16iExt** = ((int)0x8D8A), **Intensity16iExt** = ((int)0x8D8B),

Luminance16iExt = ((int)0x8D8C), **LuminanceAlpha16iExt** = ((int)0x8D8D), **Rgba8iExt** = ((int)0x8D8E), **Rgb8iExt** = ((int)0x8D8F),


```

Alpha8iExt = ((int)0x8D90), Intensity8iExt = ((int)0x8D91), Lumi-
nance8iExt = ((int)0x8D92), LuminanceAlpha8iExt = ((int)0x8D93),

RedIntegerExt = ((int)0x8D94), GreenIntegerExt = ((int)0x8D95), BlueInte-
gerExt = ((int)0x8D96), AlphaIntegerExt = ((int)0x8D97),

RgbIntegerExt = ((int)0x8D98), RgbaIntegerExt = ((int)0x8D99), BgrInte-
gerExt = ((int)0x8D9A), BgraIntegerExt = ((int)0x8D9B),

LuminanceIntegerExt = ((int)0x8D9C), LuminanceAlphaIntegerExt =
((int)0x8D9D), RgbaIntegerModeExt = ((int)0x8D9E) }

• enum ExtTextureLodBias { MaxTextureLodBiasExt = ((int)0x84FD), Tex-
tureFilterControlExt = ((int)0x8500), TextureLodBiasExt = ((int)0x8501) }

• enum ExtTextureMirrorClamp { MirrorClampExt = ((int)0x8742),
MirrorClampToEdgeExt = ((int)0x8743), MirrorClampToBorderExt =
((int)0x8912) }

• enum ExtTextureObject {
TexturePriorityExt = ((int)0x8066), TextureResidentExt = ((int)0x8067),
Texture1DBindingExt = ((int)0x8068), Texture2DBindingExt =
((int)0x8069),
Texture3DBindingExt = ((int)0x806A) }

• enum ExtTexturePerturbNormal { PerturbExt = ((int)0x85AE), TextureNor-
malExt = ((int)0x85AF) }

• enum ExtTextureSharedExponent { Rgb9E5Ext = ((int)0x8C3D),
UnsignedInt5999RevExt = ((int)0x8C3E), TextureSharedSizeExt =
((int)0x8C3F) }

• enum ExtTextureSnorm {
RgSnorm = ((int)0x8F91), RgbSnorm = ((int)0x8F92), RgbaSnorm =
((int)0x8F93), R8Snorm = ((int)0x8F94),
Rg8Snorm = ((int)0x8F95), Rgb8Snorm = ((int)0x8F96), Rgba8Snorm =
((int)0x8F97), R16Snorm = ((int)0x8F98),
Rg16Snorm = ((int)0x8F99), Rgb16Snorm = ((int)0x8F9A), Rgba16Snorm =
((int)0x8F9B), SignedNormalized = ((int)0x8F9C),
AlphaSnorm = ((int)0x9010), LuminanceSnorm = ((int)0x9011), Lumi-
nanceAlphaSnorm = ((int)0x9012), IntensitySnorm = ((int)0x9013),
Alpha8Snorm = ((int)0x9014), Luminance8Snorm = ((int)0x9015), Lumi-
nance8Alpha8Snorm = ((int)0x9016), Intensity8Snorm = ((int)0x9017),
Alpha16Snorm = ((int)0x9018), Luminance16Snorm = ((int)0x9019), Lumi-
nance16Alpha16Snorm = ((int)0x901A), Intensity16Snorm = ((int)0x901B)
}

• enum ExtTextureSrgb {
SrgbExt = ((int)0x8C40), Srgb8Ext = ((int)0x8C41), SrgbAlphaExt =
((int)0x8C42), Srgb8Alpha8Ext = ((int)0x8C43),

```

```

SluminanceAlphaExt = ((int)0x8C44), Sluminance8Alpha8Ext =
((int)0x8C45), SluminanceExt = ((int)0x8C46), Sluminance8Ext =
((int)0x8C47),

CompressedSrgbExt = ((int)0x8C48), CompressedSrgbAlphaExt =
((int)0x8C49), CompressedSluminanceExt = ((int)0x8C4A), CompressedS-
luminanceAlphaExt = ((int)0x8C4B),

CompressedSrgbS3tcDxt1Ext = ((int)0x8C4C), CompressedSrgbAl-
phaS3tcDxt1Ext = ((int)0x8C4D), CompressedSrgbAlphaS3tcDxt3Ext =
((int)0x8C4E), CompressedSrgbAlphaS3tcDxt5Ext = ((int)0x8C4F) }
• enum ExtTextureSwizzle {
TextureSwizzleRExt = ((int)0x8E42), TextureSwizzleGExt = ((int)0x8E43),
TextureSwizzleBExt = ((int)0x8E44), TextureSwizzleAExt = ((int)0x8E45),

TextureSwizzleRgbaExt = ((int)0x8E46) }
• enum ExtTimerQuery { TimeElapsedExt = ((int)0x88BF) }
• enum ExtTransformFeedback {

TransformFeedbackVaryingMaxLengthExt = ((int)0x8C76), Transform-
FeedbackBufferModeExt = ((int)0x8C7F), MaxTransformFeedbackSepa-
rateComponentsExt = ((int)0x8C80), TransformFeedbackVaryingsExt =
((int)0x8C83),

TransformFeedbackBufferStartExt = ((int)0x8C84), TransformFeedback-
BufferSizeExt = ((int)0x8C85), PrimitivesGeneratedExt = ((int)0x8C87),
TransformFeedbackPrimitivesWrittenExt = ((int)0x8C88),

RasterizerDiscardExt = ((int)0x8C89), MaxTransformFeedbackInter-
leavedComponentsExt = ((int)0x8C8A), MaxTransformFeedbackSepa-
rateAttribsExt = ((int)0x8C8B), InterleavedAttribsExt = ((int)0x8C8C),

SeparateAttribsExt = ((int)0x8C8D), TransformFeedbackBufferExt =
((int)0x8C8E), TransformFeedbackBufferBindingExt = ((int)0x8C8F) }
• enum ExtVertexArray {

VertexArrayExt = ((int)0x8074), NormalArrayExt = ((int)0x8075), ColorAr-
rayExt = ((int)0x8076), IndexArrayExt = ((int)0x8077),

TextureCoordArrayExt = ((int)0x8078), EdgeFlagArrayExt = ((int)0x8079),
VertexArraySizeExt = ((int)0x807A), VertexArrayTypeExt = ((int)0x807B),

VertexArrayStrideExt = ((int)0x807C), VertexArrayCountExt =
((int)0x807D), NormalArrayTypeExt = ((int)0x807E), NormalArrayStride-
Ext = ((int)0x807F),

NormalArrayCountExt = ((int)0x8080), ColorArraySizeExt = ((int)0x8081),
ColorArrayTypeExt = ((int)0x8082), ColorArrayStrideExt = ((int)0x8083),

ColorArrayCountExt = ((int)0x8084), IndexArrayTypeExt = ((int)0x8085),
IndexArrayStrideExt = ((int)0x8086), IndexArrayCountExt = ((int)0x8087),

TextureCoordArraySizeExt = ((int)0x8088), TextureCoordArrayTypeExt =
((int)0x8089), TextureCoordArrayStrideExt = ((int)0x808A), TextureCoord-
ArrayCountExt = ((int)0x808B),

```

```

EdgeFlagArrayStrideExt = ((int)0x808C), EdgeFlagArrayCountExt =
((int)0x808D), VertexArrayPointerExt = ((int)0x808E), NormalArrayPointerExt = ((int)0x808F),

ColorArrayPointerExt = ((int)0x8090), IndexArrayPointerExt =
((int)0x8091), TextureCoordArrayPointerExt = ((int)0x8092), EdgeFlagArrayPointerExt = ((int)0x8093) }
• enum ExtVertexArrayBgra { Bgra = ((int)0x80E1) }
• enum ExtVertexShader {
VertexShaderExt = ((int)0x8780), VertexShaderBindingExt = ((int)0x8781),
OpIndexExt = ((int)0x8782), OpNegateExt = ((int)0x8783),

OpDot3Ext = ((int)0x8784), OpDot4Ext = ((int)0x8785), OpMulExt =
((int)0x8786), OpAddExt = ((int)0x8787),

OpMaddExt = ((int)0x8788), OpFracExt = ((int)0x8789), OpMaxExt =
((int)0x878A), OpMinExt = ((int)0x878B),

OpSetGeExt = ((int)0x878C), OpSetLtExt = ((int)0x878D), OpClampExt =
((int)0x878E), OpFloorExt = ((int)0x878F),

OpRoundExt = ((int)0x8790), OpExpBase2Ext = ((int)0x8791), OpLogBase2Ext = ((int)0x8792), OpPowerExt = ((int)0x8793),

OpRecipExt = ((int)0x8794), OpRecipSqrtExt = ((int)0x8795), OpSubExt =
((int)0x8796), OpCrossProductExt = ((int)0x8797),

OpMultiplyMatrixExt = ((int)0x8798), OpMovExt = ((int)0x8799), OutputVertexExt = ((int)0x879A), OutputColor0Ext = ((int)0x879B),

OutputColor1Ext = ((int)0x879C), OutputTextureCoord0Ext =
((int)0x879D), OutputTextureCoord1Ext = ((int)0x879E), OutputTextureCoord2Ext = ((int)0x879F),

OutputTextureCoord3Ext = ((int)0x87A0), OutputTextureCoord4Ext =
((int)0x87A1), OutputTextureCoord5Ext = ((int)0x87A2), OutputTextureCoord6Ext = ((int)0x87A3),

OutputTextureCoord7Ext = ((int)0x87A4), OutputTextureCoord8Ext =
((int)0x87A5), OutputTextureCoord9Ext = ((int)0x87A6), OutputTextureCoord10Ext = ((int)0x87A7),

OutputTextureCoord11Ext = ((int)0x87A8), OutputTextureCoord12Ext =
((int)0x87A9), OutputTextureCoord13Ext = ((int)0x87AA), OutputTextureCoord14Ext = ((int)0x87AB),

OutputTextureCoord15Ext = ((int)0x87AC), OutputTextureCoord16Ext =
((int)0x87AD), OutputTextureCoord17Ext = ((int)0x87AE), OutputTextureCoord18Ext = ((int)0x87AF),

OutputTextureCoord19Ext = ((int)0x87B0), OutputTextureCoord20Ext =
((int)0x87B1), OutputTextureCoord21Ext = ((int)0x87B2), OutputTextureCoord22Ext = ((int)0x87B3),

```

OutputTextureCoord23Ext = ((int)0x87B4), **OutputTextureCoord24Ext** = ((int)0x87B5), **OutputTextureCoord25Ext** = ((int)0x87B6), **OutputTextureCoord26Ext** = ((int)0x87B7),

OutputTextureCoord27Ext = ((int)0x87B8), **OutputTextureCoord28Ext** = ((int)0x87B9), **OutputTextureCoord29Ext** = ((int)0x87BA), **OutputTextureCoord30Ext** = ((int)0x87BB),

OutputTextureCoord31Ext = ((int)0x87BC), **OutputFogExt** = ((int)0x87BD), **ScalarExt** = ((int)0x87BE), **VectorExt** = ((int)0x87BF),

MatrixExt = ((int)0x87C0), **VariantExt** = ((int)0x87C1), **InvariantExt** = ((int)0x87C2), **LocalConstantExt** = ((int)0x87C3),

LocalExt = ((int)0x87C4), **MaxVertexShaderInstructionsExt** = ((int)0x87C5), **MaxVertexShaderVariantsExt** = ((int)0x87C6), **MaxVertexShaderInvariantsExt** = ((int)0x87C7),

MaxVertexShaderLocalConstantsExt = ((int)0x87C8), **MaxVertexShaderLocalsExt** = ((int)0x87C9), **MaxOptimizedVertexShaderInstructionsExt** = ((int)0x87CA), **MaxOptimizedVertexShaderVariantsExt** = ((int)0x87CB),

MaxOptimizedVertexShaderLocalConstantsExt = ((int)0x87CC), **MaxOptimizedVertexShaderInvariantsExt** = ((int)0x87CD), **MaxOptimizedVertexShaderLocalsExt** = ((int)0x87CE), **VertexShaderInstructionsExt** = ((int)0x87CF),

VertexShaderVariantsExt = ((int)0x87D0), **VertexShaderInvariantsExt** = ((int)0x87D1), **VertexShaderLocalConstantsExt** = ((int)0x87D2), **VertexShaderLocalsExt** = ((int)0x87D3),

VertexShaderOptimizedExt = ((int)0x87D4), **XExt** = ((int)0x87D5), **YExt** = ((int)0x87D6), **ZExt** = ((int)0x87D7),

WExt = ((int)0x87D8), **NegativeXExt** = ((int)0x87D9), **NegativeYExt** = ((int)0x87DA), **NegativeZExt** = ((int)0x87DB),

NegativeWExt = ((int)0x87DC), **ZeroExt** = ((int)0x87DD), **OneExt** = ((int)0x87DE), **NegativeOneExt** = ((int)0x87DF),

NormalizedRangeExt = ((int)0x87E0), **FullRangeExt** = ((int)0x87E1), **CurrentVertexExt** = ((int)0x87E2), **MvpMatrixExt** = ((int)0x87E3),

VariantValueExt = ((int)0x87E4), **VariantDatatypeExt** = ((int)0x87E5), **VariantArrayStrideExt** = ((int)0x87E6), **VariantArrayTypeExt** = ((int)0x87E7),

VariantArrayExt = ((int)0x87E8), **VariantArrayPointerExt** = ((int)0x87E9), **InvariantValueExt** = ((int)0x87EA), **InvariantDatatypeExt** = ((int)0x87EB),

LocalConstantValueExt = ((int)0x87EC), **LocalConstantDatatypeExt** = ((int)0x87ED) }

• enum **ExtVertexWeighting** {

Modelview0StackDepthExt = ((int)0x0BA3), **Modelview0MatrixExt** = ((int)0x0BA6), **Modelview0Ext** = ((int)0x1700), **Modelview1StackDepthExt** = ((int)0x8502),

```

Modelview1MatrixExt = ((int)0x8506), VertexWeightingExt = ((int)0x8509),
Modelview1Ext = ((int)0x850A), CurrentVertexWeightExt = ((int)0x850B),

VertexWeightArrayExt = ((int)0x850C), VertexWeightArraySizeExt
= ((int)0x850D), VertexWeightArrayTypeExt = ((int)0x850E), VertexWeightArrayStrideExt = ((int)0x850F),

VertexWeightArrayPointerExt = ((int)0x8510) }

• enum FeedBackToken {

    PassThroughToken = ((int)0x0700), PointToken = ((int)0x0701), LineToken
= ((int)0x0702), PolygonToken = ((int)0x0703),

    BitmapToken = ((int)0x0704), DrawPixelToken = ((int)0x0705), CopyPixelToken
= ((int)0x0706), LineResetToken = ((int)0x0707) }

• enum FeedbackType {

    GL2D = ((int)0x0600), GL3D = ((int)0x0601), GL3DColor = ((int)0x0602),
GL3DColorTexture = ((int)0x0603),

    GL4DColorTexture = ((int)0x0604) }

• enum FfdMaskSgix { TextureDeformationBitSgix = ((int)0x00000001), GeometryDeformationBitSgix = ((int)0x00000002) }

• enum FfdTargetSgix { GeometryDeformationSgix = ((int)0x8194), TextureDeformationSgix = ((int)0x8195) }

• enum FogMode {

    Exp = ((int)0x0800), Exp2 = ((int)0x0801), Linear = ((int)0x2601), FogFuncSgis = ((int)0x812A),

    FogCoord = ((int)0x8451), FragmentDepth = ((int)0x8452) }

• enum FogParameter {

    FogIndex = ((int)0x0B61), FogDensity = ((int)0x0B62), FogStart = ((int)0x0B63), FogEnd = ((int)0x0B64),

    FogMode = ((int)0x0B65), FogColor = ((int)0x0B66), FogOffsetValueSgix = ((int)0x8199), FogCoordSrc = ((int)0x8450) }

• enum FogPointerType { Float = ((int)0x1406), Double = ((int)0x140A), HalfFloat = ((int)0x140B) }

• enum FragmentLightModelParameterSgix { FragmentLightModelLocalViewerSgix = ((int)0x8408), FragmentLightModelTwoSideSgix = ((int)0x8409), FragmentLightModelAmbientSgix = ((int)0x840A), FragmentLightModelNormalInterpolationSgix = ((int)0x840B) }

• enum FramebufferAttachment {

    DepthStencilAttachment = ((int)0x821A), ColorAttachment0 = ((int)0x8CE0), ColorAttachment0Ext = ((int)0x8CE0), ColorAttachment1 = ((int)0x8CE1),

    ColorAttachment1Ext = ((int)0x8CE1), ColorAttachment2 = ((int)0x8CE2), ColorAttachment2Ext = ((int)0x8CE2), ColorAttachment3 = ((int)0x8CE3),

```

```

ColorAttachment3Ext = ((int)0x8CE3), ColorAttachment4 = ((int)0x8CE4),
ColorAttachment4Ext = ((int)0x8CE4), ColorAttachment5 = ((int)0x8CE5),

ColorAttachment5Ext = ((int)0x8CE5), ColorAttachment6 = ((int)0x8CE6),
ColorAttachment6Ext = ((int)0x8CE6), ColorAttachment7 = ((int)0x8CE7),

ColorAttachment7Ext = ((int)0x8CE7), ColorAttachment8 = ((int)0x8CE8),
ColorAttachment8Ext = ((int)0x8CE8), ColorAttachment9 = ((int)0x8CE9),

ColorAttachment9Ext = ((int)0x8CE9), ColorAttachment10 =
((int)0x8CEA), ColorAttachment10Ext = ((int)0x8CEA), ColorAttach-
ment11 = ((int)0x8CEB),

ColorAttachment11Ext = ((int)0x8CEB), ColorAttachment12 =
((int)0x8CEC), ColorAttachment12Ext = ((int)0x8CEC), ColorAttach-
ment13 = ((int)0x8CED),

ColorAttachment13Ext = ((int)0x8CED), ColorAttachment14 =
((int)0x8CEE), ColorAttachment14Ext = ((int)0x8CEE), ColorAttach-
ment15 = ((int)0x8CEF),

ColorAttachment15Ext = ((int)0x8CEF), DepthAttachment = ((int)0x8D00),
DepthAttachmentExt = ((int)0x8D00), StencilAttachment = ((int)0x8D20),

StencilAttachmentExt = ((int)0x8D20) }

• enum FramebufferAttachmentComponentType { Int = ((int)0x1404), Float
= ((int)0x1406), UnsignedNormalized = ((int)0x8C17) }

• enum FramebufferAttachmentObjectType { None = ((int)0), Texture
= ((int)0x1702), FramebufferDefault = ((int)0x8218), Renderbuffer =
((int)0x8D41) }

• enum FramebufferErrorCode {

FramebufferUndefined = ((int)0x8219), FramebufferComplete =
((int)0x8CD5), FramebufferCompleteExt = ((int)0x8CD5), FramebufferIn-
completeAttachment = ((int)0x8CD6),

FramebufferIncompleteAttachmentExt = ((int)0x8CD6), FramebufferIn-
completeMissingAttachment = ((int)0x8CD7), FramebufferIncomplete-
MissingAttachmentExt = ((int)0x8CD7), FramebufferIncompleteDimen-
sionsExt = ((int)0x8CD9),

FramebufferIncompleteFormatsExt = ((int)0x8CDA), FramebufferIncom-
pleteDrawBuffer = ((int)0x8CDB), FramebufferIncompleteDrawBufferExt
= ((int)0x8CDB), FramebufferIncompleteReadBuffer = ((int)0x8CDC),

FramebufferIncompleteReadBufferExt = ((int)0x8CDC), FramebufferUn-
supported = ((int)0x8CDD), FramebufferUnsupportedExt = ((int)0x8CDD),
FramebufferIncompleteMultisample = ((int)0x8D56),

FramebufferIncompleteLayerTargets = ((int)0x8DA8), FramebufferIncom-
pleteLayerCount = ((int)0x8DA9) }

• enum FramebufferParameterName {

```

```

FramebufferAttachmentColorEncoding = ((int)0x8210), FramebufferAttachmentComponentType = ((int)0x8211), FramebufferAttachmentRedSize = ((int)0x8212), FramebufferAttachmentGreenSize = ((int)0x8213),

FramebufferAttachmentBlueSize = ((int)0x8214), FramebufferAttachmentAlphaSize = ((int)0x8215), FramebufferAttachmentDepthSize = ((int)0x8216), FramebufferAttachmentStencilSize = ((int)0x8217),

FramebufferAttachmentObjectType = ((int)0x8CD0), FramebufferAttachmentObjectTypeExt = ((int)0x8CD0), FramebufferAttachmentObjectName = ((int)0x8CD1), FramebufferAttachmentObjectNameExt = ((int)0x8CD1),

FramebufferAttachmentTextureLevel = ((int)0x8CD2), FramebufferAttachmentTextureLevelExt = ((int)0x8CD2), FramebufferAttachmentTextureCubeMapFace = ((int)0x8CD3), FramebufferAttachmentTextureCubeMapFaceExt = ((int)0x8CD3),

FramebufferAttachmentTexture3DZoffsetExt = ((int)0x8CD4), FramebufferAttachmentTextureLayer = ((int)0x8CD4), FramebufferAttachmentLayered = ((int)0x8DA7) }

• enum FramebufferTarget { ReadFramebuffer = ((int)0x8CA8), DrawFramebuffer = ((int)0x8CA9), Framebuffer = ((int)0x8D40), FramebufferExt = ((int)0x8D40) }

• enum FrontFaceDirection { Cw = ((int)0x0900), Ccw = ((int)0x0901) }

• enum GenerateMipmapTarget {

    Texture1D = ((int)0x0DE0), Texture2D = ((int)0x0DE1), Texture3D = ((int)0x806F), TextureCubeMap = ((int)0x8513),

    Texture1DArray = ((int)0x8C18), Texture2DArray = ((int)0x8C1A), Texture2DMultisample = ((int)0x9100), Texture2DMultisampleArray = ((int)0x9102) }

• enum GetColorTableParameterPName {

    ColorTableScale = ((int)0x80D6), ColorTableBias = ((int)0x80D7), ColorTableFormat = ((int)0x80D8), ColorTableWidth = ((int)0x80D9),

    ColorTableRedSize = ((int)0x80DA), ColorTableGreenSize = ((int)0x80DB), ColorTableBlueSize = ((int)0x80DC), ColorTableAlphaSize = ((int)0x80DD),

    ColorTableLuminanceSize = ((int)0x80DE), ColorTableIntensitySize = ((int)0x80DF) }

• enum GetColorTableParameterPNameSgi {

    ColorTableScaleSgi = ((int)0x80D6), ColorTableBiasSgi = ((int)0x80D7), ColorTableFormatSgi = ((int)0x80D8), ColorTableWidthSgi = ((int)0x80D9),

    ColorTableRedSizeSgi = ((int)0x80DA), ColorTableGreenSizeSgi = ((int)0x80DB), ColorTableBlueSizeSgi = ((int)0x80DC), ColorTableAlphaSizeSgi = ((int)0x80DD),

    ColorTableLuminanceSizeSgi = ((int)0x80DE), ColorTableIntensitySizeSgi = ((int)0x80DF) }

```

- enum **GetConvolutionParameter** {
 - ConvolutionBorderModeExt** = ((int)0x8013), **ConvolutionFilterScaleExt** = ((int)0x8014), **ConvolutionFilterBiasExt** = ((int)0x8015), **ConvolutionFormatExt** = ((int)0x8017),
 - ConvolutionWidthExt** = ((int)0x8018), **ConvolutionHeightExt** = ((int)0x8019), **MaxConvolutionWidthExt** = ((int)0x801A), **MaxConvolutionHeightExt** = ((int)0x801B) }
- enum **GetConvolutionParameterPName** {
 - ConvolutionBorderMode** = ((int)0x8013), **ConvolutionFilterScale** = ((int)0x8014), **ConvolutionFilterBias** = ((int)0x8015), **ConvolutionFormat** = ((int)0x8017),
 - ConvolutionWidth** = ((int)0x8018), **ConvolutionHeight** = ((int)0x8019), **MaxConvolutionWidth** = ((int)0x801A), **MaxConvolutionHeight** = ((int)0x801B),
 - ConvolutionBorderColor** = ((int)0x8154) }
- enum **GetHistogramParameterPName** {
 - HistogramWidth** = ((int)0x8026), **HistogramFormat** = ((int)0x8027), **HistogramRedSize** = ((int)0x8028), **HistogramGreenSize** = ((int)0x8029),
 - HistogramBlueSize** = ((int)0x802A), **HistogramAlphaSize** = ((int)0x802B), **HistogramLuminanceSize** = ((int)0x802C), **HistogramSink** = ((int)0x802D) }
- enum **GetHistogramParameterPNameExt** {
 - HistogramWidthExt** = ((int)0x8026), **HistogramFormatExt** = ((int)0x8027), **HistogramRedSizeExt** = ((int)0x8028), **HistogramGreenSizeExt** = ((int)0x8029),
 - HistogramBlueSizeExt** = ((int)0x802A), **HistogramAlphaSizeExt** = ((int)0x802B), **HistogramLuminanceSizeExt** = ((int)0x802C), **HistogramSinkExt** = ((int)0x802D) }
- enum **GetIndexedPName** {
 - UniformBufferBinding** = ((int)0x8A28), **UniformBufferStart** = ((int)0x8A29), **UniformBufferSize** = ((int)0x8A2A), **TransformFeedbackBufferStart** = ((int)0x8C84),
 - TransformFeedbackBufferSize** = ((int)0x8C85), **TransformFeedbackBufferBinding** = ((int)0x8C8F), **SampleMaskValue** = ((int)0x8E52) }
- enum **GetMapQuery** { **Coeff** = ((int)0x0A00), **Order** = ((int)0x0A01), **Domain** = ((int)0x0A02) }
- enum **GetMinmaxParameterPName** { **MinmaxFormat** = ((int)0x802F), **MinmaxSink** = ((int)0x8030) }
- enum **GetMinmaxParameterPNameExt** { **MinmaxFormatExt** = ((int)0x802F), **MinmaxSinkExt** = ((int)0x8030) }
- enum **GetMultisamplePName** { **SamplePosition** = ((int)0x8E50) }

- enum **GetPixelFormat** {
 - PixelFormatToI** = ((int)0x0C70), **PixelFormatSToS** = ((int)0x0C71), **PixelFormatToR** = ((int)0x0C72), **PixelFormatToG** = ((int)0x0C73),
 - PixelFormatToB** = ((int)0x0C74), **PixelFormatToA** = ((int)0x0C75), **PixelFormatRToR** = ((int)0x0C76), **PixelFormatGToG** = ((int)0x0C77),
 - PixelFormatBToB** = ((int)0x0C78), **PixelFormatAToA** = ((int)0x0C79) }
- enum **GetPName** {
 - CurrentColor** = ((int)0x0B00), **CurrentIndex** = ((int)0x0B01), **CurrentNormal** = ((int)0x0B02), **CurrentTextureCoords** = ((int)0x0B03),
 - CurrentRasterColor** = ((int)0x0B04), **CurrentRasterIndex** = ((int)0x0B05), **CurrentRasterTextureCoords** = ((int)0x0B06), **CurrentRasterPosition** = ((int)0x0B07),
 - CurrentRasterPositionValid** = ((int)0x0B08), **CurrentRasterDistance** = ((int)0x0B09), **PointSize** = ((int)0x0B10), **PointSize** = ((int)0x0B11),
 - PointSizeRange** = ((int)0x0B12), **SmoothPointSizeRange** = ((int)0x0B12), **PointSizeGranularity** = ((int)0x0B13), **SmoothPointSizeGranularity** = ((int)0x0B13),
 - LineSmooth** = ((int)0x0B20), **LineWidth** = ((int)0x0B21), **LineWidthRange** = ((int)0x0B22), **SmoothLineWidthRange** = ((int)0x0B22),
 - LineWidthGranularity** = ((int)0x0B23), **SmoothLineWidthGranularity** = ((int)0x0B23), **LineStipple** = ((int)0x0B24), **LineStipplePattern** = ((int)0x0B25),
 - LineStippleRepeat** = ((int)0x0B26), **ListMode** = ((int)0x0B30), **MaxListNesting** = ((int)0x0B31), **ListBase** = ((int)0x0B32),
 - ListIndex** = ((int)0x0B33), **PolygonMode** = ((int)0x0B40), **PolygonSmooth** = ((int)0x0B41), **PolygonStipple** = ((int)0x0B42),
 - EdgeFlag** = ((int)0x0B43), **CullFace** = ((int)0x0B44), **CullFaceMode** = ((int)0x0B45), **FrontFace** = ((int)0x0B46),
 - Lighting** = ((int)0x0B50), **LightModelLocalViewer** = ((int)0x0B51), **LightModelTwoSide** = ((int)0x0B52), **LightModelAmbient** = ((int)0x0B53),
 - ShadeModel** = ((int)0x0B54), **ColorMaterialFace** = ((int)0x0B55), **ColorMaterialParameter** = ((int)0x0B56), **ColorMaterial** = ((int)0x0B57),
 - Fog** = ((int)0x0B60), **FogIndex** = ((int)0x0B61), **FogDensity** = ((int)0x0B62), **FogStart** = ((int)0x0B63),
 - FogEnd** = ((int)0x0B64), **FogMode** = ((int)0x0B65), **FogColor** = ((int)0x0B66), **DepthRange** = ((int)0x0B70),
 - DepthTest** = ((int)0x0B71), **DepthWritemask** = ((int)0x0B72), **DepthClearValue** = ((int)0x0B73), **DepthFunc** = ((int)0x0B74),
 - AccumClearValue** = ((int)0x0B80), **StencilTest** = ((int)0x0B90), **StencilClearValue** = ((int)0x0B91), **StencilFunc** = ((int)0x0B92),

StencilValueMask = ((int)0x0B93), **StencilFail** = ((int)0x0B94), **StencilPass-DepthFail** = ((int)0x0B95), **StencilPassDepthPass** = ((int)0x0B96),
StencilRef = ((int)0x0B97), **StencilWritemask** = ((int)0x0B98), **MatrixMode** = ((int)0x0BA0), **Normalize** = ((int)0x0BA1),
Viewport = ((int)0x0BA2), **ModelviewStackDepth** = ((int)0x0BA3), **ProjectionStackDepth** = ((int)0x0BA4), **TextureStackDepth** = ((int)0x0BA5),
ModelviewMatrix = ((int)0x0BA6), **ProjectionMatrix** = ((int)0x0BA7), **TextureMatrix** = ((int)0x0BA8), **AttribStackDepth** = ((int)0x0BB0),
ClientAttribStackDepth = ((int)0x0BB1), **AlphaTest** = ((int)0x0BC0), **AlphaTestFunc** = ((int)0x0BC1), **AlphaTestRef** = ((int)0x0BC2),
Dither = ((int)0x0BD0), **BlendDst** = ((int)0x0BE0), **BlendSrc** = ((int)0x0BE1), **Blend** = ((int)0x0BE2),
LogicOpMode = ((int)0x0BF0), **IndexLogicOp** = ((int)0x0BF1), **LogicOp** = ((int)0x0BF1), **ColorLogicOp** = ((int)0x0BF2),
AuxBuffers = ((int)0x0C00), **DrawBuffer** = ((int)0x0C01), **ReadBuffer** = ((int)0x0C02), **ScissorBox** = ((int)0x0C10),
ScissorTest = ((int)0x0C11), **IndexClearValue** = ((int)0x0C20), **IndexWritemask** = ((int)0x0C21), **ColorClearValue** = ((int)0x0C22),
ColorWritemask = ((int)0x0C23), **IndexMode** = ((int)0x0C30), **RgbaMode** = ((int)0x0C31), **Doublebuffer** = ((int)0x0C32),
Stereo = ((int)0x0C33), **RenderMode** = ((int)0x0C40), **PerspectiveCorrectionHint** = ((int)0x0C50), **PointSmoothHint** = ((int)0x0C51),
LineSmoothHint = ((int)0x0C52), **PolygonSmoothHint** = ((int)0x0C53), **FogHint** = ((int)0x0C54), **TextureGenS** = ((int)0x0C60),
TextureGenT = ((int)0x0C61), **TextureGenR** = ((int)0x0C62), **TextureGenQ** = ((int)0x0C63), **PixelMapIToSize** = ((int)0x0CB0),
PixelMapSToSSize = ((int)0x0CB1), **PixelMapIToRSize** = ((int)0x0CB2), **PixelMapIToGSize** = ((int)0x0CB3), **PixelMapIToBSize** = ((int)0x0CB4),
PixelMapIToASize = ((int)0x0CB5), **PixelMapRToRSize** = ((int)0x0CB6), **PixelMapGToGSize** = ((int)0x0CB7), **PixelMapBToBSize** = ((int)0x0CB8),
PixelMapAToASize = ((int)0x0CB9), **UnpackSwapBytes** = ((int)0x0CF0), **UnpackLsbFirst** = ((int)0x0CF1), **UnpackRowLength** = ((int)0x0CF2),
UnpackSkipRows = ((int)0x0CF3), **UnpackSkipPixels** = ((int)0x0CF4), **UnpackAlignment** = ((int)0x0CF5), **PackSwapBytes** = ((int)0x0D00),
PackLsbFirst = ((int)0x0D01), **PackRowLength** = ((int)0x0D02), **PackSkipRows** = ((int)0x0D03), **PackSkipPixels** = ((int)0x0D04),
PackAlignment = ((int)0x0D05), **MapColor** = ((int)0x0D10), **MapStencil** = ((int)0x0D11), **IndexShift** = ((int)0x0D12),
IndexOffset = ((int)0x0D13), **RedScale** = ((int)0x0D14), **RedBias** = ((int)0x0D15), **ZoomX** = ((int)0x0D16),

ZoomY = ((int)0x0D17), **GreenScale** = ((int)0x0D18), **GreenBias** = ((int)0x0D19), **BlueScale** = ((int)0x0D1A),
BlueBias = ((int)0x0D1B), **AlphaScale** = ((int)0x0D1C), **AlphaBias** = ((int)0x0D1D), **DepthScale** = ((int)0x0D1E),
DepthBias = ((int)0x0D1F), **MaxEvalOrder** = ((int)0x0D30), **MaxLights** = ((int)0x0D31), **MaxClipPlanes** = ((int)0x0D32),
MaxTextureSize = ((int)0x0D33), **MaxPixelMapTable** = ((int)0x0D34), **MaxAttribStackDepth** = ((int)0x0D35), **MaxModelviewStackDepth** = ((int)0x0D36),
MaxNameStackDepth = ((int)0x0D37), **MaxProjectionStackDepth** = ((int)0x0D38), **MaxTextureStackDepth** = ((int)0x0D39), **MaxViewportDims** = ((int)0x0D3A),
MaxClientAttribStackDepth = ((int)0x0D3B), **SubpixelBits** = ((int)0x0D50), **IndexBits** = ((int)0x0D51), **RedBits** = ((int)0x0D52),
GreenBits = ((int)0x0D53), **BlueBits** = ((int)0x0D54), **AlphaBits** = ((int)0x0D55), **DepthBits** = ((int)0x0D56),
StencilBits = ((int)0x0D57), **AccumRedBits** = ((int)0x0D58), **AccumGreenBits** = ((int)0x0D59), **AccumBlueBits** = ((int)0x0D5A),
AccumAlphaBits = ((int)0x0D5B), **NameStackDepth** = ((int)0x0D70), **AutoNormal** = ((int)0x0D80), **Map1Color4** = ((int)0x0D90),
Map1Index = ((int)0x0D91), **Map1Normal** = ((int)0x0D92), **Map1TextureCoord1** = ((int)0x0D93), **Map1TextureCoord2** = ((int)0x0D94),
Map1TextureCoord3 = ((int)0x0D95), **Map1TextureCoord4** = ((int)0x0D96), **Map1Vertex3** = ((int)0x0D97), **Map1Vertex4** = ((int)0x0D98),
Map2Color4 = ((int)0x0DB0), **Map2Index** = ((int)0x0DB1), **Map2Normal** = ((int)0x0DB2), **Map2TextureCoord1** = ((int)0x0DB3),
Map2TextureCoord2 = ((int)0x0DB4), **Map2TextureCoord3** = ((int)0x0DB5), **Map2TextureCoord4** = ((int)0x0DB6), **Map2Vertex3** = ((int)0x0DB7),
Map2Vertex4 = ((int)0x0DB8), **Map1GridDomain** = ((int)0x0DD0), **Map1GridSegments** = ((int)0x0DD1), **Map2GridDomain** = ((int)0x0DD2),
Map2GridSegments = ((int)0x0DD3), **Texture1D** = ((int)0x0DE0), **Texture2D** = ((int)0x0DE1), **FeedbackBufferSize** = ((int)0x0DF1),
FeedbackBufferType = ((int)0x0DF2), **SelectionBufferSize** = ((int)0x0DF4), **PolygonOffsetUnits** = ((int)0x2A00), **PolygonOffsetPoint** = ((int)0x2A01),
PolygonOffsetLine = ((int)0x2A02), **ClipPlane0** = ((int)0x3000), **ClipPlane1** = ((int)0x3001), **ClipPlane2** = ((int)0x3002),
ClipPlane3 = ((int)0x3003), **ClipPlane4** = ((int)0x3004), **ClipPlane5** = ((int)0x3005), **Light0** = ((int)0x4000),

Light1 = ((int)0x4001), **Light2** = ((int)0x4002), **Light3** = ((int)0x4003), **Light4** = ((int)0x4004),
Light5 = ((int)0x4005), **Light6** = ((int)0x4006), **Light7** = ((int)0x4007), **BlendColorExt** = ((int)0x8005),
BlendEquationExt = ((int)0x8009), **BlendEquationRgb** = ((int)0x8009), **PackCmykHintExt** = ((int)0x800E), **UnpackCmykHintExt** = ((int)0x800F),
Convolution1DExt = ((int)0x8010), **Convolution2DExt** = ((int)0x8011), **Separable2DExt** = ((int)0x8012), **PostConvolutionRedScaleExt** = ((int)0x801C),
PostConvolutionGreenScaleExt = ((int)0x801D), **PostConvolutionBlueScaleExt** = ((int)0x801E), **PostConvolutionAlphaScaleExt** = ((int)0x801F), **PostConvolutionRedBiasExt** = ((int)0x8020),
PostConvolutionGreenBiasExt = ((int)0x8021), **PostConvolutionBlueBiasExt** = ((int)0x8022), **PostConvolutionAlphaBiasExt** = ((int)0x8023), **HistogramExt** = ((int)0x8024),
MinmaxExt = ((int)0x802E), **PolygonOffsetFill** = ((int)0x8037), **PolygonOffsetFactor** = ((int)0x8038), **PolygonOffsetBiasExt** = ((int)0x8039),
RescaleNormalExt = ((int)0x803A), **TextureBinding1D** = ((int)0x8068), **TextureBinding2D** = ((int)0x8069), **Texture3DBindingExt** = ((int)0x806A),
TextureBinding3D = ((int)0x806A), **PackSkipImagesExt** = ((int)0x806B), **PackImageHeightExt** = ((int)0x806C), **UnpackSkipImagesExt** = ((int)0x806D),
UnpackImageHeightExt = ((int)0x806E), **Texture3DExt** = ((int)0x806F), **Max3DTextureSize** = ((int)0x8073), **Max3DTextureSizeExt** = ((int)0x8073),
VertexArray = ((int)0x8074), **NormalArray** = ((int)0x8075), **ColorArray** = ((int)0x8076), **IndexArray** = ((int)0x8077),
TextureCoordArray = ((int)0x8078), **EdgeFlagArray** = ((int)0x8079), **VertexArraySize** = ((int)0x807A), **VertexArrayType** = ((int)0x807B),
VertexArrayStride = ((int)0x807C), **VertexArrayCountExt** = ((int)0x807D), **NormalArrayType** = ((int)0x807E), **NormalArrayStride** = ((int)0x807F),
NormalArrayCountExt = ((int)0x8080), **ColorArraySize** = ((int)0x8081), **ColorArrayType** = ((int)0x8082), **ColorArrayStride** = ((int)0x8083),
ColorArrayCountExt = ((int)0x8084), **IndexArrayType** = ((int)0x8085), **IndexArrayStride** = ((int)0x8086), **IndexArrayCountExt** = ((int)0x8087),
TextureCoordArraySize = ((int)0x8088), **TextureCoordArrayType** = ((int)0x8089), **TextureCoordArrayStride** = ((int)0x808A), **TextureCoordArrayCountExt** = ((int)0x808B),
EdgeFlagArrayStride = ((int)0x808C), **EdgeFlagArrayCountExt** = ((int)0x808D), **InterlaceSgix** = ((int)0x8094), **DetailTexture2DBindingSgix** = ((int)0x8096),
Multisample = ((int)0x809D), **MultisampleSgix** = ((int)0x809D), **SampleAlphaToCoverage** = ((int)0x809E), **SampleAlphaToMaskSgix** = ((int)0x809E),

```
SampleAlphaToOne = ((int)0x809F), SampleAlphaToOneSgis =  
((int)0x809F), SampleCoverage = ((int)0x80A0), SampleMaskSgis =  
((int)0x80A0),  
  
SampleBuffers = ((int)0x80A8), SampleBuffersSgis = ((int)0x80A8), Samples  
= ((int)0x80A9), SamplesSgis = ((int)0x80A9),  
  
SampleCoverageValue = ((int)0x80AA), SampleMaskValueSgis =  
((int)0x80AA), SampleCoverageInvert = ((int)0x80AB), SampleMask-  
InvertSgis = ((int)0x80AB),  
  
SamplePatternSgis = ((int)0x80AC), ColorMatrixSgi = ((int)0x80B1), Col-  
orMatrixStackDepthSgi = ((int)0x80B2), MaxColorMatrixStackDepthSgi =  
((int)0x80B3),  
  
PostColorMatrixRedScaleSgi = ((int)0x80B4), PostColorMatrixGreen-  
ScaleSgi = ((int)0x80B5), PostColorMatrixBlueScaleSgi = ((int)0x80B6),  
PostColorMatrixAlphaScaleSgi = ((int)0x80B7),  
  
PostColorMatrixRedBiasSgi = ((int)0x80B8), PostColorMatrixGreenBi-  
asSgi = ((int)0x80B9), PostColorMatrixBlueBiasSgi = ((int)0x80BA), Post-  
ColorMatrixAlphaBiasSgi = ((int)0x80BB),  
  
TextureColorTableSgi = ((int)0x80BC), BlendDstRgb = ((int)0x80C8),  
BlendSrcRgb = ((int)0x80C9), BlendDstAlpha = ((int)0x80CA),  
  
BlendSrcAlpha = ((int)0x80CB), ColorTableSgi = ((int)0x80D0), PostCon-  
volutionColorTableSgi = ((int)0x80D1), PostColorMatrixColorTableSgi =  
((int)0x80D2),  
  
MaxElementsVertices = ((int)0x80E8), MaxElementsIndices = ((int)0x80E9),  
PointSizeMin = ((int)0x8126), PointSizeMinSgis = ((int)0x8126),  
  
PointSizeMax = ((int)0x8127), PointSizeMaxSgis = ((int)0x8127), PointFade-  
ThresholdSize = ((int)0x8128), PointFadeThresholdSizeSgis = ((int)0x8128),  
  
DistanceAttenuationSgis = ((int)0x8129), PointDistanceAttenuation =  
((int)0x8129), FogFuncPointsSgis = ((int)0x812B), MaxFogFuncPointsSgis =  
((int)0x812C),  
  
PackSkipVolumesSgis = ((int)0x8130), PackImageDepthSgis = ((int)0x8131),  
UnpackSkipVolumesSgis = ((int)0x8132), UnpackImageDepthSgis =  
((int)0x8133),  
  
Texture4DSgis = ((int)0x8134), Max4DTextureSizeSgis = ((int)0x8138), Pix-  
elTexGenSgix = ((int)0x8139), PixelTileBestAlignmentSgix = ((int)0x813E),  
  
PixelTileCacheIncrementSgix = ((int)0x813F), PixelTileWidthSgix =  
((int)0x8140), PixelTileHeightSgix = ((int)0x8141), PixelTileGridWidthSgix  
= ((int)0x8142),  
  
PixelTileGridHeightSgix = ((int)0x8143), PixelTileGridDepthSgix =  
((int)0x8144), PixelTileCacheSizeSgix = ((int)0x8145), SpriteSgix =  
((int)0x8148),
```

SpriteModeSgix = ((int)0x8149), **SpriteAxisSgix** = ((int)0x814A), **SpriteTranslationSgix** = ((int)0x814B), **Texture4DBindingSgix** = ((int)0x814F),
MaxClipmapDepthSgix = ((int)0x8177), **MaxClipmapVirtualDepthSgix** = ((int)0x8178), **PostTextureFilterBiasRangeSgix** = ((int)0x817B), **PostTextureFilterScaleRangeSgix** = ((int)0x817C),
ReferencePlaneSgix = ((int)0x817D), **ReferencePlaneEquationSgix** = ((int)0x817E), **IrInstrument1Sgix** = ((int)0x817F), **InstrumentMeasurementsSgix** = ((int)0x8181),
CalligraphicFragmentSgix = ((int)0x8183), **FramezoomSgix** = ((int)0x818B), **FramezoomFactorSgix** = ((int)0x818C), **MaxFramezoomFactorSgix** = ((int)0x818D),
GenerateMipmapHint = ((int)0x8192), **GenerateMipmapHintSgis** = ((int)0x8192), **DeformationsMaskSgix** = ((int)0x8196), **FogOffsetSgix** = ((int)0x8198),
FogOffsetValueSgix = ((int)0x8199), **LightModelColorControl** = ((int)0x81F8), **SharedTexturePaletteExt** = ((int)0x81FB), **MajorVersion** = ((int)0x821B),
MinorVersion = ((int)0x821C), **NumExtensions** = ((int)0x821D), **ContextFlags** = ((int)0x821E), **ConvolutionHintSgix** = ((int)0x8316),
AsyncMarkerSgix = ((int)0x8329), **PixelTexGenModeSgix** = ((int)0x832B), **AsyncHistogramSgix** = ((int)0x832C), **MaxAsyncHistogramSgix** = ((int)0x832D),
PixelTextureSgis = ((int)0x8353), **AsyncTexImageSgix** = ((int)0x835C), **AsyncDrawPixelsSgix** = ((int)0x835D), **AsyncReadPixelsSgix** = ((int)0x835E),
MaxAsyncTexImageSgix = ((int)0x835F), **MaxAsyncDrawPixelsSgix** = ((int)0x8360), **MaxAsyncReadPixelsSgix** = ((int)0x8361), **VertexPreclipSgix** = ((int)0x83EE),
VertexPreclipHintSgix = ((int)0x83EF), **FragmentLightingSgix** = ((int)0x8400), **FragmentColorMaterialSgix** = ((int)0x8401), **FragmentColorMaterialFaceSgix** = ((int)0x8402),
FragmentColorMaterialParameterSgix = ((int)0x8403), **MaxFragmentLightsSgix** = ((int)0x8404), **MaxActiveLightsSgix** = ((int)0x8405), **LightEnvModeSgix** = ((int)0x8407),
FragmentLightModelLocalViewerSgix = ((int)0x8408), **FragmentLightModelTwoSideSgix** = ((int)0x8409), **FragmentLightModelAmbientSgix** = ((int)0x840A), **FragmentLightModelNormalInterpolationSgix** = ((int)0x840B),
FragmentLight0Sgix = ((int)0x840C), **PackResampleSgix** = ((int)0x842C), **UnpackResampleSgix** = ((int)0x842D), **CurrentFogCoord** = ((int)0x8453),
FogCoordArrayType = ((int)0x8454), **FogCoordArrayStride** = ((int)0x8455), **ColorSum** = ((int)0x8458), **CurrentSecondaryColor** = ((int)0x8459),

SecondaryColorArraySize = ((int)0x845A), **SecondaryColorArrayType** = ((int)0x845B), **SecondaryColorArrayStride** = ((int)0x845C), **CurrentRasterSecondaryColor** = ((int)0x845F),
AliasedPointSizeRange = ((int)0x846D), **AliasedLineWidthRange** = ((int)0x846E), **ActiveTexture** = ((int)0x84E0), **ClientActiveTexture** = ((int)0x84E1),
MaxTextureUnits = ((int)0x84E2), **TransposeModelviewMatrix** = ((int)0x84E3), **TransposeProjectionMatrix** = ((int)0x84E4), **TransposeTextureMatrix** = ((int)0x84E5),
TransposeColorMatrix = ((int)0x84E6), **MaxRenderbufferSize** = ((int)0x84E8), **MaxRenderbufferSizeExt** = ((int)0x84E8), **TextureCompressionHint** = ((int)0x84EF),
TextureBindingRectangle = ((int)0x84F6), **MaxRectangleTextureSize** = ((int)0x84F8), **MaxTextureLodBias** = ((int)0x84FD), **TextureCubeMap** = ((int)0x8513),
TextureBindingCubeMap = ((int)0x8514), **MaxCubeMapTextureSize** = ((int)0x851C), **PackSubsampleRateSgix** = ((int)0x85A0), **UnpackSubsampleRateSgix** = ((int)0x85A1),
VertexArrayBinding = ((int)0x85B5), **ProgramPointSize** = ((int)0x8642), **DepthClamp** = ((int)0x864F), **NumCompressedTextureFormats** = ((int)0x86A2),
CompressedTextureFormats = ((int)0x86A3), **StencilBackFunc** = ((int)0x8800), **StencilBackFail** = ((int)0x8801), **StencilBackPassDepthFail** = ((int)0x8802),
StencilBackPassDepthPass = ((int)0x8803), **RgbaFloatMode** = ((int)0x8820), **MaxDrawBuffers** = ((int)0x8824), **DrawBuffer0** = ((int)0x8825),
DrawBuffer1 = ((int)0x8826), **DrawBuffer2** = ((int)0x8827), **DrawBuffer3** = ((int)0x8828), **DrawBuffer4** = ((int)0x8829),
DrawBuffer5 = ((int)0x882A), **DrawBuffer6** = ((int)0x882B), **DrawBuffer7** = ((int)0x882C), **DrawBuffer8** = ((int)0x882D),
DrawBuffer9 = ((int)0x882E), **DrawBuffer10** = ((int)0x882F), **DrawBuffer11** = ((int)0x8830), **DrawBuffer12** = ((int)0x8831),
DrawBuffer13 = ((int)0x8832), **DrawBuffer14** = ((int)0x8833), **DrawBuffer15** = ((int)0x8834), **BlendEquationAlpha** = ((int)0x883D),
TextureCubeMapSeamless = ((int)0x884F), **PointSprite** = ((int)0x8861), **MaxVertexAttribs** = ((int)0x8869), **MaxTextureCoords** = ((int)0x8871),
MaxTextureImageUnits = ((int)0x8872), **ArrayBufferBinding** = ((int)0x8894), **ElementArrayBufferBinding** = ((int)0x8895), **VertexArrayBufferBinding** = ((int)0x8896),
NormalArrayBufferBinding = ((int)0x8897), **ColorArrayBufferBinding** = ((int)0x8898), **IndexArrayBufferBinding** = ((int)0x8899), **TextureCoordArrayBufferBinding** = ((int)0x889A),

EdgeFlagArrayBufferBinding = ((int)0x889B), **SecondaryColorArrayBufferBinding** = ((int)0x889C), **FogCoordArrayBufferBinding** = ((int)0x889D), **WeightArrayBufferBinding** = ((int)0x889E),

VertexAttribArrayBufferBinding = ((int)0x889F), **PixelPackBufferBinding** = ((int)0x88ED), **PixelUnpackBufferBinding** = ((int)0x88EF), **MaxArrayTextureLayers** = ((int)0x88FF),

MinProgramTexelOffset = ((int)0x8904), **MaxProgramTexelOffset** = ((int)0x8905), **ClampVertexColor** = ((int)0x891A), **ClampFragmentColor** = ((int)0x891B),

ClampReadColor = ((int)0x891C), **MaxVertexUniformBlocks** = ((int)0x8A2B), **MaxGeometryUniformBlocks** = ((int)0x8A2C), **MaxFragmentUniformBlocks** = ((int)0x8A2D),

MaxCombinedUniformBlocks = ((int)0x8A2E), **MaxUniformBufferBindings** = ((int)0x8A2F), **MaxUniformBlockSize** = ((int)0x8A30), **MaxCombinedVertexUniformComponents** = ((int)0x8A31),

MaxCombinedGeometryUniformComponents = ((int)0x8A32), **MaxCombinedFragmentUniformComponents** = ((int)0x8A33), **UniformBufferOffsetAlignment** = ((int)0x8A34), **MaxFragmentUniformComponents** = ((int)0x8B49),

MaxVertexUniformComponents = ((int)0x8B4A), **MaxVaryingComponents** = ((int)0x8B4B), **MaxVaryingFloats** = ((int)0x8B4B), **MaxVertexTextureImageUnits** = ((int)0x8B4C),

MaxCombinedTextureImageUnits = ((int)0x8B4D), **FragmentShaderDerivativeHint** = ((int)0x8B8B), **CurrentProgram** = ((int)0x8B8D), **TextureBinding1DArray** = ((int)0x8C1C),

TextureBinding2DArray = ((int)0x8C1D), **MaxGeometryTextureImageUnits** = ((int)0x8C29), **MaxTransformFeedbackSeparateComponents** = ((int)0x8C80), **MaxTransformFeedbackInterleavedComponents** = ((int)0x8C8A),

MaxTransformFeedbackSeparateAttribs = ((int)0x8C8B), **StencilBackRef** = ((int)0x8CA3), **StencilBackValueMask** = ((int)0x8CA4), **StencilBackWritemask** = ((int)0x8CA5),

DrawFramebufferBinding = ((int)0x8CA6), **FramebufferBinding** = ((int)0x8CA6), **FramebufferBindingExt** = ((int)0x8CA6), **RenderbufferBinding** = ((int)0x8CA7),

RenderbufferBindingExt = ((int)0x8CA7), **ReadFramebufferBinding** = ((int)0x8CAA), **MaxColorAttachments** = ((int)0x8CDF), **MaxColorAttachmentsExt** = ((int)0x8CDF),

MaxSamples = ((int)0x8D57), **FramebufferSrgb** = ((int)0x8DB9), **MaxGeometryVaryingComponents** = ((int)0x8DDD), **MaxVertexVaryingComponents** = ((int)0x8DDE),


```

MaxGeometryUniformComponents = ((int)0x8DDF), MaxGeometryOutputVertices = ((int)0x8DE0), MaxGeometryTotalOutputComponents = ((int)0x8DE1), QuadsFollowProvokingVertexConvention = ((int)0x8E4C),

ProvokingVertex = ((int)0x8E4F), SampleMask = ((int)0x8E51), MaxSampleMaskWords = ((int)0x8E59), TextureBinding2DMultisample = ((int)0x9104),

TextureBinding2DMultisampleArray = ((int)0x9105), MaxColorTextureSamples = ((int)0x910E), MaxDepthTextureSamples = ((int)0x910F), MaxIntegerSamples = ((int)0x9110) }

• enum GetPointervPName {

    FeedbackBufferPointer = ((int)0x0DF0), SelectionBufferPointer = ((int)0x0DF3), VertexArrayPointer = ((int)0x808E), NormalArrayPointer = ((int)0x808F),

    ColorArrayPointer = ((int)0x8090), IndexArrayPointer = ((int)0x8091), TextureCoordArrayPointer = ((int)0x8092), EdgeFlagArrayPointer = ((int)0x8093),

    InstrumentBufferPointerSgix = ((int)0x8180), FogCoordArrayPointer = ((int)0x8456), SecondaryColorArrayPointer = ((int)0x845D) }

• enum GetQueryObjectParam { QueryResult = ((int)0x8866), QueryResultAvailable = ((int)0x8867) }

• enum GetQueryParam { QueryCounterBits = ((int)0x8864), CurrentQuery = ((int)0x8865) }

• enum GetTextureParameter {

    TextureWidth = ((int)0x1000), TextureHeight = ((int)0x1001), TextureComponents = ((int)0x1003), TextureInternalFormat = ((int)0x1003),

    TextureBorderColor = ((int)0x1004), TextureBorder = ((int)0x1005), TextureMagFilter = ((int)0x2800), TextureMinFilter = ((int)0x2801),

    TextureWrapS = ((int)0x2802), TextureWrapT = ((int)0x2803), TextureRedSize = ((int)0x805C), TextureGreenSize = ((int)0x805D),

    TextureBlueSize = ((int)0x805E), TextureAlphaSize = ((int)0x805F), TextureLuminanceSize = ((int)0x8060), TextureIntensitySize = ((int)0x8061),

    TexturePriority = ((int)0x8066), TextureResident = ((int)0x8067), TextureDepth = ((int)0x8071), TextureDepthExt = ((int)0x8071),

    TextureWrapR = ((int)0x8072), TextureWrapRExt = ((int)0x8072), DetailTextureLevelSgis = ((int)0x809A), DetailTextureModeSgis = ((int)0x809B),

    DetailTextureFuncPointsSgis = ((int)0x809C), SharpenTextureFuncPointsSgis = ((int)0x80B0), ShadowAmbientSgix = ((int)0x80BF), DualTextureSelectSgis = ((int)0x8124),

    QuadTextureSelectSgis = ((int)0x8125), Texture4DsizeSgis = ((int)0x8136), TextureWrapQSgis = ((int)0x8137), TextureMinLod = ((int)0x813A),

```

TextureMinLodSgis = ((int)0x813A), **TextureMaxLod** = ((int)0x813B), **TextureMaxLodSgis** = ((int)0x813B), **TextureBaseLevel** = ((int)0x813C),

TextureBaseLevelSgis = ((int)0x813C), **TextureMaxLevel** = ((int)0x813D), **TextureMaxLevelSgis** = ((int)0x813D), **TextureFilter4SizeSgis** = ((int)0x8147),

TextureClipmapCenterSgix = ((int)0x8171), **TextureClipmapFrameSgix** = ((int)0x8172), **TextureClipmapOffsetSgix** = ((int)0x8173), **TextureClipmapVirtualDepthSgix** = ((int)0x8174),

TextureClipmapLodOffsetSgix = ((int)0x8175), **TextureClipmapDepthSgix** = ((int)0x8176), **PostTextureFilterBiasSgix** = ((int)0x8179), **PostTextureFilterScaleSgix** = ((int)0x817A),

TextureLodBiasSSgix = ((int)0x818E), **TextureLodBiasTSgix** = ((int)0x818F), **TextureLodBiasRSgix** = ((int)0x8190), **GenerateMipmap** = ((int)0x8191),

GenerateMipmapSgis = ((int)0x8191), **TextureCompareSgix** = ((int)0x819A), **TextureCompareOperatorSgix** = ((int)0x819B), **TextureLessEqualRSgix** = ((int)0x819C),

TextureGequalRSgix = ((int)0x819D), **TextureMaxClampSSgix** = ((int)0x8369), **TextureMaxClampTSgix** = ((int)0x836A), **TextureMaxClampRSgix** = ((int)0x836B),

TextureCompressedImageSize = ((int)0x86A0), **TextureCompressed** = ((int)0x86A1), **TextureDepthSize** = ((int)0x884A), **DepthTextureMode** = ((int)0x884B),

TextureCompareMode = ((int)0x884C), **TextureCompareFunc** = ((int)0x884D), **TextureStencilSize** = ((int)0x88F1), **TextureRedType** = ((int)0x8C10),

TextureGreenType = ((int)0x8C11), **TextureBlueType** = ((int)0x8C12), **TextureAlphaType** = ((int)0x8C13), **TextureLuminanceType** = ((int)0x8C14),

TextureIntensityType = ((int)0x8C15), **TextureDepthType** = ((int)0x8C16), **TextureSharedSize** = ((int)0x8C3F), **TextureSamples** = ((int)0x9106),

TextureFixedSampleLocations = ((int)0x9107) }

- enum **GL3DfxMultisample** { **MultisampleBit3Dfx** = ((int)0x20000000), **Multisample3Dfx** = ((int)0x86B2), **SampleBuffers3Dfx** = ((int)0x86B3), **Samples3Dfx** = ((int)0x86B4) }
- enum **GL3DfxTbuffer**
- enum **GL3DfxTextureCompressionFxt1** { **CompressedRgbFxt13Dfx** = ((int)0x86B0), **CompressedRgbaFxt13Dfx** = ((int)0x86B1) }
- enum **GreedyFrameTerminator**
- enum **GreedyStringMarker**
- enum **HintMode** { **DontCare** = ((int)0x1100), **Fastest** = ((int)0x1101), **Nicest** = ((int)0x1102) }

- enum **HintTarget** {
 - PerspectiveCorrectionHint** = ((int)0x0C50), **PointSmoothHint** = ((int)0x0C51), **LineSmoothHint** = ((int)0x0C52), **PolygonSmoothHint** = ((int)0x0C53),
 - FogHint** = ((int)0x0C54), **PackCmykHintExt** = ((int)0x800E), **UnpackCmykHintExt** = ((int)0x800F), **TextureMultiBufferHintSgix** = ((int)0x812E),
 - GenerateMipmapHint** = ((int)0x8192), **GenerateMipmapHintSgis** = ((int)0x8192), **ConvolutionHintSgix** = ((int)0x8316), **VertexPreclipHintSgix** = ((int)0x83EF),
 - TextureCompressionHint** = ((int)0x84EF), **FragmentShaderDerivativeHint** = ((int)0x8B8B) }
- enum **HistogramTarget** { **Histogram** = ((int)0x8024), **ProxyHistogram** = ((int)0x8025) }
- enum **HistogramTargetExt** { **HistogramExt** = ((int)0x8024), **ProxyHistogramExt** = ((int)0x8025) }
- enum **HpConvolutionBorderModes** { **IgnoreBorderHp** = ((int)0x8150), **ConstantBorderHp** = ((int)0x8151), **ReplicateBorderHp** = ((int)0x8153), **ConvolutionBorderColorHp** = ((int)0x8154) }
- enum **HpImageTransform** {
 - ImageScaleXHp** = ((int)0x8155), **ImageScaleYHp** = ((int)0x8156), **ImageTranslateXHp** = ((int)0x8157), **ImageTranslateYHp** = ((int)0x8158),
 - ImageRotateAngleHp** = ((int)0x8159), **ImageRotateOriginXHp** = ((int)0x815A), **ImageRotateOriginYHp** = ((int)0x815B), **ImageMagFilterHp** = ((int)0x815C),
 - ImageMinFilterHp** = ((int)0x815D), **ImageCubicWeightHp** = ((int)0x815E), **CubicHp** = ((int)0x815F), **AverageHp** = ((int)0x8160),
 - ImageTransform2DHp** = ((int)0x8161), **PostImageTransformColorTableHp** = ((int)0x8162), **ProxyPostImageTransformColorTableHp** = ((int)0x8163) }
- enum **HpOcclusionTest** { **OcclusionTestHp** = ((int)0x8165), **OcclusionTestResultHp** = ((int)0x8166) }
- enum **HpTextureLighting** { **TextureLightingModeHp** = ((int)0x8167), **TexturePostSpecularHp** = ((int)0x8168), **TexturePreSpecularHp** = ((int)0x8169) }
- enum **IbmCullVertex** { **CullVertexIbm** = ((int)103050) }
- enum **IbmMultimodeDrawArrays**
- enum **IbmRasterposClip** { **RasterPositionUnclippedIbm** = ((int)0x19262) }
- enum **IbmTextureMirroredRepeat** { **MirroredRepeatIbm** = ((int)0x8370) }
- enum **IbmVertexArrayLists** {
 - VertexArrayListIbm** = ((int)103070), **NormalArrayListIbm** = ((int)103071), **ColorArrayListIbm** = ((int)103072), **IndexArrayListIbm** = ((int)103073),

```

TextureCoordArrayListIbm = ((int)103074), EdgeFlagArrayListIbm =
((int)103075), FogCoordinateArrayListIbm = ((int)103076), SecondaryColorArrayListIbm = ((int)103077),

VertexArrayListStrideIbm = ((int)103080), NormalArrayListStrideIbm =
((int)103081), ColorArrayListStrideIbm = ((int)103082), IndexArrayListStrideIbm = ((int)103083),

TextureCoordArrayListStrideIbm = ((int)103084), EdgeFlagArrayListStrideIbm = ((int)103085),
FogCoordinateArrayListStrideIbm = ((int)103086), SecondaryColorArrayListStrideIbm = ((int)103087) }
• enum IndexedEnableCap { Blend = ((int)0x0BE2) }
• enum IndexPointerType { Short = ((int)0x1402), Int = ((int)0x1404), Float = ((int)0x1406), Double = ((int)0x140A) }
• enum IngrColorClamp {
    RedMinClampIngr = ((int)0x8560), GreenMinClampIngr = ((int)0x8561),
    BlueMinClampIngr = ((int)0x8562), AlphaMinClampIngr = ((int)0x8563),
    RedMaxClampIngr = ((int)0x8564), GreenMaxClampIngr = ((int)0x8565),
    BlueMaxClampIngr = ((int)0x8566), AlphaMaxClampIngr = ((int)0x8567) }
• enum IngrInterlaceRead { InterlaceReadIngr = ((int)0x8568) }
• enum IngrPaletteBuffer
• enum IntelParallelArrays {
    ParallelArraysIntel = ((int)0x83F4), VertexArrayParallelPointersIntel =
((int)0x83F5), NormalArrayParallelPointersIntel = ((int)0x83F6), ColorArrayParallelPointersIntel = ((int)0x83F7),
    TextureCoordArrayParallelPointersIntel = ((int)0x83F8) }
• enum IntelTextureScissor
• enum InterleavedArrayFormat {
    V2f = ((int)0x2A20), V3f = ((int)0x2A21), C4ubV2f = ((int)0x2A22), C4ubV3f = ((int)0x2A23),
    C3fV3f = ((int)0x2A24), N3fV3f = ((int)0x2A25), C4fN3fV3f = ((int)0x2A26), T2fV3f = ((int)0x2A27),
    T4fV4f = ((int)0x2A28), T2fC4ubV3f = ((int)0x2A29), T2fC3fV3f = ((int)0x2A2A), T2fN3fV3f = ((int)0x2A2B),
    T2fC4fN3fV3f = ((int)0x2A2C), T4fC4fN3fV4f = ((int)0x2A2D) }
• enum LightEnvModeSgix { Add = ((int)0x0104), Replace = ((int)0x1E01), Modulate = ((int)0x2100) }
• enum LightEnvParameterSgix { LightEnvModeSgix = ((int)0x8407) }
• enum LightModelColorControl { SingleColor = ((int)0x81F9), SeparateSpecularColor = ((int)0x81FA) }
• enum LightModelParameter { LightModelLocalViewer = ((int)0x0B51), LightModelTwoSide = ((int)0x0B52), LightModelAmbient = ((int)0x0B53), LightModelColorControl = ((int)0x81F8) }

```

- enum **LightName** {
 - Light0** = ((int)0x4000), **Light1** = ((int)0x4001), **Light2** = ((int)0x4002), **Light3** = ((int)0x4003),
 - Light4** = ((int)0x4004), **Light5** = ((int)0x4005), **Light6** = ((int)0x4006), **Light7** = ((int)0x4007),
 - FragmentLight0Sgix** = ((int)0x840C), **FragmentLight1Sgix** = ((int)0x840D), **FragmentLight2Sgix** = ((int)0x840E), **FragmentLight3Sgix** = ((int)0x840F),
 - FragmentLight4Sgix** = ((int)0x8410), **FragmentLight5Sgix** = ((int)0x8411), **FragmentLight6Sgix** = ((int)0x8412), **FragmentLight7Sgix** = ((int)0x8413) }
- enum **LightParameter** {
 - Ambient** = ((int)0x1200), **Diffuse** = ((int)0x1201), **Specular** = ((int)0x1202), **Position** = ((int)0x1203),
 - SpotDirection** = ((int)0x1204), **SpotExponent** = ((int)0x1205), **SpotCutoff** = ((int)0x1206), **ConstantAttenuation** = ((int)0x1207),
 - LinearAttenuation** = ((int)0x1208), **QuadraticAttenuation** = ((int)0x1209) }
- enum **ListMode** { **Compile** = ((int)0x1300), **CompileAndExecute** = ((int)0x1301) }
- enum **ListNameType** {
 - Byte** = ((int)0x1400), **UnsignedByte** = ((int)0x1401), **Short** = ((int)0x1402), **UnsignedShort** = ((int)0x1403),
 - Int** = ((int)0x1404), **UnsignedInt** = ((int)0x1405), **Float** = ((int)0x1406), **GL2Bytes** = ((int)0x1407),
 - GL3Bytes** = ((int)0x1408), **GL4Bytes** = ((int)0x1409) }
- enum **ListParameterName** { **ListPrioritySgix** = ((int)0x8182) }
- enum **LogicOp** {
 - Clear** = ((int)0x1500), **And** = ((int)0x1501), **AndReverse** = ((int)0x1502), **Copy** = ((int)0x1503),
 - AndInverted** = ((int)0x1504), **Noop** = ((int)0x1505), **Xor** = ((int)0x1506), **Or** = ((int)0x1507),
 - Nor** = ((int)0x1508), **Equiv** = ((int)0x1509), **Invert** = ((int)0x150A), **OrReverse** = ((int)0x150B),
 - CopyInverted** = ((int)0x150C), **OrInverted** = ((int)0x150D), **Nand** = ((int)0x150E), **Set** = ((int)0x150F) }
- enum **MapTarget** {
 - Map1Color4** = ((int)0x0D90), **Map1Index** = ((int)0x0D91), **Map1Normal** = ((int)0x0D92), **Map1TextureCoord1** = ((int)0x0D93),
 - Map1TextureCoord2** = ((int)0x0D94), **Map1TextureCoord3** = ((int)0x0D95), **Map1TextureCoord4** = ((int)0x0D96), **Map1Vertex3** = ((int)0x0D97),
 - Map1Vertex4** = ((int)0x0D98), **Map2Color4** = ((int)0x0DB0), **Map2Index** = ((int)0x0DB1), **Map2Normal** = ((int)0x0DB2),

```

Map2TextureCoord1 = ((int)0x0DB3), Map2TextureCoord2 =
((int)0x0DB4), Map2TextureCoord3 = ((int)0x0DB5), Map2TextureCoord4
= ((int)0x0DB6),
Map2Vertex3 = ((int)0x0DB7), Map2Vertex4 = ((int)0x0DB8), GeometryDe-
formationSgix = ((int)0x8194), TextureDeformationSgix = ((int)0x8195) }
• enum MaterialFace { Front = ((int)0x0404), Back = ((int)0x0405), FrontAnd-
Back = ((int)0x0408) }
• enum MaterialParameter {
Ambient = ((int)0x1200), Diffuse = ((int)0x1201), Specular = ((int)0x1202),
Emission = ((int)0x1600),
Shininess = ((int)0x1601), AmbientAndDiffuse = ((int)0x1602), ColorIndexes
= ((int)0x1603) }
• enum MatrixMode { Modelview = ((int)0x1700), Projection = ((int)0x1701),
Texture = ((int)0x1702), Color = ((int)0x1800) }
• enum MatrixModeArb {
Modelview = ((int)0x1700), Projection = ((int)0x1701), Texture =
((int)0x1702), Color = ((int)0x1800),
Matrix0 = ((int)0x88C0), Matrix1 = ((int)0x88C1), Matrix2 = ((int)0x88C2),
Matrix3 = ((int)0x88C3),
Matrix4 = ((int)0x88C4), Matrix5 = ((int)0x88C5), Matrix6 = ((int)0x88C6),
Matrix7 = ((int)0x88C7),
Matrix8 = ((int)0x88C8), Matrix9 = ((int)0x88C9), Matrix10 = ((int)0x88CA),
Matrix11 = ((int)0x88CB),
Matrix12 = ((int)0x88CC), Matrix13 = ((int)0x88CD), Matrix14 =
((int)0x88CE), Matrix15 = ((int)0x88CF),
Matrix16 = ((int)0x88D0), Matrix17 = ((int)0x88D1), Matrix18 =
((int)0x88D2), Matrix19 = ((int)0x88D3),
Matrix20 = ((int)0x88D4), Matrix21 = ((int)0x88D5), Matrix22 =
((int)0x88D6), Matrix23 = ((int)0x88D7),
Matrix24 = ((int)0x88D8), Matrix25 = ((int)0x88D9), Matrix26 =
((int)0x88DA), Matrix27 = ((int)0x88DB),
Matrix28 = ((int)0x88DC), Matrix29 = ((int)0x88DD), Matrix30 =
((int)0x88DE), Matrix31 = ((int)0x88DF) }
• enum MesaPackInvert { PackInvertMesa = ((int)0x8758) }
• enum MesaResizeBuffers
• enum MesaWindowPos
• enum MesaxTextureStack {
Texture1DStackMesax = ((int)0x8759), Texture2DStackMesax =
((int)0x875A), ProxyTexture1DStackMesax = ((int)0x875B), ProxyTex-
ture2DStackMesax = ((int)0x875C),
Texture1DStackBindingMesax = ((int)0x875D), Tex-
ture2DStackBindingMesax = ((int)0x875E) }

```

- enum **MesaYcbcrTexture** { **UnsignedShort88Mesa** = ((int)0x85BA), **UnsignedShort88RevMesa** = ((int)0x85BB), **YcbcrMesa** = ((int)0x8757) }
- enum **MeshMode1** { **Point** = ((int)0x1B00), **Line** = ((int)0x1B01) }
- enum **MeshMode2** { **Point** = ((int)0x1B00), **Line** = ((int)0x1B01), **Fill** = ((int)0x1B02) }
- enum **MinmaxTarget** { **Minmax** = ((int)0x802E) }
- enum **MinmaxTargetExt** { **MinmaxExt** = ((int)0x802E) }
- enum **NormalPointerType** {
Byte = ((int)0x1400), **Short** = ((int)0x1402), **Int** = ((int)0x1404), **Float** = ((int)0x1406),
Double = ((int)0x140A), **HalfFloat** = ((int)0x140B) }
- enum **NvBlendSquare**
- enum **NvConditionalRender** { **QueryWaitNv** = ((int)0x8E13), **QueryNoWaitNv** = ((int)0x8E14), **QueryByRegionWaitNv** = ((int)0x8E15), **QueryByRegionNoWaitNv** = ((int)0x8E16) }
- enum **NvCopyDepthToColor** { **DepthStencilToRgbaNv** = ((int)0x886E), **DepthStencilToBgraNv** = ((int)0x886F) }
- enum **NvDepthBufferFloat** { **DepthComponent32fNv** = ((int)0x8DAB), **Depth32fStencil8Nv** = ((int)0x8DAC), **Float32UnsignedInt248RevNv** = ((int)0x8DAD), **DepthBufferFloatModeNv** = ((int)0x8DAF) }
- enum **NvDepthClamp** { **DepthClampNv** = ((int)0x864F) }
- enum **NvEvaluators** {
Eval2DNv = ((int)0x86C0), **EvalTriangular2DNv** = ((int)0x86C1), **MapTessellationNv** = ((int)0x86C2), **MapAttribUOrderNv** = ((int)0x86C3),
MapAttribVOrderNv = ((int)0x86C4), **EvalFractionalTessellationNv** = ((int)0x86C5), **EvalVertexAttrib0Nv** = ((int)0x86C6), **EvalVertexAttrib1Nv** = ((int)0x86C7),
EvalVertexAttrib2Nv = ((int)0x86C8), **EvalVertexAttrib3Nv** = ((int)0x86C9), **EvalVertexAttrib4Nv** = ((int)0x86CA), **EvalVertexAttrib5Nv** = ((int)0x86CB),
EvalVertexAttrib6Nv = ((int)0x86CC), **EvalVertexAttrib7Nv** = ((int)0x86CD), **EvalVertexAttrib8Nv** = ((int)0x86CE), **EvalVertexAttrib9Nv** = ((int)0x86CF),
EvalVertexAttrib10Nv = ((int)0x86D0), **EvalVertexAttrib11Nv** = ((int)0x86D1), **EvalVertexAttrib12Nv** = ((int)0x86D2), **EvalVertexAttrib13Nv** = ((int)0x86D3),
EvalVertexAttrib14Nv = ((int)0x86D4), **EvalVertexAttrib15Nv** = ((int)0x86D5), **MaxMapTessellationNv** = ((int)0x86D6), **MaxRationalEvalOrderNv** = ((int)0x86D7) }
- enum **NvExplicitMultisample** {
SamplePositionNv = ((int)0x8E50), **SampleMaskNv** = ((int)0x8E51), **SampleMaskValueNv** = ((int)0x8E52), **TextureBindingRenderbufferNv** = ((int)0x8E53),

TextureRenderbufferDataStoreBindingNv = ((int)0x8E54), **TextureRenderbufferNv** = ((int)0x8E55), **SamplerRenderbufferNv** = ((int)0x8E56), **IntSamplerRenderbufferNv** = ((int)0x8E57),

UnsignedIntSamplerRenderbufferNv = ((int)0x8E58), **MaxSampleMaskWordsNv** = ((int)0x8E59) }

- enum **NvFence** { **AllCompletedNv** = ((int)0x84F2), **FenceStatusNv** = ((int)0x84F3), **FenceConditionNv** = ((int)0x84F4) }

- enum **NvFloatBuffer** {

FloatRNv = ((int)0x8880), **FloatRgNv** = ((int)0x8881), **FloatRgbNv** = ((int)0x8882), **FloatRgbaNv** = ((int)0x8883),

FloatR16Nv = ((int)0x8884), **FloatR32Nv** = ((int)0x8885), **FloatRg16Nv** = ((int)0x8886), **FloatRg32Nv** = ((int)0x8887),

FloatRgb16Nv = ((int)0x8888), **FloatRgb32Nv** = ((int)0x8889), **FloatRgba16Nv** = ((int)0x888A), **FloatRgba32Nv** = ((int)0x888B),

TextureFloatComponentsNv = ((int)0x888C), **FloatClearColorValueNv** = ((int)0x888D), **FloatRgbaModeNv** = ((int)0x888E) }

- enum **NvFogDistance** { **EyePlane** = ((int)0x2502), **FogDistanceModeNv** = ((int)0x855A), **EyeRadialNv** = ((int)0x855B), **EyePlaneAbsoluteNv** = ((int)0x855C) }

- enum **NvFragmentProgram** {

MaxFragmentProgramLocalParametersNv = ((int)0x8868), **FragmentProgramNv** = ((int)0x8870), **MaxTextureCoordsNv** = ((int)0x8871), **MaxTextureImageUnitsNv** = ((int)0x8872),

FragmentProgramBindingNv = ((int)0x8873), **ProgramErrorStringNv** = ((int)0x8874) }

- enum **NvFragmentProgram2** {

MaxProgramExecInstructionsNv = ((int)0x88F4), **MaxProgramCallDepthNv** = ((int)0x88F5), **MaxProgramIfDepthNv** = ((int)0x88F6), **MaxProgramLoopDepthNv** = ((int)0x88F7),

MaxProgramLoopCountNv = ((int)0x88F8) }

- enum **NvFragmentProgram4**

- enum **NvFragmentProgramOption**

- enum **NvFramebufferMultisampleCoverage** { **RenderbufferCoverageSamplesNv** = ((int)0x8CAB), **RenderbufferColorSamplesNv** = ((int)0x8E10), **MaxMultisampleCoverageModesNv** = ((int)0x8E11), **MultisampleCoverageModesNv** = ((int)0x8E12) }

- enum **NvGeometryProgram4** {

LinesAdjacencyExt = ((int)0x000A), **LineStripAdjacencyExt** = ((int)0x000B), **TrianglesAdjacencyExt** = ((int)0x000C), **TriangleStripAdjacencyExt** = ((int)0x000D),


```

ProgramPointSizeExt = ((int)0x8642), GeometryProgramNv =
((int)0x8C26), MaxProgramOutputVerticesNv = ((int)0x8C27), Max-
ProgramTotalOutputComponentsNv = ((int)0x8C28),

MaxGeometryTextureImageUnitsExt = ((int)0x8C29), FramebufferAt-
achmentTextureLayerExt = ((int)0x8CD4), FramebufferAttachment-
LayeredExt = ((int)0x8DA7), FramebufferIncompleteLayerTargetsExt =
((int)0x8DA8),

FramebufferIncompleteLayerCountExt = ((int)0x8DA9), GeometryVertice-
sOutExt = ((int)0x8DDA), GeometryInputTypeExt = ((int)0x8DDB), Geom-
etryOutputTypeExt = ((int)0x8DDC) }
• enum NvGeometryShader4
• enum NvGpuProgram4 {
MinProgramTexelOffsetNv = ((int)0x8904), MaxProgramTexelOffsetNv =
((int)0x8905), ProgramAttribComponentsNv = ((int)0x8906), ProgramRe-
sultComponentsNv = ((int)0x8907),

MaxProgramAttribComponentsNv = ((int)0x8908), MaxProgramRe-
sultComponentsNv = ((int)0x8909), MaxProgramGenericAttribsNv =
((int)0x8DA5), MaxProgramGenericResultsNv = ((int)0x8DA6) }
• enum NvHalfFloat { HalfFloatNv = ((int)0x140B) }
• enum NvLightMaxExponent { MaxShininessNv = ((int)0x8504), MaxSpot-
ExponentNv = ((int)0x8505) }
• enum NvMultisampleFilterHint { MultisampleFilterHintNv = ((int)0x8534)
}
• enum NvOcclusionQuery { PixelCounterBitsNv = ((int)0x8864), Curren-
tOcclusionQueryIdNv = ((int)0x8865), PixelCountNv = ((int)0x8866), Pix-
elCountAvailableNv = ((int)0x8867) }
• enum NvPackedDepthStencil { DepthStencilNv = ((int)0x84F9), Un-
signedInt248Nv = ((int)0x84FA) }
• enum NvParameterBufferObject {
MaxProgramParameterBufferBindingsNv = ((int)0x8DA0), MaxPro-
gramParameterBufferSizeNv = ((int)0x8DA1), VertexProgramParamete-
terBufferNv = ((int)0x8DA2), GeometryProgramParameterBufferNv =
((int)0x8DA3),

FragmentProgramParameterBufferNv = ((int)0x8DA4) }
• enum NvPixelDataRange {
WritePixelDataRangeNv = ((int)0x8878), ReadPixelDataRangeNv =
((int)0x8879), WritePixelDataRangeLengthNv = ((int)0x887A), ReadPixel-
DataRangeLengthNv = ((int)0x887B),

WritePixelDataRangePointerNv = ((int)0x887C), ReadPixelDataRange-
PointerNv = ((int)0x887D) }
• enum NvPointSprite { PointSpriteNv = ((int)0x8861), CoordReplaceNv =
((int)0x8862), PointSpriteRModeNv = ((int)0x8863) }

```

- enum **NvPresentVideo** {
FrameNv = ((int)0x8E26), **FieldsNv** = ((int)0x8E27), **CurrentTimeNv** = ((int)0x8E28), **NumFillStreamsNv** = ((int)0x8E29),
PresentTimeNv = ((int)0x8E2A), **PresentDurationNv** = ((int)0x8E2B) }
- enum **NvPrimitiveRestart** { **PrimitiveRestartNv** = ((int)0x8558), **PrimitiveRestartIndexNv** = ((int)0x8559) }
- enum **NvRegisterCombiners** {
None = ((int)0), **Zero** = ((int)0), **Fog** = ((int)0x0B60), **Texture0Arb** = ((int)0x84C0),
Texture1Arb = ((int)0x84C1), **RegisterCombinersNv** = ((int)0x8522), **VariableANv** = ((int)0x8523), **VariableBNv** = ((int)0x8524),
VariableCNv = ((int)0x8525), **VariableDNv** = ((int)0x8526), **VariableENv** = ((int)0x8527), **VariableFNv** = ((int)0x8528),
VariableGNv = ((int)0x8529), **ConstantColor0Nv** = ((int)0x852A), **ConstantColor1Nv** = ((int)0x852B), **PrimaryColorNv** = ((int)0x852C),
SecondaryColorNv = ((int)0x852D), **Spare0Nv** = ((int)0x852E), **Spare1Nv** = ((int)0x852F), **DiscardNv** = ((int)0x8530),
ETimesFNv = ((int)0x8531), **Spare0PlusSecondaryColorNv** = ((int)0x8532), **UnsignedIdentityNv** = ((int)0x8536), **UnsignedInvertNv** = ((int)0x8537),
ExpandNormalNv = ((int)0x8538), **ExpandNegateNv** = ((int)0x8539), **HalfBiasNormalNv** = ((int)0x853A), **HalfBiasNegateNv** = ((int)0x853B),
SignedIdentityNv = ((int)0x853C), **SignedNegateNv** = ((int)0x853D), **ScaleByTwoNv** = ((int)0x853E), **ScaleByFourNv** = ((int)0x853F),
ScaleByOneHalfNv = ((int)0x8540), **BiasByNegativeOneHalfNv** = ((int)0x8541), **CombinerInputNv** = ((int)0x8542), **CombinerMappingNv** = ((int)0x8543),
CombinerComponentUsageNv = ((int)0x8544), **CombinerAbDotProductNv** = ((int)0x8545), **CombinerCdDotProductNv** = ((int)0x8546), **CombinerMuxSumNv** = ((int)0x8547),
CombinerScaleNv = ((int)0x8548), **CombinerBiasNv** = ((int)0x8549), **CombinerAbOutputNv** = ((int)0x854A), **CombinerCdOutputNv** = ((int)0x854B),
CombinerSumOutputNv = ((int)0x854C), **MaxGeneralCombinersNv** = ((int)0x854D), **NumGeneralCombinersNv** = ((int)0x854E), **ColorSumClampNv** = ((int)0x854F),
Combiner0Nv = ((int)0x8550), **Combiner1Nv** = ((int)0x8551), **Combiner2Nv** = ((int)0x8552), **Combiner3Nv** = ((int)0x8553),
Combiner4Nv = ((int)0x8554), **Combiner5Nv** = ((int)0x8555), **Combiner6Nv** = ((int)0x8556), **Combiner7Nv** = ((int)0x8557) }
- enum **NvRegisterCombiners2** { **PerStageConstantsNv** = ((int)0x8535) }
- enum **NvTexgenEmboss** { **EmbossLightNv** = ((int)0x855D), **EmbossConstantNv** = ((int)0x855E), **EmbossMapNv** = ((int)0x855F) }

- enum **NvTexgenReflection** { **NormalMapNv** = ((int)0x8511), **ReflectionMapNv** = ((int)0x8512) }
- enum **NvTextureCompressionVtc**
- enum **NvTextureEnvCombine4** {
Combine4Nv = ((int)0x8503), **Source3RgbNv** = ((int)0x8583),
Source3AlphaNv = ((int)0x858B), **Operand3RgbNv** = ((int)0x8593),
Operand3AlphaNv = ((int)0x859B) }
- enum **NvTextureExpandNormal** { **TextureUnsignedRemapModeNv** = ((int)0x888F) }
- enum **NvTextureRectangle** { **TextureRectangleNv** = ((int)0x84F5), **TextureBindingRectangleNv** = ((int)0x84F6), **ProxyTextureRectangleNv** = ((int)0x84F7), **MaxRectangleTextureSizeNv** = ((int)0x84F8) }
- enum **NvTextureShader** {
OffsetTextureRectangleNv = ((int)0x864C), **OffsetTextureRectangleScaleNv** = ((int)0x864D), **DotProductTextureRectangleNv** = ((int)0x864E), **RgbaUnsignedDotProductMappingNv** = ((int)0x86D9),
UnsignedIntS8S888Nv = ((int)0x86DA), **UnsignedInt8S8S88RevNv** = ((int)0x86DB), **DsdtMagIntensityNv** = ((int)0x86DC), **ShaderConsistentNv** = ((int)0x86DD),
TextureShaderNv = ((int)0x86DE), **ShaderOperationNv** = ((int)0x86DF), **CullModesNv** = ((int)0x86E0), **OffsetTexture2DMatrixNv** = ((int)0x86E1),
OffsetTextureMatrixNv = ((int)0x86E1), **OffsetTexture2DScaleNv** = ((int)0x86E2), **OffsetTextureScaleNv** = ((int)0x86E2), **OffsetTexture2DBiasNv** = ((int)0x86E3),
OffsetTextureBiasNv = ((int)0x86E3), **PreviousTextureInputNv** = ((int)0x86E4), **ConstEyeNv** = ((int)0x86E5), **PassThroughNv** = ((int)0x86E6),
CullFragmentNv = ((int)0x86E7), **OffsetTexture2DNv** = ((int)0x86E8), **DependentArTexture2DNv** = ((int)0x86E9), **DependentGbTexture2DNv** = ((int)0x86EA),
DotProductNv = ((int)0x86EC), **DotProductDepthReplaceNv** = ((int)0x86ED), **DotProductTexture2DNv** = ((int)0x86EE), **DotProductTextureCubeMapNv** = ((int)0x86F0),
DotProductDiffuseCubeMapNv = ((int)0x86F1), **DotProductReflectCubeMapNv** = ((int)0x86F2), **DotProductConstEyeReflectCubeMapNv** = ((int)0x86F3), **HiloNv** = ((int)0x86F4),
DsdtNv = ((int)0x86F5), **DsdtMagNv** = ((int)0x86F6), **DsdtMagVibNv** = ((int)0x86F7), **Hilo16Nv** = ((int)0x86F8),
SignedHiloNv = ((int)0x86F9), **SignedHilo16Nv** = ((int)0x86FA), **SignedRgbaNv** = ((int)0x86FB), **SignedRgba8Nv** = ((int)0x86FC),
SignedRgbNv = ((int)0x86FE), **SignedRgb8Nv** = ((int)0x86FF), **SignedLuminanceNv** = ((int)0x8701), **SignedLuminance8Nv** = ((int)0x8702),

SignedLuminanceAlphaNv = ((int)0x8703), **SignedLuminance8Alpha8Nv** = ((int)0x8704), **SignedAlphaNv** = ((int)0x8705), **SignedAlpha8Nv** = ((int)0x8706),
SignedIntensityNv = ((int)0x8707), **SignedIntensity8Nv** = ((int)0x8708), **Dsdt8Nv** = ((int)0x8709), **Dsdt8Mag8Nv** = ((int)0x870A),
Dsdt8Mag8Intensity8Nv = ((int)0x870B), **SignedRgbUnsignedAlphaNv** = ((int)0x870C), **SignedRgb8UnsignedAlpha8Nv** = ((int)0x870D), **HiScaleNv** = ((int)0x870E),
LoScaleNv = ((int)0x870F), **DsScaleNv** = ((int)0x8710), **DtScaleNv** = ((int)0x8711), **MagnitudeScaleNv** = ((int)0x8712),
VibranceScaleNv = ((int)0x8713), **HiBiasNv** = ((int)0x8714), **LoBiasNv** = ((int)0x8715), **DsBiasNv** = ((int)0x8716),
DtBiasNv = ((int)0x8717), **MagnitudeBiasNv** = ((int)0x8718), **VibranceBiasNv** = ((int)0x8719), **TextureBorderValuesNv** = ((int)0x871A),
TextureHiSizeNv = ((int)0x871B), **TextureLoSizeNv** = ((int)0x871C), **TextureDsSizeNv** = ((int)0x871D), **TextureDtSizeNv** = ((int)0x871E),
TextureMagSizeNv = ((int)0x871F) }
• enum **NvTextureShader2** { **DotProductTexture3DNv** = ((int)0x86EF) }
• enum **NvTextureShader3** {
OffsetProjectiveTexture2DNv = ((int)0x8850), **OffsetProjectiveTexture2DScaleNv** = ((int)0x8851), **OffsetProjectiveTextureRectangleNv** = ((int)0x8852), **OffsetProjectiveTextureRectangleScaleNv** = ((int)0x8853),
OffsetHiloTexture2DNv = ((int)0x8854), **OffsetHiloTextureRectangleNv** = ((int)0x8855), **OffsetHiloProjectiveTexture2DNv** = ((int)0x8856), **OffsetHiloProjectiveTextureRectangleNv** = ((int)0x8857),
DependentHiloTexture2DNv = ((int)0x8858), **DependentRgbTexture3DNv** = ((int)0x8859), **DependentRgbTextureCubeMapNv** = ((int)0x885A), **DotProductPassThroughNv** = ((int)0x885B),
DotProductTexture1DNv = ((int)0x885C), **DotProductAffineDepthReplaceNv** = ((int)0x885D), **Hilo8Nv** = ((int)0x885E), **SignedHilo8Nv** = ((int)0x885F),
ForceBlueToOneNv = ((int)0x8860) }
• enum **NvTransformFeedback** {
BackPrimaryColorNv = ((int)0x8C77), **BackSecondaryColorNv** = ((int)0x8C78), **TextureCoordNv** = ((int)0x8C79), **ClipDistanceNv** = ((int)0x8C7A),
VertexIdNv = ((int)0x8C7B), **PrimitiveIdNv** = ((int)0x8C7C), **GenericAttribNv** = ((int)0x8C7D), **TransformFeedbackAttribsNv** = ((int)0x8C7E),
TransformFeedbackBufferModeNv = ((int)0x8C7F), **MaxTransformFeedbackSeparateComponentsNv** = ((int)0x8C80), **ActiveVaryingsNv** = ((int)0x8C81), **ActiveVaryingMaxLengthNv** = ((int)0x8C82),

```

TransformFeedbackVaryingsNv = ((int)0x8C83), TransformFeedback-
BufferStartNv = ((int)0x8C84), TransformFeedbackBufferSizeNv =
((int)0x8C85), TransformFeedbackRecordNv = ((int)0x8C86),

PrimitivesGeneratedNv = ((int)0x8C87), TransformFeedbackPrimi-
tivesWrittenNv = ((int)0x8C88), RasterizerDiscardNv = ((int)0x8C89),
MaxTransformFeedbackInterleavedAttribsNv = ((int)0x8C8A),

MaxTransformFeedbackSeparateAttribsNv = ((int)0x8C8B), InterleavedAt-
tribsNv = ((int)0x8C8C), SeparateAttribsNv = ((int)0x8C8D), Transform-
FeedbackBufferNv = ((int)0x8C8E),

TransformFeedbackBufferBindingNv = ((int)0x8C8F) }
• enum NvTransformFeedback2 { TransformFeedbackNv = ((int)0x8E22),
TransformFeedbackBufferPausedNv = ((int)0x8E23), TransformFeed-
backBufferActiveNv = ((int)0x8E24), TransformFeedbackBindingNv =
((int)0x8E25) }
• enum NvVertexArrayRange {
VertexArrayRangeNv = ((int)0x851D), VertexArrayRangeLengthNv =
((int)0x851E), VertexArrayRangeValidNv = ((int)0x851F), MaxVertexAr-
rayRangeElementNv = ((int)0x8520),

VertexArrayRangePointerNv = ((int)0x8521) }
• enum NvVertexArrayRange2 { VertexArrayRangeWithoutFlushNv =
((int)0x8533) }
• enum NvVertexProgram {
VertexProgramNv = ((int)0x8620), VertexStateProgramNv = ((int)0x8621),
AttribArraySizeNv = ((int)0x8623), AttribArrayStrideNv = ((int)0x8624),

AttribArrayTypeNv = ((int)0x8625), CurrentAttribNv = ((int)0x8626), Pro-
gramLengthNv = ((int)0x8627), ProgramStringNv = ((int)0x8628),

ModelviewProjectionNv = ((int)0x8629), IdentityNv = ((int)0x862A), In-
verseNv = ((int)0x862B), TransposeNv = ((int)0x862C),

InverseTransposeNv = ((int)0x862D), MaxTrackMatrixStackDepthNv
= ((int)0x862E), MaxTrackMatricesNv = ((int)0x862F), Matrix0Nv =
((int)0x8630),

Matrix1Nv = ((int)0x8631), Matrix2Nv = ((int)0x8632), Matrix3Nv =
((int)0x8633), Matrix4Nv = ((int)0x8634),

Matrix5Nv = ((int)0x8635), Matrix6Nv = ((int)0x8636), Matrix7Nv =
((int)0x8637), CurrentMatrixStackDepthNv = ((int)0x8640),

CurrentMatrixNv = ((int)0x8641), VertexProgramPointSizeNv =
((int)0x8642), VertexProgramTwoSideNv = ((int)0x8643), ProgramPa-
rameterNv = ((int)0x8644),

AttribArrayPointerNv = ((int)0x8645), ProgramTargetNv = ((int)0x8646),
ProgramResidentNv = ((int)0x8647), TrackMatrixNv = ((int)0x8648),

```

TrackMatrixTransformNv = ((int)0x8649), **VertexProgramBindingNv** = ((int)0x864A), **ProgramErrorPositionNv** = ((int)0x864B), **VertexAttribArray0Nv** = ((int)0x8650),

VertexAttribArray1Nv = ((int)0x8651), **VertexAttribArray2Nv** = ((int)0x8652), **VertexAttribArray3Nv** = ((int)0x8653), **VertexAttribArray4Nv** = ((int)0x8654),

VertexAttribArray5Nv = ((int)0x8655), **VertexAttribArray6Nv** = ((int)0x8656), **VertexAttribArray7Nv** = ((int)0x8657), **VertexAttribArray8Nv** = ((int)0x8658),

VertexAttribArray9Nv = ((int)0x8659), **VertexAttribArray10Nv** = ((int)0x865A), **VertexAttribArray11Nv** = ((int)0x865B), **VertexAttribArray12Nv** = ((int)0x865C),

VertexAttribArray13Nv = ((int)0x865D), **VertexAttribArray14Nv** = ((int)0x865E), **VertexAttribArray15Nv** = ((int)0x865F), **Map1VertexAttrib04Nv** = ((int)0x8660),

Map1VertexAttrib14Nv = ((int)0x8661), **Map1VertexAttrib24Nv** = ((int)0x8662), **Map1VertexAttrib34Nv** = ((int)0x8663), **Map1VertexAttrib44Nv** = ((int)0x8664),

Map1VertexAttrib54Nv = ((int)0x8665), **Map1VertexAttrib64Nv** = ((int)0x8666), **Map1VertexAttrib74Nv** = ((int)0x8667), **Map1VertexAttrib84Nv** = ((int)0x8668),

Map1VertexAttrib94Nv = ((int)0x8669), **Map1VertexAttrib104Nv** = ((int)0x866A), **Map1VertexAttrib114Nv** = ((int)0x866B), **Map1VertexAttrib124Nv** = ((int)0x866C),

Map1VertexAttrib134Nv = ((int)0x866D), **Map1VertexAttrib144Nv** = ((int)0x866E), **Map1VertexAttrib154Nv** = ((int)0x866F), **Map2VertexAttrib04Nv** = ((int)0x8670),

Map2VertexAttrib14Nv = ((int)0x8671), **Map2VertexAttrib24Nv** = ((int)0x8672), **Map2VertexAttrib34Nv** = ((int)0x8673), **Map2VertexAttrib44Nv** = ((int)0x8674),

Map2VertexAttrib54Nv = ((int)0x8675), **Map2VertexAttrib64Nv** = ((int)0x8676), **Map2VertexAttrib74Nv** = ((int)0x8677), **Map2VertexAttrib84Nv** = ((int)0x8678),

Map2VertexAttrib94Nv = ((int)0x8679), **Map2VertexAttrib104Nv** = ((int)0x867A), **Map2VertexAttrib114Nv** = ((int)0x867B), **Map2VertexAttrib124Nv** = ((int)0x867C),

Map2VertexAttrib134Nv = ((int)0x867D), **Map2VertexAttrib144Nv** = ((int)0x867E), **Map2VertexAttrib154Nv** = ((int)0x867F) }

- enum **NvVertexProgram11**
- enum **NvVertexProgram2**
- enum **NvVertexProgram2Option** { **MaxProgramExecInstructionsNv** = ((int)0x88F4), **MaxProgramCallDepthNv** = ((int)0x88F5) }

- enum **NvVertexProgram3** { **MaxVertexTextureImageUnitsArb** = ((int)0x8B4C) }
- enum **NvVertexProgram4** { **VertexAttribArrayIntegerNv** = ((int)0x88FD) }
- enum **OesReadFormat** { **ImplementationColorReadTypeOes** = ((int)0x8B9A), **ImplementationColorReadFormatOes** = ((int)0x8B9B) }
- enum **OmlInterlace** { **InterlaceOml** = ((int)0x8980), **InterlaceReadOml** = ((int)0x8981) }
- enum **OmlResample** {
PackResampleOml = ((int)0x8984), **UnpackResampleOml** = ((int)0x8985),
ResampleReplicateOml = ((int)0x8986), **ResampleZeroFillOml** = ((int)0x8987),
ResampleAverageOml = ((int)0x8988), **ResampleDecimateOml** = ((int)0x8989) }
- enum **OmlSubsample** { **FormatSubsample2424Oml** = ((int)0x8982), **FormatSubsample244244Oml** = ((int)0x8983) }
- enum **PgiMiscHints** {
PreferDoublebufferHintPgi = ((int)0x1A1F8), **ConserveMemoryHintPgi** = ((int)0x1A1FD), **ReclaimMemoryHintPgi** = ((int)0x1A1FE), **NativeGraphicsHandlePgi** = ((int)0x1A202),
NativeGraphicsBeginHintPgi = ((int)0x1A203), **NativeGraphicsEndHintPgi** = ((int)0x1A204), **AlwaysFastHintPgi** = ((int)0x1A20C), **AlwaysSoftHintPgi** = ((int)0x1A20D),
AllowDrawObjHintPgi = ((int)0x1A20E), **AllowDrawWinHintPgi** = ((int)0x1A20F), **AllowDrawFrgHintPgi** = ((int)0x1A210), **AllowDrawMemHintPgi** = ((int)0x1A211),
StrictDepthfuncHintPgi = ((int)0x1A216), **StrictLightingHintPgi** = ((int)0x1A217), **StrictScissorHintPgi** = ((int)0x1A218), **FullStippleHintPgi** = ((int)0x1A219),
ClipNearHintPgi = ((int)0x1A220), **ClipFarHintPgi** = ((int)0x1A221), **WideLineHintPgi** = ((int)0x1A222), **BackNormalsHintPgi** = ((int)0x1A223) }
- enum **PgiVertexHints** {
Vertex23BitPgi = ((int)0x00000004), **Vertex4BitPgi** = ((int)0x00000008), **Color3BitPgi** = ((int)0x00010000), **Color4BitPgi** = ((int)0x00020000),
EdgeflagBitPgi = ((int)0x00040000), **IndexBitPgi** = ((int)0x00080000), **MatAmbientBitPgi** = ((int)0x00100000), **MatAmbientAndDiffuseBitPgi** = ((int)0x00200000),
MatDiffuseBitPgi = ((int)0x00400000), **MatEmissionBitPgi** = ((int)0x00800000), **MatColorIndexesBitPgi** = ((int)0x01000000), **MatShininessBitPgi** = ((int)0x02000000),
MatSpecularBitPgi = ((int)0x04000000), **NormalBitPgi** = ((int)0x08000000), **Texcoord1BitPgi** = ((int)0x10000000), **VertexDataHintPgi** = ((int)0x1A22A),

```

VertexConsistentHintPgi = ((int)0x1A22B), MaterialSideHintPgi =
((int)0x1A22C), MaxVertexHintPgi = ((int)0x1A22D), Texcoord2BitPgi
= ((int)0x20000000),
Texcoord3BitPgi = ((int)0x40000000), Texcoord4BitPgi =
unchecked((int)0x80000000) }
• enum PixelCopyType { Color = ((int)0x1800), Depth = ((int)0x1801), Stencil
= ((int)0x1802) }
• enum PixelFormat {
ColorIndex = ((int)0x1900), StencilIndex = ((int)0x1901), DepthComponent
= ((int)0x1902), Red = ((int)0x1903),
Green = ((int)0x1904), Blue = ((int)0x1905), Alpha = ((int)0x1906), Rgb =
((int)0x1907),
Rgba = ((int)0x1908), Luminance = ((int)0x1909), LuminanceAlpha =
((int)0x190A), AbgrExt = ((int)0x8000),
CmykExt = ((int)0x800C), CmykaExt = ((int)0x800D), Bgr = ((int)0x80E0),
Bgra = ((int)0x80E1),
Ycrb422Sgix = ((int)0x81BB), Ycrb444Sgix = ((int)0x81BC), Rg =
((int)0x8227), RgInteger = ((int)0x8228),
DepthStencil = ((int)0x84F9), RedInteger = ((int)0x8D94), GreenInteger =
((int)0x8D95), BlueInteger = ((int)0x8D96),
AlphaInteger = ((int)0x8D97), RgbInteger = ((int)0x8D98), RgbaInteger =
((int)0x8D99), BgrInteger = ((int)0x8D9A),
BgraInteger = ((int)0x8D9B) }
• enum PixelInternalFormat {
DepthComponent = ((int)0x1902), Alpha = ((int)0x1906), Rgb =
((int)0x1907), Rgba = ((int)0x1908),
Luminance = ((int)0x1909), LuminanceAlpha = ((int)0x190A), R3G3B2 =
((int)0x2A10), Alpha4 = ((int)0x803B),
Alpha8 = ((int)0x803C), Alpha12 = ((int)0x803D), Alpha16 = ((int)0x803E),
Luminance4 = ((int)0x803F),
Luminance8 = ((int)0x8040), Luminance12 = ((int)0x8041), Luminance16 =
((int)0x8042), Luminance4Alpha4 = ((int)0x8043),
Luminance6Alpha2 = ((int)0x8044), Luminance8Alpha8 = ((int)0x8045),
Luminance12Alpha4 = ((int)0x8046), Luminance12Alpha12 = ((int)0x8047),
Luminance16Alpha16 = ((int)0x8048), Intensity = ((int)0x8049), Intensity4 =
((int)0x804A), Intensity8 = ((int)0x804B),
Intensity12 = ((int)0x804C), Intensity16 = ((int)0x804D), Rgb2Ext =
((int)0x804E), Rgb4 = ((int)0x804F),
Rgb5 = ((int)0x8050), Rgb8 = ((int)0x8051), Rgb10 = ((int)0x8052), Rgb12 =
((int)0x8053),

```


Rgb16 = ((int)0x8054), **Rgba2** = ((int)0x8055), **Rgba4** = ((int)0x8056),
Rgb5A1 = ((int)0x8057),
Rgba8 = ((int)0x8058), **Rgb10A2** = ((int)0x8059), **Rgba12** = ((int)0x805A),
Rgba16 = ((int)0x805B),
DualAlpha4Sgis = ((int)0x8110), **DualAlpha8Sgis** = ((int)0x8111), **DualAlpha12Sgis** = ((int)0x8112), **DualAlpha16Sgis** = ((int)0x8113),
DualLuminance4Sgis = ((int)0x8114), **DualLuminance8Sgis** = ((int)0x8115),
DualLuminance12Sgis = ((int)0x8116), **DualLuminance16Sgis** = ((int)0x8117),
DualIntensity4Sgis = ((int)0x8118), **DualIntensity8Sgis** = ((int)0x8119), **DualIntensity12Sgis** = ((int)0x811A), **DualIntensity16Sgis** = ((int)0x811B),
DualLuminanceAlpha4Sgis = ((int)0x811C), **DualLuminanceAlpha8Sgis** = ((int)0x811D), **QuadAlpha4Sgis** = ((int)0x811E), **QuadAlpha8Sgis** = ((int)0x811F),
QuadLuminance4Sgis = ((int)0x8120), **QuadLuminance8Sgis** = ((int)0x8121), **QuadIntensity4Sgis** = ((int)0x8122), **QuadIntensity8Sgis** = ((int)0x8123),
DepthComponent16 = ((int)0x81a5), **DepthComponent16Sgix** = ((int)0x81A5), **DepthComponent24** = ((int)0x81a6), **DepthComponent24Sgix** = ((int)0x81A6),
DepthComponent32 = ((int)0x81a7), **DepthComponent32Sgix** = ((int)0x81A7), **CompressedRed** = ((int)0x8225), **CompressedRg** = ((int)0x8226),
R8 = ((int)0x8229), **R16** = ((int)0x822A), **Rg8** = ((int)0x822B), **Rg16** = ((int)0x822C),
R16f = ((int)0x822D), **R32f** = ((int)0x822E), **Rg16f** = ((int)0x822F), **Rg32f** = ((int)0x8230),
R8i = ((int)0x8231), **R8ui** = ((int)0x8232), **R16i** = ((int)0x8233), **R16ui** = ((int)0x8234),
R32i = ((int)0x8235), **R32ui** = ((int)0x8236), **Rg8i** = ((int)0x8237), **Rg8ui** = ((int)0x8238),
Rg16i = ((int)0x8239), **Rg16ui** = ((int)0x823A), **Rg32i** = ((int)0x823B), **Rg32ui** = ((int)0x823C),
CompressedRgbS3tcDxt1Ext = ((int)0x83F0), **CompressedRgbaS3tcDxt1Ext** = ((int)0x83F1), **CompressedRgbaS3tcDxt3Ext** = ((int)0x83F2), **CompressedRgbaS3tcDxt5Ext** = ((int)0x83F3),
CompressedAlpha = ((int)0x84E9), **CompressedLuminance** = ((int)0x84EA), **CompressedLuminanceAlpha** = ((int)0x84EB), **CompressedIntensity** = ((int)0x84EC),
CompressedRgb = ((int)0x84ED), **CompressedRgba** = ((int)0x84EE), **DepthStencil** = ((int)0x84F9), **Rgba32f** = ((int)0x8814),

Rgb32f = ((int)0x8815), **Rgba16f** = ((int)0x881A), **Rgb16f** = ((int)0x881B),
Depth24Stencil8 = ((int)0x88F0),

R11fG11fB10f = ((int)0x8C3A), **Rgb9E5** = ((int)0x8C3D), **Srgb** =
 ((int)0x8C40), **Srgb8** = ((int)0x8C41),

SrgbAlpha = ((int)0x8C42), **Srgb8Alpha8** = ((int)0x8C43), **SluminanceAlpha**
 = ((int)0x8C44), **Sluminance8Alpha8** = ((int)0x8C45),

Sluminance = ((int)0x8C46), **Sluminance8** = ((int)0x8C47), **CompressedSrgb**
 = ((int)0x8C48), **CompressedSrgbAlpha** = ((int)0x8C49),

CompressedSluminance = ((int)0x8C4A), **CompressedSluminanceAlpha** =
 ((int)0x8C4B), **CompressedSrgbS3tcDxt1Ext** = ((int)0x8C4C), **Compressed-**
SrgbAlphaS3tcDxt1Ext = ((int)0x8C4D),

CompressedSrgbAlphaS3tcDxt3Ext = ((int)0x8C4E), **CompressedSrgbAl-**
phaS3tcDxt5Ext = ((int)0x8C4F), **DepthComponent32f** = ((int)0x8CAC),
Depth32fStencil8 = ((int)0x8CAD),

Rgba32ui = ((int)0x8D70), **Rgb32ui** = ((int)0x8D71), **Rgba16ui** =
 ((int)0x8D76), **Rgb16ui** = ((int)0x8D77),

Rgba8ui = ((int)0x8D7C), **Rgb8ui** = ((int)0x8D7D), **Rgba32i** = ((int)0x8D82),
Rgb32i = ((int)0x8D83),

Rgba16i = ((int)0x8D88), **Rgb16i** = ((int)0x8D89), **Rgba8i** = ((int)0x8D8E),
Rgb8i = ((int)0x8D8F),

Float32UnsignedInt248Rev = ((int)0x8DAD), **CompressedRedRgtc1** =
 ((int)0x8DBB), **CompressedSignedRedRgtc1** = ((int)0x8DBC), **Compressed-**
RgRgtc2 = ((int)0x8DBD),

CompressedSignedRgRgtc2 = ((int)0x8DBE), **One** = ((int)1), **Two** = ((int)2),
Three = ((int)3),

Four = ((int)4) }

- enum **PixelFormat** {

PixelFormatI = ((int)0x0C70), **PixelFormatSToS** = ((int)0x0C71), **PixelFormatI-**
ToR = ((int)0x0C72), **PixelFormatIToG** = ((int)0x0C73),

PixelFormatIToB = ((int)0x0C74), **PixelFormatIToA** = ((int)0x0C75), **Pixel-**
elMapRToR = ((int)0x0C76), **PixelFormatGToG** = ((int)0x0C77),

PixelFormatBToB = ((int)0x0C78), **PixelFormatAToA** = ((int)0x0C79) }

- enum **PixelStoreParameter** {

UnpackSwapBytes = ((int)0x0CF0), **UnpackLsbFirst** = ((int)0x0CF1), **Un-**
packRowLength = ((int)0x0CF2), **UnpackSkipRows** = ((int)0x0CF3),

UnpackSkipPixels = ((int)0x0CF4), **UnpackAlignment** = ((int)0x0CF5),
PackSwapBytes = ((int)0x0D00), **PackLsbFirst** = ((int)0x0D01),

PackRowLength = ((int)0x0D02), **PackSkipRows** = ((int)0x0D03), **PackSkip-**
Pixels = ((int)0x0D04), **PackAlignment** = ((int)0x0D05),

```

PackSkipImages = ((int)0x806B), PackSkipImagesExt = ((int)0x806B),
PackImageHeight = ((int)0x806C), PackImageHeightExt = ((int)0x806C),
UnpackSkipImages = ((int)0x806D), UnpackSkipImagesExt = ((int)0x806D),
UnpackImageHeight = ((int)0x806E), UnpackImageHeightExt =
((int)0x806E),
PackSkipVolumesSgis = ((int)0x8130), PackImageDepthSgis = ((int)0x8131),
UnpackSkipVolumesSgis = ((int)0x8132), UnpackImageDepthSgis =
((int)0x8133),
PixelTileWidthSgix = ((int)0x8140), PixelTileHeightSgix = ((int)0x8141),
PixelTileGridWidthSgix = ((int)0x8142), PixelTileGridHeightSgix =
((int)0x8143),
PixelTileGridDepthSgix = ((int)0x8144), PixelTileCacheSizeSgix =
((int)0x8145), PackResampleSgix = ((int)0x842C), UnpackResampleS-
gix = ((int)0x842D),
PackSubsampleRateSgix = ((int)0x85A0), UnpackSubsampleRateSgix =
((int)0x85A1) }
• enum PixelStoreResampleMode { ResampleReplicateSgix = ((int)0x842E),
ResampleZeroFillSgix = ((int)0x842F), ResampleDecimateSgix =
((int)0x8430) }
• enum PixelStoreSubsampleRate { PixelSubsample4444Sgix = ((int)0x85A2),
PixelSubsample2424Sgix = ((int)0x85A3), PixelSubsample4242Sgix =
((int)0x85A4) }
• enum PixelTexGenMode {
None = ((int)0), Rgb = ((int)0x1907), Rgba = ((int)0x1908), Luminance =
((int)0x1909),
LuminanceAlpha = ((int)0x190A), PixelTexGenAlphaReplaceSgix =
((int)0x8187), PixelTexGenAlphaNoReplaceSgix = ((int)0x8188), PixelTex-
GenAlphaLsSgix = ((int)0x8189),
PixelTexGenAlphaMsSgix = ((int)0x818A) }
• enum PixelTexGenParameterNameSgis { PixelFragmentRgbSourceSgis =
((int)0x8354), PixelFragmentAlphaSourceSgis = ((int)0x8355) }
• enum PixelTransferParameter {
MapColor = ((int)0x0D10), MapStencil = ((int)0x0D11), IndexShift =
((int)0x0D12), IndexOffset = ((int)0x0D13),
RedScale = ((int)0x0D14), RedBias = ((int)0x0D15), GreenScale =
((int)0x0D18), GreenBias = ((int)0x0D19),
BlueScale = ((int)0x0D1A), BlueBias = ((int)0x0D1B), AlphaScale =
((int)0x0D1C), AlphaBias = ((int)0x0D1D),
DepthScale = ((int)0x0D1E), DepthBias = ((int)0x0D1F), PostConvo-
lutionRedScaleExt = ((int)0x801C), PostConvolutionGreenScaleExt =
((int)0x801D),

```

```

PostConvolutionBlueScaleExt = ((int)0x801E), PostConvolutionAlphaScaleExt = ((int)0x801F), PostConvolutionRedBiasExt = ((int)0x8020), PostConvolutionGreenBiasExt = ((int)0x8021),

PostConvolutionBlueBiasExt = ((int)0x8022), PostConvolutionAlphaBiasExt = ((int)0x8023), PostColorMatrixRedScaleSgi = ((int)0x80B4), PostColorMatrixGreenScaleSgi = ((int)0x80B5),

PostColorMatrixBlueScaleSgi = ((int)0x80B6), PostColorMatrixAlphaScaleSgi = ((int)0x80B7), PostColorMatrixRedBiasSgi = ((int)0x80B8), PostColorMatrixGreenBiasSgi = ((int)0x80B9),

PostColorMatrixBlueBiasSgi = ((int)0x80BA), PostColorMatrixAlphaBiasSgi = ((int)0x80BB) }

• enum PixelFormat {

    Byte = ((int)0x1400), UnsignedByte = ((int)0x1401), Short = ((int)0x1402), UnsignedShort = ((int)0x1403),

    Int = ((int)0x1404), UnsignedInt = ((int)0x1405), Float = ((int)0x1406), HalfFloat = ((int)0x140B),

    Bitmap = ((int)0x1A00), UnsignedByte332 = ((int)0x8032), UnsignedByte332Ext = ((int)0x8032), UnsignedShort4444 = ((int)0x8033),

    UnsignedShort4444Ext = ((int)0x8033), UnsignedShort5551 = ((int)0x8034), UnsignedShort5551Ext = ((int)0x8034), UnsignedInt8888 = ((int)0x8035),

    UnsignedInt8888Ext = ((int)0x8035), UnsignedInt1010102 = ((int)0x8036), UnsignedInt1010102Ext = ((int)0x8036), UnsignedByte233Reversed = ((int)0x8362),

    UnsignedShort565 = ((int)0x8363), UnsignedShort565Reversed = ((int)0x8364), UnsignedShort4444Reversed = ((int)0x8365), UnsignedShort1555Reversed = ((int)0x8366),

    UnsignedInt8888Reversed = ((int)0x8367), UnsignedInt2101010Reversed = ((int)0x8368), UnsignedInt248 = ((int)0x84FA), UnsignedInt10F11F11FRev = ((int)0x8C3B),

    UnsignedInt5999Rev = ((int)0x8C3E), Float32UnsignedInt248Rev = ((int)0x8DAD) }

• enum PointParameterName {

    PointSizeMin = ((int)0x8126), PointSizeMax = ((int)0x8127), PointFadeThresholdSize = ((int)0x8128), PointDistanceAttenuation = ((int)0x8129),

    PointSpriteCoordOrigin = ((int)0x8CA0) }

• enum PointParameterNameSgis { PointSizeMinSgis = ((int)0x8126), PointSizeMaxSgis = ((int)0x8127), PointFadeThresholdSizeSgis = ((int)0x8128), DistanceAttenuationSgis = ((int)0x8129) }

• enum PointSpriteCoordOriginParameter { LowerLeft = ((int)0x8CA1), UpperLeft = ((int)0x8CA2) }

```

- enum **PolygonMode** { **Point** = ((int)0x1B00), **Line** = ((int)0x1B01), **Fill** = ((int)0x1B02) }
- enum **ProgramParameter** {
 - ActiveUniformBlockMaxNameLength** = ((int)0x8A35), **ActiveUniformBlocks** = ((int)0x8A36), **DeleteStatus** = ((int)0x8B80), **LinkStatus** = ((int)0x8B82),
 - ValidateStatus** = ((int)0x8B83), **InfoLogLength** = ((int)0x8B84), **AttachedShaders** = ((int)0x8B85), **ActiveUniforms** = ((int)0x8B86),
 - ActiveUniformMaxLength** = ((int)0x8B87), **ActiveAttributes** = ((int)0x8B89), **ActiveAttributeMaxLength** = ((int)0x8B8A), **TransformFeedbackVaryingMaxLength** = ((int)0x8C76),
 - TransformFeedbackBufferMode** = ((int)0x8C7F), **TransformFeedbackVaryings** = ((int)0x8C83), **GeometryVerticesOut** = ((int)0x8DDA), **GeometryInputType** = ((int)0x8DDB),
 - GeometryOutputType** = ((int)0x8DDC) }
- enum **ProvokingVertexMode** { **FirstVertexConvention** = ((int)0x8E4D), **LastVertexConvention** = ((int)0x8E4E) }
- enum **QueryTarget** { **SamplesPassed** = ((int)0x8914), **PrimitivesGenerated** = ((int)0x8C87), **TransformFeedbackPrimitivesWritten** = ((int)0x8C88) }
- enum **ReadBufferMode** {
 - FrontLeft** = ((int)0x0400), **FrontRight** = ((int)0x0401), **BackLeft** = ((int)0x0402), **BackRight** = ((int)0x0403),
 - Front** = ((int)0x0404), **Back** = ((int)0x0405), **Left** = ((int)0x0406), **Right** = ((int)0x0407),
 - Aux0** = ((int)0x0409), **Aux1** = ((int)0x040A), **Aux2** = ((int)0x040B), **Aux3** = ((int)0x040C),
 - ColorAttachment0** = ((int)0x8CE0), **ColorAttachment1** = ((int)0x8CE1), **ColorAttachment2** = ((int)0x8CE2), **ColorAttachment3** = ((int)0x8CE3),
 - ColorAttachment4** = ((int)0x8CE4), **ColorAttachment5** = ((int)0x8CE5), **ColorAttachment6** = ((int)0x8CE6), **ColorAttachment7** = ((int)0x8CE7),
 - ColorAttachment8** = ((int)0x8CE8), **ColorAttachment9** = ((int)0x8CE9), **ColorAttachment10** = ((int)0x8CEA), **ColorAttachment11** = ((int)0x8CEB),
 - ColorAttachment12** = ((int)0x8CEC), **ColorAttachment13** = ((int)0x8CED), **ColorAttachment14** = ((int)0x8CEE), **ColorAttachment15** = ((int)0x8CEF) }
- enum **RenderbufferParameterName** {
 - RenderbufferSamples** = ((int)0x8CAB), **RenderbufferWidth** = ((int)0x8D42), **RenderbufferWidthExt** = ((int)0x8D42), **RenderbufferHeight** = ((int)0x8D43),
 - RenderbufferHeightExt** = ((int)0x8D43), **RenderbufferInternalFormat** = ((int)0x8D44), **RenderbufferInternalFormatExt** = ((int)0x8D44), **RenderbufferRedSize** = ((int)0x8D50),

RenderbufferRedSizeExt = ((int)0x8D50), **RenderbufferGreenSize** = ((int)0x8D51), **RenderbufferGreenSizeExt** = ((int)0x8D51), **RenderbufferBlueSize** = ((int)0x8D52),

RenderbufferBlueSizeExt = ((int)0x8D52), **RenderbufferAlphaSize** = ((int)0x8D53), **RenderbufferAlphaSizeExt** = ((int)0x8D53), **RenderbufferDepthSize** = ((int)0x8D54),

RenderbufferDepthSizeExt = ((int)0x8D54), **RenderbufferStencilSize** = ((int)0x8D55), **RenderbufferStencilSizeExt** = ((int)0x8D55) }

• enum **RenderbufferStorage** {

R3G3B2 = ((int)0x2A10), **Alpha4** = ((int)0x803B), **Alpha8** = ((int)0x803C), **Alpha12** = ((int)0x803D),

Alpha16 = ((int)0x803E), **Rgb4** = ((int)0x804F), **Rgb5** = ((int)0x8050), **Rgb8** = ((int)0x8051),

Rgb10 = ((int)0x8052), **Rgb12** = ((int)0x8053), **Rgb16** = ((int)0x8054), **Rgba2** = ((int)0x8055),

Rgba4 = ((int)0x8056), **Rgba8** = ((int)0x8058), **Rgb10A2** = ((int)0x8059), **Rgba12** = ((int)0x805A),

Rgba16 = ((int)0x805B), **DepthComponent16** = ((int)0x81a5), **DepthComponent24** = ((int)0x81a6), **DepthComponent32** = ((int)0x81a7),

R8 = ((int)0x8229), **R16** = ((int)0x822A), **Rg8** = ((int)0x822B), **Rg16** = ((int)0x822C),

R16f = ((int)0x822D), **R32f** = ((int)0x822E), **Rg16f** = ((int)0x822F), **Rg32f** = ((int)0x8230),

R8i = ((int)0x8231), **R8ui** = ((int)0x8232), **R16i** = ((int)0x8233), **R16ui** = ((int)0x8234),

R32i = ((int)0x8235), **R32ui** = ((int)0x8236), **Rg8i** = ((int)0x8237), **Rg8ui** = ((int)0x8238),

Rg16i = ((int)0x8239), **Rg16ui** = ((int)0x823A), **Rg32i** = ((int)0x823B), **Rg32ui** = ((int)0x823C),

Rgba32f = ((int)0x8814), **Rgb32f** = ((int)0x8815), **Rgba16f** = ((int)0x881A), **Rgb16f** = ((int)0x881B),

Depth24Stencil8 = ((int)0x88F0), **R11fG11fB10f** = ((int)0x8C3A), **Rgb9E5** = ((int)0x8C3D), **Srgb8** = ((int)0x8C41),

Srgb8Alpha8 = ((int)0x8C43), **DepthComponent32f** = ((int)0x8CAC), **Depth32fStencil8** = ((int)0x8CAD), **StencilIndex1** = ((int)0x8D46),

StencilIndex1Ext = ((int)0x8D46), **StencilIndex4** = ((int)0x8D47), **StencilIndex4Ext** = ((int)0x8D47), **StencilIndex8** = ((int)0x8D48),

StencilIndex8Ext = ((int)0x8D48), **StencilIndex16** = ((int)0x8D49), **StencilIndex16Ext** = ((int)0x8D49), **Rgba32ui** = ((int)0x8D70),

```

Rgb32ui = ((int)0x8D71), Rgba16ui = ((int)0x8D76), Rgb16ui =
((int)0x8D77), Rgba8ui = ((int)0x8D7C),

Rgb8ui = ((int)0x8D7D), Rgba32i = ((int)0x8D82), Rgb32i = ((int)0x8D83),
Rgba16i = ((int)0x8D88),

Rgb16i = ((int)0x8D89), Rgba8i = ((int)0x8D8E), Rgb8i = ((int)0x8D8F) }
• enum RenderbufferTarget { Renderbuffer = ((int)0x8D41), RenderbufferExt =
((int)0x8D41) }
• enum RenderingMode { Render = ((int)0x1C00), Feedback = ((int)0x1C01),
Select = ((int)0x1C02) }
• enum RendScreenCoordinates { ScreenCoordinatesRend = ((int)0x8490),
InvertedScreenWRend = ((int)0x8491) }
• enum S3S3tc { RgbS3tc = ((int)0x83A0), Rgb4S3tc = ((int)0x83A1), Rg-
baS3tc = ((int)0x83A2), Rgba4S3tc = ((int)0x83A3) }
• enum SamplePatternSgis {

Gl1PassSgis = ((int)0x80A1), Gl2Pass0Sgis = ((int)0x80A2), Gl2Pass1Sgis =
((int)0x80A3), Gl4Pass0Sgis = ((int)0x80A4),

Gl4Pass1Sgis = ((int)0x80A5), Gl4Pass2Sgis = ((int)0x80A6), Gl4Pass3Sgis =
((int)0x80A7) }
• enum SeparableTarget { Separable2D = ((int)0x8012) }
• enum SeparableTargetExt { Separable2DExt = ((int)0x8012) }
• enum SgiColorMatrix {

ColorMatrixSgi = ((int)0x80B1), ColorMatrixStackDepthSgi =
((int)0x80B2), MaxColorMatrixStackDepthSgi = ((int)0x80B3), Post-
ColorMatrixRedScaleSgi = ((int)0x80B4),

PostColorMatrixGreenScaleSgi = ((int)0x80B5), PostColorMa-
trixBlueScaleSgi = ((int)0x80B6), PostColorMatrixAlphaScaleSgi =
((int)0x80B7), PostColorMatrixRedBiasSgi = ((int)0x80B8),

PostColorMatrixGreenBiasSgi = ((int)0x80B9), PostColorMatrixBlueBi-
asSgi = ((int)0x80BA), PostColorMatrixAlphaBiasSgi = ((int)0x80BB) }
• enum SgiColorTable {

ColorTableSgi = ((int)0x80D0), PostConvolutionColorTableSgi =
((int)0x80D1), PostColorMatrixColorTableSgi = ((int)0x80D2), Proxy-
ColorTableSgi = ((int)0x80D3),

ProxyPostConvolutionColorTableSgi = ((int)0x80D4), ProxyPostColorMa-
trixColorTableSgi = ((int)0x80D5), ColorTableScaleSgi = ((int)0x80D6), Col-
orTableBiasSgi = ((int)0x80D7),

ColorTableFormatSgi = ((int)0x80D8), ColorTableWidthSgi = ((int)0x80D9),
ColorTableRedSizeSgi = ((int)0x80DA), ColorTableGreenSizeSgi =
((int)0x80DB),

ColorTableBlueSizeSgi = ((int)0x80DC), ColorTableAlphaSizeSgi =
((int)0x80DD), ColorTableLuminanceSizeSgi = ((int)0x80DE), Col-
orTableIntensitySizeSgi = ((int)0x80DF) }

```

- enum **SgiDepthPassInstrument** { **DepthPassInstrumentSgix** = ((int)0x8310), **DepthPassInstrumentCountersSgix** = ((int)0x8311), **DepthPassInstrumentMaxSgix** = ((int)0x8312) }
- enum **SgisDetailTexture** {
DetailTexture2DSgis = ((int)0x8095), **DetailTexture2DBindingSgis** = ((int)0x8096), **LinearDetailSgis** = ((int)0x8097), **LinearDetailAlphaSgis** = ((int)0x8098),
LinearDetailColorSgis = ((int)0x8099), **DetailTextureLevelSgis** = ((int)0x809A), **DetailTextureModeSgis** = ((int)0x809B), **DetailTextureFuncPointsSgis** = ((int)0x809C) }
- enum **SgisFogFunction** { **FogFuncSgis** = ((int)0x812A), **FogFuncPointsSgis** = ((int)0x812B), **MaxFogFuncPointsSgis** = ((int)0x812C) }
- enum **SgisGenerateMipmap** {
GenerateMipmapSgis = ((int)0x8191), **GenerateMipmapHintSgis** = ((int)0x8192), **GeometryDeformationSgix** = ((int)0x8194), **TextureDeformationSgix** = ((int)0x8195),
DeformationsMaskSgix = ((int)0x8196), **MaxDeformationOrderSgix** = ((int)0x8197) }
- enum **SgisMultisample** {
MultisampleSgis = ((int)0x809D), **SampleAlphaToMaskSgis** = ((int)0x809E), **SampleAlphaToOneSgis** = ((int)0x809F), **SampleMaskSgis** = ((int)0x80A0),
GL1PassSgis = ((int)0x80A1), **GL2Pass0Sgis** = ((int)0x80A2), **GL2Pass1Sgis** = ((int)0x80A3), **GL4Pass0Sgis** = ((int)0x80A4),
GL4Pass1Sgis = ((int)0x80A5), **GL4Pass2Sgis** = ((int)0x80A6), **GL4Pass3Sgis** = ((int)0x80A7), **SampleBuffersSgis** = ((int)0x80A8),
SamplesSgis = ((int)0x80A9), **SampleMaskValueSgis** = ((int)0x80AA), **SampleMaskInvertSgis** = ((int)0x80AB), **SamplePatternSgis** = ((int)0x80AC) }
- enum **SgisPixelTexture** { **PixelTextureSgis** = ((int)0x8353), **PixelFragmentRgbSourceSgis** = ((int)0x8354), **PixelFragmentAlphaSourceSgis** = ((int)0x8355), **PixelGroupColorSgis** = ((int)0x8356) }
- enum **SgisPointLineTexgen** {
EyeDistanceToPointSgis = ((int)0x81F0), **ObjectDistanceToPointSgis** = ((int)0x81F1), **EyeDistanceToLineSgis** = ((int)0x81F2), **ObjectDistanceToLineSgis** = ((int)0x81F3),
EyePointSgis = ((int)0x81F4), **ObjectPointSgis** = ((int)0x81F5), **EyeLineSgis** = ((int)0x81F6), **ObjectLineSgis** = ((int)0x81F7) }
- enum **SgisPointParameters** { **PointSizeMinSgis** = ((int)0x8126), **PointSizeMaxSgis** = ((int)0x8127), **PointFadeThresholdSizeSgis** = ((int)0x8128), **DistanceAttenuationSgis** = ((int)0x8129) }
- enum **SgisSharpenTexture** { **LinearSharpenSgis** = ((int)0x80AD), **LinearSharpenAlphaSgis** = ((int)0x80AE), **LinearSharpenColorSgis** = ((int)0x80AF), **SharpenTextureFuncPointsSgis** = ((int)0x80B0) }

- enum **SgisTexture4D** {
PackSkipVolumesSgis = ((int)0x8130), **PackImageDepthSgis** = ((int)0x8131),
UnpackSkipVolumesSgis = ((int)0x8132), **UnpackImageDepthSgis** = ((int)0x8133),
Texture4DSgis = ((int)0x8134), **ProxyTexture4DSgis** = ((int)0x8135), **Texture4DsizeSgis** = ((int)0x8136), **TextureWrapQSgis** = ((int)0x8137),
Max4DTextureSizeSgis = ((int)0x8138), **Texture4DBindingSgis** = ((int)0x814F) }
- enum **SgisTextureBorderClamp** { **ClampToBorderSgis** = ((int)0x812D) }
- enum **SgisTextureColorMask** { **TextureColorWritemaskSgis** = ((int)0x81EF) }
- enum **SgisTextureEdgeClamp** { **ClampToEdgeSgis** = ((int)0x812F) }
- enum **SgisTextureFilter4** { **Filter4Sgis** = ((int)0x8146), **TextureFilter4SizeSgis** = ((int)0x8147) }
- enum **SgisTextureLod** { **TextureMinLodSgis** = ((int)0x813A), **TextureMaxLodSgis** = ((int)0x813B), **TextureBaseLevelSgis** = ((int)0x813C), **TextureMaxLevelSgis** = ((int)0x813D) }
- enum **SgisTextureSelect** {
DualAlpha4Sgis = ((int)0x8110), **DualAlpha8Sgis** = ((int)0x8111), **DualAlpha12Sgis** = ((int)0x8112), **DualAlpha16Sgis** = ((int)0x8113),
DualLuminance4Sgis = ((int)0x8114), **DualLuminance8Sgis** = ((int)0x8115), **DualLuminance12Sgis** = ((int)0x8116), **DualLuminance16Sgis** = ((int)0x8117),
DualIntensity4Sgis = ((int)0x8118), **DualIntensity8Sgis** = ((int)0x8119), **DualIntensity12Sgis** = ((int)0x811A), **DualIntensity16Sgis** = ((int)0x811B),
DualLuminanceAlpha4Sgis = ((int)0x811C), **DualLuminanceAlpha8Sgis** = ((int)0x811D), **QuadAlpha4Sgis** = ((int)0x811E), **QuadAlpha8Sgis** = ((int)0x811F),
QuadLuminance4Sgis = ((int)0x8120), **QuadLuminance8Sgis** = ((int)0x8121), **QuadIntensity4Sgis** = ((int)0x8122), **QuadIntensity8Sgis** = ((int)0x8123),
DualTextureSelectSgis = ((int)0x8124), **QuadTextureSelectSgis** = ((int)0x8125) }
- enum **SgiTextureColorTable** { **TextureColorTableSgi** = ((int)0x80BC), **ProxyTextureColorTableSgi** = ((int)0x80BD) }
- enum **SgixAsync** { **AsyncMarkerSgix** = ((int)0x8329) }
- enum **SgixAsyncHistogram** { **AsyncHistogramSgix** = ((int)0x832C), **MaxAsyncHistogramSgix** = ((int)0x832D) }
- enum **SgixAsyncPixel** {
AsyncTexImageSgix = ((int)0x835C), **AsyncDrawPixelsSgix** = ((int)0x835D),
AsyncReadPixelsSgix = ((int)0x835E), **MaxAsyncTexImageSgix** = ((int)0x835F),

```

MaxAsyncDrawPixelsSgix = ((int)0x8360), MaxAsyncReadPixelsSgix =
((int)0x8361) }
• enum SgixBlendAlphaMinmax { AlphaMinSgix = ((int)0x8320), AlphaMaxSgix = ((int)0x8321), AsyncMarkerSgix = ((int)0x8329) }
• enum SgixCalligraphicFragment { CalligraphicFragmentSgix = ((int)0x8183) }
• enum SgixClipmap {
LinearClipmapLinearSgix = ((int)0x8170), TextureClipmapCenterSgix = ((int)0x8171), TextureClipmapFrameSgix = ((int)0x8172), TextureClipmapOffsetSgix = ((int)0x8173),
TextureClipmapVirtualDepthSgix = ((int)0x8174), TextureClipmapLodOffsetSgix = ((int)0x8175), TextureClipmapDepthSgix = ((int)0x8176), MaxClipmapDepthSgix = ((int)0x8177),
MaxClipmapVirtualDepthSgix = ((int)0x8178), NearestClipmapNearestSgix = ((int)0x844D), NearestClipmapLinearSgix = ((int)0x844E), LinearClipmapNearestSgix = ((int)0x844F) }
• enum SgixConvolutionAccuracy { ConvolutionHintSgix = ((int)0x8316) }
• enum SgixDepthTexture { DepthComponent16Sgix = ((int)0x81A5), DepthComponent24Sgix = ((int)0x81A6), DepthComponent32Sgix = ((int)0x81A7) }
• enum SgixFlushRaster
• enum SgixFogOffset { FogOffsetSgix = ((int)0x8198), FogOffsetValueSgix = ((int)0x8199) }
• enum SgixFogScale { FogScaleSgix = ((int)0x81FC), FogScaleValueSgix = ((int)0x81FD) }
• enum SgixFragmentLighting {
FragmentLightingSgix = ((int)0x8400), FragmentColorMaterialSgix = ((int)0x8401), FragmentColorMaterialFaceSgix = ((int)0x8402), FragmentColorMaterialParameterSgix = ((int)0x8403),
MaxFragmentLightsSgix = ((int)0x8404), MaxActiveLightsSgix = ((int)0x8405), CurrentRasterNormalSgix = ((int)0x8406), LightEnvModeSgix = ((int)0x8407),
FragmentLightModelLocalViewerSgix = ((int)0x8408), FragmentLightModelTwoSideSgix = ((int)0x8409), FragmentLightModelAmbientSgix = ((int)0x840A), FragmentLightModelNormalInterpolationSgix = ((int)0x840B),
FragmentLight0Sgix = ((int)0x840C), FragmentLight1Sgix = ((int)0x840D), FragmentLight2Sgix = ((int)0x840E), FragmentLight3Sgix = ((int)0x840F),
FragmentLight4Sgix = ((int)0x8410), FragmentLight5Sgix = ((int)0x8411), FragmentLight6Sgix = ((int)0x8412), FragmentLight7Sgix = ((int)0x8413) }
• enum SgixFramezoom { FramezoomSgix = ((int)0x818B), FramezoomFactorSgix = ((int)0x818C), MaxFramezoomFactorSgix = ((int)0x818D) }

```

- enum **SgixImpactPixelTexture** {
PixelTexGenQCeilingSgix = ((int)0x8184), **PixelTexGenQRoundSgix** = ((int)0x8185), **PixelTexGenQFloorSgix** = ((int)0x8186), **PixelTexGenAlphaReplaceSgix** = ((int)0x8187),
PixelTexGenAlphaNoReplaceSgix = ((int)0x8188), **PixelTexGenAlphaLsSgix** = ((int)0x8189), **PixelTexGenAlphaMsSgix** = ((int)0x818A) }
- enum **SgixInstruments** { **InstrumentBufferPointerSgix** = ((int)0x8180), **InstrumentMeasurementsSgix** = ((int)0x8181) }
- enum **SgixInterlace** { **InterlaceSgix** = ((int)0x8094) }
- enum **SgixIrInstrument1** { **IrInstrument1Sgix** = ((int)0x817F) }
- enum **SgixListPriority** { **ListPrioritySgix** = ((int)0x8182) }
- enum **SgixPixelTexture** { **PixelTexGenSgix** = ((int)0x8139), **PixelTexGenModeSgix** = ((int)0x832B) }
- enum **SgixPixelTiles** {
PixelTileBestAlignmentSgix = ((int)0x813E), **PixelTileCacheIncrementSgix** = ((int)0x813F), **PixelTileWidthSgix** = ((int)0x8140), **PixelTileHeightSgix** = ((int)0x8141),
PixelTileGridWidthSgix = ((int)0x8142), **PixelTileGridHeightSgix** = ((int)0x8143), **PixelTileGridDepthSgix** = ((int)0x8144), **PixelTileCacheSizeSgix** = ((int)0x8145) }
- enum **SgixPolynomialFfd** { **GeometryDeformationSgix** = ((int)0x8194), **TextureDeformationSgix** = ((int)0x8195), **DeformationsMaskSgix** = ((int)0x8196), **MaxDeformationOrderSgix** = ((int)0x8197) }
- enum **SgixReferencePlane** { **ReferencePlaneSgix** = ((int)0x817D), **ReferencePlaneEquationSgix** = ((int)0x817E) }
- enum **SgixResample** {
PackResampleSgix = ((int)0x842C), **UnpackResampleSgix** = ((int)0x842D), **ResampleReplicateSgix** = ((int)0x842E), **ResampleZeroFillSgix** = ((int)0x842F),
ResampleDecimateSgix = ((int)0x8430) }
- enum **SgixScalebiasHint** { **ScalebiasHintSgix** = ((int)0x8322) }
- enum **SgixShadow** { **TextureCompareSgix** = ((int)0x819A), **TextureCompareOperatorSgix** = ((int)0x819B), **TextureLequalRSgix** = ((int)0x819C), **TextureGequalRSgix** = ((int)0x819D) }
- enum **SgixShadowAmbient** { **ShadowAmbientSgix** = ((int)0x80BF) }
- enum **SgixSprite** {
SpriteSgix = ((int)0x8148), **SpriteModeSgix** = ((int)0x8149), **SpriteAxisSgix** = ((int)0x814A), **SpriteTranslationSgix** = ((int)0x814B),
SpriteAxialSgix = ((int)0x814C), **SpriteObjectAlignedSgix** = ((int)0x814D), **SpriteEyeAlignedSgix** = ((int)0x814E) }

- enum **SgixSubsample** {
 - PackSubsampleRateSgix** = ((int)0x85A0), **UnpackSubsampleRateSgix** = ((int)0x85A1), **PixelSubsample4444Sgix** = ((int)0x85A2), **PixelSubsample2424Sgix** = ((int)0x85A3),
 - PixelSubsample4242Sgix** = ((int)0x85A4) }
- enum **SgixTagSampleBuffer**
- enum **SgixTextureAddEnv** { **TextureEnvBiasSgix** = ((int)0x80BE) }
- enum **SgixTextureCoordinateClamp** { **TextureMaxClampSSgix** = ((int)0x8369), **TextureMaxClampTSgix** = ((int)0x836A), **TextureMaxClampRSgix** = ((int)0x836B), **FogFactorToAlphaSgix** = ((int)0x836F) }
- enum **SgixTextureLodBias** { **TextureLodBiasSSgix** = ((int)0x818E), **TextureLodBiasTSgix** = ((int)0x818F), **TextureLodBiasRSgix** = ((int)0x8190) }
- enum **SgixTextureMultiBuffer** { **TextureMultiBufferHintSgix** = ((int)0x812E) }
- enum **SgixTextureScaleBias** { **PostTextureFilterBiasSgix** = ((int)0x8179), **PostTextureFilterScaleSgix** = ((int)0x817A), **PostTextureFilterBiasRangeSgix** = ((int)0x817B), **PostTextureFilterScaleRangeSgix** = ((int)0x817C) }
- enum **SgixVertexPreclip** { **VertexPreclipSgix** = ((int)0x83EE), **VertexPreclipHintSgix** = ((int)0x83EF) }
- enum **SgixYcrCb** { **YcrCb422Sgix** = ((int)0x81BB), **YcrCb444Sgix** = ((int)0x81BC) }
- enum **SgixYcrCbA** { **YcrCbSgix** = ((int)0x8318), **YcrCbASgix** = ((int)0x8319) }
- enum **SgixYcrCbSubsample** {
 - PackSubsampleRateSgix** = ((int)0x85A0), **UnpackSubsampleRateSgix** = ((int)0x85A1), **PixelSubsample4444Sgix** = ((int)0x85A2), **PixelSubsample2424Sgix** = ((int)0x85A3),
 - PixelSubsample4242Sgix** = ((int)0x85A4) }
- enum **ShaderParameter** {
 - ShaderType** = ((int)0x8B4F), **DeleteStatus** = ((int)0x8B80), **CompileStatus** = ((int)0x8B81), **InfoLogLength** = ((int)0x8B84),
 - ShaderSourceLength** = ((int)0x8B88) }
- enum **ShaderType** { **FragmentShader** = ((int)0x8B30), **VertexShader** = ((int)0x8B31), **GeometryShader** = ((int)0x8DD9), **GeometryShaderExt** = ((int)0x8DD9) }
- enum **ShadingModel** { **Flat** = ((int)0x1D00), **Smooth** = ((int)0x1D01) }
- enum **SizedInternalFormat** {
 - Rgba8** = ((int)0x8058), **Rgba16** = ((int)0x805B), **R8** = ((int)0x8229), **R16** = ((int)0x822A),
 - Rg8** = ((int)0x822B), **Rg16** = ((int)0x822C), **R16f** = ((int)0x822D), **R32f** = ((int)0x822E),

```

Rg16f = ((int)0x822F), Rg32f = ((int)0x8230), R8i = ((int)0x8231), R8ui =
((int)0x8232),

R16i = ((int)0x8233), R16ui = ((int)0x8234), R32i = ((int)0x8235), R32ui =
((int)0x8236),

Rg8i = ((int)0x8237), Rg8ui = ((int)0x8238), Rg16i = ((int)0x8239), Rg16ui =
((int)0x823A),

Rg32i = ((int)0x823B), Rg32ui = ((int)0x823C), Rgba32f = ((int)0x8814),
Rgba16f = ((int)0x881A),

Rgba32ui = ((int)0x8D70), Rgba16ui = ((int)0x8D76), Rgba8ui =
((int)0x8D7C), Rgba32i = ((int)0x8D82),

Rgba16i = ((int)0x8D88), Rgba8i = ((int)0x8D8E) }
• enum StencilFace { Front = ((int)0x0404), Back = ((int)0x0405), FrontAnd-
Back = ((int)0x0408) }
• enum StencilFunction {

Never = ((int)0x0200), Less = ((int)0x0201), Equal = ((int)0x0202), Lequal =
((int)0x0203),

Greater = ((int)0x0204), Notequal = ((int)0x0205), Gequal = ((int)0x0206),
Always = ((int)0x0207) }
• enum StencilOp {

Zero = ((int)0), Invert = ((int)0x150A), Keep = ((int)0x1E00), Replace =
((int)0x1E01),

Incr = ((int)0x1E02), Decr = ((int)0x1E03), IncrWrap = ((int)0x8507), Decr-
Wrap = ((int)0x8508) }
• enum StringName {

Vendor = ((int)0x1F00), Renderer = ((int)0x1F01), Version = ((int)0x1F02),
Extensions = ((int)0x1F03),

ShadingLanguageVersion = ((int)0x8B8C) }
• enum SunConvolutionBorderModes { WrapBorderSun = ((int)0x81D4) }
• enum SunGlobalAlpha { GlobalAlphaSun = ((int)0x81D9), GlobalAlphaFac-
torSun = ((int)0x81DA) }
• enum SunMeshArray { QuadMeshSun = ((int)0x8614), TriangleMeshSun =
((int)0x8615) }
• enum SunSliceAccum { SliceAccumSun = ((int)0x85CC) }
• enum SunTriangleList {

RestartSun = ((int)0x0001), ReplaceMiddleSun = ((int)0x0002), ReplaceOld-
estSun = ((int)0x0003), TriangleListSun = ((int)0x81D7),

ReplacementCodeSun = ((int)0x81D8), ReplacementCodeArraySun =
((int)0x85C0), ReplacementCodeArrayTypeSun = ((int)0x85C1), Replace-
mentCodeArrayStrideSun = ((int)0x85C2),

```

ReplacementCodeArrayPointerSun = ((int)0x85C3), **R1uiV3fSun** = ((int)0x85C4), **R1uiC4ubV3fSun** = ((int)0x85C5), **R1uiC3fV3fSun** = ((int)0x85C6),

R1uiN3fV3fSun = ((int)0x85C7), **R1uiC4fN3fV3fSun** = ((int)0x85C8), **R1uiT2fV3fSun** = ((int)0x85C9), **R1uiT2fN3fV3fSun** = ((int)0x85CA),

R1uiT2fC4fN3fV3fSun = ((int)0x85CB) }

- enum **SunVertex**
- enum **SunxConstantData** { **UnpackConstantDataSunx** = ((int)0x81D5), **TextureConstantDataSunx** = ((int)0x81D6) }
- enum **TexCoordPointerType** {

Short = ((int)0x1402), **Int** = ((int)0x1404), **Float** = ((int)0x1406), **Double** = ((int)0x140A),

HalfFloat = ((int)0x140B) }
- enum **TextureBufferTarget** { **TextureBuffer** = ((int)0x8C2A) }
- enum **TextureCompareMode** { **CompareRefToTexture** = ((int)0x884E), **CompareRToTexture** = ((int)0x884E) }
- enum **TextureCoordName** { **S** = ((int)0x2000), **T** = ((int)0x2001), **R** = ((int)0x2002), **Q** = ((int)0x2003) }
- enum **TextureEnvMode** {

Add = ((int)0x0104), **Blend** = ((int)0x0BE2), **Replace** = ((int)0x1e01), **Modulate** = ((int)0x2100),

Decal = ((int)0x2101), **ReplaceExt** = ((int)0x8062), **TextureEnvBiasSgix** = ((int)0x80BE), **Combine** = ((int)0x8570) }
- enum **TextureEnvModeCombine** {

Add = ((int)0x0104), **Replace** = ((int)0x1E01), **Modulate** = ((int)0x2100), **Subtract** = ((int)0x84E7),

AddSigned = ((int)0x8574), **Interpolate** = ((int)0x8575), **Dot3Rgb** = ((int)0x86AE), **Dot3Rgba** = ((int)0x86AF) }
- enum **TextureEnvModeOperandAlpha** { **SrcAlpha** = ((int)0x0302), **OneMinusSrcAlpha** = ((int)0x0303) }
- enum **TextureEnvModeOperandRgb** { **SrcColor** = ((int)0x0300), **OneMinusSrcColor** = ((int)0x0301), **SrcAlpha** = ((int)0x0302), **OneMinusSrcAlpha** = ((int)0x0303) }
- enum **TextureEnvModePointSprite** { **False** = ((int)0), **True** = ((int)1) }
- enum **TextureEnvModeScale** { **One** = ((int)1), **Two** = ((int)2), **Four** = ((int)4) }
- enum **TextureEnvModeSource** {

Texture = ((int)0x1702), **Texture0** = ((int)0x84C0), **Texture1** = ((int)0x84C1), **Texture2** = ((int)0x84C2),

Texture3 = ((int)0x84C3), **Texture4** = ((int)0x84C4), **Texture5** = ((int)0x84C5), **Texture6** = ((int)0x84C6),

```

Texture7 = ((int)0x84C7), Texture8 = ((int)0x84C8), Texture9 = ((int)0x84C9),
Texture10 = ((int)0x84CA),

Texture11 = ((int)0x84CB), Texture12 = ((int)0x84CC), Texture13 =
((int)0x84CD), Texture14 = ((int)0x84CE),

Texture15 = ((int)0x84CF), Texture16 = ((int)0x84D0), Texture17 =
((int)0x84D1), Texture18 = ((int)0x84D2),

Texture19 = ((int)0x84D3), Texture20 = ((int)0x84D4), Texture21 =
((int)0x84D5), Texture22 = ((int)0x84D6),

Texture23 = ((int)0x84D7), Texture24 = ((int)0x84D8), Texture25 =
((int)0x84D9), Texture26 = ((int)0x84DA),

Texture27 = ((int)0x84DB), Texture28 = ((int)0x84DC), Texture29 =
((int)0x84DD), Texture30 = ((int)0x84DE),

Texture31 = ((int)0x84DF), Constant = ((int)0x8576), PrimaryColor =
((int)0x8577), Previous = ((int)0x8578) }
• enum TextureEnvParameter {
    AlphaScale = ((int)0x0D1C), TextureEnvMode = ((int)0x2200), TextureEnv-
Color = ((int)0x2201), TextureLodBias = ((int)0x8501),

    CombineRgb = ((int)0x8571), CombineAlpha = ((int)0x8572), RgbScale =
((int)0x8573), Source0Rgb = ((int)0x8580),

    Src1Rgb = ((int)0x8581), Src2Rgb = ((int)0x8582), Src0Alpha =
((int)0x8588), Src1Alpha = ((int)0x8589),

    Src2Alpha = ((int)0x858A), Operand0Rgb = ((int)0x8590), Operand1Rgb =
((int)0x8591), Operand2Rgb = ((int)0x8592),

    Operand0Alpha = ((int)0x8598), Operand1Alpha = ((int)0x8599),
Operand2Alpha = ((int)0x859A), CoordReplace = ((int)0x8862) }
• enum TextureEnvTarget { TextureEnv = ((int)0x2300), TextureFilterControl
= ((int)0x8500), PointSprite = ((int)0x8861) }
• enum TextureFilterFuncSgis { Filter4Sgis = ((int)0x8146) }
• enum TextureGenMode {
    EyeLinear = ((int)0x2400), ObjectLinear = ((int)0x2401), SphereMap =
((int)0x2402), EyeDistanceToPointSgis = ((int)0x81F0),

    ObjectDistanceToPointSgis = ((int)0x81F1), EyeDistanceToLineSgis =
((int)0x81F2), ObjectDistanceToLineSgis = ((int)0x81F3), NormalMap =
((int)0x8511),

    ReflectionMap = ((int)0x8512) }
• enum TextureGenParameter {
    TextureGenMode = ((int)0x2500), ObjectPlane = ((int)0x2501), EyePlane =
((int)0x2502), EyePointSgis = ((int)0x81F4),

    ObjectPointSgis = ((int)0x81F5), EyeLineSgis = ((int)0x81F6), ObjectLineS-
gis = ((int)0x81F7) }

```

- enum **TextureMagFilter** {
 - Nearest** = ((int)0x2600), **Linear** = ((int)0x2601), **LinearDetailSgis** = ((int)0x8097), **LinearDetailAlphaSgis** = ((int)0x8098),
 - LinearDetailColorSgis** = ((int)0x8099), **LinearSharpenSgis** = ((int)0x80AD), **LinearSharpenAlphaSgis** = ((int)0x80AE), **LinearSharpenColorSgis** = ((int)0x80AF),
 - Filter4Sgis** = ((int)0x8146), **PixelTexGenQCeilingSgix** = ((int)0x8184), **PixelTexGenQRoundSgix** = ((int)0x8185), **PixelTexGenQFloorSgix** = ((int)0x8186) }
- enum **TextureMinFilter** {
 - Nearest** = ((int)0x2600), **Linear** = ((int)0x2601), **NearestMipmapNearest** = ((int)0x2700), **LinearMipmapNearest** = ((int)0x2701),
 - NearestMipmapLinear** = ((int)0x2702), **LinearMipmapLinear** = ((int)0x2703), **Filter4Sgis** = ((int)0x8146), **LinearClipmapLinearSgix** = ((int)0x8170),
 - PixelTexGenQCeilingSgix** = ((int)0x8184), **PixelTexGenQRoundSgix** = ((int)0x8185), **PixelTexGenQFloorSgix** = ((int)0x8186), **NearestClipmapNearestSgix** = ((int)0x844D),
 - NearestClipmapLinearSgix** = ((int)0x844E), **LinearClipmapNearestSgix** = ((int)0x844F) }
- enum **TextureParameterName** {
 - TextureBorderColor** = ((int)0x1004), **Red** = ((int)0x1903), **TextureMagFilter** = ((int)0x2800), **TextureMinFilter** = ((int)0x2801),
 - TextureWrapS** = ((int)0x2802), **TextureWrapT** = ((int)0x2803), **TexturePriority** = ((int)0x8066), **TextureDepth** = ((int)0x8071),
 - TextureWrapR** = ((int)0x8072), **TextureWrapRExt** = ((int)0x8072), **DetailTextureLevelSgis** = ((int)0x809A), **DetailTextureModeSgis** = ((int)0x809B),
 - ShadowAmbientSgix** = ((int)0x80BF), **TextureCompareFailValue** = ((int)0x80BF), **DualTextureSelectSgis** = ((int)0x8124), **QuadTextureSelectSgis** = ((int)0x8125),
 - ClampToBorder** = ((int)0x812D), **ClampToEdge** = ((int)0x812F), **TextureWrapQSgis** = ((int)0x8137), **TextureMinLod** = ((int)0x813A),
 - TextureMaxLod** = ((int)0x813B), **TextureBaseLevel** = ((int)0x813C), **TextureMaxLevel** = ((int)0x813D), **TextureClipmapCenterSgix** = ((int)0x8171),
 - TextureClipmapFrameSgix** = ((int)0x8172), **TextureClipmapOffsetSgix** = ((int)0x8173), **TextureClipmapVirtualDepthSgix** = ((int)0x8174), **TextureClipmapLodOffsetSgix** = ((int)0x8175),
 - TextureClipmapDepthSgix** = ((int)0x8176), **PostTextureFilterBiasSgix** = ((int)0x8179), **PostTextureFilterScaleSgix** = ((int)0x817A), **TextureLodBiasSSgix** = ((int)0x818E),


```

TextureLodBiasTSgix = ((int)0x818F), TextureLodBiasRSgix =
((int)0x8190), GenerateMipmap = ((int)0x8191), GenerateMipmapSgis
= ((int)0x8191),

TextureCompareSgix = ((int)0x819A), TextureCompareOperatorSgix
= ((int)0x819B), TextureMaxClampSSgix = ((int)0x8369), TextureMax-
ClampTSgix = ((int)0x836A),

TextureMaxClampRSgix = ((int)0x836B), TextureLodBias = ((int)0x8501),
DepthTextureMode = ((int)0x884B), TextureCompareMode = ((int)0x884C),
TextureCompareFunc = ((int)0x884D) }
• enum TextureTarget {
Texture1D = ((int)0x0DE0), Texture2D = ((int)0x0DE1), ProxyTexture1D =
((int)0x8063), ProxyTexture2D = ((int)0x8064),

Texture3D = ((int)0x806F), ProxyTexture3D = ((int)0x8070), DetailTex-
ture2DSgis = ((int)0x8095), Texture4DSgis = ((int)0x8134),

ProxyTexture4DSgis = ((int)0x8135), TextureMinLod = ((int)0x813A), Tex-
tureMaxLod = ((int)0x813B), TextureBaseLevel = ((int)0x813C),

TextureMaxLevel = ((int)0x813D), TextureRectangle = ((int)0x84F5), Tex-
tureRectangleArb = ((int)0x84F5), TextureRectangleNv = ((int)0x84F5),

ProxyTextureRectangle = ((int)0x84F7), TextureCubeMap = ((int)0x8513),
TextureBindingCubeMap = ((int)0x8514), TextureCubeMapPositiveX =
((int)0x8515),

TextureCubeMapNegativeX = ((int)0x8516), TextureCubeMapPositiveY =
((int)0x8517), TextureCubeMapNegativeY = ((int)0x8518), TextureCube-
MapPositiveZ = ((int)0x8519),

TextureCubeMapNegativeZ = ((int)0x851A), ProxyTextureCubeMap =
((int)0x851B), Texture1DArray = ((int)0x8C18), ProxyTexture1DArray =
((int)0x8C19),

Texture2DArray = ((int)0x8C1A), ProxyTexture2DArray = ((int)0x8C1B),
TextureBuffer = ((int)0x8C2A), Texture2DMultisample = ((int)0x9100),

ProxyTexture2DMultisample = ((int)0x9101), Texture2DMultisampleArray
= ((int)0x9102), ProxyTexture2DMultisampleArray = ((int)0x9103) }
• enum TextureTargetMultisample { Texture2DMultisample = ((int)0x9100),
ProxyTexture2DMultisample = ((int)0x9101), Texture2DMultisampleArray
= ((int)0x9102), ProxyTexture2DMultisampleArray = ((int)0x9103) }
• enum TextureUnit {
Texture0 = ((int)0x84C0), Texture1 = ((int)0x84C1), Texture2 = ((int)0x84C2),
Texture3 = ((int)0x84C3),

Texture4 = ((int)0x84C4), Texture5 = ((int)0x84C5), Texture6 = ((int)0x84C6),
Texture7 = ((int)0x84C7),

Texture8 = ((int)0x84C8), Texture9 = ((int)0x84C9), Texture10 =
((int)0x84CA), Texture11 = ((int)0x84CB),

```

```

Texture12 = ((int)0x84CC), Texture13 = ((int)0x84CD), Texture14 =
((int)0x84CE), Texture15 = ((int)0x84CF),
Texture16 = ((int)0x84D0), Texture17 = ((int)0x84D1), Texture18 =
((int)0x84D2), Texture19 = ((int)0x84D3),
Texture20 = ((int)0x84D4), Texture21 = ((int)0x84D5), Texture22 =
((int)0x84D6), Texture23 = ((int)0x84D7),
Texture24 = ((int)0x84D8), Texture25 = ((int)0x84D9), Texture26 =
((int)0x84DA), Texture27 = ((int)0x84DB),
Texture28 = ((int)0x84DC), Texture29 = ((int)0x84DD), Texture30 =
((int)0x84DE), Texture31 = ((int)0x84DF) }
• enum TextureWrapMode {
Clamp = ((int)0x2900), Repeat = ((int)0x2901), ClampToBorder =
((int)0x812D), ClampToEdge = ((int)0x812F),
MirroredRepeat = ((int)0x8370) }
• enum TransformFeedbackMode { InterleavedAttribs = ((int)0x8C8C), SeperateAttribs = ((int)0x8C8D) }
• enum Version11 {
False = ((int)0), NoError = ((int)0), None = ((int)0), Zero = ((int)0),
Points = ((int)0x0000), DepthBufferBit = ((int)0x00000100), StencilBufferBit
= ((int)0x00000400), ColorBufferBit = ((int)0x00004000),
Lines = ((int)0x0001), LineLoop = ((int)0x0002), LineStrip = ((int)0x0003),
Triangles = ((int)0x0004),
TriangleStrip = ((int)0x0005), TriangleFan = ((int)0x0006), Never =
((int)0x0200), Less = ((int)0x0201),
Equal = ((int)0x0202), Lequal = ((int)0x0203), Greater = ((int)0x0204), Notequal
= ((int)0x0205),
Gequal = ((int)0x0206), Always = ((int)0x0207), SrcColor = ((int)0x0300),
OneMinusSrcColor = ((int)0x0301),
SrcAlpha = ((int)0x0302), OneMinusSrcAlpha = ((int)0x0303), DstAlpha =
((int)0x0304), OneMinusDstAlpha = ((int)0x0305),
DstColor = ((int)0x0306), OneMinusDstColor = ((int)0x0307), SrcAlphaSaturate
= ((int)0x0308), FrontLeft = ((int)0x0400),
FrontRight = ((int)0x0401), BackLeft = ((int)0x0402), BackRight =
((int)0x0403), Front = ((int)0x0404),
Back = ((int)0x0405), Left = ((int)0x0406), Right = ((int)0x0407), FrontAndBack
= ((int)0x0408),
InvalidEnum = ((int)0x0500), InvalidValue = ((int)0x0501), InvalidOperation
= ((int)0x0502), OutOfMemory = ((int)0x0505),
Cw = ((int)0x0900), Ccw = ((int)0x0901), PointSize = ((int)0x0B11), PointSizeRange
= ((int)0x0B12),

```

PointSizeGranularity = ((int)0x0B13), **LineSmooth** = ((int)0x0B20),
LineWidth = ((int)0x0B21), **LineWidthRange** = ((int)0x0B22),
LineWidthGranularity = ((int)0x0B23), **PolygonSmooth** = ((int)0x0B41),
CullFace = ((int)0x0B44), **CullFaceMode** = ((int)0x0B45),
FrontFace = ((int)0x0B46), **DepthRange** = ((int)0x0B70), **DepthTest** =
((int)0x0B71), **DepthWritemask** = ((int)0x0B72),
DepthClearValue = ((int)0x0B73), **DepthFunc** = ((int)0x0B74), **StencilTest** =
((int)0x0B90), **StencilClearValue** = ((int)0x0B91),
StencilFunc = ((int)0x0B92), **StencilValueMask** = ((int)0x0B93), **StencilFail** =
((int)0x0B94), **StencilPassDepthFail** = ((int)0x0B95),
StencilPassDepthPass = ((int)0x0B96), **StencilRef** = ((int)0x0B97), **Stencil-**
Writemask = ((int)0x0B98), **Viewport** = ((int)0x0BA2),
Dither = ((int)0x0BD0), **BlendDst** = ((int)0x0BE0), **BlendSrc** = ((int)0x0BE1),
Blend = ((int)0x0BE2),
LogicOpMode = ((int)0x0BF0), **ColorLogicOp** = ((int)0x0BF2), **DrawBuffer** =
((int)0x0C01), **ReadBuffer** = ((int)0x0C02),
ScissorBox = ((int)0x0C10), **ScissorTest** = ((int)0x0C11), **ColorClearValue** =
((int)0x0C22), **ColorWritemask** = ((int)0x0C23),
Doublebuffer = ((int)0x0C32), **Stereo** = ((int)0x0C33), **LineSmoothHint** =
((int)0x0C52), **PolygonSmoothHint** = ((int)0x0C53),
UnpackSwapBytes = ((int)0x0CF0), **UnpackLsbFirst** = ((int)0x0CF1), **Un-**
packRowLength = ((int)0x0CF2), **UnpackSkipRows** = ((int)0x0CF3),
UnpackSkipPixels = ((int)0x0CF4), **UnpackAlignment** = ((int)0x0CF5),
PackSwapBytes = ((int)0x0D00), **PackLsbFirst** = ((int)0x0D01),
PackRowLength = ((int)0x0D02), **PackSkipRows** = ((int)0x0D03), **PackSkip-**
Pixels = ((int)0x0D04), **PackAlignment** = ((int)0x0D05),
MaxTextureSize = ((int)0x0D33), **MaxViewportDims** = ((int)0x0D3A), **Sub-**
pixelBits = ((int)0x0D50), **Texture1D** = ((int)0x0DE0),
Texture2D = ((int)0x0DE1), **TextureWidth** = ((int)0x1000), **TextureHeight** =
((int)0x1001), **TextureInternalFormat** = ((int)0x1003),
TextureBorderColor = ((int)0x1004), **TextureBorder** = ((int)0x1005), **Dont-**
Care = ((int)0x1100), **Fastest** = ((int)0x1101),
Nicest = ((int)0x1102), **Byte** = ((int)0x1400), **UnsignedByte** = ((int)0x1401),
Short = ((int)0x1402),
UnsignedShort = ((int)0x1403), **Int** = ((int)0x1404), **UnsignedInt** =
((int)0x1405), **Float** = ((int)0x1406),
Double = ((int)0x140A), **Clear** = ((int)0x1500), **And** = ((int)0x1501), **AndRe-**
verse = ((int)0x1502),
Copy = ((int)0x1503), **AndInverted** = ((int)0x1504), **Noop** = ((int)0x1505),
Xor = ((int)0x1506),

```

Or = ((int)0x1507), Nor = ((int)0x1508), Equiv = ((int)0x1509), Invert =
((int)0x150A),
OrReverse = ((int)0x150B), CopyInverted = ((int)0x150C), OrInverted =
((int)0x150D), Nand = ((int)0x150E),
Set = ((int)0x150F), Texture = ((int)0x1702), Color = ((int)0x1800), Depth =
((int)0x1801),
Stencil = ((int)0x1802), StencilIndex = ((int)0x1901), DepthComponent =
((int)0x1902), Red = ((int)0x1903),
Green = ((int)0x1904), Blue = ((int)0x1905), Alpha = ((int)0x1906), Rgb =
((int)0x1907),
Rgba = ((int)0x1908), Point = ((int)0x1B00), Line = ((int)0x1B01), Fill =
((int)0x1B02),
Keep = ((int)0x1E00), Replace = ((int)0x1E01), Incr = ((int)0x1E02), Decr =
((int)0x1E03),
Vendor = ((int)0x1F00), Renderer = ((int)0x1F01), Version = ((int)0x1F02),
Extensions = ((int)0x1F03),
Nearest = ((int)0x2600), Linear = ((int)0x2601), NearestMipmapNearest =
((int)0x2700), LinearMipmapNearest = ((int)0x2701),
NearestMipmapLinear = ((int)0x2702), LinearMipmapLinear =
((int)0x2703), TextureMagFilter = ((int)0x2800), TextureMinFilter =
((int)0x2801),
TextureWrapS = ((int)0x2802), TextureWrapT = ((int)0x2803), Repeat =
((int)0x2901), PolygonOffsetUnits = ((int)0x2A00),
PolygonOffsetPoint = ((int)0x2A01), PolygonOffsetLine = ((int)0x2A02),
R3G3B2 = ((int)0x2A10), PolygonOffsetFill = ((int)0x8037),
PolygonOffsetFactor = ((int)0x8038), Rgb4 = ((int)0x804F), Rgb5 =
((int)0x8050), Rgb8 = ((int)0x8051),
Rgb10 = ((int)0x8052), Rgb12 = ((int)0x8053), Rgb16 = ((int)0x8054), Rgba2
= ((int)0x8055),
Rgba4 = ((int)0x8056), Rgb5A1 = ((int)0x8057), Rgba8 = ((int)0x8058),
Rgb10A2 = ((int)0x8059),
Rgba12 = ((int)0x805A), Rgba16 = ((int)0x805B), TextureRedSize =
((int)0x805C), TextureGreenSize = ((int)0x805D),
TextureBlueSize = ((int)0x805E), TextureAlphaSize = ((int)0x805F), Proxy-
Texture1D = ((int)0x8063), ProxyTexture2D = ((int)0x8064),
TextureBinding1D = ((int)0x8068), TextureBinding2D = ((int)0x8069), One
= ((int)1), True = ((int)1) }
• enum Version11Deprecated {
ClientPixelStoreBit = ((int)0x00000001), CurrentBit = ((int)0x00000001),
ClientVertexArrayBit = ((int)0x00000002), PointBit = ((int)0x00000002),

```

LineBit = ((int)0x00000004), **PolygonBit** = ((int)0x00000008), **PolygonStippleBit** = ((int)0x00000010), **PixelModeBit** = ((int)0x00000020),
LightingBit = ((int)0x00000040), **FogBit** = ((int)0x00000080), **AccumBufferBit** = ((int)0x00000200), **ViewportBit** = ((int)0x00000800),
TransformBit = ((int)0x00001000), **EnableBit** = ((int)0x00002000), **HintBit** = ((int)0x00008000), **EvalBit** = ((int)0x00010000),
ListBit = ((int)0x00020000), **TextureBit** = ((int)0x00040000), **Quads** = ((int)0x0007), **QuadStrip** = ((int)0x0008),
ScissorBit = ((int)0x00080000), **Polygon** = ((int)0x0009), **Accum** = ((int)0x0100), **Load** = ((int)0x0101),
Return = ((int)0x0102), **Mult** = ((int)0x0103), **Add** = ((int)0x0104), **Aux0** = ((int)0x0409),
Aux1 = ((int)0x040A), **Aux2** = ((int)0x040B), **Aux3** = ((int)0x040C), **StackOverflow** = ((int)0x0503),
StackUnderflow = ((int)0x0504), **GL2D** = ((int)0x0600), **GL3D** = ((int)0x0601), **GL3DColor** = ((int)0x0602),
GL3DColorTexture = ((int)0x0603), **GL4DColorTexture** = ((int)0x0604), **PassThroughToken** = ((int)0x0700), **PointToken** = ((int)0x0701),
LineToken = ((int)0x0702), **PolygonToken** = ((int)0x0703), **BitmapToken** = ((int)0x0704), **DrawPixelToken** = ((int)0x0705),
CopyPixelToken = ((int)0x0706), **LineResetToken** = ((int)0x0707), **Exp** = ((int)0x0800), **Exp2** = ((int)0x0801),
Coeff = ((int)0x0A00), **Order** = ((int)0x0A01), **Domain** = ((int)0x0A02), **CurrentColor** = ((int)0x0B00),
CurrentIndex = ((int)0x0B01), **CurrentNormal** = ((int)0x0B02), **CurrentTextureCoords** = ((int)0x0B03), **CurrentRasterColor** = ((int)0x0B04),
CurrentRasterIndex = ((int)0x0B05), **CurrentRasterTextureCoords** = ((int)0x0B06), **CurrentRasterPosition** = ((int)0x0B07), **CurrentRasterPositionValid** = ((int)0x0B08),
CurrentRasterDistance = ((int)0x0B09), **PointSmooth** = ((int)0x0B10), **LineStipple** = ((int)0x0B24), **LineStipplePattern** = ((int)0x0B25),
LineStippleRepeat = ((int)0x0B26), **ListMode** = ((int)0x0B30), **MaxListNesting** = ((int)0x0B31), **ListBase** = ((int)0x0B32),
ListIndex = ((int)0x0B33), **PolygonMode** = ((int)0x0B40), **PolygonStipple** = ((int)0x0B42), **EdgeFlag** = ((int)0x0B43),
Lighting = ((int)0x0B50), **LightModelLocalViewer** = ((int)0x0B51), **LightModelTwoSide** = ((int)0x0B52), **LightModelAmbient** = ((int)0x0B53),
ShadeModel = ((int)0x0B54), **ColorMaterialFace** = ((int)0x0B55), **ColorMaterialParameter** = ((int)0x0B56), **ColorMaterial** = ((int)0x0B57),

Fog = ((int)0x0B60), **FogIndex** = ((int)0x0B61), **FogDensity** = ((int)0x0B62),
FogStart = ((int)0x0B63),
FogEnd = ((int)0x0B64), **FogMode** = ((int)0x0B65), **FogColor** =
((int)0x0B66), **AccumClearValue** = ((int)0x0B80),
MatrixMode = ((int)0x0BA0), **Normalize** = ((int)0x0BA1), **ModelviewStack-**
Depth = ((int)0x0BA3), **ProjectionStackDepth** = ((int)0x0BA4),
TextureStackDepth = ((int)0x0BA5), **ModelviewMatrix** = ((int)0x0BA6),
ProjectionMatrix = ((int)0x0BA7), **TextureMatrix** = ((int)0x0BA8),
AttribStackDepth = ((int)0x0BB0), **ClientAttribStackDepth** = ((int)0x0BB1),
AlphaTest = ((int)0x0BC0), **AlphaTestFunc** = ((int)0x0BC1),
AlphaTestRef = ((int)0x0BC2), **IndexLogicOp** = ((int)0x0BF1), **LogicOp** =
((int)0x0BF1), **AuxBuffers** = ((int)0x0C00),
IndexClearValue = ((int)0x0C20), **IndexWritemask** = ((int)0x0C21), **Index-**
Mode = ((int)0x0C30), **RgbaMode** = ((int)0x0C31),
RenderMode = ((int)0x0C40), **PerspectiveCorrectionHint** = ((int)0x0C50),
PointSmoothHint = ((int)0x0C51), **FogHint** = ((int)0x0C54),
TextureGenS = ((int)0x0C60), **TextureGenT** = ((int)0x0C61), **TextureGenR** =
((int)0x0C62), **TextureGenQ** = ((int)0x0C63),
PixelMapIToI = ((int)0x0C70), **PixelMapSToS** = ((int)0x0C71), **PixelMapI-**
ToR = ((int)0x0C72), **PixelMapIToG** = ((int)0x0C73),
PixelMapIToB = ((int)0x0C74), **PixelMapIToA** = ((int)0x0C75), **Pixel-**
MapRToR = ((int)0x0C76), **PixelMapGToG** = ((int)0x0C77),
PixelMapBToB = ((int)0x0C78), **PixelMapAToA** = ((int)0x0C79), **PixelMapI-**
ToSize = ((int)0x0CB0), **PixelMapSToSSize** = ((int)0x0CB1),
PixelMapIToRSize = ((int)0x0CB2), **PixelMapIToGSize** = ((int)0x0CB3),
PixelMapIToBSize = ((int)0x0CB4), **PixelMapIToASize** = ((int)0x0CB5),
PixelMapRToRSize = ((int)0x0CB6), **PixelMapGToGSize** = ((int)0x0CB7),
PixelMapBToBSize = ((int)0x0CB8), **PixelMapAToASize** = ((int)0x0CB9),
MapColor = ((int)0x0D10), **MapStencil** = ((int)0x0D11), **IndexShift** =
((int)0x0D12), **IndexOffset** = ((int)0x0D13),
RedScale = ((int)0x0D14), **RedBias** = ((int)0x0D15), **ZoomX** = ((int)0x0D16),
ZoomY = ((int)0x0D17),
GreenScale = ((int)0x0D18), **GreenBias** = ((int)0x0D19), **BlueScale** =
((int)0x0D1A), **BlueBias** = ((int)0x0D1B),
AlphaScale = ((int)0x0D1C), **AlphaBias** = ((int)0x0D1D), **DepthScale** =
((int)0x0D1E), **DepthBias** = ((int)0x0D1F),
MaxEvalOrder = ((int)0x0D30), **MaxLights** = ((int)0x0D31), **MaxClipPlanes**
= ((int)0x0D32), **MaxPixelMapTable** = ((int)0x0D34),

MaxAttribStackDepth = ((int)0x0D35), **MaxModelviewStackDepth** = ((int)0x0D36), **MaxNameStackDepth** = ((int)0x0D37), **MaxProjectionStackDepth** = ((int)0x0D38),
MaxTextureStackDepth = ((int)0x0D39), **MaxClientAttribStackDepth** = ((int)0x0D3B), **IndexBits** = ((int)0x0D51), **RedBits** = ((int)0x0D52),
GreenBits = ((int)0x0D53), **BlueBits** = ((int)0x0D54), **AlphaBits** = ((int)0x0D55), **DepthBits** = ((int)0x0D56),
StencilBits = ((int)0x0D57), **AccumRedBits** = ((int)0x0D58), **AccumGreenBits** = ((int)0x0D59), **AccumBlueBits** = ((int)0x0D5A),
AccumAlphaBits = ((int)0x0D5B), **NameStackDepth** = ((int)0x0D70), **AutoNormal** = ((int)0x0D80), **Map1Color4** = ((int)0x0D90),
Map1Index = ((int)0x0D91), **Map1Normal** = ((int)0x0D92), **Map1TextureCoord1** = ((int)0x0D93), **Map1TextureCoord2** = ((int)0x0D94),
Map1TextureCoord3 = ((int)0x0D95), **Map1TextureCoord4** = ((int)0x0D96), **Map1Vertex3** = ((int)0x0D97), **Map1Vertex4** = ((int)0x0D98),
Map2Color4 = ((int)0x0DB0), **Map2Index** = ((int)0x0DB1), **Map2Normal** = ((int)0x0DB2), **Map2TextureCoord1** = ((int)0x0DB3),
Map2TextureCoord2 = ((int)0x0DB4), **Map2TextureCoord3** = ((int)0x0DB5), **Map2TextureCoord4** = ((int)0x0DB6), **Map2Vertex3** = ((int)0x0DB7),
Map2Vertex4 = ((int)0x0DB8), **Map1GridDomain** = ((int)0x0DD0), **Map1GridSegments** = ((int)0x0DD1), **Map2GridDomain** = ((int)0x0DD2),
Map2GridSegments = ((int)0x0DD3), **FeedbackBufferPointer** = ((int)0x0DF0), **FeedbackBufferSize** = ((int)0x0DF1), **FeedbackBufferType** = ((int)0x0DF2),
SelectionBufferPointer = ((int)0x0DF3), **SelectionBufferSize** = ((int)0x0DF4), **TextureComponents** = ((int)0x1003), **Ambient** = ((int)0x1200),
Diffuse = ((int)0x1201), **Specular** = ((int)0x1202), **Position** = ((int)0x1203), **SpotDirection** = ((int)0x1204),
SpotExponent = ((int)0x1205), **SpotCutoff** = ((int)0x1206), **ConstantAttenuation** = ((int)0x1207), **LinearAttenuation** = ((int)0x1208),
QuadraticAttenuation = ((int)0x1209), **Compile** = ((int)0x1300), **CompileAndExecute** = ((int)0x1301), **GL2Bytes** = ((int)0x1407),
GL3Bytes = ((int)0x1408), **GL4Bytes** = ((int)0x1409), **Emission** = ((int)0x1600), **Shininess** = ((int)0x1601),
AmbientAndDiffuse = ((int)0x1602), **ColorIndexes** = ((int)0x1603), **Modelview** = ((int)0x1700), **Projection** = ((int)0x1701),
ColorIndex = ((int)0x1900), **Luminance** = ((int)0x1909), **LuminanceAlpha** = ((int)0x190A), **Bitmap** = ((int)0x1A00),

Render = ((int)0x1C00), **Feedback** = ((int)0x1C01), **Select** = ((int)0x1C02),
Flat = ((int)0x1D00),
Smooth = ((int)0x1D01), **S** = ((int)0x2000), **T** = ((int)0x2001), **R** =
((int)0x2002),
Q = ((int)0x2003), **Modulate** = ((int)0x2100), **Decal** = ((int)0x2101), **TextureEnvMode** = ((int)0x2200),
TextureEnvColor = ((int)0x2201), **TextureEnv** = ((int)0x2300), **EyeLinear** =
((int)0x2400), **ObjectLinear** = ((int)0x2401),
SphereMap = ((int)0x2402), **TextureGenMode** = ((int)0x2500), **ObjectPlane**
= ((int)0x2501), **EyePlane** = ((int)0x2502),
Clamp = ((int)0x2900), **V2f** = ((int)0x2A20), **V3f** = ((int)0x2A21), **C4ubV2f** =
((int)0x2A22),
C4ubV3f = ((int)0x2A23), **C3fV3f** = ((int)0x2A24), **N3fV3f** = ((int)0x2A25),
C4fN3fV3f = ((int)0x2A26),
T2fV3f = ((int)0x2A27), **T4fV4f** = ((int)0x2A28), **T2fC4ubV3f** =
((int)0x2A29), **T2fC3fV3f** = ((int)0x2A2A),
T2fN3fV3f = ((int)0x2A2B), **T2fC4fN3fV3f** = ((int)0x2A2C), **T4fC4fN3fV4f**
= ((int)0x2A2D), **ClipPlane0** = ((int)0x3000),
ClipPlane1 = ((int)0x3001), **ClipPlane2** = ((int)0x3002), **ClipPlane3** =
((int)0x3003), **ClipPlane4** = ((int)0x3004),
ClipPlane5 = ((int)0x3005), **Light0** = ((int)0x4000), **Light1** = ((int)0x4001),
Light2 = ((int)0x4002),
Light3 = ((int)0x4003), **Light4** = ((int)0x4004), **Light5** = ((int)0x4005), **Light6**
= ((int)0x4006),
Light7 = ((int)0x4007), **Alpha4** = ((int)0x803B), **Alpha8** = ((int)0x803C), **Alpha12** = ((int)0x803D),
Alpha16 = ((int)0x803E), **Luminance4** = ((int)0x803F), **Luminance8** =
((int)0x8040), **Luminance12** = ((int)0x8041),
Luminance16 = ((int)0x8042), **Luminance4Alpha4** = ((int)0x8043), **Luminance6Alpha2** = ((int)0x8044), **Luminance8Alpha8** = ((int)0x8045),
Luminance12Alpha4 = ((int)0x8046), **Luminance12Alpha12** = ((int)0x8047),
Luminance16Alpha16 = ((int)0x8048), **Intensity** = ((int)0x8049),
Intensity4 = ((int)0x804A), **Intensity8** = ((int)0x804B), **Intensity12** =
((int)0x804C), **Intensity16** = ((int)0x804D),
TextureLuminanceSize = ((int)0x8060), **TextureIntensitySize** = ((int)0x8061),
TexturePriority = ((int)0x8066), **TextureResident** = ((int)0x8067),
VertexArray = ((int)0x8074), **NormalArray** = ((int)0x8075), **ColorArray** =
((int)0x8076), **IndexArray** = ((int)0x8077),
TextureCoordArray = ((int)0x8078), **EdgeFlagArray** = ((int)0x8079), **VertexArraySize** = ((int)0x807A), **VertexArrayType** = ((int)0x807B),

VertexArrayStride = ((int)0x807C), **NormalArrayType** = ((int)0x807E), **NormalArrayStride** = ((int)0x807F), **ColorArraySize** = ((int)0x8081),
ColorArrayType = ((int)0x8082), **ColorArrayStride** = ((int)0x8083), **IndexArrayType** = ((int)0x8085), **IndexArrayStride** = ((int)0x8086),
TextureCoordArraySize = ((int)0x8088), **TextureCoordArrayType** = ((int)0x8089), **TextureCoordArrayStride** = ((int)0x808A), **EdgeFlagArrayStride** = ((int)0x808C),
VertexArrayPointer = ((int)0x808E), **NormalArrayPointer** = ((int)0x808F), **ColorArrayPointer** = ((int)0x8090), **IndexArrayPointer** = ((int)0x8091),
TextureCoordArrayPointer = ((int)0x8092), **EdgeFlagArrayPointer** = ((int)0x8093), **AllAttribBits** = unchecked((int)0xFFFFFFFF), **ClientAllAttribBits** = unchecked((int)0xFFFFFFFF) }

- enum **Version12** {

SmoothPointSizeRange = ((int)0x0B12), **SmoothPointSizeGranularity** = ((int)0x0B13), **SmoothLineWidthRange** = ((int)0x0B22), **SmoothLineWidthGranularity** = ((int)0x0B23),
ConstantColor = ((int)0x8001), **OneMinusConstantColor** = ((int)0x8002), **ConstantAlpha** = ((int)0x8003), **OneMinusConstantAlpha** = ((int)0x8004),
BlendColor = ((int)0x8005), **Convolution1D** = ((int)0x8010), **Convolution2D** = ((int)0x8011), **Separable2D** = ((int)0x8012),
ConvolutionBorderMode = ((int)0x8013), **ConvolutionFilterScale** = ((int)0x8014), **ConvolutionFilterBias** = ((int)0x8015), **Reduce** = ((int)0x8016),
ConvolutionFormat = ((int)0x8017), **ConvolutionWidth** = ((int)0x8018), **ConvolutionHeight** = ((int)0x8019), **MaxConvolutionWidth** = ((int)0x801A),
MaxConvolutionHeight = ((int)0x801B), **PostConvolutionRedScale** = ((int)0x801C), **PostConvolutionGreenScale** = ((int)0x801D), **PostConvolutionBlueScale** = ((int)0x801E),
PostConvolutionAlphaScale = ((int)0x801F), **PostConvolutionRedBias** = ((int)0x8020), **PostConvolutionGreenBias** = ((int)0x8021), **PostConvolutionBlueBias** = ((int)0x8022),
PostConvolutionAlphaBias = ((int)0x8023), **Histogram** = ((int)0x8024), **ProxyHistogram** = ((int)0x8025), **HistogramWidth** = ((int)0x8026),
HistogramFormat = ((int)0x8027), **HistogramRedSize** = ((int)0x8028), **HistogramGreenSize** = ((int)0x8029), **HistogramBlueSize** = ((int)0x802A),
HistogramAlphaSize = ((int)0x802B), **HistogramSink** = ((int)0x802D), **Minmax** = ((int)0x802E), **MinmaxFormat** = ((int)0x802F),
MinmaxSink = ((int)0x8030), **TableTooLarge** = ((int)0x8031), **UnsignedByte32** = ((int)0x8032), **UnsignedShort4444** = ((int)0x8033),
UnsignedShort5551 = ((int)0x8034), **UnsignedInt8888** = ((int)0x8035), **UnsignedInt1010102** = ((int)0x8036), **RescaleNormal** = ((int)0x803A),

TextureBinding3D = ((int)0x806A), **PackSkipImages** = ((int)0x806B),
PackImageHeight = ((int)0x806C), **UnpackSkipImages** = ((int)0x806D),
UnpackImageHeight = ((int)0x806E), **Texture3D** = ((int)0x806F), **ProxyTexture3D** = ((int)0x8070), **TextureDepth** = ((int)0x8071),
TextureWrapR = ((int)0x8072), **Max3DTextureSize** = ((int)0x8073), **ColorMatrix** = ((int)0x80B1), **ColorMatrixStackDepth** = ((int)0x80B2),
MaxColorMatrixStackDepth = ((int)0x80B3), **PostColorMatrixRedScale** = ((int)0x80B4), **PostColorMatrixGreenScale** = ((int)0x80B5), **PostColorMatrixBlueScale** = ((int)0x80B6),
PostColorMatrixAlphaScale = ((int)0x80B7), **PostColorMatrixRedBias** = ((int)0x80B8), **PostColorMatrixGreenBias** = ((int)0x80B9), **PostColorMatrixBlueBias** = ((int)0x80BA),
PostColorMatrixAlphaBias = ((int)0x80BB), **ColorTable** = ((int)0x80D0), **PostConvolutionColorTable** = ((int)0x80D1), **PostColorMatrixColorTable** = ((int)0x80D2),
ProxyColorTable = ((int)0x80D3), **ProxyPostConvolutionColorTable** = ((int)0x80D4), **ProxyPostColorMatrixColorTable** = ((int)0x80D5), **ColorTableScale** = ((int)0x80D6),
ColorTableBias = ((int)0x80D7), **ColorTableFormat** = ((int)0x80D8), **ColorTableWidth** = ((int)0x80D9), **ColorTableRedSize** = ((int)0x80DA),
ColorTableGreenSize = ((int)0x80DB), **ColorTableBlueSize** = ((int)0x80DC), **ColorTableAlphaSize** = ((int)0x80DD), **ColorTableLuminanceSize** = ((int)0x80DE),
ColorTableIntensitySize = ((int)0x80DF), **Bgr** = ((int)0x80E0), **Bgra** = ((int)0x80E1), **MaxElementsVertices** = ((int)0x80E8),
MaxElementsIndices = ((int)0x80E9), **ClampToEdge** = ((int)0x812F), **TextureMinLod** = ((int)0x813A), **TextureMaxLod** = ((int)0x813B),
TextureBaseLevel = ((int)0x813C), **TextureMaxLevel** = ((int)0x813D), **ConstantBorder** = ((int)0x8151), **ReplicateBorder** = ((int)0x8153),
ConvolutionBorderColor = ((int)0x8154), **LightModelColorControl** = ((int)0x81F8), **SingleColor** = ((int)0x81F9), **SeparateSpecularColor** = ((int)0x81FA),
UnsignedByte233Rev = ((int)0x8362), **UnsignedShort565** = ((int)0x8363), **UnsignedShort565Rev** = ((int)0x8364), **UnsignedShort4444Rev** = ((int)0x8365),
UnsignedShort1555Rev = ((int)0x8366), **UnsignedInt8888Rev** = ((int)0x8367), **UnsignedInt2101010Rev** = ((int)0x8368), **AliasedPointSizeRange** = ((int)0x846D),
AliasedLineWidthRange = ((int)0x846E) }

- enum **Version12Deprecated** {
 - RescaleNormal** = ((int)0x803A), **LightModelColorControl** = ((int)0x81F8),
 - SingleColor** = ((int)0x81F9), **SeparateSpecularColor** = ((int)0x81FA),
 - AliasedPointSizeRange** = ((int)0x846D) }
- enum **Version13** {
 - Multisample** = ((int)0x809D), **SampleAlphaToCoverage** = ((int)0x809E),
 - SampleAlphaToOne** = ((int)0x809F), **SampleCoverage** = ((int)0x80A0),
 - SampleBuffers** = ((int)0x80A8), **Samples** = ((int)0x80A9), **SampleCoverageValue** = ((int)0x80AA), **SampleCoverageInvert** = ((int)0x80AB),
 - ClampToBorder** = ((int)0x812D), **Texture0** = ((int)0x84C0), **Texture1** = ((int)0x84C1), **Texture2** = ((int)0x84C2),
 - Texture3** = ((int)0x84C3), **Texture4** = ((int)0x84C4), **Texture5** = ((int)0x84C5), **Texture6** = ((int)0x84C6),
 - Texture7** = ((int)0x84C7), **Texture8** = ((int)0x84C8), **Texture9** = ((int)0x84C9), **Texture10** = ((int)0x84CA),
 - Texture11** = ((int)0x84CB), **Texture12** = ((int)0x84CC), **Texture13** = ((int)0x84CD), **Texture14** = ((int)0x84CE),
 - Texture15** = ((int)0x84CF), **Texture16** = ((int)0x84D0), **Texture17** = ((int)0x84D1), **Texture18** = ((int)0x84D2),
 - Texture19** = ((int)0x84D3), **Texture20** = ((int)0x84D4), **Texture21** = ((int)0x84D5), **Texture22** = ((int)0x84D6),
 - Texture23** = ((int)0x84D7), **Texture24** = ((int)0x84D8), **Texture25** = ((int)0x84D9), **Texture26** = ((int)0x84DA),
 - Texture27** = ((int)0x84DB), **Texture28** = ((int)0x84DC), **Texture29** = ((int)0x84DD), **Texture30** = ((int)0x84DE),
 - Texture31** = ((int)0x84DF), **ActiveTexture** = ((int)0x84E0), **CompressedRgb** = ((int)0x84ED), **CompressedRgba** = ((int)0x84EE),
 - TextureCompressionHint** = ((int)0x84EF), **TextureCubeMap** = ((int)0x8513), **TextureBindingCubeMap** = ((int)0x8514), **TextureCubeMapPositiveX** = ((int)0x8515),
 - TextureCubeMapNegativeX** = ((int)0x8516), **TextureCubeMapPositiveY** = ((int)0x8517), **TextureCubeMapNegativeY** = ((int)0x8518), **TextureCubeMapPositiveZ** = ((int)0x8519),
 - TextureCubeMapNegativeZ** = ((int)0x851A), **ProxyTextureCubeMap** = ((int)0x851B), **MaxCubeMapTextureSize** = ((int)0x851C), **TextureCompressedImageSize** = ((int)0x86A0),
 - TextureCompressed** = ((int)0x86A1), **NumCompressedTextureFormats** = ((int)0x86A2), **CompressedTextureFormats** = ((int)0x86A3) }

- enum **Version13Deprecated** {
 - MultisampleBit** = ((int)0x20000000), **ClientActiveTexture** = ((int)0x84E1),
MaxTextureUnits = ((int)0x84E2), **TransposeModelviewMatrix** = ((int)0x84E3),
 - TransposeProjectionMatrix** = ((int)0x84E4), **TransposeTextureMatrix** = ((int)0x84E5), **TransposeColorMatrix** = ((int)0x84E6), **Subtract** = ((int)0x84E7),
 - CompressedAlpha** = ((int)0x84E9), **CompressedLuminance** = ((int)0x84EA),
CompressedLuminanceAlpha = ((int)0x84EB), **CompressedIntensity** = ((int)0x84EC),
 - NormalMap** = ((int)0x8511), **ReflectionMap** = ((int)0x8512), **Combine** = ((int)0x8570), **CombineRgb** = ((int)0x8571),
 - CombineAlpha** = ((int)0x8572), **RgbScale** = ((int)0x8573), **AddSigned** = ((int)0x8574), **Interpolate** = ((int)0x8575),
 - Constant** = ((int)0x8576), **PrimaryColor** = ((int)0x8577), **Previous** = ((int)0x8578), **Source0Rgb** = ((int)0x8580),
 - Source1Rgb** = ((int)0x8581), **Source2Rgb** = ((int)0x8582), **Source0Alpha** = ((int)0x8588), **Source1Alpha** = ((int)0x8589),
 - Source2Alpha** = ((int)0x858A), **Operand0Rgb** = ((int)0x8590),
Operand1Rgb = ((int)0x8591), **Operand2Rgb** = ((int)0x8592),
 - Operand0Alpha** = ((int)0x8598), **Operand1Alpha** = ((int)0x8599),
Operand2Alpha = ((int)0x859A), **Dot3Rgb** = ((int)0x86AE),
 - Dot3Rgba** = ((int)0x86AF) }
- enum **Version14** {
 - BlendDstRgb** = ((int)0x80C8), **BlendSrcRgb** = ((int)0x80C9), **BlendDstAlpha** = ((int)0x80CA), **BlendSrcAlpha** = ((int)0x80CB),
 - PointSizeMin** = ((int)0x8126), **PointSizeMax** = ((int)0x8127), **PointFadeThresholdSize** = ((int)0x8128), **PointDistanceAttenuation** = ((int)0x8129),
 - GenerateMipmap** = ((int)0x8191), **GenerateMipmapHint** = ((int)0x8192),
DepthComponent16 = ((int)0x81A5), **DepthComponent24** = ((int)0x81A6),
 - DepthComponent32** = ((int)0x81A7), **MirroredRepeat** = ((int)0x8370), **MaxTextureLodBias** = ((int)0x84FD), **TextureLodBias** = ((int)0x8501),
 - IncrWrap** = ((int)0x8507), **DecrWrap** = ((int)0x8508), **TextureDepthSize** = ((int)0x884A), **TextureCompareMode** = ((int)0x884C),
 - TextureCompareFunc** = ((int)0x884D) }
- enum **Version14Deprecated** {
 - PointSizeMin** = ((int)0x8126), **PointSizeMax** = ((int)0x8127), **PointDistanceAttenuation** = ((int)0x8129), **GenerateMipmap** = ((int)0x8191),

```

GenerateMipmapHint = ((int)0x8192), FogCoordinateSource =
((int)0x8450), FogCoordinate = ((int)0x8451), FragmentDepth =
((int)0x8452),

CurrentFogCoordinate = ((int)0x8453), FogCoordinateArrayType =
((int)0x8454), FogCoordinateArrayStride = ((int)0x8455), FogCoordi-
nateArrayPointer = ((int)0x8456),

FogCoordinateArray = ((int)0x8457), ColorSum = ((int)0x8458), Cur-
rentSecondaryColor = ((int)0x8459), SecondaryColorArraySize =
((int)0x845A),

SecondaryColorArrayType = ((int)0x845B), SecondaryColorArrayStride =
((int)0x845C), SecondaryColorArrayPointer = ((int)0x845D), Secondary-
ColorArray = ((int)0x845E),

TextureFilterControl = ((int)0x8500), DepthTextureMode = ((int)0x884B),
CompareRToTexture = ((int)0x884E) }
• enum Version15 {
    BufferSize = ((int)0x8764), BufferUsage = ((int)0x8765), QueryCounterBits
    = ((int)0x8864), CurrentQuery = ((int)0x8865),

    QueryResult = ((int)0x8866), QueryResultAvailable = ((int)0x8867), Array-
    Buffer = ((int)0x8892), ElementArrayBuffer = ((int)0x8893),

    ArrayBufferBinding = ((int)0x8894), ElementArrayBufferBinding =
    ((int)0x8895), VertexAttribArrayBufferBinding = ((int)0x889F), ReadOnly
    = ((int)0x88B8),

    WriteOnly = ((int)0x88B9), ReadWrite = ((int)0x88BA), BufferAccess =
    ((int)0x88BB), BufferMapped = ((int)0x88BC),

    BufferMapPointer = ((int)0x88BD), StreamDraw = ((int)0x88E0), Stream-
    Read = ((int)0x88E1), StreamCopy = ((int)0x88E2),

    StaticDraw = ((int)0x88E4), StaticRead = ((int)0x88E5), StaticCopy =
    ((int)0x88E6), DynamicDraw = ((int)0x88E8),

    DynamicRead = ((int)0x88E9), DynamicCopy = ((int)0x88EA), Sam-
    plesPassed = ((int)0x8914) }
• enum Version15Deprecated {
    FogCoordSrc = ((int)0x8450), FogCoord = ((int)0x8451), CurrentFogCoord
    = ((int)0x8453), FogCoordArrayType = ((int)0x8454),

    FogCoordArrayStride = ((int)0x8455), FogCoordArrayPointer =
    ((int)0x8456), FogCoordArray = ((int)0x8457), Src0Rgb = ((int)0x8580),

    Src1Rgb = ((int)0x8581), Src2Rgb = ((int)0x8582), Src0Alpha =
    ((int)0x8588), Src1Alpha = ((int)0x8589),

    Src2Alpha = ((int)0x858A), VertexArrayBufferBinding = ((int)0x8896),
    NormalArrayBufferBinding = ((int)0x8897), ColorArrayBufferBinding =
    ((int)0x8898),

```

IndexArrayBufferBinding = ((int)0x8899), **TextureCoordArrayBufferBinding** = ((int)0x889A), **EdgeFlagArrayBufferBinding** = ((int)0x889B), **SecondaryColorArrayBufferBinding** = ((int)0x889C),

FogCoordArrayBufferBinding = ((int)0x889D), **FogCoordinateArrayBufferBinding** = ((int)0x889D), **WeightArrayBufferBinding** = ((int)0x889E)
}

• enum **Version20** {

BlendEquationRgb = ((int)0x8009), **VertexAttribArrayEnabled** = ((int)0x8622), **VertexAttribArraySize** = ((int)0x8623), **VertexAttribArrayStride** = ((int)0x8624),

VertexAttribArrayType = ((int)0x8625), **CurrentVertexAttrib** = ((int)0x8626), **VertexProgramPointSize** = ((int)0x8642), **VertexAttribArrayPointer** = ((int)0x8645),

StencilBackFunc = ((int)0x8800), **StencilBackFail** = ((int)0x8801), **StencilBackPassDepthFail** = ((int)0x8802), **StencilBackPassDepthPass** = ((int)0x8803),

MaxDrawBuffers = ((int)0x8824), **DrawBuffer0** = ((int)0x8825), **DrawBuffer1** = ((int)0x8826), **DrawBuffer2** = ((int)0x8827),

DrawBuffer3 = ((int)0x8828), **DrawBuffer4** = ((int)0x8829), **DrawBuffer5** = ((int)0x882A), **DrawBuffer6** = ((int)0x882B),

DrawBuffer7 = ((int)0x882C), **DrawBuffer8** = ((int)0x882D), **DrawBuffer9** = ((int)0x882E), **DrawBuffer10** = ((int)0x882F),

DrawBuffer11 = ((int)0x8830), **DrawBuffer12** = ((int)0x8831), **DrawBuffer13** = ((int)0x8832), **DrawBuffer14** = ((int)0x8833),

DrawBuffer15 = ((int)0x8834), **BlendEquationAlpha** = ((int)0x883D), **MaxVertexAttribs** = ((int)0x8869), **VertexAttribArrayNormalized** = ((int)0x886A),

MaxTextureImageUnits = ((int)0x8872), **FragmentShader** = ((int)0x8B30), **VertexShader** = ((int)0x8B31), **MaxFragmentUniformComponents** = ((int)0x8B49),

MaxVertexUniformComponents = ((int)0x8B4A), **MaxVaryingFloats** = ((int)0x8B4B), **MaxVertexTextureImageUnits** = ((int)0x8B4C), **MaxCombinedTextureImageUnits** = ((int)0x8B4D),

ShaderType = ((int)0x8B4F), **FloatVec2** = ((int)0x8B50), **FloatVec3** = ((int)0x8B51), **FloatVec4** = ((int)0x8B52),

IntVec2 = ((int)0x8B53), **IntVec3** = ((int)0x8B54), **IntVec4** = ((int)0x8B55), **Bool** = ((int)0x8B56),

BoolVec2 = ((int)0x8B57), **BoolVec3** = ((int)0x8B58), **BoolVec4** = ((int)0x8B59), **FloatMat2** = ((int)0x8B5A),

FloatMat3 = ((int)0x8B5B), **FloatMat4** = ((int)0x8B5C), **Sampler1D** = ((int)0x8B5D), **Sampler2D** = ((int)0x8B5E),

```

Sampler3D = ((int)0x8B5F), SamplerCube = ((int)0x8B60), Sampler1DShadow = ((int)0x8B61), Sampler2DShadow = ((int)0x8B62),

DeleteStatus = ((int)0x8B80), CompileStatus = ((int)0x8B81), LinkStatus = ((int)0x8B82), ValidateStatus = ((int)0x8B83),

InfoLogLength = ((int)0x8B84), AttachedShaders = ((int)0x8B85), ActiveUniforms = ((int)0x8B86), ActiveUniformMaxLength = ((int)0x8B87),

ShaderSourceLength = ((int)0x8B88), ActiveAttributes = ((int)0x8B89), ActiveAttributeMaxLength = ((int)0x8B8A), FragmentShaderDerivativeHint = ((int)0x8B8B),

ShadingLanguageVersion = ((int)0x8B8C), CurrentProgram = ((int)0x8B8D), PointSpriteCoordOrigin = ((int)0x8CA0), LowerLeft = ((int)0x8CA1),

UpperLeft = ((int)0x8CA2), StencilBackRef = ((int)0x8CA3), StencilBackValueMask = ((int)0x8CA4), StencilBackWritemask = ((int)0x8CA5) }

• enum Version20Deprecated { VertexProgramTwoSide = ((int)0x8643), PointSprite = ((int)0x8861), CoordReplace = ((int)0x8862), MaxTextureCoords = ((int)0x8871) }

• enum Version21 {

PixelPackBuffer = ((int)0x88EB), PixelUnpackBuffer = ((int)0x88EC), PixelPackBufferBinding = ((int)0x88ED), PixelUnpackBufferBinding = ((int)0x88EF),

FloatMat2x3 = ((int)0x8B65), FloatMat2x4 = ((int)0x8B66), FloatMat3x2 = ((int)0x8B67), FloatMat3x4 = ((int)0x8B68),

FloatMat4x2 = ((int)0x8B69), FloatMat4x3 = ((int)0x8B6A), Srgb = ((int)0x8C40), Srgb8 = ((int)0x8C41),

SrgbAlpha = ((int)0x8C42), Srgb8Alpha8 = ((int)0x8C43), CompressedSrgb = ((int)0x8C48), CompressedSrgbAlpha = ((int)0x8C49) }

• enum Version21Deprecated {

CurrentRasterSecondaryColor = ((int)0x845F), SluminanceAlpha = ((int)0x8C44), Sluminance8Alpha8 = ((int)0x8C45), Sluminance = ((int)0x8C46),

Sluminance8 = ((int)0x8C47), CompressedSluminance = ((int)0x8C4A), CompressedSluminanceAlpha = ((int)0x8C4B) }

• enum Version30 {

ContextFlagForwardCompatibleBit = ((int)0x0001), MapReadBit = ((int)0x0001), MapWriteBit = ((int)0x0002), MapInvalidateRangeBit = ((int)0x0004),

MapInvalidateBufferBit = ((int)0x0008), MapFlushExplicitBit = ((int)0x0010), MapUnsynchronizedBit = ((int)0x0020), InvalidFramebufferOperation = ((int)0x0506),

```

MaxClipDistances = ((int)0x0D32), **HalfFloat** = ((int)0x140B), **ClipDistance0** = ((int)0x3000), **ClipDistance1** = ((int)0x3001),
ClipDistance2 = ((int)0x3002), **ClipDistance3** = ((int)0x3003), **ClipDistance4** = ((int)0x3004), **ClipDistance5** = ((int)0x3005),
ClipDistance6 = ((int)0x3006), **ClipDistance7** = ((int)0x3007), **FramebufferAttachmentColorEncoding** = ((int)0x8210), **FramebufferAttachmentComponentType** = ((int)0x8211),
FramebufferAttachmentRedSize = ((int)0x8212), **FramebufferAttachmentGreenSize** = ((int)0x8213), **FramebufferAttachmentBlueSize** = ((int)0x8214), **FramebufferAttachmentAlphaSize** = ((int)0x8215),
FramebufferAttachmentDepthSize = ((int)0x8216), **FramebufferAttachmentStencilSize** = ((int)0x8217), **FramebufferDefault** = ((int)0x8218), **FramebufferUndefined** = ((int)0x8219),
DepthStencilAttachment = ((int)0x821A), **MajorVersion** = ((int)0x821B), **MinorVersion** = ((int)0x821C), **NumExtensions** = ((int)0x821D),
ContextFlags = ((int)0x821E), **DepthBuffer** = ((int)0x8223), **StencilBuffer** = ((int)0x8224), **CompressedRed** = ((int)0x8225),
CompressedRg = ((int)0x8226), **Rg** = ((int)0x8227), **RgInteger** = ((int)0x8228), **R8** = ((int)0x8229),
R16 = ((int)0x822A), **Rg8** = ((int)0x822B), **Rg16** = ((int)0x822C), **R16f** = ((int)0x822D),
R32f = ((int)0x822E), **Rg16f** = ((int)0x822F), **Rg32f** = ((int)0x8230), **R8i** = ((int)0x8231),
R8ui = ((int)0x8232), **R16i** = ((int)0x8233), **R16ui** = ((int)0x8234), **R32i** = ((int)0x8235),
R32ui = ((int)0x8236), **Rg8i** = ((int)0x8237), **Rg8ui** = ((int)0x8238), **Rg16i** = ((int)0x8239),
Rg16ui = ((int)0x823A), **Rg32i** = ((int)0x823B), **Rg32ui** = ((int)0x823C), **MaxRenderbufferSize** = ((int)0x84E8),
DepthStencil = ((int)0x84F9), **UnsignedInt248** = ((int)0x84FA), **VertexArrayBinding** = ((int)0x85B5), **Rgba32f** = ((int)0x8814),
Rgb32f = ((int)0x8815), **Rgba16f** = ((int)0x881A), **Rgb16f** = ((int)0x881B), **CompareRefToTexture** = ((int)0x884E),
Depth24Stencil8 = ((int)0x88F0), **TextureStencilSize** = ((int)0x88F1), **VertexAttribArrayInteger** = ((int)0x88FD), **MaxArrayTextureLayers** = ((int)0x88FF),
MinProgramTexelOffset = ((int)0x8904), **MaxProgramTexelOffset** = ((int)0x8905), **ClampReadColor** = ((int)0x891C), **FixedOnly** = ((int)0x891D),
MaxVaryingComponents = ((int)0x8B4B), **TextureRedType** = ((int)0x8C10), **TextureGreenType** = ((int)0x8C11), **TextureBlueType** = ((int)0x8C12),

TextureAlphaType = ((int)0x8C13), **TextureDepthType** = ((int)0x8C16), **UnsignedNormalized** = ((int)0x8C17), **Texture1DArray** = ((int)0x8C18),
ProxyTexture1DArray = ((int)0x8C19), **Texture2DArray** = ((int)0x8C1A),
ProxyTexture2DArray = ((int)0x8C1B), **TextureBinding1DArray** = ((int)0x8C1C),
TextureBinding2DArray = ((int)0x8C1D), **R11fG11fB10f** = ((int)0x8C3A),
UnsignedInt10F11F11FRev = ((int)0x8C3B), **Rgb9E5** = ((int)0x8C3D),
UnsignedInt5999Rev = ((int)0x8C3E), **TextureSharedSize** = ((int)0x8C3F),
TransformFeedbackVaryingMaxLength = ((int)0x8C76), **TransformFeedbackBufferMode** = ((int)0x8C7F),
MaxTransformFeedbackSeparateComponents = ((int)0x8C80), **TransformFeedbackVaryings** = ((int)0x8C83), **TransformFeedbackBufferStart** = ((int)0x8C84), **TransformFeedbackBufferSize** = ((int)0x8C85),
PrimitivesGenerated = ((int)0x8C87), **TransformFeedbackPrimitivesWritten** = ((int)0x8C88), **RasterizerDiscard** = ((int)0x8C89), **MaxTransformFeedbackInterleavedComponents** = ((int)0x8C8A),
MaxTransformFeedbackSeparateAttribs = ((int)0x8C8B), **InterleavedAttribs** = ((int)0x8C8C), **SeparateAttribs** = ((int)0x8C8D), **TransformFeedbackBuffer** = ((int)0x8C8E),
TransformFeedbackBufferBinding = ((int)0x8C8F), **DrawFramebufferBinding** = ((int)0x8CA6), **FramebufferBinding** = ((int)0x8CA6), **RenderbufferBinding** = ((int)0x8CA7),
ReadFramebuffer = ((int)0x8CA8), **DrawFramebuffer** = ((int)0x8CA9), **ReadFramebufferBinding** = ((int)0x8CAA), **RenderbufferSamples** = ((int)0x8CAB),
DepthComponent32f = ((int)0x8CAC), **Depth32fStencil8** = ((int)0x8CAD), **FramebufferAttachmentObjectType** = ((int)0x8CD0), **FramebufferAttachmentObjectName** = ((int)0x8CD1),
FramebufferAttachmentTextureLevel = ((int)0x8CD2), **FramebufferAttachmentTextureCubeMapFace** = ((int)0x8CD3), **FramebufferAttachmentTextureLayer** = ((int)0x8CD4), **FramebufferComplete** = ((int)0x8CD5),
FramebufferIncompleteAttachment = ((int)0x8CD6), **FramebufferIncompleteMissingAttachment** = ((int)0x8CD7), **FramebufferIncompleteDrawBuffer** = ((int)0x8CDB), **FramebufferIncompleteReadBuffer** = ((int)0x8CDC),
FramebufferUnsupported = ((int)0x8CDD), **MaxColorAttachments** = ((int)0x8CDF), **ColorAttachment0** = ((int)0x8CE0), **ColorAttachment1** = ((int)0x8CE1),
ColorAttachment2 = ((int)0x8CE2), **ColorAttachment3** = ((int)0x8CE3), **ColorAttachment4** = ((int)0x8CE4), **ColorAttachment5** = ((int)0x8CE5),
ColorAttachment6 = ((int)0x8CE6), **ColorAttachment7** = ((int)0x8CE7), **ColorAttachment8** = ((int)0x8CE8), **ColorAttachment9** = ((int)0x8CE9),

ColorAttachment10 = ((int)0x8CEA), **ColorAttachment11** = ((int)0x8CEB),
ColorAttachment12 = ((int)0x8CEC), **ColorAttachment13** = ((int)0x8CED),
ColorAttachment14 = ((int)0x8CEE), **ColorAttachment15** = ((int)0x8CEF),
DepthAttachment = ((int)0x8D00), **StencilAttachment** = ((int)0x8D20),
Framebuffer = ((int)0x8D40), **Renderbuffer** = ((int)0x8D41), **Renderbuffer-Width** = ((int)0x8D42), **RenderbufferHeight** = ((int)0x8D43),
RenderbufferInternalFormat = ((int)0x8D44), **StencilIndex1** = ((int)0x8D46), **StencilIndex4** = ((int)0x8D47), **StencilIndex8** = ((int)0x8D48),
StencilIndex16 = ((int)0x8D49), **RenderbufferRedSize** = ((int)0x8D50), **RenderbufferGreenSize** = ((int)0x8D51), **RenderbufferBlueSize** = ((int)0x8D52),
RenderbufferAlphaSize = ((int)0x8D53), **RenderbufferDepthSize** = ((int)0x8D54), **RenderbufferStencilSize** = ((int)0x8D55), **FramebufferIncompleteMultisample** = ((int)0x8D56),
MaxSamples = ((int)0x8D57), **Rgba32ui** = ((int)0x8D70), **Rgb32ui** = ((int)0x8D71), **Rgba16ui** = ((int)0x8D76),
Rgb16ui = ((int)0x8D77), **Rgba8ui** = ((int)0x8D7C), **Rgb8ui** = ((int)0x8D7D),
Rgba32i = ((int)0x8D82),
Rgb32i = ((int)0x8D83), **Rgba16i** = ((int)0x8D88), **Rgb16i** = ((int)0x8D89),
Rgba8i = ((int)0x8D8E),
Rgb8i = ((int)0x8D8F), **RedInteger** = ((int)0x8D94), **GreenInteger** = ((int)0x8D95), **BlueInteger** = ((int)0x8D96),
RgbInteger = ((int)0x8D98), **RgbaInteger** = ((int)0x8D99), **BgrInteger** = ((int)0x8D9A), **BgraInteger** = ((int)0x8D9B),
Float32UnsignedInt248Rev = ((int)0x8DAD), **FramebufferSrgb** = ((int)0x8DB9), **CompressedRedRgtc1** = ((int)0x8DBB), **CompressedSignedRedRgtc1** = ((int)0x8DBC),
CompressedRgRgtc2 = ((int)0x8DBD), **CompressedSignedRgRgtc2** = ((int)0x8DBE), **Sampler1DArray** = ((int)0x8DC0), **Sampler2DArray** = ((int)0x8DC1),
Sampler1DArrayShadow = ((int)0x8DC3), **Sampler2DArrayShadow** = ((int)0x8DC4), **SamplerCubeShadow** = ((int)0x8DC5), **UnsignedIntVec2** = ((int)0x8DC6),
UnsignedIntVec3 = ((int)0x8DC7), **UnsignedIntVec4** = ((int)0x8DC8),
IntSampler1D = ((int)0x8DC9), **IntSampler2D** = ((int)0x8DCA),
IntSampler3D = ((int)0x8DCB), **IntSamplerCube** = ((int)0x8DCC), **IntSampler1DArray** = ((int)0x8DCE), **IntSampler2DArray** = ((int)0x8DCF),
UnsignedIntSampler1D = ((int)0x8DD1), **UnsignedIntSampler2D** = ((int)0x8DD2), **UnsignedIntSampler3D** = ((int)0x8DD3), **UnsignedIntSamplerCube** = ((int)0x8DD4),

```

UnsignedIntSampler1DArray = ((int)0x8DD6), UnsignedIntSampler2DArray = ((int)0x8DD7), QueryWait = ((int)0x8E13), QueryNoWait = ((int)0x8E14),
QueryByRegionWait = ((int)0x8E15), QueryByRegionNoWait = ((int)0x8E16), BufferAccessFlags = ((int)0x911F), BufferMapLength = ((int)0x9120),
BufferMapOffset = ((int)0x9121) }
• enum Version30Deprecated { ClampVertexColor = ((int)0x891A), ClampFragmentColor = ((int)0x891B), AlphaInteger = ((int)0x8D97) }
• enum Version31 {
TextureRectangle = ((int)0x84F5), TextureBindingRectangle = ((int)0x84F6), ProxyTextureRectangle = ((int)0x84F7), MaxRectangleTextureSize = ((int)0x84F8),
UniformBuffer = ((int)0x8A11), UniformBufferBinding = ((int)0x8A28), UniformBufferStart = ((int)0x8A29), UniformBufferSize = ((int)0x8A2A),
MaxVertexUniformBlocks = ((int)0x8A2B), MaxFragmentUniformBlocks = ((int)0x8A2D), MaxCombinedUniformBlocks = ((int)0x8A2E), MaxUniformBufferBindings = ((int)0x8A2F),
MaxUniformBlockSize = ((int)0x8A30), MaxCombinedVertexUniformComponents = ((int)0x8A31), MaxCombinedFragmentUniformComponents = ((int)0x8A33), UniformBufferOffsetAlignment = ((int)0x8A34),
ActiveUniformBlockMaxNameLength = ((int)0x8A35), ActiveUniformBlocks = ((int)0x8A36), UniformType = ((int)0x8A37), UniformSize = ((int)0x8A38),
UniformNameLength = ((int)0x8A39), UniformBlockIndex = ((int)0x8A3A), UniformOffset = ((int)0x8A3B), UniformArrayStride = ((int)0x8A3C),
UniformMatrixStride = ((int)0x8A3D), UniformIsRowMajor = ((int)0x8A3E), UniformBlockBinding = ((int)0x8A3F), UniformBlockDataSize = ((int)0x8A40),
UniformBlockNameLength = ((int)0x8A41), UniformBlockActiveUniforms = ((int)0x8A42), UniformBlockActiveUniformIndices = ((int)0x8A43), UniformBlockReferencedByVertexShader = ((int)0x8A44),
UniformBlockReferencedByFragmentShader = ((int)0x8A46), Sampler2DRect = ((int)0x8B63), Sampler2DRectShadow = ((int)0x8B64), TextureBuffer = ((int)0x8C2A),
MaxTextureBufferSize = ((int)0x8C2B), TextureBindingBuffer = ((int)0x8C2C), TextureBufferDataStoreBinding = ((int)0x8C2D), TextureBufferFormat = ((int)0x8C2E),
SamplerBuffer = ((int)0x8DC2), IntSampler2DRect = ((int)0x8DCD), IntSamplerBuffer = ((int)0x8DD0), UnsignedIntSampler2DRect = ((int)0x8DD5),

```

```

UnsignedIntSamplerBuffer = ((int)0x8DD8), CopyReadBuffer =
((int)0x8F36), CopyWriteBuffer = ((int)0x8F37), RedSnorm = ((int)0x8F90),
RgSnorm = ((int)0x8F91), RgbSnorm = ((int)0x8F92), RgbaSnorm =
((int)0x8F93), R8Snorm = ((int)0x8F94),
Rg8Snorm = ((int)0x8F95), Rgb8Snorm = ((int)0x8F96), Rgba8Snorm =
((int)0x8F97), R16Snorm = ((int)0x8F98),
Rg16Snorm = ((int)0x8F99), Rgb16Snorm = ((int)0x8F9A), Rgba16Snorm =
((int)0x8F9B), SignedNormalized = ((int)0x8F9C),
PrimitiveRestart = ((int)0x8F9D), PrimitiveRestartIndex = ((int)0x8F9E), In-
validIndex = unchecked((int)0xFFFFFFFF) }
• enum Version32 {
ContextCoreProfileBit = ((int)0x00000001), SyncFlushCommandsBit =
((int)0x00000001), ContextCompatibilityProfileBit = ((int)0x00000002),
LinesAdjacency = ((int)0x000A),
LineStripAdjacency = ((int)0x000B), TrianglesAdjacency = ((int)0x000C),
TriangleStripAdjacency = ((int)0x000D), ProgramPointSize = ((int)0x8642),
DepthClamp = ((int)0x864F), TextureCubeMapSeamless = ((int)0x884F),
GeometryVerticesOut = ((int)0x8916), GeometryInputType = ((int)0x8917),
GeometryOutputType = ((int)0x8918), MaxVaryingComponents =
((int)0x8B4B), MaxGeometryTextureImageUnits = ((int)0x8C29), Frame-
bufferAttachmentTextureLayer = ((int)0x8CD4),
FramebufferAttachmentLayered = ((int)0x8DA7), FramebufferIncomplete-
LayerTargets = ((int)0x8DA8), GeometryShader = ((int)0x8DD9), MaxGe-
ometryUniformComponents = ((int)0x8DDF),
MaxGeometryOutputVertices = ((int)0x8DE0), MaxGeometryTotalOutput-
Components = ((int)0x8DE1), QuadsFollowProvokingVertexConvention =
((int)0x8E4C), FirstVertexConvention = ((int)0x8E4D),
LastVertexConvention = ((int)0x8E4E), ProvokingVertex = ((int)0x8E4F),
SamplePosition = ((int)0x8E50), SampleMask = ((int)0x8E51),
SampleMaskValue = ((int)0x8E52), MaxSampleMaskWords = ((int)0x8E59),
Texture2DMultisample = ((int)0x9100), ProxyTexture2DMultisample =
((int)0x9101),
Texture2DMultisampleArray = ((int)0x9102), ProxyTex-
ture2DMultisampleArray = ((int)0x9103), TextureBinding2DMultisample =
((int)0x9104), TextureBinding2DMultisampleArray = ((int)0x9105),
TextureSamples = ((int)0x9106), TextureFixedSampleLocations =
((int)0x9107), Sampler2DMultisample = ((int)0x9108), IntSam-
pler2DMultisample = ((int)0x9109),
UnsignedIntSampler2DMultisample = ((int)0x910A), Sam-
pler2DMultisampleArray = ((int)0x910B), IntSampler2DMultisampleArray
= ((int)0x910C), UnsignedIntSampler2DMultisampleArray = ((int)0x910D),

```

```

MaxColorTextureSamples = ((int)0x910E), MaxDepthTextureSamples =
((int)0x910F), MaxIntegerSamples = ((int)0x9110), MaxServerWaitTimeout
= ((int)0x9111),

ObjectType = ((int)0x9112), SyncCondition = ((int)0x9113), SyncStatus =
((int)0x9114), SyncFlags = ((int)0x9115),

SyncFence = ((int)0x9116), SyncGpuCommandsComplete = ((int)0x9117),
Unsignaled = ((int)0x9118), Signaled = ((int)0x9119),

AlreadySignaled = ((int)0x911A), TimeoutExpired = ((int)0x911B), Condi-
tionSatisfied = ((int)0x911C), WaitFailed = ((int)0x911D),

MaxVertexOutputComponents = ((int)0x9122), MaxGeometryInputCom-
ponents = ((int)0x9123), MaxGeometryOutputComponents = ((int)0x9124),
MaxFragmentInputComponents = ((int)0x9125),

ContextProfileMask = ((int)0x9126), TimeoutIgnored =
unchecked((int)0xFFFFFFFFFFFFFFFF) }

• enum VertexAttribPointerType {
Byte = ((int)0x1400), UnsignedByte = ((int)0x1401), Short = ((int)0x1402),
UnsignedShort = ((int)0x1403),

Int = ((int)0x1404), UnsignedInt = ((int)0x1405) }

• enum VertexAttribParameter {
ArrayEnabled = ((int)0x8622), ArraySize = ((int)0x8623), ArrayStride =
((int)0x8624), ArrayType = ((int)0x8625),

CurrentVertexAttrib = ((int)0x8626), ArrayNormalized = ((int)0x886A),
VertexAttribArrayInteger = ((int)0x88FD) }

• enum VertexAttribParameterArb {
ArrayEnabled = ((int)0x8622), ArraySize = ((int)0x8623), ArrayStride =
((int)0x8624), ArrayType = ((int)0x8625),

CurrentVertexAttrib = ((int)0x8626), ArrayNormalized = ((int)0x886A), Ar-
rayDivisor = ((int)0x88FE) }

• enum VertexAttribPointerParameter { ArrayPointer = ((int)0x8645) }
• enum VertexAttribPointerParameterArb { ArrayPointer = ((int)0x8645) }
• enum VertexAttribPointerType {
Byte = ((int)0x1400), UnsignedByte = ((int)0x1401), Short = ((int)0x1402),
UnsignedShort = ((int)0x1403),

Int = ((int)0x1404), UnsignedInt = ((int)0x1405), Float = ((int)0x1406), Dou-
ble = ((int)0x140A),

HalfFloat = ((int)0x140B) }

• enum VertexAttribPointerTypeArb {
Byte = ((int)0x1400), UnsignedByte = ((int)0x1401), Short = ((int)0x1402),
UnsignedShort = ((int)0x1403),

Int = ((int)0x1404), UnsignedInt = ((int)0x1405), Float = ((int)0x1406), Dou-
ble = ((int)0x140A) }

```

- enum **VertexPointerType** {
 Short = ((int)0x1402), **Int** = ((int)0x1404), **Float** = ((int)0x1406), **Double** = ((int)0x140A),
 HalfFloat = ((int)0x140B) }
- enum **WinPhongShading** { **PhongWin** = ((int)0x80EA), **PhongHintWin** = ((int)0x80EB) }
- enum **WinSpecularFog** { **FogSpecularTextureWin** = ((int)0x80EC) }

4.9 Package OpenTK.Input

Classes

- class [GamePad](#)
Provides access to [GamePad](#) devices. Note: this API is not implemented yet.
- struct [GamePadState](#)
Encapsulates the state of a [GamePad](#) device.
- interface [IInputDevice](#)
Defines a common interface for all input devices.
- interface [IInputDriver](#)
Defines the interface for an input driver.
- interface [IJoystickDriver](#)
Defines the interface for [JoystickDevice](#) drivers.
- interface [IKeyboardDriver](#)
Defines the interface for [KeyboardDevice](#) drivers.
- interface [IMouseDriver](#)
Defines the interface for [MouseDevice](#) drivers.
- class [JoystickDevice](#)
Represents a joystick device and provides methods to query its status.
- class [JoystickEventArgs](#)
The base class for [JoystickDevice](#) event arguments.
- class [JoystickButtonEventArgs](#)

Provides data for the *JoystickDevice.ButtonDown* and *JoystickDevice.ButtonUp* events. This class is cached for performance reasons - avoid storing references outside the scope of the event.

- class [JoystickMoveEventArgs](#)
*Provides data for the *JoystickDevice.Move* event. This class is cached for performance reasons - avoid storing references outside the scope of the event.*
- class [JoystickButtonCollection](#)
Defines a collection of JoystickButtons.
- class [JoystickAxisCollection](#)
Defines a collection of JoystickAxes.
- class [KeyboardDevice](#)
Represents a keyboard device and provides methods to query its status.
- class [KeyboardKeyEventArgs](#)
*Defines the event data for *KeyboardDevice* events.*
- struct [KeyboardState](#)
Encapsulates the state of a Keyboard device.
- class [MouseDevice](#)
Represents a mouse device and provides methods to query its status.
- class [MouseEventArgs](#)
*Defines the event data for *MouseDevice* events.*
- class [MouseMoveEventArgs](#)
*Defines the event data for *MouseDevice.Move* events.*
- class [MouseButtonEventArgs](#)
*Defines the event data for *MouseDevice.ButtonDown* and *MouseDevice.ButtonUp* events.*
- class [MouseWheelEventArgs](#)
*Defines the event data for *MouseDevice.WheelChanged* events.*
- struct [MouseState](#)
Encapsulates the state of a mouse device.

Enumerations

- enum [InputDeviceType](#) { [Keyboard](#), [Mouse](#), [Hid](#) }
The type of the input device.
- enum [JoystickButton](#) {
[Button0](#) = 0, [Button1](#), [Button2](#), [Button3](#),
[Button4](#), [Button5](#), [Button6](#), [Button7](#),
[Button8](#), [Button9](#), [Button10](#), [Button11](#),
[Button12](#), [Button13](#), [Button14](#), [Button15](#) }
Defines available JoystickDevice buttons.
- enum [JoystickAxis](#) {
[Axis0](#) = 0, [Axis1](#), [Axis2](#), [Axis3](#),
[Axis4](#), [Axis5](#), [Axis6](#), [Axis7](#),
[Axis8](#), [Axis9](#) }
Defines available JoystickDevice axes.
- enum [Key](#) {
[Unknown](#) = 0, [ShiftLeft](#), [LShift](#) = [ShiftLeft](#), [ShiftRight](#),
[RShift](#) = [ShiftRight](#), [ControlLeft](#), [LControl](#) = [ControlLeft](#), [ControlRight](#),
[RControl](#) = [ControlRight](#), [AltLeft](#), [LAlt](#) = [AltLeft](#), [AltRight](#),
[RAlt](#) = [AltRight](#), [WinLeft](#), [LWin](#) = [WinLeft](#), [WinRight](#),
[RWin](#) = [WinRight](#), [Menu](#), [F1](#), [F2](#),
[F3](#), [F4](#), [F5](#), [F6](#),
[F7](#), [F8](#), [F9](#), [F10](#),
[F11](#), [F12](#), [F13](#), [F14](#),
[F15](#), [F16](#), [F17](#), [F18](#),
[F19](#), [F20](#), [F21](#), [F22](#),
[F23](#), [F24](#), [F25](#), [F26](#),
[F27](#), [F28](#), [F29](#), [F30](#),
[F31](#), [F32](#), [F33](#), [F34](#),
[F35](#), [Up](#), [Down](#), [Left](#),
[Right](#), [Enter](#), [Escape](#), [Space](#),
[Tab](#), [BackSpace](#), [Back](#) = [BackSpace](#), [Insert](#),
[Delete](#), [PageUp](#), [PageDown](#), [Home](#),
[End](#), [CapsLock](#), [ScrollLock](#), [PrintScreen](#),


```

Pause, NumLock, Clear, Sleep,
Keypad0, Keypad1, Keypad2, Keypad3,
Keypad4, Keypad5, Keypad6, Keypad7,
Keypad8, Keypad9, KeypadDivide, KeypadMultiply,
KeypadSubtract, KeypadMinus = KeypadSubtract, KeypadAdd, KeypadPlus =
KeypadAdd,
KeypadDecimal, KeypadEnter, A, B,
C, D, E, F,
G, H, I, J,
K, L, M, N,
O, P, Q, R,
S, T, U, V,
W, X, Y, Z,
Number0, Number1, Number2, Number3,
Number4, Number5, Number6, Number7,
Number8, Number9, Tilde, Minus,
Plus, BracketLeft, LBracket = BracketLeft, BracketRight,
RBracket = BracketRight, Semicolon, Quote, Comma,
Period, Slash, BackSlash, LastKey }

```

The available keyboard keys.

- enum MouseButton {
Left = 0, Middle, Right, Button1,
Button2, Button3, Button4, Button5,
Button6, Button7, Button8, Button9,
LastButton }

Enumerates all possible mouse buttons.

4.9.1 Enumeration Type Documentation

4.9.1.1 enum OpenTK::Input::InputDeviceType

The type of the input device.

Enumerator:

Keyboard Device is a keyboard.

Mouse Device is a mouse.

Hid Device is a Human Interface Device. Joysticks, joypads, pens and some specific usb keyboards/mice fall into this category.

4.9.1.2 enum OpenTK::Input::JoystickAxis

Defines available [JoystickDevice](#) axes.

Enumerator:

- Axis0* The first axis of the [JoystickDevice](#).
- Axis1* The second axis of the [JoystickDevice](#).
- Axis2* The third axis of the [JoystickDevice](#).
- Axis3* The fourth axis of the [JoystickDevice](#).
- Axis4* The fifth axis of the [JoystickDevice](#).
- Axis5* The sixth axis of the [JoystickDevice](#).
- Axis6* The seventh axis of the [JoystickDevice](#).
- Axis7* The eighth axis of the [JoystickDevice](#).
- Axis8* The ninth axis of the [JoystickDevice](#).
- Axis9* The tenth axis of the [JoystickDevice](#).

4.9.1.3 enum OpenTK::Input::JoystickButton

Defines available [JoystickDevice](#) buttons.

Enumerator:

- Button0* The first button of the [JoystickDevice](#).
- Button1* The second button of the [JoystickDevice](#).
- Button2* The third button of the [JoystickDevice](#).
- Button3* The fourth button of the [JoystickDevice](#).
- Button4* The fifth button of the [JoystickDevice](#).
- Button5* The sixth button of the [JoystickDevice](#).
- Button6* The seventh button of the [JoystickDevice](#).
- Button7* The eighth button of the [JoystickDevice](#).
- Button8* The ninth button of the [JoystickDevice](#).
- Button9* The tenth button of the [JoystickDevice](#).

- Button10** The eleventh button of the [JoystickDevice](#).
- Button11** The twelfth button of the [JoystickDevice](#).
- Button12** The thirteenth button of the [JoystickDevice](#).
- Button13** The fourteenth button of the [JoystickDevice](#).
- Button14** The fifteenth button of the [JoystickDevice](#).
- Button15** The sixteenth button of the [JoystickDevice](#).

4.9.1.4 enum OpenTK::Input::Key

The available keyboard keys.

Enumerator:

- Unknown** A key outside the known keys.
- ShiftLeft** The left shift key.
- LShift** The left shift key (equivalent to ShiftLeft).
- ShiftRight** The right shift key.
- RShift** The right shift key (equivalent to ShiftRight).
- ControlLeft** The left control key.
- LControl** The left control key (equivalent to ControlLeft).
- ControlRight** The right control key.
- RControl** The right control key (equivalent to ControlRight).
- AltLeft** The left alt key.
- LAlt** The left alt key (equivalent to AltLeft).
- AltRight** The right alt key.
- RAlt** The right alt key (equivalent to AltRight).
- WinLeft** The left win key.
- LWin** The left win key (equivalent to WinLeft).
- WinRight** The right win key.
- RWin** The right win key (equivalent to WinRight).
- Menu** The menu key.
- F1** The F1 key.
- F2** The F2 key.
- F3** The F3 key.
- F4** The F4 key.
- F5** The F5 key.

F6 The F6 key.

F7 The F7 key.

F8 The F8 key.

F9 The F9 key.

F10 The F10 key.

F11 The F11 key.

F12 The F12 key.

F13 The F13 key.

F14 The F14 key.

F15 The F15 key.

F16 The F16 key.

F17 The F17 key.

F18 The F18 key.

F19 The F19 key.

F20 The F20 key.

F21 The F21 key.

F22 The F22 key.

F23 The F23 key.

F24 The F24 key.

F25 The F25 key.

F26 The F26 key.

F27 The F27 key.

F28 The F28 key.

F29 The F29 key.

F30 The F30 key.

F31 The F31 key.

F32 The F32 key.

F33 The F33 key.

F34 The F34 key.

F35 The F35 key.

Up The up arrow key.

Down The down arrow key.

Left The left arrow key.

Right The right arrow key.

Enter The enter key.

Escape The escape key.

Space The space key.

Tab The tab key.

BackSpace The backspace key.

Back The backspace key (equivalent to BackSpace).

Insert The insert key.

Delete The delete key.

PageUp The page up key.

PageDown The page down key.

Home The home key.

End The end key.

CapsLock The caps lock key.

ScrollLock The scroll lock key.

PrintScreen The print screen key.

Pause The pause key.

NumLock The num lock key.

Clear The clear key (Keypad5 with NumLock disabled, on typical keyboards).

Sleep The sleep key.

Keypad0 The keypad 0 key.

Keypad1 The keypad 1 key.

Keypad2 The keypad 2 key.

Keypad3 The keypad 3 key.

Keypad4 The keypad 4 key.

Keypad5 The keypad 5 key.

Keypad6 The keypad 6 key.

Keypad7 The keypad 7 key.

Keypad8 The keypad 8 key.

Keypad9 The keypad 9 key.

KeypadDivide The keypad divide key.

KeypadMultiply The keypad multiply key.

KeypadSubtract The keypad subtract key.

KeypadMinus The keypad minus key (equivalent to KeypadSubtract).

KeypadAdd The keypad add key.

KeypadPlus The keypad plus key (equivalent to KeypadAdd).

KeypadDecimal The keypad decimal key.

KeypadEnter The keypad enter key.

A The A key.

B The B key.

C The C key.

D The D key.

E The E key.

F The F key.

G The G key.

H The H key.

I The I key.

J The J key.

K The K key.

L The L key.

M The M key.

N The N key.

O The O key.

P The P key.

Q The Q key.

R The R key.

S The S key.

T The T key.

U The U key.

V The V key.

W The W key.

X The X key.

Y The Y key.

Z The Z key.

Number0 The number 0 key.

Number1 The number 1 key.

Number2 The number 2 key.

Number3 The number 3 key.

Number4 The number 4 key.

Number5 The number 5 key.

Number6 The number 6 key.

Number7 The number 7 key.

Number8 The number 8 key.
Number9 The number 9 key.
Tilde The tilde key.
Minus The minus key.
Plus The plus key.
BracketLeft The left bracket key.
LBracket The left bracket key (equivalent to BracketLeft).
BracketRight The right bracket key.
RBracket The right bracket key (equivalent to BracketRight).
Semicolon The semicolon key.
Quote The quote key.
Comma The comma key.
Period The period key.
Slash The slash key.
BackSlash The backslash key.
LastKey Indicates the last available keyboard key.

4.9.1.5 enum OpenTK::Input::MouseButton

Enumerates all possible mouse buttons.

Enumerator:

Left The left mouse button.
Middle The middle mouse button.
Right The right mouse button.
Button1 The first extra mouse button.
Button2 The second extra mouse button.
Button3 The third extra mouse button.
Button4 The fourth extra mouse button.
Button5 The fifth extra mouse button.
Button6 The sixth extra mouse button.
Button7 The seventh extra mouse button.
Button8 The eighth extra mouse button.
Button9 The ninth extra mouse button.
LastButton Indicates the last available mouse button.

4.10 Package OpenTK.Platform

Packages

- package [Dummy](#)
- package [Egl](#)
- package [MacOS](#)
- package [Windows](#)
- package [X11](#)

Classes

- interface [IGameWindow](#)
Defines the interface for a [GameWindow](#).
- interface [IWindowInfo](#)
Describes an OS window.

Functions

- internal delegate void **CreateEvent** (object sender, EventArgs e)
- internal delegate void **DestroyEvent** (object sender, EventArgs e)

4.11 Package OpenTK.Platform.Dummy

4.12 Package OpenTK.Platform.Egl

4.13 Package OpenTK.Platform.MacOS

Packages

- package [Carbon](#)

Enumerations

- enum **OSStatus** {
 NoError = 0, ParameterError = -50, NoHardwareError = -200, NotEnough-
 HardwareError = -201,

UserCanceledError = -128, **QueueError** = -1, **VTypeErr** = -2, **CorErr** = -3,
UnimpErr = -4, **SlpTypeErr** = -5, **SeNoDB** = -8, **ControlErr** = -17,
StatusErr = -18, **ReadErr** = -19, **WritErr** = -20, **BadUnitErr** = -21,
UnitEmptyErr = -22, **OpenErr** = -23, **ClosErr** = -24, **DRemovErr** = -25,
DInstErr = -26, **InvalidWindowPtr** = -5600, **UnsupportedWindowAt-**
tributesForClass = -5601, **WindowDoesNotHaveProxy** = -5602,
WindowPropertyNotFound = -5604, **UnrecognizedWindowClass** = -5605,
CorruptWindowDescription = -5606, **UserWantsToDragWindow** = -5607,
WindowsAlreadyInitialized = -5608, **FloatingWindowsNotInitialized** = -
5609, **WindowNotFound** = -5610, **WindowDoesNotFitOnscreen** = -5611,
WindowAttributeImmutable = -5612, **WindowAttributesConflict** = -5613,
WindowManagerInternalError = -5614, **WindowWrongState** = -5615,
WindowGroupInvalid = -5616, **WindowAppModalStateAlreadyExists** = -
5617, **WindowNoAppModalState** = -5618, **WindowDoesntSupportFocus** =
-30583,
WindowRegionCodeInvalid = -30593, **EventAlreadyPosted** = -9860, **Event-**
TargetBusy = -9861, **EventDeferAccessibilityEvent** = -9865,
EventInternalError = -9868, **EventParameterNotFound** = -9870, **Event-**
NotHandled = -9874, **EventLoopTimedOut** = -9875,
EventLoopQuit = -9876, **EventNotInQueue** = -9877, **HotKeyExists** = -9878,
EventPassToNextTarget = -9880 }

4.14 Package OpenTK.Platform.MacOS.Carbon

Enumerations

- enum **EventAttributes** { **kEventAttributeNone** = 0, **kEventAttributeUserEvent** = (1 << 0), **kEventAttributeMonitored** = 1 << 3 }
- enum **EventClass** {
Mouse = 0x6d6f7573, **Keyboard** = 0x6b657962, **Application** = 0x6170706c,
AppleEvent = 0x65707063,
Menu = 0x6d656e75, **Window** = 0x77696e64 }
- enum **WindowEventKind** {
WindowUpdate = 1, **WindowDrawContent** = 2, **WindowDrawStructure** = 3,
WindowEraseContent = 4,
WindowActivate = 5, **WindowDeactivate** = 6, **WindowSizeChanged** = 23,
WindowBoundsChanging = 26,
WindowBoundsChanged = 27, **WindowClickDragRgn** = 32, **WindowClick-**
ResizeRgn = 33, **WindowClickCollapseRgn** = 34,

WindowClickCloseRgn = 35, **WindowClickZoomRgn** = 36, **WindowClickContentRgn** = 37, **WindowClickProxyIconRgn** = 38,

WindowClose = 72, **WindowClosed** = 73 }

- enum **MouseEventKind** {
MouseDown = 1, **MouseUp** = 2, **MouseMoved** = 5, **MouseDragged** = 6,
MouseEntered = 8, **MouseExited** = 9, **WheelMoved** = 10 }
- enum **MouseButton** { **Primary** = 1, **Secondary** = 2, **Tertiary** = 3 }
- enum **KeyboardEventKind** { **RawKeyDown** = 1, **RawKeyRepeat** = 2, **RawKeyUp** = 3, **RawKeyModifiersChanged** = 4 }
- enum **AppEventKind** { **AppActivated** = 1, **AppDeactivated** = 2, **AppQuit** = 3, **AppLaunchNotification** = 4 }
- enum **AppleEventKind** { **AppleEvent** = 1 }
- enum **EventParamName** {
WindowRef = 0x77696e64, **MouseLocation** = 0x6d6c6f63, **WindowMouseLocation** = 0x776d6f75, **MouseButton** = 0x6d62746e,
ClickCount = 0x63636e74, **MouseWheelAxis** = 0x6d776178, **MouseWheelDelta** = 0x6d77646c, **MouseDelta** = 0x6d647461,
KeyCode = 0x6b636f64, **KeyMacCharCode** = 0x6b636872, **KeyModifiers** = 0x6b6d6f64 }
- enum **EventParamType** {
typeWindowRef = 0x77696e64, **typeMouseButton** = 0x6d62746e, **typeMouseWheelAxis** = 0x6d776178, **typeHIPoint** = 0x68697074,
typeHISize = 0x6869737a, **typeHIRect** = 0x68697263, **typeChar** = 0x54455854, **typeUInt32** = 0x6d61676e,
typeSInt32 = 0x6c6f6e67, **typeSInt16** = 0x73686f72, **typeSInt64** = 0x636f6d70, **typeIEEE32BitFloatingPoint** = 0x73696e67,
typeIEEE64BitFloatingPoint = 0x646f7562 }
- enum **EventMouseButton** { **Primary** = 0, **Secondary** = 1, **Tertiary** = 2 }
- enum **WindowRegionCode** {
TitleBarRegion = 0, **TitleTextRegion** = 1, **CloseBoxRegion** = 2, **ZoomBoxRegion** = 3,
DragRegion = 5, **GrowRegion** = 6, **CollapseBoxRegion** = 7, **TitleProxyIconRegion** = 8,
StructureRegion = 32, **ContentRegion** = 33, **UpdateRegion** = 34, **OpaqueRegion** = 35,
GlobalPortRegion = 40, **ToolbarButtonRegion** = 41 }
- enum **WindowClass** {
Alert = 1, **MovableAlert** = 2, **Modal** = 3, **MovableModal** = 4,
Floating = 5, **Document** = 6, **Desktop** = 7, **Utility** = 8,
Help = 10, **Sheet** = 11, **Toolbar** = 12, **Plain** = 13,

```

    Overlay = 14, SheetAlert = 15, AltPlain = 16, Drawer = 20,
    All = 0xFFFFFFFFu }
• enum WindowAttributes {
    NoAttributes = 0u, CloseBox = (1u << 0), HorizontalZoom = (1u << 1),
    VerticalZoom = (1u << 2),
    FullZoom = (VerticalZoom | HorizontalZoom), CollapseBox = (1u << 3), Re-
    sizable = (1u << 4), SideTitlebar = (1u << 5),
    NoUpdates = (1u << 16), NoActivates = (1u << 17), NoBuffering = (1u <<
    20), StandardHandler = (1u << 25),
    InWindowMenu = (1u << 27), LiveResize = (1u << 28), StandardDocu-
    ment = (CloseBox | FullZoom | CollapseBox | Resizable), StandardFloating =
    (CloseBox | CollapseBox) }
• enum WindowPositionMethod {
    CenterOnMainScreen = 1, CenterOnParentWindow = 2, CenterOnParen-
    tWindowScreen = 3, CascadeOnMainScreen = 4,
    CascadeOnParentWindow = 5, CascadeOnParentWindowScreen = 6, Cas-
    cadeStartAtParentWindowScreen = 10, AlertPositionOnMainScreen = 7,
    AlertPositionOnParentWindow = 8, AlertPositionOnParentWindowScreen
    = 9 }
• enum WindowPartCode {
    inDesk = 0, inNoWindow = 0, inMenuBar = 1, inSysWindow = 2,
    inContent = 3, inDrag = 4, inGrow = 5, inGoAway = 6,
    inZoomIn = 7, inZoomOut = 8, inCollapseBox = 11, inProxyIcon = 12,
    inToolbarButton = 13, inStructure = 15 }
• enum GestaltSelector { SystemVersion = 0x73797376, SystemVersionMajor
    = 0x73797331, SystemVersionMinor = 0x73797332, SystemVersionBugFix =
    0x73797333 }
• enum ProcessApplicationTransformState { kProcessTransformToFore-
    groundApplication = 1 }
• enum HICoordinateSpace { _72DPIGlobal = 1, ScreenPixel = 2, Window =
    3, View = 4 }
• enum MacOSKeyCode {
    A = 0, B = 11, C = 8, D = 2,
    E = 14, F = 3, G = 5, H = 4,
    I = 34, J = 38, K = 40, L = 37,
    M = 46, N = 45, O = 31, P = 35,
    Q = 12, R = 15, S = 1, T = 17,
    U = 32, V = 9, W = 13, X = 7,
    Y = 16, Z = 6, Key_1 = 18, Key_2 = 19,

```

Key_3 = 20, **Key_4** = 21, **Key_5** = 23, **Key_6** = 22,
Key_7 = 26, **Key_8** = 28, **Key_9** = 25, **Key_0** = 29,
Space = 49, **Tilde** = 50, **Minus** = 27, **Equals** = 24,
BracketLeft = 33, **BracketRight** = 30, **Backslash** = 42, **Semicolon** = 41,
Quote = 39, **Comma** = 43, **Period** = 47, **Slash** = 44,
Enter = 36, **Tab** = 48, **Backspace** = 51, **Return** = 52,
Esc = 53, **KeyPad_Decimal** = 65, **KeyPad_Multiply** = 67, **KeyPad_Add** = 69,
KeyPad_Divide = 75, **KeyPad_Enter** = 76, **KeyPad_Subtract** = 78, **KeyPad_**
Equal = 81,
KeyPad_0 = 82, **KeyPad_1** = 83, **KeyPad_2** = 84, **KeyPad_3** = 85,
KeyPad_4 = 86, **KeyPad_5** = 87, **KeyPad_6** = 88, **KeyPad_7** = 89,
KeyPad_8 = 91, **KeyPad_9** = 92, **F1** = 122, **F2** = 120,
F3 = 99, **F4** = 118, **F5** = 96, **F6** = 97,
F7 = 98, **F8** = 100, **F9** = 101, **F10** = 109,
F11 = 103, **F12** = 111, **F13** = 105, **F14** = 107,
F15 = 113, **Menu** = 110, **Insert** = 114, **Home** = 115,
Pageup = 116, **Del** = 117, **End** = 119, **Pagedown** = 121,
Up = 126, **Down** = 125, **Left** = 123, **Right** = 124 }

- enum **MacOSKeyModifiers** {
None = 0, **Shift** = 0x0200, **CapsLock** = 0x0400, **Control** = 0x1000,
Command = 0x0100, **Option** = 0x0800 }
- enum **CGDisplayErr**

Functions

- internal delegate OSStatus **MacOSEventHandler** (IntPtr inCaller, IntPtr in-Event, IntPtr userData)

4.15 Package OpenTK.Platform.Windows

Enumerations

- enum **SetWindowPosFlags** {
NOSIZE = 0x0001, **NOMOVE** = 0x0002, **NOZORDER** = 0x0004, **NORE-**
DRAW = 0x0008,
NOACTIVATE = 0x0010, **FRAMECHANGED** = 0x0020, **SHOWWINDOW** =
0x0040, **HIDEWINDOW** = 0x0080,

NOCOPYBITS = 0x0100, **NOOWNERZORDER** = 0x0200, **NOSEND-CHANGING** = 0x0400, **DRAWFRAME** = FRAMECHANGED,
NOREPOSITION = NOOWNERZORDER, **DEFERERASE** = 0x2000,
ASYNCHWINDOWPOS = 0x4000 }

- enum **GWL** {
WNDPROC = (-4), **HINSTANCE** = (-6), **HWNDPARENT** = (-8), **STYLE** = (-16),
EXSTYLE = (-20), **USERDATA** = (-21), **ID** = (-12) }

Window field offsets for `GetWindowLong()` and `GetWindowLongPtr()`.

- enum **SizeMessage** {
MAXHIDE = 4, **MAXIMIZED** = 2, **MAXSHOW** = 3, **MINIMIZED** = 1,
RESTORED = 0 }
- enum **NcCalcSizeOptions** {
ALIGNTOP = 0x10, **ALIGNRIGHT** = 0x80, **ALIGNLEFT** = 0x20, **ALIGN-BOTTOM** = 0x40,
HREDRAW = 0x100, **VREDRAW** = 0x200, **REDRAW** = (HREDRAW | VREDRAW), **VALIDRECTS** = 0x400 }
- enum **DisplayModeSettingsEnum** { **CurrentSettings** = -1, **RegistrySettings** = -2 }
- enum **DisplayDeviceStateFlags** {
None = 0x00000000, **AttachedToDesktop** = 0x00000001, **MultiDriver** = 0x00000002, **PrimaryDevice** = 0x00000004,
MirroringDriver = 0x00000008, **VgaCompatible** = 0x00000010, **Removable** = 0x00000020, **ModesPruned** = 0x08000000,
Remote = 0x04000000, **Disconnect** = 0x02000000, **Active** = 0x00000001, **Attached** = 0x00000002 }
- enum **ChangeDisplaySettingsEnum** { **UpdateRegistry** = 0x00000001, **Test** = 0x00000002, **Fullscreen** = 0x00000004 }
- enum **WindowStyle** {
Overlapped = 0x00000000, **Popup** = 0x80000000, **Child** = 0x40000000, **Minimize** = 0x20000000,
Visible = 0x10000000, **Disabled** = 0x08000000, **ClipSiblings** = 0x04000000, **ClipChildren** = 0x02000000,
Maximize = 0x01000000, **Caption** = 0x00C00000, **Border** = 0x00800000, **DialogFrame** = 0x00400000,
VScroll = 0x00200000, **HScreen** = 0x00100000, **SystemMenu** = 0x00080000, **ThickFrame** = 0x00040000,
Group = 0x00020000, **TabStop** = 0x00010000, **MinimizeBox** = 0x00002000, **MaximizeBox** = 0x00001000,

Tiled = Overlapped, **Iconic** = Minimize, **SizeBox** = ThickFrame, **TiledWindow** = OverlappedWindow,

OverlappedWindow = Overlapped | Caption | SystemMenu | ThickFrame | MinimizeBox | MaximizeBox, **PopupWindow** = Popup | Border | SystemMenu, **ChildWindow** = Child }

- enum **ExtendedWindowStyle** {

DialogModalFrame = 0x00000001, **NoParentNotify** = 0x00000004, **Topmost** = 0x00000008, **AcceptFiles** = 0x00000010,

Transparent = 0x00000020, **MdiChild** = 0x00000040, **ToolWindow** = 0x00000080, **WindowEdge** = 0x00000100,

ClientEdge = 0x00000200, **ContextHelp** = 0x00000400, **Right** = 0x00001000, **Left** = 0x00000000,

RightToLeftReading = 0x00002000, **LeftToRightReading** = 0x00000000, **LeftScrollbar** = 0x00004000, **RightScrollbar** = 0x00000000,

ControlParent = 0x00010000, **StaticEdge** = 0x00020000, **ApplicationWindow** = 0x00040000, **OverlappedWindow** = WindowEdge | ClientEdge,

PaletteWindow = WindowEdge | ToolWindow | Topmost, **Layered** = 0x00080000, **NoInheritLayout** = 0x00100000, **RightToLeftLayout** = 0x00400000,

Composited = 0x02000000, **NoActivate** = 0x08000000 }

- enum **GetWindowLongOffsets** {

WNDPROC = (-4), **HINSTANCE** = (-6), **HWNDPARENT** = (-8), **STYLE** = (-16),

EXSTYLE = (-20), **USERDATA** = (-21), **ID** = (-12) }

- enum **PixelFormatDescriptorFlags** {

DOUBLEBUFFER = 0x01, **STEREO** = 0x02, **DRAW_TO_WINDOW** = 0x04, **DRAW_TO_BITMAP** = 0x08,

SUPPORT_GDI = 0x10, **SUPPORT_OPENGL** = 0x20, **GENERIC_FORMAT** = 0x40, **NEED_PALETTE** = 0x80,

NEED_SYSTEM_PALETTE = 0x100, **SWAP_EXCHANGE** = 0x200, **SWAP_COPY** = 0x400, **SWAP_LAYER_BUFFERS** = 0x800,

GENERIC_ACCELERATED = 0x1000, **SUPPORT_DIRECTDRAW** = 0x2000, **DEPTH_DONTCARE** = unchecked((int)0x20000000), **DOUBLEBUFFER_DONTCARE** = unchecked((int)0x40000000),

STEREO_DONTCARE = unchecked((int)0x80000000) }

- enum **PixelFormat** { **RGBA** = 0, **INDEXED** = 1 }

- enum **WindowPlacementOptions** { **TOP** = 0, **BOTTOM** = 1, **TOPMOST** = -1, **NOTOPMOST** = -2 }

- enum **ClassStyle** {
 - VRedraw** = 0x0001, **HRedraw** = 0x0002, **DoubleClicks** = 0x0008, **OwnDC** = 0x0020,
 - ClassDC** = 0x0040, **ParentDC** = 0x0080, **NoClose** = 0x0200, **SaveBits** = 0x0800,
 - ByteAlignClient** = 0x1000, **ByteAlignWindow** = 0x2000, **GlobalClass** = 0x4000, **Ime** = 0x00010000,
 - DropShadow** = 0x00020000 }
- enum **RawInputDeviceFlags** {
 - REMOVE** = 0x00000001, **EXCLUDE** = 0x00000010, **PAGEONLY** = 0x00000020, **NOLEGACY** = 0x00000030,
 - INPUTSINK** = 0x00000100, **CAPTUREMOUSE** = 0x00000200, **NO-HOTKEYS** = 0x00000200, **APPKEYS** = 0x00000400,
 - EXINPUTSINK** = 0x00001000, **DEVNOTIFY** = 0x00002000 }
- enum **GetRawInputDataEnum** { **INPUT** = 0x10000003, **HEADER** = 0x10000005 }
- enum **RawInputDeviceInfoEnum** { **PREPARED** = 0x20000005, **DEVICENAME** = 0x20000007, **DEVICEINFO** = 0x2000000b }
- enum **RawInputMouseState** {
 - LEFT_BUTTON_DOWN** = 0x0001, **LEFT_BUTTON_UP** = 0x0002, **RIGHT_BUTTON_DOWN** = 0x0004, **RIGHT_BUTTON_UP** = 0x0008,
 - MIDDLE_BUTTON_DOWN** = 0x0010, **MIDDLE_BUTTON_UP** = 0x0020, **BUTTON_1_DOWN** = **LEFT_BUTTON_DOWN**, **BUTTON_1_UP** = **LEFT_BUTTON_UP**,
 - BUTTON_2_DOWN** = **RIGHT_BUTTON_DOWN**, **BUTTON_2_UP** = **RIGHT_BUTTON_UP**, **BUTTON_3_DOWN** = **MIDDLE_BUTTON_DOWN**, **BUTTON_3_UP** = **MIDDLE_BUTTON_UP**,
 - BUTTON_4_DOWN** = 0x0040, **BUTTON_4_UP** = 0x0080, **BUTTON_5_DOWN** = 0x0100, **BUTTON_5_UP** = 0x0200,
 - WHEEL** = 0x0400 }
- enum **RawInputKeyboardDataFlags** {
 - MAKE** = 0, **BREAK** = 1, **E0** = 2, **E1** = 4,
 - TERMSRV_SET_LED** = 8, **TERMSRV_SHADOW** = 0x10 }
- enum **RawInputDeviceType** { **MOUSE** = 0, **KEYBOARD** = 1, **HID** = 2 }
- enum **RawMouseFlags** { **MOUSE_MOVE_RELATIVE** = 0x00, **MOUSE_MOVE_ABSOLUTE** = 0x01, **MOUSE_VIRTUAL_DESKTOP** = 0x02, **MOUSE_ATTRIBUTES_CHANGED** = 0x04 }

Mouse indicator flags (found in winuser.h).

- enum **VirtualKeys** {
 - LBUTTON** = 0x01, **RBUTTON** = 0x02, **CANCEL** = 0x03, **MBUTTON** = 0x04,
 - XBUTTON1** = 0x05, **XBUTTON2** = 0x06, **BACK** = 0x08, **TAB** = 0x09,
 - CLEAR** = 0x0C, **RETURN** = 0x0D, **SHIFT** = 0x10, **CONTROL** = 0x11,
 - MENU** = 0x12, **PAUSE** = 0x13, **CAPITAL** = 0x14, **KANA** = 0x15,
 - HANGEUL** = 0x15, **HANGUL** = 0x15, **JUNJA** = 0x17, **FINAL** = 0x18,
 - HANJA** = 0x19, **KANJI** = 0x19, **ESCAPE** = 0x1B, **CONVERT** = 0x1C,
 - NONCONVERT** = 0x1D, **ACCEPT** = 0x1E, **MODECHANGE** = 0x1F,
 - SPACE** = 0x20,
 - PRIOR** = 0x21, **NEXT** = 0x22, **END** = 0x23, **HOME** = 0x24,
 - LEFT** = 0x25, **UP** = 0x26, **RIGHT** = 0x27, **DOWN** = 0x28,
 - SELECT** = 0x29, **PRINT** = 0x2A, **EXECUTE** = 0x2B, **SNAPSHOT** = 0x2C,
 - INSERT** = 0x2D, **DELETE** = 0x2E, **HELP** = 0x2F, **LWIN** = 0x5B,
 - RWIN** = 0x5C, **APPS** = 0x5D, **SLEEP** = 0x5F, **NUMPAD0** = 0x60,
 - NUMPAD1** = 0x61, **NUMPAD2** = 0x62, **NUMPAD3** = 0x63, **NUMPAD4** = 0x64,
 - NUMPAD5** = 0x65, **NUMPAD6** = 0x66, **NUMPAD7** = 0x67, **NUMPAD8** = 0x68,
 - NUMPAD9** = 0x69, **MULTIPLY** = 0x6A, **ADD** = 0x6B, **SEPARATOR** = 0x6C,
 - SUBTRACT** = 0x6D, **DECIMAL** = 0x6E, **DIVIDE** = 0x6F, **F1** = 0x70,
 - F2** = 0x71, **F3** = 0x72, **F4** = 0x73, **F5** = 0x74,
 - F6** = 0x75, **F7** = 0x76, **F8** = 0x77, **F9** = 0x78,
 - F10** = 0x79, **F11** = 0x7A, **F12** = 0x7B, **F13** = 0x7C,
 - F14** = 0x7D, **F15** = 0x7E, **F16** = 0x7F, **F17** = 0x80,
 - F18** = 0x81, **F19** = 0x82, **F20** = 0x83, **F21** = 0x84,
 - F22** = 0x85, **F23** = 0x86, **F24** = 0x87, **NUMLOCK** = 0x90,
 - SCROLL** = 0x91, **OEM_NEC_EQUAL** = 0x92, **OEM_FJ_JISHO** = 0x92,
 - OEM_FJ_MASSHOU** = 0x93,
 - OEM_FJ_TOUROKU** = 0x94, **OEM_FJ_LOYA** = 0x95, **OEM_FJ_ROYA** = 0x96, **LSHIFT** = 0xA0,
 - RSHIFT** = 0xA1, **LCONTROL** = 0xA2, **RCONTROL** = 0xA3, **LMENU** = 0xA4,
 - RMENU** = 0xA5, **BROWSER_BACK** = 0xA6, **BROWSER_FORWARD** = 0xA7,
 - BROWSER_REFRESH** = 0xA8,

BROWSER_STOP = 0xA9, **BROWSER_SEARCH** = 0xAA, **BROWSER_FAVORITES** = 0xAB, **BROWSER_HOME** = 0xAC,
VOLUME_MUTE = 0xAD, **VOLUME_DOWN** = 0xAE, **VOLUME_UP** = 0xAF, **MEDIA_NEXT_TRACK** = 0xB0,
MEDIA_PREV_TRACK = 0xB1, **MEDIA_STOP** = 0xB2, **MEDIA_PLAY_PAUSE** = 0xB3, **LAUNCH_MAIL** = 0xB4,
LAUNCH_MEDIA_SELECT = 0xB5, **LAUNCH_APP1** = 0xB6, **LAUNCH_APP2** = 0xB7, **OEM_1** = 0xBA,
OEM_PLUS = 0xBB, **OEM_COMMA** = 0xBC, **OEM_MINUS** = 0xBD, **OEM_PERIOD** = 0xBE,
OEM_2 = 0xBF, **OEM_3** = 0xC0, **OEM_4** = 0xDB, **OEM_5** = 0xDC,
OEM_6 = 0xDD, **OEM_7** = 0xDE, **OEM_8** = 0xDF, **OEM_AX** = 0xE1,
OEM_102 = 0xE2, **ICO_HELP** = 0xE3, **ICO_00** = 0xE4, **PROCESSKEY** = 0xE5,
ICO_CLEAR = 0xE6, **PACKET** = 0xE7, **OEM_RESET** = 0xE9, **OEM_JUMP** = 0xEA,
OEM_PA1 = 0xEB, **OEM_PA2** = 0xEC, **OEM_PA3** = 0xED, **OEM_WSCtrl** = 0xEE,
OEM_CUSEL = 0xEF, **OEM_ATTN** = 0xF0, **OEM_FINISH** = 0xF1, **OEM_COPY** = 0xF2,
OEM_AUTO = 0xF3, **OEM_ENLW** = 0xF4, **OEM_BACKTAB** = 0xF5, **ATTN** = 0xF6,
CRSEL = 0xF7, **EXSEL** = 0xF8, **EREOF** = 0xF9, **PLAY** = 0xFA,
ZOOM = 0xFB, **NONAME** = 0xFC, **PA1** = 0xFD, **OEM_CLEAR** = 0xFE,
Last }

- enum [MouseKeys](#) {

None = 0, **Left** = 0x0001, **Right** = 0x0002, **Middle** = 0x0010,
XButton1 = 0x0020, **XButton2** = 0x0040 }

Enumerates available mouse keys (suitable for use in WM_MOUSEMOVE messages).
- enum [QueueStatusFlags](#) {

KEY = 0x0001, **MOUSEMOVE** = 0x0002, **MOUSEBUTTON** = 0x0004,
POSTMESSAGE = 0x0008,
TIMER = 0x0010, **PAINT** = 0x0020, **SENDMESSAGE** = 0x0040, **HOTKEY** = 0x0080,
ALLPOSTMESSAGE = 0x0100, **RAWINPUT** = 0x0400, **MOUSE** = MOUSEMOVE | MOUSEBUTTON, **INPUT** = MOUSE | KEY | RAWINPUT,
INPUT_LEGACY = MOUSE | KEY, **ALLEVENTS** = INPUT | POSTMESSAGE | TIMER | PAINT | HOTKEY, **ALLINPUT** = INPUT | POSTMESSAGE | TIMER | PAINT | HOTKEY | SENDMESSAGE }

- enum [WindowMessage](#) {
 - NULL** = 0x0000, **CREATE** = 0x0001, **DESTROY** = 0x0002, **MOVE** = 0x0003,
 - SIZE** = 0x0005, **ACTIVATE** = 0x0006, **SETFOCUS** = 0x0007, **KILLFOCUS** = 0x0008,
 - ENABLE** = 0x000A, **SETREDRAW** = 0x000B, **SETTEXT** = 0x000C, **GETTEXT** = 0x000D,
 - GETTEXTLENGTH** = 0x000E, **PAINT** = 0x000F, **CLOSE** = 0x0010, **QUERYENDSESSION** = 0x0011,
 - QUIT** = 0x0012, **QUERYOPEN** = 0x0013, **ERASEBKGD** = 0x0014, **SYS-COLORCHANGE** = 0x0015,
 - ENDSESSION** = 0x0016, **SHOWWINDOW** = 0x0018, **CTLCOLOR** = 0x0019, **WININICHANGE** = 0x001A,
 - SETTINGCHANGE** = 0x001A, **DEVMODECHANGE** = 0x001B, **ACTIVATEAPP** = 0x001C, **FONTCHANGE** = 0x001D,
 - TIMECHANGE** = 0x001E, **CANCELMODE** = 0x001F, **SETCURSOR** = 0x0020, **MOUSEACTIVATE** = 0x0021,
 - CHILDACTIVATE** = 0x0022, **QUEUESYNC** = 0x0023, **GETMINMAX-INFO** = 0x0024, **PAINTICON** = 0x0026,
 - ICONERASEBKGD** = 0x0027, **NEXTDLGCTL** = 0x0028, **SPOOLER-STATUS** = 0x002A, **DRAWITEM** = 0x002B,
 - MEASUREITEM** = 0x002C, **DELETEITEM** = 0x002D, **VKEYTOITEM** = 0x002E, **CHARTOITEM** = 0x002F,
 - SETFONT** = 0x0030, **GETFONT** = 0x0031, **SETHOTKEY** = 0x0032, **GETHOTKEY** = 0x0033,
 - QUERYDRAGICON** = 0x0037, **COMPAREITEM** = 0x0039, **GETOBJECT** = 0x003D, **COMPACTING** = 0x0041,
 - COMMNOTIFY** = 0x0044, **WINDOWPOSCHANGING** = 0x0046, **WINDOWPOSCHANGED** = 0x0047, **POWER** = 0x0048,
 - COPYDATA** = 0x004A, **CANCELJOURNAL** = 0x004B, **NOTIFY** = 0x004E, **INPUTLANGCHANGEREQUEST** = 0x0050,
 - INPUTLANGCHANGE** = 0x0051, **TCARD** = 0x0052, **HELP** = 0x0053, **USERCHANGED** = 0x0054,
 - NOTIFYFORMAT** = 0x0055, **CONTEXTMENU** = 0x007B,
 - STYLECHANGING** = 0x007C, **STYLECHANGED** = 0x007D,
 - DISPLAYCHANGE** = 0x007E, **GETICON** = 0x007F, **SETICON** = 0x0080, **NCCREATE** = 0x0081,
 - NCDESTROY** = 0x0082, **NCCALCSIZE** = 0x0083, **NCHITTEST** = 0x0084, **NCPAINT** = 0x0085,

NCACTIVATE = 0x0086, **GETDLGCODE** = 0x0087, **SYNCPAINT** = 0x0088, **NCMOUSEMOVE** = 0x00A0,
NCLBUTTONDOWN = 0x00A1, **NCLBUTTONUP** = 0x00A2, **NCLBUTTONDBLCLK** = 0x00A3, **NCRBUTTONDOWN** = 0x00A4,
NCRBUTTONUP = 0x00A5, **NCRBUTTONDBLCLK** = 0x00A6, **NCMBUTTONDOWN** = 0x00A7, **NCMBUTTONUP** = 0x00A8,
NCMBUTTONDBLCLK = 0x00A9, **NCXBUTTONDOWN** = 0x00ab, **NCXBUTTONUP** = 0x00ac, **NCXBUTTONDBLCLK** = 0x00ad,
INPUT = 0x00FF, **KEYDOWN** = 0x0100, **KEYFIRST** = 0x0100, **KEYUP** = 0x0101,
CHAR = 0x0102, **DEADCHAR** = 0x0103, **SYSKEYDOWN** = 0x0104, **SYSKEYUP** = 0x0105,
SYSCHAR = 0x0106, **SYSDEADCHAR** = 0x0107, **KEYLAST** = 0x0108, **IME_STARTCOMPOSITION** = 0x010D,
IME_ENDCOMPOSITION = 0x010E, **IME_COMPOSITION** = 0x010F, **IME_KEYLAST** = 0x010F, **INITDIALOG** = 0x0110,
COMMAND = 0x0111, **SYSCOMMAND** = 0x0112, **TIMER** = 0x0113, **HSCROLL** = 0x0114,
VSCROLL = 0x0115, **INITMENU** = 0x0116, **INITMENUPOPUP** = 0x0117, **MENUSELECT** = 0x011F,
MENUCHAR = 0x0120, **ENTERIDLE** = 0x0121, **MENURBUTTONUP** = 0x0122, **MENUDRAG** = 0x0123,
MENUGETOBJECT = 0x0124, **UNINITMENUPOPUP** = 0x0125, **MENUCOMMAND** = 0x0126, **CHANGEUISTATE** = 0x0127,
UPDATEUISTATE = 0x0128, **QUERYUISTATE** = 0x0129, **CTLCOLORMSGBOX** = 0x0132, **CTLCOLOREDIT** = 0x0133,
CTLCOLORLISTBOX = 0x0134, **CTLCOLORBTN** = 0x0135, **CTLCOLORDLG** = 0x0136, **CTLCOLORSCROLLBAR** = 0x0137,
CTLCOLORSTATIC = 0x0138, **MOUSEMOVE** = 0x0200, **MOUSEFIRST** = 0x0200, **LBUTTONDOWN** = 0x0201,
LBUTTONUP = 0x0202, **LBUTTONDBLCLK** = 0x0203, **RBUTTONDOWN** = 0x0204, **RBUTTONUP** = 0x0205,
RBUTTONDBLCLK = 0x0206, **MBUTTONDOWN** = 0x0207, **MBUTTONUP** = 0x0208, **MBUTTONDBLCLK** = 0x0209,
MOUSEWHEEL = 0x020A, **MOUSELAST** = 0x020D, **XBUTTONDOWN** = 0x020B, **XBUTTONUP** = 0x020C,
XBUTTONDBLCLK = 0x020D, **PARENTNOTIFY** = 0x0210, **ENTERMENULOOP** = 0x0211, **EXITMENULOOP** = 0x0212,
NEXTMENU = 0x0213, **SIZING** = 0x0214, **CAPTURECHANGED** = 0x0215, **MOVING** = 0x0216,

DEVICECHANGE = 0x0219, **MDICREATE** = 0x0220, **MDIDESTROY** = 0x0221, **MDIACTIVATE** = 0x0222,

MDIRESTORE = 0x0223, **MDINEXT** = 0x0224, **MDIMAXIMIZE** = 0x0225, **MDITILE** = 0x0226,

MDICASCADE = 0x0227, **MDIICONARRANGE** = 0x0228, **MDIGETACTIVE** = 0x0229, **MDISETMENU** = 0x0230,

ENTERSIZEMOVE = 0x0231, **EXITSIZEMOVE** = 0x0232, **DROPPFILES** = 0x0233, **MDIREFRESHMENU** = 0x0234,

IME_SETCONTEXT = 0x0281, **IME_NOTIFY** = 0x0282, **IME_CONTROL** = 0x0283, **IME_COMPOSITIONFULL** = 0x0284,

IME_SELECT = 0x0285, **IME_CHAR** = 0x0286, **IME_REQUEST** = 0x0288, **IME_KEYDOWN** = 0x0290,

IME_KEYUP = 0x0291, **NCMOUSEHOVER** = 0x02A0, **MOUSEHOVER** = 0x02A1, **NCMOUSELEAVE** = 0x02A2,

MOUSELEAVE = 0x02A3, **CUT** = 0x0300, **COPY** = 0x0301, **PASTE** = 0x0302,

CLEAR = 0x0303, **UNDO** = 0x0304, **RENDERFORMAT** = 0x0305, **RENDERALLFORMATS** = 0x0306,

DESTROYCLIPBOARD = 0x0307, **DRAWCLIPBOARD** = 0x0308, **PAINTCLIPBOARD** = 0x0309, **VSCROLLCLIPBOARD** = 0x030A,

SIZECLIPBOARD = 0x030B, **ASKCBFORMATNAME** = 0x030C, **CHANGECBCHAIN** = 0x030D, **HSCROLLCLIPBOARD** = 0x030E,

QUERYNEWPALETTE = 0x030F, **PALETTEISCHANGING** = 0x0310, **PALETTECHANGED** = 0x0311, **HOTKEY** = 0x0312,

PRINT = 0x0317, **PRINTCLIENT** = 0x0318, **HANDHELDFIRST** = 0x0358, **HANDHELDLAST** = 0x035F,

AFXFIRST = 0x0360, **AFXLAST** = 0x037F, **PENWINFIRST** = 0x0380, **PENWINLAST** = 0x038F,

APP = 0x8000, **USER** = 0x0400, **MOUSE_ENTER** = 0x0401, **ASYN_-MESSAGE** = 0x0403,

REFLECT = **USER** + 0x1c00, **CLOSE_INTERNAL** = **USER** + 0x1c01, **BALLOONSHOW** = **USER** + 0x0002, **BALLOONHIDE** = **USER** + 0x0003,

BALLOONTIMEOUT = **USER** + 0x0004, **BALLOONUSERCLICK** = **USER** + 0x0005 }

- enum [ShowWindowCommand](#) {

[HIDE](#) = 0, [SHOWNORMAL](#) = 1, **NORMAL** = 1, [SHOWMINIMIZED](#) = 2, [SHOWMAXIMIZED](#) = 3, **MAXIMIZE** = 3, [SHOWNOACTIVATE](#) = 4, [SHOW](#) = 5,

[MINIMIZE](#) = 6, [SHOWMINNOACTIVE](#) = 7, [SHOWNA](#) = 8, [RESTORE](#) = 9, [SHOWDEFAULT](#) = 10, [FORCEMINIMIZE](#) = 11 }

ShowWindow() Commands.

- enum [ShowWindowMessageIdentifiers](#) { **PARENTCLOSING** = 1, **OTHERZOOM** = 2, **PARENTOPENING** = 3, **OTHERUNZOOM** = 4 }

Identifiers for the WM_SHOWWINDOW message.

- enum [GdiCharset](#) {
Ansi = 0, **Default** = 1, **Symbol** = 2, **ShiftJIS** = 128,
Hangeul = 129, **Hangul** = 129, **GB2312** = 134, **ChineseBig5** = 136,
OEM = 255, **Johab** = 130, **Hebrew** = 177, **Arabic** = 178,
Greek = 161, **Turkish** = 162, **Vietnamese** = 163, **Thai** = 222,
EastEurope = 238, **Russian** = 204, **Mac** = 77, **Baltic** = 186 }

Enumerates the available character sets.

- enum [MapVirtualKeyType](#) {
VirtualKeyToScanCode = 0, **ScanCodeToVirtualKey** = 1, **VirtualKeyToCharacter** = 2, **ScanCodeToVirtualKeyExtended** = 3,
VirtualKeyToScanCodeExtended = 4 }
- enum **DwmWindowAttribute** {
NCRENDERING_ENABLED = 1, **NCRENDERING_POLICY**,
TRANSITIONS_FORCEDISABLED, **ALLOW_NCPAINT**,
CAPTION_BUTTON_BOUNDS, **NONCLIENT_RTL_LAYOUT**,
FORCE_ICONIC_REPRESENTATION, **FLIP3D_POLICY**,
EXTENDED_FRAME_BOUNDS, **HAS_ICONIC_BITMAP**,
DISALLOW_PEEK, **EXCLUDED_FROM_PEEK**,
LAST }
- enum [ShGetFileIconFlags](#) {
Icon = 0x000000100, **DisplayName** = 0x000000200, **TypeName** =
0x000000400, **Attributes** = 0x000000800,
IconLocation = 0x000001000, **ExeType** = 0x000002000, **SysIconIndex** =
0x000004000, **LinkOverlay** = 0x000008000,
Selected = 0x000010000, **Attr_Specified** = 0x000020000, **LargeIcon** =
0x000000000, **SmallIcon** = 0x000000001,
OpenIcon = 0x000000002, **ShellIconSize** = 0x000000004, **PIDL** =
0x000000008, **UseFileAttributes** = 0x000000010,
AddOverlays = 0x000000020, **OverlayIndex** = 0x000000040 }
- enum **MonitorFrom** { **Null** = 0, **Primary** = 1, **Nearest** = 2 }
- enum **CursorName** { **Arrow** = 32512 }

- enum **TrackMouseEventFlags** {
HOVER = 0x00000001, **LEAVE** = 0x00000002, **NONCLIENT** = 0x00000010,
QUERY = 0x40000000,
CANCEL = 0x80000000 }
- enum **MouseActivate** { **ACTIVATE** = 1, **ACTIVATEANDEAT** = 2, **NOACTIVATE** = 3, **NOACTIVATEANDEAT** = 4 }
- enum **ArbCreateContext** {
DebugBit = 0x0001, **ForwardCompatibleBit** = 0x0002, **MajorVersion** = 0x2091, **MinorVersion** = 0x2092,
LayerPlane = 0x2093, **Flags** = 0x2094, **ErrorInvalidVersion** = 0x2095 }
- enum **WGL_ARB_buffer_region** { **BackColorBufferBitArb** = ((int)0x00000002), **StencilBufferBitArb** = ((int)0x00000008), **FrontColorBufferBitArb** = ((int)0x00000001), **DepthBufferBitArb** = ((int)0x00000004) }
- enum **WGL_EXT_pixel_format** {
SupportGdiExt = ((int)0x200f), **TypeColorindexExt** = ((int)0x202c), **AccelerationExt** = ((int)0x2003), **GreenBitsExt** = ((int)0x2017),
DrawToWindowExt = ((int)0x2001), **SwapCopyExt** = ((int)0x2029), **DrawToBitmapExt** = ((int)0x2002), **TransparentExt** = ((int)0x200a),
SwapMethodExt = ((int)0x2007), **SwapLayerBuffersExt** = ((int)0x2006), **PixelFormatExt** = ((int)0x2013), **AlphaShiftExt** = ((int)0x201c),
AccumRedBitsExt = ((int)0x201e), **FullAccelerationExt** = ((int)0x2027), **SupportOpenGLExt** = ((int)0x2010), **BlueShiftExt** = ((int)0x201a),
RedBitsExt = ((int)0x2015), **NoAccelerationExt** = ((int)0x2025), **StereoExt** = ((int)0x2012), **GreenShiftExt** = ((int)0x2018),
BlueBitsExt = ((int)0x2019), **AlphaBitsExt** = ((int)0x201b), **RedShiftExt** = ((int)0x2016), **DepthBitsExt** = ((int)0x2022),
TypeRgbaExt = ((int)0x202b), **GenericAccelerationExt** = ((int)0x2026), **AccumAlphaBitsExt** = ((int)0x2021), **AccumGreenBitsExt** = ((int)0x201f),
TransparentValueExt = ((int)0x200b), **AccumBlueBitsExt** = ((int)0x2020), **ShareDepthExt** = ((int)0x200c), **ShareAccumExt** = ((int)0x200e),
SwapExchangeExt = ((int)0x2028), **AccumBitsExt** = ((int)0x201d), **NumberUnderlaysExt** = ((int)0x2009), **StencilBitsExt** = ((int)0x2023),
DoubleBufferExt = ((int)0x2011), **NeedPaletteExt** = ((int)0x2004), **ColorBitsExt** = ((int)0x2014), **SwapUndefinedExt** = ((int)0x202a),
NeedSystemPaletteExt = ((int)0x2005), **NumberOverlaysExt** = ((int)0x2008), **AuxBuffersExt** = ((int)0x2024), **NumberPixelFormatsExt** = ((int)0x2000),
ShareStencilExt = ((int)0x200d) }

- enum **WGL_ARB_pixel_format** {
 - ShareStencilArb** = ((int)0x200d), **AccumBitsArb** = ((int)0x201d), **NumberUnderlaysArb** = ((int)0x2009), **StereoArb** = ((int)0x2012),
 - MaxPbufferHeightArb** = ((int)0x2030), **TypeRgbaArb** = ((int)0x202b), **SupportGdiArb** = ((int)0x200f), **NeedSystemPaletteArb** = ((int)0x2005),
 - AlphaBitsArb** = ((int)0x201b), **ShareDepthArb** = ((int)0x200c), **SupportOpenglArb** = ((int)0x2010), **ColorBitsArb** = ((int)0x2014),
 - AccumRedBitsArb** = ((int)0x201e), **MaxPbufferWidthArb** = ((int)0x202f), **NumberOverlaysArb** = ((int)0x2008), **MaxPbufferPixelsArb** = ((int)0x202e),
 - NeedPaletteArb** = ((int)0x2004), **RedShiftArb** = ((int)0x2016), **AccelerationArb** = ((int)0x2003), **GreenBitsArb** = ((int)0x2017),
 - TransparentGreenValueArb** = ((int)0x2038), **PixelTypeArb** = ((int)0x2013), **AuxBuffersArb** = ((int)0x2024), **DrawToWindowArb** = ((int)0x2001),
 - RedBitsArb** = ((int)0x2015), **NumberPixelFormatsArb** = ((int)0x2000), **GenericAccelerationArb** = ((int)0x2026), **BlueBitsArb** = ((int)0x2019),
 - PbufferLargestArb** = ((int)0x2033), **AccumAlphaBitsArb** = ((int)0x2021), **TransparentArb** = ((int)0x200a), **FullAccelerationArb** = ((int)0x2027),
 - ShareAccumArb** = ((int)0x200e), **SwapExchangeArb** = ((int)0x2028), **SwapUndefinedArb** = ((int)0x202a), **TransparentAlphaValueArb** = ((int)0x203a),
 - PbufferHeightArb** = ((int)0x2035), **TransparentBlueValueArb** = ((int)0x2039), **SwapMethodArb** = ((int)0x2007), **StencilBitsArb** = ((int)0x2023),
 - DepthBitsArb** = ((int)0x2022), **GreenShiftArb** = ((int)0x2018), **TransparentRedValueArb** = ((int)0x2037), **DoubleBufferArb** = ((int)0x2011),
 - NoAccelerationArb** = ((int)0x2025), **TypeColorindexArb** = ((int)0x202c), **SwapLayerBuffersArb** = ((int)0x2006), **AccumBlueBitsArb** = ((int)0x2020),
 - DrawToPbufferArb** = ((int)0x202d), **AccumGreenBitsArb** = ((int)0x201f), **PbufferWidthArb** = ((int)0x2034), **TransparentIndexValueArb** = ((int)0x203b),
 - AlphaShiftArb** = ((int)0x201c), **DrawToBitmapArb** = ((int)0x2002), **BlueShiftArb** = ((int)0x201a), **SwapCopyArb** = ((int)0x2029) }
- enum **WGL_EXT_pbuffer** {
 - DrawToPbufferExt** = ((int)0x202d), **PbufferLargestExt** = ((int)0x2033), **OptimalPbufferWidthExt** = ((int)0x2031), **MaxPbufferPixelsExt** = ((int)0x202e),
 - MaxPbufferHeightExt** = ((int)0x2030), **PbufferWidthExt** = ((int)0x2034), **MaxPbufferWidthExt** = ((int)0x202f), **OptimalPbufferHeightExt** = ((int)0x2032),
 - PbufferHeightExt** = ((int)0x2035) }

- enum **WGL_ARB_pbuffer** {
 - PbufferWidthArb** = ((int)0x2034), **TransparentGreenValueArb** = ((int)0x2038), **PbufferHeightArb** = ((int)0x2035), **PbufferLostArb** = ((int)0x2036),
 - DrawToPbufferArb** = ((int)0x202d), **TransparentIndexValueArb** = ((int)0x203b), **TransparentRedValueArb** = ((int)0x2037), **MaxPbufferPixelsArb** = ((int)0x202e),
 - TransparentAlphaValueArb** = ((int)0x203a), **MaxPbufferWidthArb** = ((int)0x202f), **MaxPbufferHeightArb** = ((int)0x2030), **TransparentBlueValueArb** = ((int)0x2039),
 - PbufferLargestArb** = ((int)0x2033) }
- enum **WGL_EXT_depth_float** { **DepthFloatExt** = ((int)0x2040) }
- enum **WGL_EXT_multisample** { **SampleBuffersExt** = ((int)0x2041), **SamplesExt** = ((int)0x2042) }
- enum **WGL_ARB_multisample** { **SampleBuffersArb** = ((int)0x2041), **SamplesArb** = ((int)0x2042) }
- enum **WGL_EXT_make_current_read** { **ErrorInvalidPixelFormatExt** = ((int)0x2043) }
- enum **WGL_ARB_make_current_read** { **ErrorInvalidPixelFormatArb** = ((int)0x2043), **ErrorIncompatibleDeviceContextsArb** = ((int)0x2054) }
- enum **WGL_I3D_genlock** {
 - GenlockSourceMultiviewI3d** = ((int)0x2044), **GenlockSourceEdgeBothI3d** = ((int)0x204c), **GenlockSourceEdgeRisingI3d** = ((int)0x204b), **GenlockSourceDigitalSyncI3d** = ((int)0x2048),
 - GenlockSourceExternalFieldI3d** = ((int)0x2046), **GenlockSourceDigitalFieldI3d** = ((int)0x2049), **GenlockSourceExternalSyncI3d** = ((int)0x2045), **GenlockSourceEdgeFallingI3d** = ((int)0x204a),
 - GenlockSourceExternalTtlI3d** = ((int)0x2047) }
- enum **WGL_I3D_gamma** { **GammaExcludeDesktopI3d** = ((int)0x204f), **GammaTableSizeI3d** = ((int)0x204e) }
- enum **WGL_I3D_digital_video_control** { **DigitalVideoCursorAlphaFramebufferI3d** = ((int)0x2050), **DigitalVideoGammaCorrectedI3d** = ((int)0x2053), **DigitalVideoCursorAlphaValueI3d** = ((int)0x2051), **DigitalVideoCursorIncludedI3d** = ((int)0x2052) }
- enum **WGL_3DFX_multisample** { **SampleBuffers3dfx** = ((int)0x2060), **Samples3dfx** = ((int)0x2061) }
- enum **WGL_ARB_render_texture** {
 - TextureCubeMapPositiveXArb** = ((int)0x207d), **TextureCubeMapPositiveYArb** = ((int)0x207f), **Aux0Arb** = ((int)0x2087), **TextureIdArb** = ((int)0x2079),
 - Aux6Arb** = ((int)0x208d), **TextureCubeMapArb** = ((int)0x2078), **TextureFormatArb** = ((int)0x2072), **BackRightArb** = ((int)0x2086),


```

BindToTextureRgbArb = ((int)0x2070), MipmapLevelArb = ((int)0x207b),
CubeMapFaceArb = ((int)0x207c), TextureCubeMapNegativeXArb =
((int)0x207e),

Aux7Arb = ((int)0x208e), Aux8Arb = ((int)0x208f), MipmapTextureArb =
((int)0x2074), NoTextureArb = ((int)0x2077),

Aux3Arb = ((int)0x208a), Texture2DArb = ((int)0x207a), Aux1Arb =
((int)0x2088), TextureCubeMapPositiveZArb = ((int)0x2081),

BindToTextureRgbaArb = ((int)0x2071), TextureCubeMapNegativeYArb
= ((int)0x2080), TextureRgbaArb = ((int)0x2076), FrontRightArb =
((int)0x2084),

Aux5Arb = ((int)0x208c), Aux4Arb = ((int)0x208b), TextureTargetArb =
((int)0x2073), FrontLeftArb = ((int)0x2083),

Aux9Arb = ((int)0x2090), TextureRgbArb = ((int)0x2075), BackLeftArb =
((int)0x2085), TextureCubeMapNegativeZArb = ((int)0x2082),

Aux2Arb = ((int)0x2089) }
• enum WGL_NV_render_texture_rectangle { BindToTextureRectang-
leRgbNv = ((int)0x20a0), BindToTextureRectangleRgbaNv = ((int)0x20a1),
TextureRectangleNv = ((int)0x20a2) }
• enum WGL_NV_render_depth_texture {

DepthTextureFormatNv = ((int)0x20a5), TextureDepthComponentNv =
((int)0x20a6), BindToTextureDepthNv = ((int)0x20a3), DepthComponentNv
= ((int)0x20a7),

BindToTextureRectangleDepthNv = ((int)0x20a4) }
• enum WGL_NV_float_buffer {

BindToTextureRectangleFloatRNv = ((int)0x20b1), TextureFloatRNv =
((int)0x20b5), TextureFloatRgbNv = ((int)0x20b7), TextureFloatRgNv =
((int)0x20b6),

TextureFloatRgbaNv = ((int)0x20b8), BindToTextureRectangleFloatRg-
baNv = ((int)0x20b4), FloatComponentsNv = ((int)0x20b0), BindToTextur-
eRectangleFloatRgNv = ((int)0x20b2),

BindToTextureRectangleFloatRgbNv = ((int)0x20b3) }
• enum WGL_ARB_pixel_format_float { TypeRgbaFloatArb = ((int)0x21a0)
}
• enum WGL_ATI_pixel_format_float { TypeRgbaFloatAti = ((int)0x21a0) }
• enum WGL_font_type { FontLines = ((int)0) }
• enum All {

SwapCopyExt = ((int)0x2029), BackColorBufferBitArb = ((int)0x00000002),
FullAccelerationArb = ((int)0x2027), AccelerationExt = ((int)0x2003),

GenlockSourceMultiviewI3d = ((int)0x2044), Aux3Arb = ((int)0x208a),
TextureCubeMapNegativeYArb = ((int)0x2080), DoubleBufferArb =
((int)0x2011),

```

SwapUndefinedExt = ((int)0x202a), **SupportGdiArb** = ((int)0x200f),
Aux2Arb = ((int)0x2089), **TextureCubeMapArb** = ((int)0x2078),

SwapLayerBuffersExt = ((int)0x2006), **SwapCopyArb** = ((int)0x2029), **ErrorIncompatibleDeviceContextsArb** = ((int)0x2054), **TypeColorindexArb** = ((int)0x202c),

DigitalVideoCursorIncludedI3d = ((int)0x2052), **NeedPaletteExt** = ((int)0x2004), **RedBitsArb** = ((int)0x2015), **TextureCubeMapNegativeXArb** = ((int)0x207e),

SampleBuffersExt = ((int)0x2041), **GenericAccelerationExt** = ((int)0x2026),
BindToTextureRectangleRgbaNv = ((int)0x20a1), **NoTextureArb** = ((int)0x2077),

FrontColorBufferBitArb = ((int)0x00000001), **TransparentValueExt** = ((int)0x200b), **AlphaBitsArb** = ((int)0x201b), **RedBitsExt** = ((int)0x2015),

PbufferHeightArb = ((int)0x2035), **BindToTextureRectangleFloatRgbaNv** = ((int)0x20b4), **SampleBuffersArb** = ((int)0x2041), **MipmapLevelArb** = ((int)0x207b),

NeedSystemPaletteExt = ((int)0x2005), **Aux4Arb** = ((int)0x208b), **TextureFormatArb** = ((int)0x2072), **AccumBitsExt** = ((int)0x201d),

AccumBlueBitsExt = ((int)0x2020), **BackLeftArb** = ((int)0x2085), **AlphaBitsExt** = ((int)0x201b), **StencilBitsArb** = ((int)0x2023),

DrawToPbufferExt = ((int)0x202d), **FullAccelerationExt** = ((int)0x2027),
ColorBitsExt = ((int)0x2014), **BindToTextureRectangleFloatRgNv** = ((int)0x20b2),

DepthBufferBitArb = ((int)0x00000004), **BindToTextureRgbaArb** = ((int)0x2071), **AccumGreenBitsArb** = ((int)0x201f), **AccumBitsArb** = ((int)0x201d),

TypeRgbaFloatArb = ((int)0x21a0), **NeedPaletteArb** = ((int)0x2004),
ShareAccumArb = ((int)0x200e), **TransparentArb** = ((int)0x200a),

ShareStencilArb = ((int)0x200d), **Aux5Arb** = ((int)0x208c), **ImageBufferLockI3d** = ((int)0x00000002), **TextureFloatRNv** = ((int)0x20b5),

DepthComponentNv = ((int)0x20a7), **FloatComponentsNv** = ((int)0x20b0),
TransparentGreenValueArb = ((int)0x2038), **GenlockSourceExternalI3d** = ((int)0x2047),

NeedSystemPaletteArb = ((int)0x2005), **BlueBitsExt** = ((int)0x2019), **GreenShiftExt** = ((int)0x2018), **OptimalPbufferWidthExt** = ((int)0x2031),

AuxBuffersExt = ((int)0x2024), **TypeRgbaFloatAti** = ((int)0x21a0), **FrontRightArb** = ((int)0x2084), **DepthBitsExt** = ((int)0x2022),

GammaTableSizeI3d = ((int)0x204e), **AccumAlphaBitsArb** = ((int)0x2021),
Aux0Arb = ((int)0x2087), **TransparentIndexValueArb** = ((int)0x203b),

AccumGreenBitsExt = ((int)0x201f), **TransparentBlueValueArb** = ((int)0x2039), **NoAccelerationArb** = ((int)0x2025), **MaxPbufferPixelsArb** = ((int)0x202e),

GammaExcludeDesktopI3d = ((int)0x204f), **MaxPbufferPixelsExt** = ((int)0x202e), **AccumBlueBitsArb** = ((int)0x2020), **SwapUndefinedArb** = ((int)0x202a),

ShareDepthExt = ((int)0x200c), **GenlockSourceEdgeBothI3d** = ((int)0x204c), **Samples3dfx** = ((int)0x2061), **DoubleBufferExt** = ((int)0x2011),

BindToTextureRectangleFloatRgbNv = ((int)0x20b3), **SwapMethodExt** = ((int)0x2007), **ErrorInvalidPixelTypeArb** = ((int)0x2043), **GreenShiftArb** = ((int)0x2018),

TextureFloatRgbaNv = ((int)0x20b8), **Aux1Arb** = ((int)0x2088), **GreenBitsArb** = ((int)0x2017), **NumberPixelFormatsExt** = ((int)0x2000),

NumberOverlaysExt = ((int)0x2008), **PixelTypeArb** = ((int)0x2013), **SwapLayerBuffersArb** = ((int)0x2006), **DrawToBitmapArb** = ((int)0x2002),

NumberPixelFormatsArb = ((int)0x2000), **PbufferLostArb** = ((int)0x2036), **Aux9Arb** = ((int)0x2090), **TextureCubeMapPositiveZArb** = ((int)0x2081),

MaxPbufferHeightArb = ((int)0x2030), **TransparentExt** = ((int)0x200a), **PbufferLargestArb** = ((int)0x2033), **SwapMethodArb** = ((int)0x2007),

TextureRgbaArb = ((int)0x2076), **PbufferWidthExt** = ((int)0x2034), **OptimalPbufferHeightExt** = ((int)0x2032), **StencilBitsExt** = ((int)0x2023),

ShareStencilExt = ((int)0x200d), **DepthFloatExt** = ((int)0x2040), **BindToTextureRgbArb** = ((int)0x2070), **BindToTextureRectangleRgbNv** = ((int)0x20a0),

GenlockSourceDigitalSyncI3d = ((int)0x2048), **AccumAlphaBitsExt** = ((int)0x2021), **GenlockSourceExtenalSyncI3d** = ((int)0x2045), **RedShiftExt** = ((int)0x2016),

GenlockSourceDigitalFieldI3d = ((int)0x2049), **FrontLeftArb** = ((int)0x2083), **BlueShiftArb** = ((int)0x201a), **PbufferWidthArb** = ((int)0x2034),

CubeMapFaceArb = ((int)0x207c), **StencilBufferBitArb** = ((int)0x00000008), **NumberOverlaysArb** = ((int)0x2008), **SwapExchangeExt** = ((int)0x2028),

BackRightArb = ((int)0x2086), **DepthTextureFormatNv** = ((int)0x20a5), **TextureFloatRgNv** = ((int)0x20b6), **Texture1dArb** = ((int)0x2079),

DepthBitsArb = ((int)0x2022), **BindToTextureDepthNv** = ((int)0x20a3), **DrawToWindowArb** = ((int)0x2001), **TypeRgbaExt** = ((int)0x202b),

DigitalVideoCursorAlphaValueI3d = ((int)0x2051), **ErrorInvalidPixelTypeExt** = ((int)0x2043), **AccumRedBitsExt** = ((int)0x201e), **GreenBitsExt** = ((int)0x2017),

TypeRgbaArb = ((int)0x202b), **DigitalVideoCursorAlphaFramebufferI3d** = ((int)0x2050), **AuxBuffersArb** = ((int)0x2024), **AccumRedBitsArb** = ((int)0x201e),

TextureFloatRgbNv = ((int)0x20b7), **TypeColorindexExt** = ((int)0x202c), **TransparentAlphaValueArb** = ((int)0x203a), **BlueShiftExt** = ((int)0x201a),

RedShiftArb = ((int)0x2016), **PbufferHeightExt** = ((int)0x2035), **GenlockSourceEdgeRisingI3d** = ((int)0x204b), **Texture2DArb** = ((int)0x207a),

NumberUnderlaysArb = ((int)0x2009), **NumberUnderlaysExt** = ((int)0x2009), **DrawToBitmapExt** = ((int)0x2002), **ShareDepthArb** = ((int)0x200c),

TextureDepthComponentNv = ((int)0x20a6), **NoAccelerationExt** = ((int)0x2025), **PixelTypeExt** = ((int)0x2013), **SupportOpenglArb** = ((int)0x2010),

TextureCubeMapPositiveYArb = ((int)0x207f), **DrawToWindowExt** = ((int)0x2001), **PbufferLargestExt** = ((int)0x2033), **DrawToPbufferArb** = ((int)0x202d),

SupportOpenglExt = ((int)0x2010), **SampleBuffers3dfx** = ((int)0x2060), **GenlockSourceExtenalFieldI3d** = ((int)0x2046), **MaxPbufferHeightExt** = ((int)0x2030),

SupportGdiExt = ((int)0x200f), **Aux7Arb** = ((int)0x208e), **DigitalVideoGammaCorrectedI3d** = ((int)0x2053), **ColorBitsArb** = ((int)0x2014),

Aux6Arb = ((int)0x208d), **ShareAccumExt** = ((int)0x200e), **StereoArb** = ((int)0x2012), **TextureRgbArb** = ((int)0x2075),

AccelerationArb = ((int)0x2003), **TextureCubeMapPositiveXArb** = ((int)0x207d), **TransparentRedValueArb** = ((int)0x2037), **BlueBitsArb** = ((int)0x2019),

SwapExchangeArb = ((int)0x2028), **SamplesExt** = ((int)0x2042), **AlphaShiftExt** = ((int)0x201c), **SamplesArb** = ((int)0x2042),

TextureTargetArb = ((int)0x2073), **BindToTextureRectangleDepthNv** = ((int)0x20a4), **AlphaShiftArb** = ((int)0x201c), **Aux8Arb** = ((int)0x208f),

MaxPbufferWidthExt = ((int)0x202f), **GenlockSourceEdgeFallingI3d** = ((int)0x204a), **StereoExt** = ((int)0x2012), **MaxPbufferWidthArb** = ((int)0x202f),

TextureRectangleNv = ((int)0x20a2), **ImageBufferMinAccessI3d** = ((int)0x00000001), **TextureCubeMapNegativeZArb** = ((int)0x2082), **MipmapTextureArb** = ((int)0x2074),

GenericAccelerationArb = ((int)0x2026), **BindToTextureRectangleFloatRNv** = ((int)0x20b1), **FontLines** = ((int)0) }

- enum **WGL_ARB_extensions_string**
- enum **WGL_I3D_image_buffer** { **ImageBufferMinAccessI3d** = ((int)0x00000001), **ImageBufferLockI3d** = ((int)0x00000002) }
- enum **WGL_I3D_swap_frame_lock**

Functions

- internal delegate IntPtr **WindowProcedure** (IntPtr handle, [WindowMessage](#) message, IntPtr wParam, IntPtr lParam)

4.15.1 Enumeration Type Documentation

4.15.1.1 enum OpenTK::Platform::Windows::GdiCharset

Enumerates the available character sets.

4.15.1.2 enum OpenTK::Platform::Windows::GWL

Window field offsets for `GetWindowLong()` and `GetWindowLongPtr()`.

4.15.1.3 enum OpenTK::Platform::Windows::MapVirtualKeyType

Enumerator:

VirtualKeyToScanCode uCode is a virtual-key code and is translated into a scan code. If it is a virtual-key code that does not distinguish between left- and right-hand keys, the left-hand scan code is returned. If there is no translation, the function returns 0.

ScanCodeToVirtualKey uCode is a scan code and is translated into a virtual-key code that does not distinguish between left- and right-hand keys. If there is no translation, the function returns 0.

VirtualKeyToCharacter uCode is a virtual-key code and is translated into an unshifted character value in the low-order word of the return value. Dead keys (diacritics) are indicated by setting the top bit of the return value. If there is no translation, the function returns 0.

ScanCodeToVirtualKeyExtended [Windows](#) NT/2000/XP: uCode is a scan code and is translated into a virtual-key code that distinguishes between left- and right-hand keys. If there is no translation, the function returns 0.

4.15.1.4 enum OpenTK::Platform::Windows::MouseKeys

Enumerates available mouse keys (suitable for use in WM_MOUSEMOVE messages).

4.15.1.5 enum OpenTK::Platform::Windows::QueueStatusFlags

Enumerator:

KEY A WM_KEYUP, WM_KEYDOWN, WM_SYSKEYUP, or WM_SYSKEYDOWN message is in the queue.

MOUSEMOVE A WM_MOUSEMOVE message is in the queue.

MOUSEBUTTON A mouse-button message (WM_LBUTTONDOWN, WM_RBUTTONDOWN, and so on).

POSTMESSAGE A posted message (other than those listed here) is in the queue.

TIMER A WM_TIMER message is in the queue.

PAINT A WM_PAINT message is in the queue.

SENDMESSAGE A message sent by another thread or application is in the queue.

HOTKEY A WM_HOTKEY message is in the queue.

ALLPOSTMESSAGE A posted message (other than those listed here) is in the queue.

RAWINPUT A raw input message is in the queue. For more information, see [Raw Input](#). [Windows](#) XP and higher only.

MOUSE A WM_MOUSEMOVE message or mouse-button message (WM_LBUTTONDOWN, WM_RBUTTONDOWN, and so on).

INPUT An input message is in the queue. This is composed of KEY, MOUSE and RAWINPUT. [Windows](#) XP and higher only.

INPUT_LEGACY An input message is in the queue. This is composed of QS_KEY and QS_MOUSE. [Windows](#) 2000 and earlier.

ALLEVENTS An input, WM_TIMER, WM_PAINT, WM_HOTKEY, or posted message is in the queue.

ALLINPUT Any message is in the queue.

4.15.1.6 enum OpenTK::Platform::Windows::RawInputDeviceFlags

Enumerator:

REMOVE If set, this removes the top level collection from the inclusion list. This tells the operating system to stop reading from a device which matches the top level collection.

EXCLUDE If set, this specifies the top level collections to exclude when reading a complete usage page. This flag only affects a TLC whose usage page is already specified with RawInputDeviceEnum.PAGEONLY.

PAGEONLY If set, this specifies all devices whose top level collection is from the specified UsagePage. Note that usUsage must be zero. To exclude a particular top level collection, use EXCLUDE.

NOLEGACY If set, this prevents any devices specified by UsagePage or Usage from generating legacy messages. This is only for the mouse and keyboard. See RawInputDevice Remarks.

INPUTSINK If set, this enables the caller to receive the input even when the caller is not in the foreground. Note that Target must be specified in RawInputDevice.

CAPTUREMOUSE If set, the mouse button click does not activate the other window.

NOHOTKEYS If set, the application-defined keyboard device hotkeys are not handled. However, the system hotkeys; for example, ALT+TAB and CTRL+ALT+DEL, are still handled. By default, all keyboard hotkeys are handled. NOHOTKEYS can be specified even if NOLEGACY is not specified and Target is NULL in RawInputDevice.

APPKEYS Microsoft Windows XP Service Pack 1 (SP1): If set, the application command keys are handled. APPKEYS can be specified only if NOLEGACY is specified for a keyboard device.

EXINPUTSINK If set, this enables the caller to receive input in the background only if the foreground application does not process it. In other words, if the foreground application is not registered for raw input, then the background application that is registered will receive the input.

4.15.1.7 enum OpenTK::Platform::Windows::RawMouseFlags

Mouse indicator flags (found in winuser.h).

Enumerator:

MOUSE_MOVE_RELATIVE LastX/Y indicate relative motion.

MOUSE_MOVE_ABSOLUTE LastX/Y indicate absolute motion.

MOUSE_VIRTUAL_DESKTOP The coordinates are mapped to the virtual desktop.

MOUSE_ATTRIBUTES_CHANGED Requery for mouse attributes.

4.15.1.8 enum OpenTK::Platform::Windows::SetWindowPosFlags

Enumerator:

NOSIZE Retains the current size (ignores the cx and cy parameters).

NOMOVE Retains the current position (ignores the x and y parameters).

NOZORDER Retains the current Z order (ignores the hwndInsertAfter parameter).

NOREDRAW Does not redraw changes. If this flag is set, no repainting of any kind occurs. This applies to the client area, the nonclient area (including the title bar and scroll bars), and any part of the parent window uncovered as a result of the window being moved. When this flag is set, the application must explicitly invalidate or redraw any parts of the window and parent window that need redrawing.

NOACTIVATE Does not activate the window. If this flag is not set, the window is activated and moved to the top of either the topmost or non-topmost group (depending on the setting of the hwndInsertAfter member).

FRAMECHANGED Sends a WM_NCCALCSIZE message to the window, even if the window's size is not being changed. If this flag is not specified, WM_NCCALCSIZE is sent only when the window's size is being changed.

SHOWWINDOW Displays the window.

HIDEWINDOW Hides the window.

NOCOPYBITS Discards the entire contents of the client area. If this flag is not specified, the valid contents of the client area are saved and copied back into the client area after the window is sized or repositioned.

NOOWNERZORDER Does not change the owner window's position in the Z order.

NOSENDCHANGING Prevents the window from receiving the WM_WINDOWPOSCHANGING message.

DRAWFRAME Draws a frame (defined in the window's class description) around the window.

NOREPOSITION Same as the NOOWNERZORDER flag.

4.15.1.9 enum OpenTK::Platform::Windows::ShGetFileIconFlags

Enumerator:

Icon get icon

DisplayName get display name

TypeName get type name

Attributes get attributes

IconLocation get icon location

ExeType return exe type

SysIconIndex get system icon index

LinkOverlay put a link overlay on icon
Selected show icon in selected state
Attr_Specified get only specified attributes
LargeIcon get large icon
SmallIcon get small icon
OpenIcon get open icon
ShellIconSize get shell size icon
PIDL pszPath is a pidl
UseFileAttributes use passed dwFileAttribute
AddOverlays apply the appropriate overlays
OverlayIndex Get the index of the overlay in the upper 8 bits of the iIcon.

4.15.1.10 enum OpenTK::Platform::Windows::ShowWindowCommand

ShowWindow() Commands.

Enumerator:

HIDE Hides the window and activates another window.
SHOWNORMAL Activates and displays a window. If the window is minimized or maximized, the system restores it to its original size and position. An application should specify this flag when displaying the window for the first time.
SHOWMINIMIZED Activates the window and displays it as a minimized window.
SHOWMAXIMIZED Activates the window and displays it as a maximized window.
SHOWNOACTIVATE Displays the window as a minimized window. This value is similar to SW_SHOWMINIMIZED, except the window is not activated.
SHOW Activates the window and displays it in its current size and position.
MINIMIZE Minimizes the specified window and activates the next top-level window in the Z order.
SHOWMINNOACTIVE Displays the window as a minimized window. This value is similar to SW_SHOWMINIMIZED, except the window is not activated.
SHOWNA Displays the window in its current size and position. This value is similar to SW_SHOW, except the window is not activated.
RESTORE Activates and displays the window. If the window is minimized or maximized, the system restores it to its original size and position. An application should specify this flag when restoring a minimized window.

SHOWDEFAULT Sets the show state based on the `SW_` value specified in the `STARTUPINFO` structure passed to the `CreateProcess` function by the program that started the application.

FORCEMINIMIZE [Windows](#) 2000/XP: Minimizes a window, even if the thread that owns the window is not responding. This flag should only be used when minimizing windows from a different thread.

4.15.1.11 enum `OpenTK::Platform::Windows::ShowWindowMessageIdentifiers`

Identifiers for the `WM_SHOWWINDOW` message.

4.15.1.12 enum `OpenTK::Platform::Windows::WindowMessage`

Enumerator:

NCXBUTTONDOWN [Windows](#) 2000 and higher only.

NCXBUTTONUP [Windows](#) 2000 and higher only.

NCXBUTTONDBLCLK [Windows](#) 2000 and higher only.

XBUTTONDOWN [Windows](#) 2000 and higher only.

XBUTTONUP [Windows](#) 2000 and higher only.

XBUTTONDBLCLK [Windows](#) 2000 and higher only.

4.16 Package `OpenTK.Platform.X11`

Enumerations

- enum `MotifFlags` { `Functions` = 1, `Decorations` = 2, `InputMode` = 4, `Status` = 8 }
- enum `MotifFunctions` {
`All` = 0x01, `Resize` = 0x02, `Move` = 0x04, `Minimize` = 0x08,
`Maximize` = 0x10, `Close` = 0x20 }
- enum `MotifDecorations` {
`All` = 0x01, `Border` = 0x02, `ResizeH` = 0x04, `Title` = 0x08,
`Menu` = 0x10, `Minimize` = 0x20, `Maximize` = 0x40 }
- enum `MotifInputMode` { `Modeless` = 0, `ApplicationModal` = 1, `SystemModal` = 2, `FullApplicationModal` = 3 }

- enum **WindowLayer** {
Desktop = 0, **Below** = 2, **Normal** = 4, **OnTop** = 6,
Dock = 8, **AboveDock** = 10, **Menu** = 12 }
- enum **WindowState** {
Sticky = (1<<0), **Minimized** = (1<<1), **MaximizedVertically** = (1<<2),
MaximizedHorizontally = (1<<3),
Hidden = (1<<4), **Shaded** = (1<<5), **HID_WORKSPACE** = (1<<6), **HID_ - TRANSIENT** = (1<<7),
FixedPosition = (1<<8), **ArrangeIgnore** = (1<<9) }
- enum **WindowHints** {
SkipFocus = (1<<0), **SkipWinlist** = (1<<1), **SkipTaskbar** = (1<<2), **Group-Transient** = (1<<3),
FocusOnClick = (1<<4), **DoNotCover** = (1<<5) }
- enum **ErrorCodes** {
Success = 0, **BadRequest** = 1, **BadValue** = 2, **BadWindow** = 3,
BadPixmap = 4, **BadAtom** = 5, **BadCursor** = 6, **BadFont** = 7,
BadMatch = 8, **BadDrawable** = 9, **BadAccess** = 10, **BadAlloc** = 11,
BadColor = 12, **BadGC** = 13, **BadIDChoice** = 14, **BadName** = 15,
BadLength = 16, **BadImplementation** = 17 }
- enum **CreateWindowMask** {
CWBackPixmap = (1L<<0), **CWBackPixel** = (1L<<1), **CWSaveUnder** = (1L<<10), **CWEventMask** = (1L<<11),
CWDontPropagate = (1L<<12), **CWColormap** = (1L<<13), **CWCursor** = (1L<<14), **CWBorderPixmap** = (1L<<2),
CWBorderPixel = (1L<<3), **CWBitGravity** = (1L<<4), **CWWinGravity** = (1L<<5), **CWBackingStore** = (1L<<6),
CWBackingPlanes = (1L<<7), **CWBackingPixel** = (1L<<8), **CWOverrideRedirect** = (1L<<9) }
- enum **XKey** {
BackSpace = 0xff08, **Tab** = 0xff09, **Linefeed** = 0xff0a, **Clear** = 0xff0b,
Return = 0xff0d, **Pause** = 0xff13, **Scroll_Lock** = 0xff14, **Sys_Req** = 0xff15,
Escape = 0xff1b, **Delete** = 0xffff, **Multi_key** = 0xff20, **Codeinput** = 0xff37,
SingleCandidate = 0xff3c, **MultipleCandidate** = 0xff3d, **PreviousCandidate** = 0xff3e, **Kanji** = 0xff21,
Muhenkan = 0xff22, **Henkan_Mode** = 0xff23, **Henkan** = 0xff23, **Romaji** = 0xff24,
Hiragana = 0xff25, **Katakana** = 0xff26, **Hiragana_Katakana** = 0xff27,
Zenkaku = 0xff28,

Hankaku = 0xff29, **Zenkaku_Hankaku** = 0xff2a, **Touroku** = 0xff2b, **Massyo** = 0xff2c,
Kana_Lock = 0xff2d, **Kana_Shift** = 0xff2e, **Eisu_Shift** = 0xff2f, **Eisu_toggle** = 0xff30,
Kanji_Bangou = 0xff37, **Zen_Koho** = 0xff3d, **Mae_Koho** = 0xff3e, **Home** = 0xff50,
Left = 0xff51, **Up** = 0xff52, **Right** = 0xff53, **Down** = 0xff54,
Prior = 0xff55, **Page_Up** = 0xff55, **Next** = 0xff56, **Page_Down** = 0xff56,
End = 0xff57, **Begin** = 0xff58, **Select** = 0xff60, **Print** = 0xff61,
Execute = 0xff62, **Insert** = 0xff63, **Undo** = 0xff65, **Redo** = 0xff66,
Menu = 0xff67, **Find** = 0xff68, **Cancel** = 0xff69, **Help** = 0xff6a,
Break = 0xff6b, **Mode_switch** = 0xff7e, **script_switch** = 0xff7e, **Num_Lock** = 0xff7f,
KP_Space = 0xff80, **KP_Tab** = 0xff89, **KP_Enter** = 0xff8d, **KP_F1** = 0xff91,
KP_F2 = 0xff92, **KP_F3** = 0xff93, **KP_F4** = 0xff94, **KP_Home** = 0xff95,
KP_Left = 0xff96, **KP_Up** = 0xff97, **KP_Right** = 0xff98, **KP_Down** = 0xff99,
KP_Prior = 0xff9a, **KP_Page_Up** = 0xff9a, **KP_Next** = 0xff9b, **KP_Page_Down** = 0xff9b,
KP_End = 0xff9c, **KP_Begin** = 0xff9d, **KP_Insert** = 0xff9e, **KP_Delete** = 0xff9f,
KP_Equal = 0xffbd, **KP_Multiply** = 0xffaa, **KP_Add** = 0xffab, **KP_Separator** = 0xffac,
KP_Subtract = 0xffad, **KP_Decimal** = 0xffae, **KP_Divide** = 0xffaf, **KP_0** = 0xffb0,
KP_1 = 0xffb1, **KP_2** = 0xffb2, **KP_3** = 0xffb3, **KP_4** = 0xffb4,
KP_5 = 0xffb5, **KP_6** = 0xffb6, **KP_7** = 0xffb7, **KP_8** = 0xffb8,
KP_9 = 0xffb9, **F1** = 0xffbe, **F2** = 0xffbf, **F3** = 0xffc0,
F4 = 0xffc1, **F5** = 0xffc2, **F6** = 0xffc3, **F7** = 0xffc4,
F8 = 0xffc5, **F9** = 0xffc6, **F10** = 0xffc7, **F11** = 0xffc8,
L1 = 0xffc8, **F12** = 0xffc9, **L2** = 0xffc9, **F13** = 0xffca,
L3 = 0xffca, **F14** = 0xffcb, **L4** = 0xffcb, **F15** = 0xffcc,
L5 = 0xffcc, **F16** = 0xffcd, **L6** = 0xffcd, **F17** = 0xffce,
L7 = 0xffce, **F18** = 0xffcf, **L8** = 0xffcf, **F19** = 0xffd0,
L9 = 0xffd0, **F20** = 0xffd1, **L10** = 0xffd1, **F21** = 0xffd2,
R1 = 0xffd2, **F22** = 0xffd3, **R2** = 0xffd3, **F23** = 0xffd4,
R3 = 0xffd4, **F24** = 0xffd5, **R4** = 0xffd5, **F25** = 0xffd6,
R5 = 0xffd6, **F26** = 0xffd7, **R6** = 0xffd7, **F27** = 0xffd8,

R7 = 0xffd8, **F28** = 0xffd9, **R8** = 0xffd9, **F29** = 0xffda,
R9 = 0xffda, **F30** = 0xffdb, **R10** = 0xffdb, **F31** = 0xffdc,
R11 = 0xffdc, **F32** = 0xffdd, **R12** = 0xffdd, **F33** = 0xffde,
R13 = 0xffde, **F34** = 0xffdf, **R14** = 0xffdf, **F35** = 0xffe0,
R15 = 0xffe0, **Shift_L** = 0xffe1, **Shift_R** = 0xffe2, **Control_L** = 0xffe3,
Control_R = 0xffe4, **Caps_Lock** = 0xffe5, **Shift_Lock** = 0xffe6, **Meta_L** =
0xffe7,
Meta_R = 0xffe8, **Alt_L** = 0xffe9, **Alt_R** = 0xffea, **Super_L** = 0xffeb,
Super_R = 0xffec, **Hyper_L** = 0xffed, **Hyper_R** = 0xffee, **space** = 0x0020,
exclam = 0x0021, **quotedbl** = 0x0022, **numbersign** = 0x0023, **dollar** = 0x0024,
percent = 0x0025, **ampersand** = 0x0026, **apostrophe** = 0x0027, **quoteright** =
0x0027,
parenleft = 0x0028, **parenright** = 0x0029, **asterisk** = 0x002a, **plus** = 0x002b,
comma = 0x002c, **minus** = 0x002d, **period** = 0x002e, **slash** = 0x002f,
Number0 = 0x0030, **Number1** = 0x0031, **Number2** = 0x0032, **Number3** =
0x0033,
Number4 = 0x0034, **Number5** = 0x0035, **Number6** = 0x0036, **Number7** =
0x0037,
Number8 = 0x0038, **Number9** = 0x0039, **colon** = 0x003a, **semicolon** = 0x003b,
less = 0x003c, **equal** = 0x003d, **greater** = 0x003e, **question** = 0x003f,
at = 0x0040, **A** = 0x0041, **B** = 0x0042, **C** = 0x0043,
D = 0x0044, **E** = 0x0045, **F** = 0x0046, **G** = 0x0047,
H = 0x0048, **I** = 0x0049, **J** = 0x004a, **K** = 0x004b,
L = 0x004c, **M** = 0x004d, **N** = 0x004e, **O** = 0x004f,
P = 0x0050, **Q** = 0x0051, **R** = 0x0052, **S** = 0x0053,
T = 0x0054, **U** = 0x0055, **V** = 0x0056, **W** = 0x0057,
X = 0x0058, **Y** = 0x0059, **Z** = 0x005a, **bracketleft** = 0x005b,
backslash = 0x005c, **bracketright** = 0x005d, **asciicircum** = 0x005e, **under-**
score = 0x005f,
grave = 0x0060, **quoteleft** = 0x0060, **a** = 0x0061, **b** = 0x0062,
c = 0x0063, **d** = 0x0064, **e** = 0x0065, **f** = 0x0066,
g = 0x0067, **h** = 0x0068, **i** = 0x0069, **j** = 0x006a,
k = 0x006b, **l** = 0x006c, **m** = 0x006d, **n** = 0x006e,
o = 0x006f, **p** = 0x0070, **q** = 0x0071, **r** = 0x0072,
s = 0x0073, **t** = 0x0074, **u** = 0x0075, **v** = 0x0076,
w = 0x0077, **x** = 0x0078, **y** = 0x0079, **z** = 0x007a,

```
braceleft = 0x007b, bar = 0x007c, braceright = 0x007d, asciitilde = 0x007e
}
```

Defines LATIN-1 and miscellaneous keys.

- enum **XVisualClass** {
StaticGray = 0, **GrayScale** = 1, **StaticColor** = 2, **PseudoColor** = 3,
TrueColor = 4, **DirectColor** = 5 }
- enum **XVisualInfoMask** {
No = 0x0, **ID** = 0x1, **Screen** = 0x2, **Depth** = 0x4,
Class = 0x8, **Red** = 0x10, **Green** = 0x20, **Blue** = 0x40,
ColormapSize = 0x80, **BitsPerRGB** = 0x100, **All** = 0x1FF }
- enum **MouseMask** {
Button1MotionMask = (1 << 8), **Button2MotionMask** = (1 << 9), **Button3MotionMask** = (1 << 10), **Button4MotionMask** = (1 << 11),
Button5MotionMask = (1 << 12), **Button1Mask** = (1 << 8), **Button2Mask** = (1 << 9), **Button3Mask** = (1 << 10),
Button4Mask = (1 << 11), **Button5Mask** = (1 << 12), **Button6Mask** = (1 << 13), **Button7Mask** = (1 << 14),
Button8Mask = (1 << 15), **ShiftMask** = (1 << 0), **LockMask** = (1 << 1), **ControlMask** = (1 << 2),
Mod1Mask = (1 << 3), **Mod2Mask** = (1 << 4), **Mod3Mask** = (1 << 5), **Mod4Mask** = (1 << 6),
Mod5Mask = (1 << 7) }
- enum **GLXAttribute** {
TRANSPARENT_BLUE_VALUE_EXT = 0x27, **GRAY_SCALE** = 0x8006,
RGBA_TYPE = 0x8014, **TRANSPARENT_RGB_EXT** = 0x8008,
ACCUM_BLUE_SIZE = 16, **SHARE_CONTEXT_EXT** = 0x800A,
STEREO = 6, **ALPHA_SIZE** = 11,
FLOAT_COMPONENTS_NV = 0x20B0, **NONE** = 0x8000, **DEPTH_SIZE** = 12, **TRANSPARENT_INDEX_VALUE_EXT** = 0x24,
MAX_PBUFFER_WIDTH_SGIX = 0x8016, **GREEN_SIZE** = 9, **X_RENDERABLE_SGIX** = 0x8012, **LARGEST_PBUFFER** = 0x801C,
DONT_CARE = unchecked((int)0xFFFFFFFF), **TRANSPARENT_ALPHA_VALUE_EXT** = 0x28, **PSEUDO_COLOR_EXT** = 0x8004, **USE_GL** = 1,
SAMPLE_BUFFERS_SGIS = 100000, **TRANSPARENT_GREEN_VALUE_EXT** = 0x26, **HYPERPIPE_ID_SGIX** = 0x8030, **COLOR_INDEX_TYPE_SGIX** = 0x8015,
SLOW_CONFIG = 0x8001, **PRESERVED_CONTENTS** = 0x801B, **ACCUM_RED_SIZE** = 14, **EVENT_MASK** = 0x801F,

VISUAL_ID_EXT = 0x800B, **EVENT_MASK_SGIX** = 0x801F, **SLOW_VISUAL_EXT** = 0x8001, **TRANSPARENT_GREEN_VALUE** = 0x26,
MAX_PBUFFER_WIDTH = 0x8016, **DIRECT_COLOR_EXT** = 0x8003,
VISUAL_ID = 0x800B, **ACCUM_GREEN_SIZE** = 15,
DRAWABLE_TYPE_SGIX = 0x8010, **SCREEN_EXT** = 0x800C, **SAMPLES** = 100001, **HEIGHT** = 0x801E,
TRANSPARENT_INDEX_VALUE = 0x24, **SAMPLE_BUFFERS_ARB** = 100000, **PBUFFER** = 0x8023, **RGBA_TYPE_SGIX** = 0x8014,
MAX_PBUFFER_HEIGHT = 0x8017, **FBCONFIG_ID_SGIX** = 0x8013, **DRAWABLE_TYPE** = 0x8010, **SCREEN** = 0x800C,
RED_SIZE = 8, **VISUAL_SELECT_GROUP_SGIX** = 0x8028, **VISUAL_CAVEAT_EXT** = 0x20, **PSEUDO_COLOR** = 0x8004,
PBUFFER_HEIGHT = 0x8040, **STATIC_GRAY** = 0x8007, **PRESERVED_CONTENTS_SGIX** = 0x801B, **RGBA_FLOAT_TYPE_ARB** = 0x20B9,
TRANSPARENT_RED_VALUE = 0x25, **TRANSPARENT_ALPHA_VALUE** = 0x28, **WINDOW** = 0x8022, **X_RENDERABLE** = 0x8012,
STENCIL_SIZE = 13, **TRANSPARENT_RGB** = 0x8008, **LARGEST_PBUFFER_SGIX** = 0x801C, **STATIC_GRAY_EXT** = 0x8007,
TRANSPARENT_BLUE_VALUE = 0x27, **DIGITAL_MEDIA_PBUFFER_SGIX** = 0x8024, **BLENDED_RGBA_SGIS** = 0x8025, **NON_CONFORMANT_VISUAL_EXT** = 0x800D,
COLOR_INDEX_TYPE = 0x8015, **TRANSPARENT_RED_VALUE_EXT** = 0x25, **GRAY_SCALE_EXT** = 0x8006, **WINDOW_SGIX** = 0x8022,
X_VISUAL_TYPE = 0x22, **MAX_PBUFFER_HEIGHT_SGIX** = 0x8017, **DOUBLEBUFFER** = 5, **OPTIMAL_PBUFFER_WIDTH_SGIX** = 0x8019,
X_VISUAL_TYPE_EXT = 0x22, **WIDTH_SGIX** = 0x801D, **STATIC_COLOR_EXT** = 0x8005, **BUFFER_SIZE** = 2,
DIRECT_COLOR = 0x8003, **MAX_PBUFFER_PIXELS** = 0x8018, **NONE_EXT** = 0x8000, **HEIGHT_SGIX** = 0x801E,
RENDER_TYPE = 0x8011, **FBCONFIG_ID** = 0x8013, **TRANSPARENT_INDEX_EXT** = 0x8009, **TRANSPARENT_INDEX** = 0x8009,
TRANSPARENT_TYPE_EXT = 0x23, **ACCUM_ALPHA_SIZE** = 17, **PBUFFER_SGIX** = 0x8023, **MAX_PBUFFER_PIXELS_SGIX** = 0x8018,
OPTIMAL_PBUFFER_HEIGHT_SGIX = 0x801A, **DAMAGED** = 0x8020, **SAVED_SGIX** = 0x8021, **TRANSPARENT_TYPE** = 0x23,
MULTISAMPLE_SUB_RECT_WIDTH_SGIS = 0x8026, **NON_CONFORMANT_CONFIG** = 0x800D, **BLUE_SIZE** = 10, **TRUE_COLOR_EXT** = 0x8002,
SAMPLES_SGIS = 100001, **SAMPLES_ARB** = 100001, **TRUE_COLOR** = 0x8002, **RGBA** = 4,

```

AUX_BUFFERS = 7, SAMPLE_BUFFERS = 100000, SAVED = 0x8021,
MULTISAMPLE_SUB_RECT_HEIGHT_SGIS = 0x8027,
DAMAGED_SGIX = 0x8020, STATIC_COLOR = 0x8005, PBUFFER_-
WIDTH = 0x8041, WIDTH = 0x801D,
LEVEL = 3, CONFIG_CAVEAT = 0x20, RENDER_TYPE_SGIX = 0x8011
}
• enum GLXHyperpipeAttrib { PIPE_RECT_LIMITS_SGIX = 0x00000002,
PIPE_RECT_SGIX = 0x00000001, HYPERPIPE_STEREO_SGIX =
0x00000003, HYPERPIPE_PIXEL_AVERAGE_SGIX = 0x00000004 }
• enum GLXStringName { EXTENSIONS = 0x3, VERSION = 0x2, VENDOR
= 0x1 }
• enum GLXEventMask { PBUFFER_CLOBBER_MASK = 0x08000000,
BUFFER_CLOBBER_MASK_SGIX = 0x08000000 }
• enum GLXRenderTypeMask {
COLOR_INDEX_BIT_SGIX = 0x00000002, RGBA_BIT = 0x00000001,
RGBA_FLOAT_BIT_ARB = 0x00000004, RGBA_BIT_SGIX =
0x00000001,
COLOR_INDEX_BIT = 0x00000002 }
• enum GLXHyperpipeTypeMask { HYPERPIPE_RENDER_PIPE_SGIX =
0x00000002, HYPERPIPE_DISPLAY_PIPE_SGIX = 0x00000001 }
• enum GLXPbufferClobberMask {
ACCUM_BUFFER_BIT_SGIX = 0x00000080, FRONT_LEFT_BUFFER_-
BIT = 0x00000001, BACK_RIGHT_BUFFER_BIT = 0x00000008,
FRONT_RIGHT_BUFFER_BIT_SGIX = 0x00000002,
STENCIL_BUFFER_BIT_SGIX = 0x00000040, SAMPLE_BUFFERS_-
BIT_SGIX = 0x00000100, STENCIL_BUFFER_BIT = 0x00000040,
BACK_RIGHT_BUFFER_BIT_SGIX = 0x00000008,
BACK_LEFT_BUFFER_BIT_SGIX = 0x00000004, AUX_BUFFERS_BIT
= 0x00000010, DEPTH_BUFFER_BIT_SGIX = 0x00000020, ACCUM_-
BUFFER_BIT = 0x00000080,
AUX_BUFFERS_BIT_SGIX = 0x00000010, DEPTH_BUFFER_BIT =
0x00000020, FRONT_LEFT_BUFFER_BIT_SGIX = 0x00000001, BACK_-
LEFT_BUFFER_BIT = 0x00000004,
FRONT_RIGHT_BUFFER_BIT = 0x00000002 }
• enum GLXHyperpipeMisc { HYPERPIPE_PIPE_NAME_LENGTH_SGIX
= 80 }
• enum GLXErrorCode {
BAD_CONTEXT = 5, NO_EXTENSION = 3, BAD_HYPERPIPE_SGIX =
92, BAD_ENUM = 7,
BAD_SCREEN = 1, BAD_VALUE = 6, BAD_ATTRIBUTE = 2, BAD_-
VISUAL = 4,
BAD_HYPERPIPE_CONFIG_SGIX = 91 }

```


- enum **GLXSyncType** { **SYNC_SWAP_SGIX** = 0x00000001, **SYNC_FRAME_SGIX** = 0x00000000 }
- enum **GLXDrawableTypeMask** {
WINDOW_BIT = 0x00000001, **PIXMAP_BIT** = 0x00000002, **PBUFFER_BIT_SGIX** = 0x00000004, **PBUFFER_BIT** = 0x00000004,
WINDOW_BIT_SGIX = 0x00000001, **PIXMAP_BIT_SGIX** = 0x00000002
}
- enum **ArbCreateContext** {
DebugBit = 0x0001, **ForwardCompatibleBit** = 0x0002, **MajorVersion** = 0x2091, **MinorVersion** = 0x2092,
LayerPlane = 0x2093, **Flags** = 0x2094, **ErrorInvalidVersion** = 0x2095 }
- enum **ErrorCode** {
NO_ERROR = 0, **BAD_SCREEN** = 1, **BAD_ATTRIBUTE** = 2, **NO_EXTENSION** = 3,
BAD_VISUAL = 4, **BAD_CONTEXT** = 5, **BAD_VALUE** = 6, **BAD_ENUM** = 7 }
- enum **XWindowClass** { **InputOutput** = 1, **InputOnly** = 2 }
- enum **XEventName** {
KeyPress = 2, **KeyRelease** = 3, **ButtonPress** = 4, **ButtonRelease** = 5,
MotionNotify = 6, **EnterNotify** = 7, **LeaveNotify** = 8, **FocusIn** = 9,
FocusOut = 10, **KeymapNotify** = 11, **Expose** = 12, **GraphicsExpose** = 13,
NoExpose = 14, **VisibilityNotify** = 15, **CreateNotify** = 16, **DestroyNotify** = 17,
UnmapNotify = 18, **MapNotify** = 19, **MapRequest** = 20, **ReparentNotify** = 21,
ConfigureNotify = 22, **ConfigureRequest** = 23, **GravityNotify** = 24, **ResizeRequest** = 25,
CirculateNotify = 26, **CirculateRequest** = 27, **PropertyNotify** = 28, **SelectionClear** = 29,
SelectionRequest = 30, **SelectionNotify** = 31, **ColormapNotify** = 32, **ClientMessage** = 33,
MappingNotify = 34, **LASTEvent** }
- enum **SetWindowValuemask** {
Nothing = 0, **BackPixmap** = 1, **BackPixel** = 2, **BorderPixmap** = 4,
BorderPixel = 8, **BitGravity** = 16, **WinGravity** = 32, **BackingStore** = 64,
BackingPlanes = 128, **BackingPixel** = 256, **OverrideRedirect** = 512, **SaveUnder** = 1024,
EventMask = 2048, **DontPropagate** = 4096, **ColorMap** = 8192, **Cursor** = 16384 }

- enum **CreateWindowArgs** { **CopyFromParent** = 0, **ParentRelative** = 1, **InputOutput** = 1, **InputOnly** = 2 }
- enum **Gravity** {
ForgetGravity = 0, **NorthWestGravity** = 1, **NorthGravity** = 2, **NorthEastGravity** = 3,
WestGravity = 4, **CenterGravity** = 5, **EastGravity** = 6, **SouthWestGravity** = 7,
SouthGravity = 8, **SouthEastGravity** = 9, **StaticGravity** = 10 }
- enum **XKeySym** {
XK_BackSpace = 0xFF08, **XK_Tab** = 0xFF09, **XK_Clear** = 0xFF0B, **XK_Return** = 0xFF0D,
XK_Home = 0xFF50, **XK_Left** = 0xFF51, **XK_Up** = 0xFF52, **XK_Right** = 0xFF53,
XK_Down = 0xFF54, **XK_Page_Up** = 0xFF55, **XK_Page_Down** = 0xFF56, **XK_End** = 0xFF57,
XK_Begin = 0xFF58, **XK_Menu** = 0xFF67, **XK_Shift_L** = 0xFFE1, **XK_Shift_R** = 0xFFE2,
XK_Control_L = 0xFFE3, **XK_Control_R** = 0xFFE4, **XK_Caps_Lock** = 0xFFE5, **XK_Shift_Lock** = 0xFFE6,
XK_Meta_L = 0xFFE7, **XK_Meta_R** = 0xFFE8, **XK_Alt_L** = 0xFFE9, **XK_Alt_R** = 0xFFEA,
XK_Super_L = 0xFFEB, **XK_Super_R** = 0xFFEC, **XK_Hyper_L** = 0xFFED, **XK_Hyper_R** = 0xFFEE }
- enum **EventMask** {
NoEventMask = 0, **KeyPressMask** = 1 << 0, **KeyReleaseMask** = 1 << 1, **ButtonPressMask** = 1 << 2,
ButtonReleaseMask = 1 << 3, **EnterWindowMask** = 1 << 4, **LeaveWindowMask** = 1 << 5, **PointerMotionMask** = 1 << 6,
PointerMotionHintMask = 1 << 7, **Button1MotionMask** = 1 << 8, **Button2MotionMask** = 1 << 9, **Button3MotionMask** = 1 << 10,
Button4MotionMask = 1 << 11, **Button5MotionMask** = 1 << 12, **ButtonMotionMask** = 1 << 13, **KeymapStateMask** = 1 << 14,
ExposureMask = 1 << 15, **VisibilityChangeMask** = 1 << 16, **StructureNotifyMask** = 1 << 17, **ResizeRedirectMask** = 1 << 18,
SubstructureNotifyMask = 1 << 19, **SubstructureRedirectMask** = 1 << 20, **FocusChangeMask** = 1 << 21, **PropertyChangeMask** = 1 << 22,
ColormapChangeMask = 1 << 23, **OwnerGrabButtonMask** = 1 << 24 }
- enum **GrabMode** { **GrabModeSync** = 0, **GrabModeAsync** = 1 }

- enum **Atom** {
 - AnyPropertyType** = 0, **XA_PRIMARY** = 1, **XA_SECONDARY** = 2, **XA_ARC** = 3,
 - XA_ATOM** = 4, **XA_BITMAP** = 5, **XA_CARDINAL** = 6, **XA_COLORMAP** = 7,
 - XA_CURSOR** = 8, **XA_CUT_BUFFER0** = 9, **XA_CUT_BUFFER1** = 10, **XA_CUT_BUFFER2** = 11,
 - XA_CUT_BUFFER3** = 12, **XA_CUT_BUFFER4** = 13, **XA_CUT_BUFFER5** = 14, **XA_CUT_BUFFER6** = 15,
 - XA_CUT_BUFFER7** = 16, **XA_DRAWABLE** = 17, **XA_FONT** = 18, **XA_INTEGER** = 19,
 - XA_PIXMAP** = 20, **XA_POINT** = 21, **XA_RECTANGLE** = 22, **XA_RESOURCE_MANAGER** = 23,
 - XA_RGB_COLOR_MAP** = 24, **XA_RGB_BEST_MAP** = 25, **XA_RGB_BLUE_MAP** = 26, **XA_RGB_DEFAULT_MAP** = 27,
 - XA_RGB_GRAY_MAP** = 28, **XA_RGB_GREEN_MAP** = 29, **XA_RGB_RED_MAP** = 30, **XA_STRING** = 31,
 - XA_VISUALID** = 32, **XA_WINDOW** = 33, **XA_WM_COMMAND** = 34, **XA_WM_HINTS** = 35,
 - XA_WM_CLIENT_MACHINE** = 36, **XA_WM_ICON_NAME** = 37, **XA_WM_ICON_SIZE** = 38, **XA_WM_NAME** = 39,
 - XA_WM_NORMAL_HINTS** = 40, **XA_WM_SIZE_HINTS** = 41, **XA_WM_ZOOM_HINTS** = 42, **XA_MIN_SPACE** = 43,
 - XA_NORM_SPACE** = 44, **XA_MAX_SPACE** = 45, **XA_END_SPACE** = 46, **XA_SUPERSCRIPT_X** = 47,
 - XA_SUPERSCRIPT_Y** = 48, **XA_SUBSCRIPT_X** = 49, **XA_SUBSCRIPT_Y** = 50, **XA_UNDERLINE_POSITION** = 51,
 - XA_UNDERLINE_THICKNESS** = 52, **XA_STRIKEOUT_ASCENT** = 53, **XA_STRIKEOUT_DESCENT** = 54, **XA_ITALIC_ANGLE** = 55,
 - XA_X_HEIGHT** = 56, **XA_QUAD_WIDTH** = 57, **XA_WEIGHT** = 58, **XA_POINT_SIZE** = 59,
 - XA_RESOLUTION** = 60, **XA_COPYRIGHT** = 61, **XA_NOTICE** = 62, **XA_FONT_NAME** = 63,
 - XA_FAMILY_NAME** = 64, **XA_FULL_NAME** = 65, **XA_CAP_HEIGHT** = 66, **XA_WM_CLASS** = 67,
 - XA_WM_TRANSIENT_FOR** = 68, **XA_LAST_PREDEFINED** = 68 }
- enum **ChangeWindowAttributes** {
 - X** = 1 << 0, **Y** = 1 << 1, **Width** = 1 << 2, **Height** = 1 << 3,
 - BorderWidth** = 1 << 4, **Sibling** = 1 << 5, **StackMode** = 1 << 6, **OverrideRedirect** = 1 << 9 }

- enum **StackMode** {
Above = 0, **Below** = 1, **TopIf** = 2, **BottomIf** = 3,
Opposite = 4 }
- enum **ColorFlags** { **DoRed** = 1 << 0, **DoGreen** = 1 << 1, **DoBlue** = 1 << 2 }
- enum **NotifyMode** { **NotifyNormal** = 0, **NotifyGrab** = 1, **NotifyUngrab** = 2 }
- enum **NotifyDetail** {
NotifyAncestor = 0, **NotifyVirtual** = 1, **NotifyInferior** = 2, **NotifyNonlinear** = 3,
NotifyNonlinearVirtual = 4, **NotifyPointer** = 5, **NotifyPointerRoot** = 6, **NotifyDetailNone** = 7 }
- enum **KeyMasks** {
ShiftMask = (1 << 0), **LockMask** = (1 << 1), **ControlMask** = (1 << 2),
Mod1Mask = (1 << 3),
Mod2Mask = (1 << 4), **Mod3Mask** = (1 << 5), **Mod4Mask** = (1 << 6),
Mod5Mask = (1 << 7),
ModMasks = Mod1Mask | Mod2Mask | Mod3Mask | Mod4Mask | Mod5Mask
}
- enum **PropertyMode** { **Replace** = 0, **Prepend** = 1, **Append** = 2 }
- enum **GCFunction** {
GCFunction = 1 << 0, **GCPlaneMask** = 1 << 1, **GCForeground** = 1 << 2,
GCBgground = 1 << 3,
GCLineWidth = 1 << 4, **GCLineStyle** = 1 << 5, **GCCapStyle** = 1 << 6,
GCJoinStyle = 1 << 7,
GCFillStyle = 1 << 8, **GCFillRule** = 1 << 9, **GCTile** = 1 << 10, **GCStipple** = 1 << 11,
GCTileStipXOrigin = 1 << 12, **GCTileStipYOrigin** = 1 << 13, **GCFont** = 1 << 14,
GCSubwindowMode = 1 << 15,
GCGraphicsExposures = 1 << 16, **GCClipXOrigin** = 1 << 17, **GCClipYOrigin** = 1 << 18,
GCClipMask = 1 << 19,
GCDashOffset = 1 << 20, **GCDashList** = 1 << 21, **GCArcMode** = 1 << 22
}
- enum **GCJoinStyle** { **JoinMiter** = 0, **JoinRound** = 1, **JoinBevel** = 2 }
- enum **GCLineStyle** { **LineSolid** = 0, **LineOnOffDash** = 1, **LineDoubleDash** = 2 }
- enum **GCCapStyle** { **CapNotLast** = 0, **CapButt** = 1, **CapRound** = 2, **CapProjecting** = 3 }
- enum **GCFillStyle** { **FillSolid** = 0, **FillTiled** = 1, **FillStippled** = 2, **FillOpaqueStippled** = 3 }
- enum **GCFillRule** { **EvenOddRule** = 0, **WindingRule** = 1 }
- enum **GCArcMode** { **ArcChord** = 0, **ArcPieSlice** = 1 }

- enum **GCSubwindowMode** { **ClipByChildren** = 0, **IncludeInferiors** = 1 }
- enum **GXFunction** {
 - GXclear** = 0x0, **GXand** = 0x1, **GXandReverse** = 0x2, **GXcopy** = 0x3,
 - GXandInverted** = 0x4, **GXnoop** = 0x5, **GXxor** = 0x6, **GXor** = 0x7,
 - GXnor** = 0x8, **GXequiv** = 0x9, **GXinvert** = 0xa, **GXorReverse** = 0xb,
 - GXcopyInverted** = 0xc, **GXorInverted** = 0xd, **GXnand** = 0xe, **GXset** = 0xf }
- enum **NetWindowManagerState** { **Remove** = 0, **Add** = 1, **Toggle** = 2 }
- enum **RevertTo** { **None** = 0, **PointerRoot** = 1, **Parent** = 2 }
- enum **MapState** { **IsUnmapped** = 0, **IsUnviewable** = 1, **IsViewable** = 2 }
- enum **CursorFontShape** {
 - XC_X_cursor** = 0, **XC_arrow** = 2, **XC_based_arrow_down** = 4, **XC_based_arrow_up** = 6,
 - XC_boat** = 8, **XC_bogosity** = 10, **XC_bottom_left_corner** = 12, **XC_bottom_right_corner** = 14,
 - XC_bottom_side** = 16, **XC_bottom_tee** = 18, **XC_box_spiral** = 20, **XC_center_ptr** = 22,
 - XC_circle** = 24, **XC_clock** = 26, **XC_coffee_mug** = 28, **XC_cross** = 30,
 - XC_cross_reverse** = 32, **XC_crosshair** = 34, **XC_diamond_cross** = 36, **XC_dot** = 38,
 - XC_dotbox** = 40, **XC_double_arrow** = 42, **XC_draft_large** = 44, **XC_draft_small** = 46,
 - XC_draped_box** = 48, **XC_exchange** = 50, **XC_fleur** = 52, **XC_gobbler** = 54,
 - XC_gumby** = 56, **XC_hand1** = 58, **XC_hand2** = 60, **XC_heart** = 62,
 - XC_icon** = 64, **XC_iron_cross** = 66, **XC_left_ptr** = 68, **XC_left_side** = 70,
 - XC_left_tee** = 72, **XC_left_button** = 74, **XC_ll_angle** = 76, **XC_lr_angle** = 78,
 - XC_man** = 80, **XC_middlebutton** = 82, **XC_mouse** = 84, **XC_pencil** = 86,
 - XC_pirate** = 88, **XC_plus** = 90, **XC_question_arrow** = 92, **XC_right_ptr** = 94,
 - XC_right_side** = 96, **XC_right_tee** = 98, **XC_rightbutton** = 100, **XC_rtl_logo** = 102,
 - XC_sailboat** = 104, **XC_sb_down_arrow** = 106, **XC_sb_h_double_arrow** = 108, **XC_sb_left_arrow** = 110,
 - XC_sb_right_arrow** = 112, **XC_sb_up_arrow** = 114, **XC_sb_v_double_arrow** = 116, **XC_sb_shuttle** = 118,
 - XC_sizing** = 120, **XC_spider** = 122, **XC_spraycan** = 124, **XC_star** = 126,
 - XC_target** = 128, **XC_tcross** = 130, **XC_top_left_arrow** = 132, **XC_top_left_corner** = 134,

```

XC_top_right_corner = 136, XC_top_side = 138, XC_top_tee = 140, XC_trek = 142,
XC_ul_angle = 144, XC_umbrella = 146, XC_ur_angle = 148, XC_watch = 150,
XC_xterm = 152, XC_num_glyphs = 154 }
• enum SystrayRequest { SYSTEM_TRAY_REQUEST_DOCK = 0,
SYSTEM_TRAY_BEGIN_MESSAGE = 1, SYSTEM_TRAY_CANCEL_MESSAGE = 2 }
• enum XSizeHintsFlags {
USPosition = (1 << 0), USSize = (1 << 1), PPosition = (1 << 2), PSize = (1
<< 3),
PMinSize = (1 << 4), PMaxSize = (1 << 5), PResizeInc = (1 << 6), PAspect
= (1 << 7),
PAllHints = (PPosition | PSize | PMinSize | PMaxSize | PResizeInc | PAspect),
PBaseSize = (1 << 8), PWinGravity = (1 << 9) }
• enum XWMHintsFlags {
InputHint = (1 << 0), StateHint = (1 << 1), IconPixmapHint = (1 << 2),
IconWindowHint = (1 << 3),
IconPositionHint = (1 << 4), IconMaskHint = (1 << 5), WindowGroupHint
= (1 << 6), AllHints = (InputHint | StateHint | IconPixmapHint | IconWin-
dowHint | IconPositionHint | IconMaskHint | WindowGroupHint) }
• enum XInitialState {
DontCareState = 0, NormalState = 1, ZoomState = 2, IconicState = 3,
InactiveState = 4 }
• enum XRequest {
X_CreateWindow = 1, X_ChangeWindowAttributes = 2, X_
GetWindowAttributes = 3, X_DestroyWindow = 4,
X_DestroySubwindows = 5, X_ChangeSaveSet = 6, X_ReparentWindow =
7, X_MapWindow = 8,
X_MapSubwindows = 9, X_UnmapWindow = 10, X_UnmapSubwindows =
11, X_ConfigureWindow = 12,
X_CirculateWindow = 13, X_GetGeometry = 14, X_QueryTree = 15, X_
InternAtom = 16,
X_GetAtomName = 17, X_ChangeProperty = 18, X_DeleteProperty = 19,
X_GetProperty = 20,
X_ListProperties = 21, X_SetSelectionOwner = 22, X_GetSelectionOwner =
23, X_ConvertSelection = 24,
X_SendEvent = 25, X_GrabPointer = 26, X_UngrabPointer = 27, X_
GrabButton = 28,

```

X_UngrabButton = 29, **X_ChangeActivePointerGrab** = 30, **X_GrabKeyboard** = 31, **X_UngrabKeyboard** = 32,
X_GrabKey = 33, **X_UngrabKey** = 34, **X_AllowEvents** = 35, **X_GrabServer** = 36,
X_UngrabServer = 37, **X_QueryPointer** = 38, **X_GetMotionEvents** = 39, **X_TranslateCoords** = 40,
X_WarpPointer = 41, **X_SetInputFocus** = 42, **X_GetInputFocus** = 43, **X_QueryKeymap** = 44,
X_OpenFont = 45, **X_CloseFont** = 46, **X_QueryFont** = 47, **X_QueryTextExtents** = 48,
X_ListFonts = 49, **X_ListFontsWithInfo** = 50, **X_SetFontPath** = 51, **X_GetFontPath** = 52,
X_CreatePixmap = 53, **X_FreePixmap** = 54, **X_CreateGC** = 55, **X_ChangeGC** = 56,
X_CopyGC = 57, **X_SetDashes** = 58, **X_SetClipRectangles** = 59, **X_FreeGC** = 60,
X_ClearArea = 61, **X_CopyArea** = 62, **X_CopyPlane** = 63, **X_PolyPoint** = 64,
X_PolyLine = 65, **X_PolySegment** = 66, **X_PolyRectangle** = 67, **X_PolyArc** = 68,
X_FillPoly = 69, **X_PolyFillRectangle** = 70, **X_PolyFillArc** = 71, **X_PutImage** = 72,
X_GetImage = 73, **X_PolyText8** = 74, **X_PolyText16** = 75, **X_ImageText8** = 76,
X_ImageText16 = 77, **X_CreateColormap** = 78, **X_FreeColormap** = 79, **X_CopyColormapAndFree** = 80,
X_InstallColormap = 81, **X_UninstallColormap** = 82, **X_ListInstalledColormaps** = 83, **X_AllocColor** = 84,
X_AllocNamedColor = 85, **X_AllocColorCells** = 86, **X_AllocColorPlanes** = 87, **X_FreeColors** = 88,
X_StoreColors = 89, **X_StoreNamedColor** = 90, **X_QueryColors** = 91, **X_LookupColor** = 92,
X_CreateCursor = 93, **X_CreateGlyphCursor** = 94, **X_FreeCursor** = 95, **X_RecolorCursor** = 96,
X_QueryBestSize = 97, **X_QueryExtension** = 98, **X_ListExtensions** = 99, **X_ChangeKeyboardMapping** = 100,
X_GetKeyboardMapping = 101, **X_ChangeKeyboardControl** = 102, **X_GetKeyboardControl** = 103, **X_Bell** = 104,
X_ChangePointerControl = 105, **X_GetPointerControl** = 106, **X_SetScreenSaver** = 107, **X_GetScreenSaver** = 108,

X_ChangeHosts = 109, **X_ListHosts** = 110, **X_SetAccessControl** = 111, **X_SetCloseDownMode** = 112,
X_KillClient = 113, **X_RotateProperties** = 114, **X_ForceScreenSaver** = 115,
X_SetPointerMapping = 116,
X_GetPointerMapping = 117, **X_SetModifierMapping** = 118, **X_GetModifierMapping** = 119, **X_NoOperation** = 127 }
 • enum **XIMProperties** {
 XIMPreeditArea = 0x0001, **XIMPreeditCallbacks** = 0x0002, **XIMPreeditPosition** = 0x0004, **XIMPreeditNothing** = 0x0008,
 XIMPreeditNone = 0x0010, **XIMStatusArea** = 0x0100, **XIMStatusCallbacks** = 0x0200, **XIMStatusNothing** = 0x0400,
 XIMStatusNone = 0x0800 }
 • enum **WindowType** { **Client** = 1, **Whole** = 2, **Both** = 3 }
 • enum **XEmbedMessage** {
 EmbeddedNotify = 0, **WindowActivate** = 1, **WindowDeactivate** = 2, **RequestFocus** = 3,
 FocusIn = 4, **FocusOut** = 5, **FocusNext** = 6, **FocusPrev** = 7,
 ModalityOn = 10, **ModalityOff** = 11, **RegisterAccelerator** = 12, **UnregisterAccelerator** = 13,
 ActivateAccelerator = 14 }
 • enum **ImageFormat** { **XPixmap** = 1, **ZPixmap** }

Functions

- internal delegate int **XErrorHandler** (IntPtr DisplayHandle, ref XErrorEvent error_event)

4.16.1 Enumeration Type Documentation

4.16.1.1 enum OpenTK::Platform::X11::XKey

Defines LATIN-1 and miscellaneous keys.

4.17 Package OpenTK.Properties

Classes

- class [Resources](#)
A strongly-typed resource class, for looking up localized strings, etc.

Chapter 5

Class Documentation

5.1 OpenTK.Audio.AudioCapture Class Reference

Provides methods to instantiate, use and destroy an audio device for recording. Static methods are provided to list available devices known by the driver.

Public Member Functions

- [AudioCapture](#) ()
Opens the default device for audio recording. Implicitly set parameters are: 22050Hz, 16Bit Mono, 4096 samples ringbuffer.
- [AudioCapture](#) (string deviceName, int frequency, [ALFormat](#) sampleFormat, int bufferSize)
Opens a device for audio recording.
- void [CheckErrors](#) ()
Checks for ALC error conditions.
- void [Start](#) ()
Start recording samples. The number of available samples can be obtained through the [AvailableSamples](#) property. The data can be queried with any [ReadSamples\(IntPtr, int\)](#) method.
- void [Stop](#) ()
Stop recording samples. This will not clear previously recorded samples.
- void [ReadSamples](#) (IntPtr buffer, int sampleCount)

Fills the specified buffer with samples from the internal capture ring-buffer. This method does not block: it is an error to specify a sampleCount larger than AvailableSamples.

- void [ReadSamples< TBuffer >](#) (TBuffer[] buffer, int sampleCount)
Fills the specified buffer with samples from the internal capture ring-buffer. This method does not block: it is an error to specify a sampleCount larger than AvailableSamples.
- void [Dispose](#) ()
Closes the device and disposes the instance.

Properties

- string [CurrentDevice](#) [get]
The name of the device associated with this instance.
- static IList< string > [AvailableDevices](#) [get]
Returns a list of strings containing all known recording devices.
- static string [DefaultDevice](#) [get]
Returns the name of the device that will be used as recording default.
- [AlcError](#) [CurrentError](#) [get]
Returns the ALC error code for this device.
- int [AvailableSamples](#) [get]
Returns the number of available samples for capture.
- [ALFormat](#) [SampleFormat](#) [get, set]
Gets the OpenTK.Audio.ALFormat for this instance.
- int [SampleFrequency](#) [get, set]
Gets the sampling rate for this instance.
- bool [IsRunning](#) [get]
Gets a value indicating whether this instance is currently capturing samples.

5.1.1 Detailed Description

Provides methods to instantiate, use and destroy an audio device for recording. Static methods are provided to list available devices known by the driver.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 OpenTK.Audio.AudioCapture.AudioCapture ()

Opens the default device for audio recording. Implicitly set parameters are: 22050Hz, 16Bit Mono, 4096 samples ringbuffer.

5.1.2.2 OpenTK.Audio.AudioCapture.AudioCapture (string *deviceName*, int *frequency*, ALFormat *sampleFormat*, int *bufferSize*)

Opens a device for audio recording.

Parameters

deviceName The device name.

frequency The frequency that the data should be captured at.

sampleFormat The requested capture buffer format.

bufferSize The size of OpenAL's capture internal ring-buffer. This value expects number of samples, not bytes.

5.1.3 Member Function Documentation

5.1.3.1 void OpenTK.Audio.AudioCapture.CheckErrors ()

Checks for ALC error conditions.

Exceptions

OutOfMemoryException Raised when an out of memory error is detected.

AudioValueException Raised when an invalid value is detected.

AudioDeviceException Raised when an invalid device is detected.

AudioContextException Raised when an invalid context is detected.

5.1.3.2 void OpenTK.Audio.AudioCapture.Dispose ()

Closes the device and disposes the instance.

5.1.3.3 void OpenTK.Audio.AudioCapture.ReadSamples (IntPtr *buffer*, int *sampleCount*)

Fills the specified buffer with samples from the internal capture ring-buffer. This method does not block: it is an error to specify a sampleCount larger than AvailableSamples.

Parameters

buffer A pointer to a previously initialized and pinned array.

sampleCount The number of samples to be written to the buffer.

5.1.3.4 void OpenTK.Audio.AudioCapture.ReadSamples< TBuffer > (TBuffer[] *buffer*, int *sampleCount*)

Fills the specified buffer with samples from the internal capture ring-buffer. This method does not block: it is an error to specify a sampleCount larger than AvailableSamples.

Parameters

buffer The buffer to fill.

sampleCount The number of samples to be written to the buffer.

Exceptions

System.ArgumentNullException Raised when buffer is null.

System.ArgumentOutOfRangeException Raised when sampleCount is larger than the buffer.

Type Constraints

TBuffer : struct

5.1.3.5 void OpenTK.Audio.AudioCapture.Start ()

Start recording samples. The number of available samples can be obtained through the [AvailableSamples](#) property. The data can be queried with any [ReadSamples\(IntPtr, int\)](#) method.

5.1.3.6 void OpenTK.Audio.AudioCapture.Stop ()

Stop recording samples. This will not clear previously recorded samples.

5.1.4 Property Documentation

5.1.4.1 IList<string> OpenTK.Audio.AudioCapture.AvailableDevices [static, get]

Returns a list of strings containing all known recording devices.

5.1.4.2 int OpenTK.Audio.AudioCapture.AvailableSamples [get]

Returns the number of available samples for capture.

5.1.4.3 string OpenTK.Audio.AudioCapture.CurrentDevice [get]

The name of the device associated with this instance.

5.1.4.4 AlcError OpenTK.Audio.AudioCapture.CurrentError [get]

Returns the ALC error code for this device.

5.1.4.5 string OpenTK.Audio.AudioCapture.DefaultDevice [static, get]

Returns the name of the device that will be used as recording default.

5.1.4.6 bool OpenTK.Audio.AudioCapture.IsRunning [get]

Gets a value indicating whether this instance is currently capturing samples.

5.1.4.7 ALFormat OpenTK.Audio.AudioCapture.SampleFormat [get, set]

Gets the OpenTK.Audio.ALFormat for this instance.

5.1.4.8 int OpenTK.Audio.AudioCapture.SampleFrequency [get, set]

Gets the sampling rate for this instance.

5.2 OpenTK.Audio.AudioContext Class Reference

Provides methods to instantiate, use and destroy an audio context for playback. Static methods are provided to list available devices known by the driver.

Public Types

- enum [MaxAuxiliarySends](#) {
 [UseDriverDefault](#) = 0, [One](#) = 1, [Two](#) = 2, [Three](#) = 3,
 [Four](#) = 4 }

May be passed at context construction time to indicate the number of desired auxiliary effect slot sends per source.

Public Member Functions

- [AudioContext](#) ()
Constructs a new [AudioContext](#), using the default audio device.
- [AudioContext](#) (string device)
Constructs a new [AudioContext](#) instance.
- [AudioContext](#) (string device, int freq)
Constructs a new [AudioContext](#), using the specified audio device and device parameters.
- [AudioContext](#) (string device, int freq, int refresh)
Constructs a new [AudioContext](#), using the specified audio device and device parameters.
- [AudioContext](#) (string device, int freq, int refresh, bool sync)
Constructs a new [AudioContext](#), using the specified audio device and device parameters.
- [AudioContext](#) (string device, int freq, int refresh, bool sync, bool enableEfx, [MaxAuxiliarySends](#) efxMaxAuxSends)
Creates the audio context using the specified device and device parameters.
- void [CheckErrors](#) ()
Checks for ALC error conditions.
- void [MakeCurrent](#) ()
Makes the [AudioContext](#) current in the calling thread.
- void [Process](#) ()
Processes queued audio events.
- void [Suspend](#) ()
Suspends processing of audio events.

- bool [SupportsExtension](#) (string extension)
Checks whether the specified [OpenAL](#) extension is supported.
- void [Dispose](#) ()
Disposes of the [AudioContext](#), cleaning up all resources consumed by it.
- override int [GetHashCode](#) ()
Calculates the hash code for this instance.
- override bool [Equals](#) (object obj)
Compares this instance with another.
- override string [ToString](#) ()
Returns a System.String that describes this instance.

Properties

- [AlcError CurrentError](#) [get]
Returns the ALC error code for this instance.
- bool [IsProcessing](#) [get, set]
Gets a System.Boolean indicating whether the [AudioContext](#) is currently processing audio events.
- bool [IsSynchronized](#) [get, set]
Gets a System.Boolean indicating whether the [AudioContext](#) is synchronized.
- string [CurrentDevice](#) [get]
Gets a System.String with the name of the device used in this context.
- static [AudioContext CurrentContext](#) [get]
Gets the [OpenTK.Audio.AudioContext](#) which is current in the application.
- static IList< string > [AvailableDevices](#) [get]
Returns a list of strings containing all known playback devices.
- static string [DefaultDevice](#) [get]
Returns the name of the device that will be used as playback default.

5.2.1 Detailed Description

Provides methods to instantiate, use and destroy an audio context for playback. Static methods are provided to list available devices known by the driver.

5.2.2 Member Enumeration Documentation

5.2.2.1 `enum OpenTK::Audio::AudioContext::MaxAuxiliarySends`

May be passed at context construction time to indicate the number of desired auxiliary effect slot sends per source.

Enumerator:

UseDriverDefault Will chose a reliably working parameter.

One One send per source.

Two Two sends per source.

Three Three sends per source.

Four Four sends per source.

5.2.3 Constructor & Destructor Documentation

5.2.3.1 `OpenTK.Audio.AudioContext.AudioContext ()`

Constructs a new [AudioContext](#), using the default audio device.

5.2.3.2 `OpenTK.Audio.AudioContext.AudioContext (string device)`

Constructs a new [AudioContext](#) instance.

Parameters

device The device name that will host this instance.

5.2.3.3 `OpenTK.Audio.AudioContext.AudioContext (string device, int freq)`

Constructs a new [AudioContext](#), using the specified audio device and device parameters.

Parameters

device The name of the audio device to use.

freq Frequency for mixing output buffer, in units of Hz. Pass 0 for driver default.

Use [AudioContext.AvailableDevices](#) to obtain a list of all available audio devices. devices.

5.2.3.4 OpenTK.Audio.AudioContext.AudioContext (string device, int freq, int refresh)

Constructs a new [AudioContext](#), using the specified audio device and device parameters.

Parameters

device The name of the audio device to use.

freq Frequency for mixing output buffer, in units of Hz. Pass 0 for driver default.

refresh Refresh intervals, in units of Hz. Pass 0 for driver default.

Use [AudioContext.AvailableDevices](#) to obtain a list of all available audio devices. devices.

5.2.3.5 OpenTK.Audio.AudioContext.AudioContext (string device, int freq, int refresh, bool sync)

Constructs a new [AudioContext](#), using the specified audio device and device parameters.

Parameters

device The name of the audio device to use.

freq Frequency for mixing output buffer, in units of Hz. Pass 0 for driver default.

refresh Refresh intervals, in units of Hz. Pass 0 for driver default.

sync Flag, indicating a synchronous context.

Use [AudioContext.AvailableDevices](#) to obtain a list of all available audio devices. devices.

5.2.3.6 OpenTK.Audio.AudioContext.AudioContext (string device, int freq, int refresh, bool sync, bool enableEfx)

Creates the audio context using the specified device and device parameters.

Parameters

device The device descriptor obtained through [AudioContext.AvailableDevices](#).

freq Frequency for mixing output buffer, in units of Hz. Pass 0 for driver default.

refresh Refresh intervals, in units of Hz. Pass 0 for driver default.

sync Flag, indicating a synchronous context.

enableEfx Indicates whether the EFX extension should be initialized, if present.

Exceptions

ArgumentNullException Occurs when the device string is invalid.

ArgumentOutOfRangeException Occurs when a specified parameter is invalid.

AudioDeviceException Occurs when the specified device is not available, or is in use by another program.

AudioContextException Occurs when an audio context could not be created with the specified parameters.

NotSupportedException Occurs when an [AudioContext](#) already exists.

For maximum compatibility, you are strongly recommended to use the default constructor.

Multiple AudioContexts are not supported at this point.

The number of auxilliary EFX sends depends on the audio hardware and drivers. Most Realtek devices, as well as the Creative SB Live!, support 1 auxilliary send. Creative's Audigy and X-Fi series support 4 sends. Values higher than supported will be clamped by the driver.

5.2.3.7 OpenTK.Audio.AudioContext.AudioContext (string device, int freq, int refresh, bool sync, bool enableEfx, MaxAuxiliarySends efxMaxAuxSends)

Creates the audio context using the specified device and device parameters.

Parameters

device The device descriptor obtained through [AudioContext.AvailableDevices](#).

freq Frequency for mixing output buffer, in units of Hz. Pass 0 for driver default.

refresh Refresh intervals, in units of Hz. Pass 0 for driver default.

sync Flag, indicating a synchronous context.

enableEfx Indicates whether the EFX extension should be initialized, if present.

efxMaxAuxSends Requires EFX enabled. The number of desired Auxiliary Sends per source.

Exceptions

ArgumentNullException Occurs when the device string is invalid.

ArgumentOutOfRangeException Occurs when a specified parameter is invalid.

AudioDeviceException Occurs when the specified device is not available, or is in use by another program.

AudioContextException Occurs when an audio context could not be created with the specified parameters.

NotSupportedException Occurs when an [AudioContext](#) already exists.

For maximum compatibility, you are strongly recommended to use the default constructor.

Multiple AudioContexts are not supported at this point.

The number of auxilliary EFX sends depends on the audio hardware and drivers. Most Realtek devices, as well as the Creative SB Live!, support 1 auxilliary send. Creative's Audigy and X-Fi series support 4 sends. Values higher than supported will be clamped by the driver.

5.2.4 Member Function Documentation

5.2.4.1 void OpenTK.Audio.AudioContext.CheckErrors ()

Checks for ALC error conditions.

Exceptions

OutOfMemoryException Raised when an out of memory error is detected.

AudioValueException Raised when an invalid value is detected.

AudioDeviceException Raised when an invalid device is detected.

AudioContextException Raised when an invalid context is detected.

5.2.4.2 void OpenTK.Audio.AudioContext.Dispose ()

Disposes of the [AudioContext](#), cleaning up all resources consumed by it.

5.2.4.3 override bool OpenTK.Audio.AudioContext.Equals (object obj)

Compares this instance with another.

Parameters

obj The instance to compare to.

Returns

True, if obj refers to this instance; false otherwise.

5.2.4.4 `override int OpenTK.Audio.AudioContext.GetHashCode ()`

Calculates the hash code for this instance.

Returns

5.2.4.5 `void OpenTK.Audio.AudioContext.MakeCurrent ()`

Makes the [AudioContext](#) current in the calling thread.

Exceptions

ObjectDisposedException Occurs if this function is called after the [AudioContext](#) has been disposed.

AudioContextException Occurs when the [AudioContext](#) could not be made current.

Only one [AudioContext](#) can be current in the application at any time, **regardless of the number of threads**.

5.2.4.6 `void OpenTK.Audio.AudioContext.Process ()`

Processes queued audio events.

If [AudioContext.IsSynchronized](#) is true, this function will resume the internal audio processing thread. If [AudioContext.IsSynchronized](#) is false, you will need to call this function multiple times per second to process audio events.

In some implementations this function may have no effect.

Exceptions

ObjectDisposedException Occurs when this function is called after the [AudioContext](#) had been disposed.

See also

[Suspend](#), [IsProcessing](#), [IsSynchronized](#)

5.2.4.7 `bool OpenTK.Audio.AudioContext.SupportsExtension (string extension)`

Checks whether the specified [OpenAL](#) extension is supported.

Parameters

extension The name of the extension to check (e.g. "ALC_EXT_EFX").

Returns

true if the extension is supported; false otherwise.

5.2.4.8 void OpenTK.Audio.AudioContext.Suspend ()

Suspends processing of audio events.

To avoid audio artifacts when calling this function, set audio gain to zero before suspending an [AudioContext](#).

In some implementations, it can be faster to suspend processing before changing [AudioContext](#) state.

In some implementations this function may have no effect.

Exceptions

ObjectDisposedException Occurs when this function is called after the [AudioContext](#) had been disposed.

See also

[Process](#), [IsProcessing](#), [IsSynchronized](#)

5.2.4.9 override string OpenTK.Audio.AudioContext.ToString ()

Returns a System.String that describes this instance.

Returns

A System.String that describes this instance.

5.2.5 Property Documentation**5.2.5.1 IList<string> OpenTK.Audio.AudioContext.AvailableDevices [static, get]**

Returns a list of strings containing all known playback devices.

5.2.5.2 **AudioContext** `OpenTK.Audio.AudioContext.CurrentContext` `[static, get]`

Gets the [OpenTK.Audio.AudioContext](#) which is current in the application.

Only one [AudioContext](#) can be current in the application at any time, **regardless of the number of threads**.

5.2.5.3 **string** `OpenTK.Audio.AudioContext.CurrentDevice` `[get]`

Gets a `System.String` with the name of the device used in this context.

5.2.5.4 **AlcError** `OpenTK.Audio.AudioContext.CurrentError` `[get]`

Returns the ALC error code for this instance.

5.2.5.5 **string** `OpenTK.Audio.AudioContext.DefaultDevice` `[static, get]`

Returns the name of the device that will be used as playback default.

5.2.5.6 **bool** `OpenTK.Audio.AudioContext.IsProcessing` `[get, set]`

Gets a `System.Boolean` indicating whether the [AudioContext](#) is currently processing audio events.

See also

[Process](#), [Suspend](#)

5.2.5.7 **bool** `OpenTK.Audio.AudioContext.IsSynchronized` `[get, set]`

Gets a `System.Boolean` indicating whether the [AudioContext](#) is synchronized.

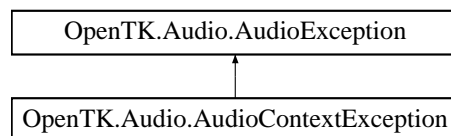
See also

[Process](#)

5.3 **OpenTK.Audio.AudioContextException** Class Reference

Represents exceptions related to an [OpenTK.Audio.AudioContext](#).

Inheritance diagram for OpenTK.Audio.AudioContextException:



Public Member Functions

- [AudioContextException](#) ()
Constructs a new [AudioContextException](#).
- [AudioContextException](#) (string message)
Constructs a new [AudioContextException](#) with the specified error message.

5.3.1 Detailed Description

Represents exceptions related to an [OpenTK.Audio.AudioContext](#).

5.3.2 Constructor & Destructor Documentation

5.3.2.1 OpenTK.Audio.AudioContextException.AudioContextException ()

Constructs a new [AudioContextException](#).

5.3.2.2 OpenTK.Audio.AudioContextException.AudioContextException (string message)

Constructs a new [AudioContextException](#) with the specified error message.

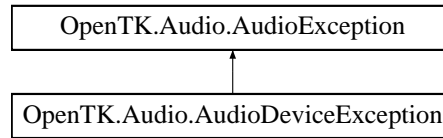
Parameters

message The error message of the [AudioContextException](#).

5.4 OpenTK.Audio.AudioDeviceException Class Reference

Represents exceptions related to an [OpenTK.Audio](#) device.

Inheritance diagram for OpenTK.Audio.AudioDeviceException:



Public Member Functions

- [AudioDeviceException](#) ()
Constructs a new [AudioDeviceException](#).
- [AudioDeviceException](#) (string message)
Constructs a new [AudioDeviceException](#) with the specified error message.

5.4.1 Detailed Description

Represents exceptions related to an [OpenTK.Audio](#) device.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 OpenTK.Audio.AudioDeviceException.AudioDeviceException ()

Constructs a new [AudioDeviceException](#).

5.4.2.2 OpenTK.Audio.AudioDeviceException.AudioDeviceException (string message)

Constructs a new [AudioDeviceException](#) with the specified error message.

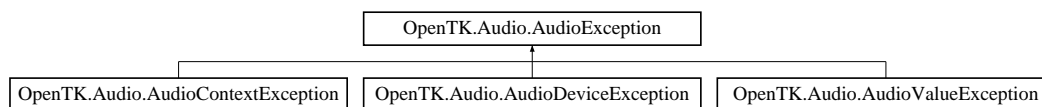
Parameters

message The error message of the [AudioDeviceException](#).

5.5 OpenTK.Audio.AudioException Class Reference

Represents exceptions related to the [OpenTK.Audio](#) subsystem.

Inheritance diagram for OpenTK.Audio.AudioException:



Public Member Functions

- [AudioException](#) ()
Constructs a new [AudioException](#).
- [AudioException](#) (string message)
Constructs a new [AudioException](#) with the specified error message.

5.5.1 Detailed Description

Represents exceptions related to the [OpenTK.Audio](#) subsystem.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 OpenTK.Audio.AudioException.AudioException ()

Constructs a new [AudioException](#).

5.5.2.2 OpenTK.Audio.AudioException.AudioException (string message)

Constructs a new [AudioException](#) with the specified error message.

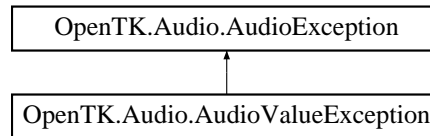
Parameters

message The error message of the [AudioException](#).

5.6 OpenTK.Audio.AudioValueException Class Reference

Represents exceptions related to invalid values.

Inheritance diagram for OpenTK.Audio.AudioValueException:



Public Member Functions

- [AudioValueException](#) ()
Constructs a new instance.
- [AudioValueException](#) (string message)
Constructs a new instance with the specified error message.

5.6.1 Detailed Description

Represents exceptions related to invalid values.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 `OpenTK.Audio.AudioValueException.AudioValueException ()`

Constructs a new instance.

5.6.2.2 `OpenTK.Audio.AudioValueException.AudioValueException (string message)`

Constructs a new instance with the specified error message.

Parameters

message The error message of the [AudioContextException](#).

5.7 OpenTK.Audio.OpenAL.EffectsExtension Class Reference

Provides access to the [OpenAL](#) effects extension.

Public Member Functions

- void **BindEffect** (uint eid, **EfxEffectType** type)
(Helper) Selects the Effect type used by this Effect handle.
- void **BindEffect** (int eid, **EfxEffectType** type)
(Helper) Selects the Effect type used by this Effect handle.
- void **BindFilterToSource** (uint source, uint filter)
(Helper) reroutes the output of a Source through a Filter.
- void **BindFilterToSource** (int source, int filter)
(Helper) reroutes the output of a Source through a Filter.
- void **BindEffectToAuxiliarySlot** (uint auxiliaryeffectslot, uint effect)
(Helper) Attaches an Effect to an Auxiliary Effect Slot.
- void **BindEffectToAuxiliarySlot** (int auxiliaryeffectslot, int effect)
(Helper) Attaches an Effect to an Auxiliary Effect Slot.
- void **BindSourceToAuxiliarySlot** (uint source, uint slot, int slotnumber, uint filter)
(Helper) Reroutes a Source's output into an Auxiliary Effect Slot.
- void **BindSourceToAuxiliarySlot** (int source, int slot, int slotnumber, int filter)
(Helper) Reroutes a Source's output into an Auxiliary Effect Slot.
- void **GenEffects** (int n, out uint effects)
The GenEffects function is used to create one or more Effect objects. An Effect object stores an effect type and a set of parameter values to control that Effect. In order to use an Effect it must be attached to an Auxiliary Effect Slot object.
- void **GenEffects** (int n, out int effects)
The GenEffects function is used to create one or more Effect objects. An Effect object stores an effect type and a set of parameter values to control that Effect. In order to use an Effect it must be attached to an Auxiliary Effect Slot object.
- int[] **GenEffects** (int n)
Generates one or more effect objects.
- int **GenEffect** ()
Generates a single effect object.

- void [GenEffect](#) (out uint effect)
Generates a single effect object.
- void [DeleteEffects](#) (int n, ref uint effects)
The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.
- void [DeleteEffects](#) (int n, ref int effects)
The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.
- void [DeleteEffects](#) (int[] effects)
The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.
- void [DeleteEffects](#) (uint[] effects)
The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.
- void [DeleteEffect](#) (int effect)
This function deletes one Effect only.
- void [DeleteEffect](#) (ref uint effect)
This function deletes one Effect only.
- bool [IsEffect](#) (uint eid)
The IsEffect function is used to determine if an object identifier is a valid Effect object.
- bool [IsEffect](#) (int eid)
The IsEffect function is used to determine if an object identifier is a valid Effect object.
- void [Effect](#) (uint eid, [EfxEffecti](#) param, int value)
This function is used to set integer properties on Effect objects.
- void [Effect](#) (int eid, [EfxEffecti](#) param, int value)
This function is used to set integer properties on Effect objects.
- void [Effect](#) (uint eid, [EfxEffectf](#) param, float value)
This function is used to set floating-point properties on Effect objects.
- void [Effect](#) (int eid, [EfxEffectf](#) param, float value)
This function is used to set floating-point properties on Effect objects.

- void [Effect](#) (uint eid, [EfxEffect3f](#) param, ref [Vector3](#) values)
This function is used to set 3 floating-point properties on Effect objects.
- void [Effect](#) (int eid, [EfxEffect3f](#) param, ref [Vector3](#) values)
This function is used to set 3 floating-point properties on Effect objects.
- void [GetEffect](#) (uint eid, [EfxEffecti](#) pname, out int value)
This function is used to retrieve integer properties from Effect objects.
- void [GetEffect](#) (int eid, [EfxEffecti](#) pname, out int value)
This function is used to retrieve integer properties from Effect objects.
- void [GetEffect](#) (uint eid, [EfxEffectf](#) pname, out float value)
This function is used to retrieve floating-point properties from Effect objects.
- void [GetEffect](#) (int eid, [EfxEffectf](#) pname, out float value)
This function is used to retrieve floating-point properties from Effect objects.
- void [GetEffect](#) (uint eid, [EfxEffect3f](#) param, out [Vector3](#) values)
This function is used to retrieve 3 floating-point properties from Effect objects.
- void [GetEffect](#) (int eid, [EfxEffect3f](#) param, out [Vector3](#) values)
This function is used to retrieve 3 floating-point properties from Effect objects.
- void [GenFilters](#) (int n, out uint filters)
The GenFilters function is used to create one or more Filter objects. A Filter object stores a filter type and a set of parameter values to control that Filter. Filter objects can be attached to Sources as Direct Filters or Auxiliary Send Filters.
- void [GenFilters](#) (int n, out int filters)
The GenFilters function is used to create one or more Filter objects. A Filter object stores a filter type and a set of parameter values to control that Filter. Filter objects can be attached to Sources as Direct Filters or Auxiliary Send Filters.
- int[] [GenFilters](#) (int n)
The GenFilters function is used to create one or more Filter objects. A Filter object stores a filter type and a set of parameter values to control that Filter. Filter objects can be attached to Sources as Direct Filters or Auxiliary Send Filters.
- int [GenFilter](#) ()
This function generates only one Filter.
- unsafe void [GenFilter](#) (out uint filter)

This function generates only one Filter.

- void [DeleteFilters](#) (int n, ref uint filters)

The DeleteFilters function is used to delete and free resources for Filter objects previously created with GenFilters.

- void [DeleteFilters](#) (int n, ref int filters)

The DeleteFilters function is used to delete and free resources for Filter objects previously created with GenFilters.

- void [DeleteFilters](#) (uint[] filters)

This function deletes one Filter only.

- void [DeleteFilters](#) (int[] filters)

This function deletes one Filter only.

- void [DeleteFilter](#) (int filter)

This function deletes one Filter only.

- void [DeleteFilter](#) (ref uint filter)

This function deletes one Filter only.

- bool [IsFilter](#) (uint fid)

The IsFilter function is used to determine if an object identifier is a valid Filter object.

- bool [IsFilter](#) (int fid)

The IsFilter function is used to determine if an object identifier is a valid Filter object.

- void [Filter](#) (uint fid, [EfxFilteri](#) param, int value)

This function is used to set integer properties on Filter objects.

- void [Filter](#) (int fid, [EfxFilteri](#) param, int value)

This function is used to set integer properties on Filter objects.

- void [Filter](#) (uint fid, [EfxFilterf](#) param, float value)

This function is used to set floating-point properties on Filter objects.

- void [Filter](#) (int fid, [EfxFilterf](#) param, float value)

This function is used to set floating-point properties on Filter objects.

- void [GetFilter](#) (uint fid, [EfxFilteri](#) pname, out int value)

This function is used to retrieve integer properties from Filter objects.

- void [GetFilter](#) (int fid, [EfxFilteri](#) pname, out int value)
This function is used to retrieve integer properties from Filter objects.
- void [GetFilter](#) (uint fid, [EfxFilterf](#) pname, out float value)
This function is used to retrieve floating-point properties from Filter objects.
- void [GetFilter](#) (int fid, [EfxFilterf](#) pname, out float value)
This function is used to retrieve floating-point properties from Filter objects.
- void [GenAuxiliaryEffectSlots](#) (int n, out uint slots)
The GenAuxiliaryEffectSlots function is used to create one or more Auxiliary Effect Slots. The number of slots that can be created will be dependant upon the Open AL device used.
- void [GenAuxiliaryEffectSlots](#) (int n, out int slots)
The GenAuxiliaryEffectSlots function is used to create one or more Auxiliary Effect Slots. The number of slots that can be created will be dependant upon the Open AL device used.
- int[] [GenAuxiliaryEffectSlots](#) (int n)
The GenAuxiliaryEffectSlots function is used to create one or more Auxiliary Effect Slots. The number of slots that can be created will be dependant upon the Open AL device used.
- int [GenAuxiliaryEffectSlot](#) ()
This function generates only one Auxiliary Effect Slot.
- void [GenAuxiliaryEffectSlot](#) (out uint slot)
This function generates only one Auxiliary Effect Slot.
- void [DeleteAuxiliaryEffectSlots](#) (int n, ref uint slots)
The DeleteAuxiliaryEffectSlots function is used to delete and free resources for Auxiliary Effect Slots previously created with GenAuxiliaryEffectSlots.
- void [DeleteAuxiliaryEffectSlots](#) (int n, ref int slots)
The DeleteAuxiliaryEffectSlots function is used to delete and free resources for Auxiliary Effect Slots previously created with GenAuxiliaryEffectSlots.
- void [DeleteAuxiliaryEffectSlots](#) (int[] slots)
The DeleteAuxiliaryEffectSlots function is used to delete and free resources for Auxiliary Effect Slots previously created with GenAuxiliaryEffectSlots.
- void [DeleteAuxiliaryEffectSlots](#) (uint[] slots)
This function deletes one AuxiliaryEffectSlot only.

- void [DeleteAuxiliaryEffectSlot](#) (int slot)
This function deletes one AuxiliaryEffectSlot only.
- void [DeleteAuxiliaryEffectSlot](#) (ref uint slot)
This function deletes one AuxiliaryEffectSlot only.
- bool [IsAuxiliaryEffectSlot](#) (uint slot)
The IsAuxiliaryEffectSlot function is used to determine if an object identifier is a valid Auxiliary Effect Slot object.
- bool [IsAuxiliaryEffectSlot](#) (int slot)
The IsAuxiliaryEffectSlot function is used to determine if an object identifier is a valid Auxiliary Effect Slot object.
- void [AuxiliaryEffectSlot](#) (uint asid, [EfxAuxiliaryi](#) param, int value)
This function is used to set integer properties on Auxiliary Effect Slot objects.
- void [AuxiliaryEffectSlot](#) (int asid, [EfxAuxiliaryi](#) param, int value)
This function is used to set integer properties on Auxiliary Effect Slot objects.
- void [AuxiliaryEffectSlot](#) (uint asid, [EfxAuxiliaryf](#) param, float value)
This function is used to set floating-point properties on Auxiliary Effect Slot objects.
- void [AuxiliaryEffectSlot](#) (int asid, [EfxAuxiliaryf](#) param, float value)
This function is used to set floating-point properties on Auxiliary Effect Slot objects.
- void [GetAuxiliaryEffectSlot](#) (uint asid, [EfxAuxiliaryi](#) pname, out int value)
This function is used to retrieve integer properties on Auxiliary Effect Slot objects.
- void [GetAuxiliaryEffectSlot](#) (int asid, [EfxAuxiliaryi](#) pname, out int value)
This function is used to retrieve integer properties on Auxiliary Effect Slot objects.
- void [GetAuxiliaryEffectSlot](#) (uint asid, [EfxAuxiliaryf](#) pname, out float value)
This function is used to retrieve floating properties on Auxiliary Effect Slot objects.
- void [GetAuxiliaryEffectSlot](#) (int asid, [EfxAuxiliaryf](#) pname, out float value)
This function is used to retrieve floating properties on Auxiliary Effect Slot objects.
- [EffectsExtension](#) ()
Constructs a new [EffectsExtension](#) instance.

Static Public Member Functions

- static void **GetEaxFromEfxEax** (ref EaxReverb input, out EfxEaxReverb output)

Properties

- bool **IsInitialized** [get]
Returns True if the EFX Extension has been found and could be initialized.

5.7.1 Detailed Description

Provides access to the [OpenAL](#) effects extension.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 OpenTK.Audio.OpenAL.EffectsExtension.EffectsExtension ()

Constructs a new [EffectsExtension](#) instance.

5.7.3 Member Function Documentation

5.7.3.1 void OpenTK.Audio.OpenAL.EffectsExtension.AuxiliaryEffectSlot (uint *asid*, EfxAuxiliary *param*, int *value*)

This function is used to set integer properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.
param Auxiliary Effect Slot property to set.
value Integer value.

5.7.3.2 void OpenTK.Audio.OpenAL.EffectsExtension.AuxiliaryEffectSlot (int *asid*, EfxAuxiliary *param*, int *value*)

This function is used to set integer properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.

param Auxiliary Effect Slot property to set.

value Integer value.

5.7.3.3 void OpenTK.Audio.OpenAL.EffectsExtension.AuxiliaryEffectSlot (uint *asid*, EfxAuxiliaryf *param*, float *value*)

This function is used to set floating-point properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.

param Auxiliary Effect Slot property to set.

value Floating-point value.

5.7.3.4 void OpenTK.Audio.OpenAL.EffectsExtension.AuxiliaryEffectSlot (int *asid*, EfxAuxiliaryf *param*, float *value*)

This function is used to set floating-point properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.

param Auxiliary Effect Slot property to set.

value Floating-point value.

5.7.3.5 void OpenTK.Audio.OpenAL.EffectsExtension.BindEffect (uint *eid*, EfxEffectType *type*)

(Helper) Selects the Effect type used by this Effect handle.

Parameters

eid Effect id returned from a successful call to GenEffects.

type Effect type.

5.7.3.6 void OpenTK.Audio.OpenAL.EffectsExtension.BindEffect (int *eid*, EfxEffectType *type*)

(Helper) Selects the Effect type used by this Effect handle.

Parameters

eid Effect id returned from a successful call to GenEffects.

type Effect type.

5.7.3.7 void OpenTK.Audio.OpenAL.EffectsExtension.BindEffectToAuxiliarySlot (uint *auxiliaryeffects*slot, uint *effect*)

(Helper) Attaches an Effect to an Auxiliary Effect Slot.

Parameters

*auxiliaryeffects*slot The slot handle to attach the Effect to.

effect The Effect handle that is being attached.

5.7.3.8 void OpenTK.Audio.OpenAL.EffectsExtension.BindEffectToAuxiliarySlot (int *auxiliaryeffects*slot, int *effect*)

(Helper) Attaches an Effect to an Auxiliary Effect Slot.

Parameters

*auxiliaryeffects*slot The slot handle to attach the Effect to.

effect The Effect handle that is being attached.

5.7.3.9 void OpenTK.Audio.OpenAL.EffectsExtension.BindFilterToSource (int *source*, int *filter*)

(Helper) reroutes the output of a Source through a Filter.

Parameters

source A valid Source handle.

filter A valid Filter handle.

5.7.3.10 void OpenTK.Audio.OpenAL.EffectsExtension.BindFilterToSource (uint *source*, uint *filter*)

(Helper) reroutes the output of a Source through a Filter.

Parameters

source A valid Source handle.

filter A valid Filter handle.

5.7.3.11 void OpenTK.Audio.OpenAL.EffectsExtension.BindSourceToAuxiliarySlot (int *source*, int *slot*, int *slotnumber*, int *filter*)

(Helper) Reroutes a Source's output into an Auxiliary Effect Slot.

Parameters

source The Source handle who's output is forwarded.

slot The Auxiliary Effect Slot handle that receives input from the Source.

slotnumber Every Source has only a limited number of slots it can feed buffer to. The number must stay below AlcContextAttributes.EfxMaxAuxiliarySends

filter Filter handle to be attached between Source output and Auxiliary Slot input. Use 0 or EfxFilterType.FilterNull for no filter.

5.7.3.12 void OpenTK.Audio.OpenAL.EffectsExtension.BindSourceToAuxiliarySlot (uint *source*, uint *slot*, int *slotnumber*, uint *filter*)

(Helper) Reroutes a Source's output into an Auxiliary Effect Slot.

Parameters

source The Source handle who's output is forwarded.

slot The Auxiliary Effect Slot handle that receives input from the Source.

slotnumber Every Source has only a limited number of slots it can feed buffer to. The number must stay below AlcContextAttributes.EfxMaxAuxiliarySends

filter Filter handle to be attached between Source output and Auxiliary Slot input. Use 0 or EfxFilterType.FilterNull for no filter.

5.7.3.13 void
OpenTK.Audio.OpenAL.EffectsExtension.DeleteAuxiliaryEffectSlot (
int *slot*)

This function deletes one AuxiliaryEffectSlot only.

Parameters

slot Pointer to an auxiliary effect slot name/handle identifying the Auxiliary Effect Slot Object to be deleted.

5.7.3.14 void
OpenTK.Audio.OpenAL.EffectsExtension.DeleteAuxiliaryEffectSlot (
ref uint *slot*)

This function deletes one AuxiliaryEffectSlot only.

Parameters

slot Pointer to an auxiliary effect slot name/handle identifying the Auxiliary Effect Slot Object to be deleted.

5.7.3.15 void
OpenTK.Audio.OpenAL.EffectsExtension.DeleteAuxiliaryEffectSlots (
int *n*, ref uint *slots*)

The DeleteAuxiliaryEffectSlots function is used to delete and free resources for Auxiliary Effect Slots previously created with GenAuxiliaryEffectSlots.

Parameters

n Number of Auxiliary Effect Slots to be deleted.

slots Pointer to n Effect Slot object identifiers.

5.7.3.16 void
OpenTK.Audio.OpenAL.EffectsExtension.DeleteAuxiliaryEffectSlots (
int *n*, ref int *slots*)

The DeleteAuxiliaryEffectSlots function is used to delete and free resources for Auxiliary Effect Slots previously created with GenAuxiliaryEffectSlots.

Parameters

n Number of Auxiliary Effect Slots to be deleted.

slots Pointer to n Effect Slot object identifiers.

5.7.3.17 void
OpenTK.Audio.OpenAL.EffectsExtension.DeleteAuxiliaryEffectSlots (
int[] *slots*)

The DeleteAuxiliaryEffectSlots function is used to delete and free resources for Auxiliary Effect Slots previously created with GenAuxiliaryEffectSlots.

Parameters

slots Pointer to n Effect Slot object identifiers.

5.7.3.18 void
OpenTK.Audio.OpenAL.EffectsExtension.DeleteAuxiliaryEffectSlots (
uint[] *slots*)

This function deletes one AuxiliaryEffectSlot only.

Parameters

slots Pointer to an auxiliary effect slot name/handle identifying the Auxiliary Effect Slot Object to be deleted.

5.7.3.19 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteEffect (int
***effect*)**

This function deletes one Effect only.

Parameters

effect Pointer to an effect name/handle identifying the Effect Object to be deleted.

5.7.3.20 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteEffect (ref uint
***effect*)**

This function deletes one Effect only.

Parameters

effect Pointer to an effect name/handle identifying the Effect Object to be deleted.

5.7.3.21 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteEffects (int[] *effects*)

The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.

Parameters

effects Pointer to n Effect object identifiers.

5.7.3.22 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteEffects (uint[] *effects*)

The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.

Parameters

effects Pointer to n Effect object identifiers.

5.7.3.23 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteEffects (int *n*, ref uint *effects*)

The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.

Parameters

n Number of Effects to be deleted.

effects Pointer to n Effect object identifiers.

5.7.3.24 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteEffects (int *n*, ref int *effects*)

The DeleteEffects function is used to delete and free resources for Effect objects previously created with GenEffects.

Parameters

n Number of Effects to be deleted.

effects Pointer to n Effect object identifiers.

5.7.3.25 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteFilter (int *filter*)

This function deletes one Filter only.

Parameters

filter Pointer to an filter name/handle identifying the Filter Object to be deleted.

5.7.3.26 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteFilter (ref uint *filter*)

This function deletes one Filter only.

Parameters

filter Pointer to an filter name/handle identifying the Filter Object to be deleted.

5.7.3.27 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteFilters (int *n*, ref uint *filters*)

The DeleteFilters function is used to delete and free resources for Filter objects previously created with GenFilters.

Parameters

n Number of Filters to be deleted.

filters Pointer to n Filter object identifiers.

5.7.3.28 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteFilters (uint[] *filters*)

This function deletes one Filter only.

Parameters

filters Pointer to an filter name/handle identifying the Filter Object to be deleted.

5.7.3.29 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteFilters (int *n*, ref int *filters*)

The DeleteFilters function is used to delete and free resources for Filter objects previously created with GenFilters.

Parameters

- n* Number of Filters to be deleted.
filters Pointer to n Filter object identifiers.

5.7.3.30 void OpenTK.Audio.OpenAL.EffectsExtension.DeleteFilters (int[] *filters*)

This function deletes one Filter only.

Parameters

- filters* Pointer to an filter name/handle identifying the Filter Object to be deleted.

5.7.3.31 void OpenTK.Audio.OpenAL.EffectsExtension.Effect (uint *eid*, EfxEffecti *param*, int *value*)

This function is used to set integer properties on Effect objects.

Parameters

- eid* Effect object identifier.
param Effect property to set.
value Integer value.

5.7.3.32 void OpenTK.Audio.OpenAL.EffectsExtension.Effect (int *eid*, EfxEffecti *param*, int *value*)

This function is used to set integer properties on Effect objects.

Parameters

- eid* Effect object identifier.
param Effect property to set.
value Integer value.

5.7.3.33 void OpenTK.Audio.OpenAL.EffectsExtension.Effect (uint *eid*, EfxEffectf *param*, float *value*)

This function is used to set floating-point properties on Effect objects.

Parameters

eid Effect object identifier.
param Effect property to set.
value Floating-point value.

5.7.3.34 void OpenTK.Audio.OpenAL.EffectsExtension.Effect (int *eid*, EfxEffectf *param*, float *value*)

This function is used to set floating-point properties on Effect objects.

Parameters

eid Effect object identifier.
param Effect property to set.
value Floating-point value.

5.7.3.35 void OpenTK.Audio.OpenAL.EffectsExtension.Effect (uint *eid*, EfxEffect3f *param*, ref Vector3 *values*)

This function is used to set 3 floating-point properties on Effect objects.

Parameters

eid Effect object identifier.
param Effect property to set.
values Pointer to Math.Vector3.

5.7.3.36 void OpenTK.Audio.OpenAL.EffectsExtension.Effect (int *eid*, EfxEffect3f *param*, ref Vector3 *values*)

This function is used to set 3 floating-point properties on Effect objects.

Parameters

eid Effect object identifier.
param Effect property to set.
values Pointer to Math.Vector3.

**5.7.3.37 void OpenTK.Audio.OpenAL.EffectsExtension.Filter (uint *fid*,
EfxFilteri *param*, int *value*)**

This function is used to set integer properties on Filter objects.

Parameters

fid Filter object identifier.
param Effect property to set.
value Integer value.

**5.7.3.38 void OpenTK.Audio.OpenAL.EffectsExtension.Filter (int *fid*,
EfxFilteri *param*, int *value*)**

This function is used to set integer properties on Filter objects.

Parameters

fid Filter object identifier.
param Effect property to set.
value Integer value.

**5.7.3.39 void OpenTK.Audio.OpenAL.EffectsExtension.Filter (uint *fid*,
EfxFilterf *param*, float *value*)**

This function is used to set floating-point properties on Filter objects.

Parameters

fid Filter object identifier.
param Effect property to set.
value Floating-point value.

**5.7.3.40 void OpenTK.Audio.OpenAL.EffectsExtension.Filter (int *fid*,
EfxFilterf *param*, float *value*)**

This function is used to set floating-point properties on Filter objects.

Parameters

fid Filter object identifier.
param Effect property to set.
value Floating-point value.

5.7.3.41 `int OpenTK.Audio.OpenAL.EffectsExtension.GenAuxiliaryEffectSlot ()`

This function generates only one Auxiliary Effect Slot.

Returns

Storage Int32 for the new auxiliary effect slot name/handle.

5.7.3.42 `void OpenTK.Audio.OpenAL.EffectsExtension.GenAuxiliaryEffectSlot (out uint slot)`

This function generates only one Auxiliary Effect Slot.

Returns

Storage UInt32 for the new auxiliary effect slot name/handle.

5.7.3.43 `int [] OpenTK.Audio.OpenAL.EffectsExtension.GenAuxiliaryEffectSlots (int n)`

The GenAuxiliaryEffectSlots function is used to create one or more Auxiliary Effect Slots. The number of slots that can be created will be dependant upon the Open AL device used.

An application should check the [OpenAL](#) error state after making this call to determine if the Effect Slot was successfully created. If the function call fails then none of the requested Effect Slots are created. A good strategy for creating any [OpenAL](#) object is to use a for-loop and generate one object each loop iteration and then check for an error condition. If an error is set then the loop can be broken and the application can determine if sufficient resources are available.

Parameters

n Number of Auxiliary Effect Slots to be created.

Returns

Pointer addressing sufficient memory to store n Effect Slot object identifiers.

5.7.3.44 void

OpenTK.Audio.OpenAL.EffectsExtension.GenAuxiliaryEffectSlots (
int *n*, out int *slots*)

The GenAuxiliaryEffectSlots function is used to create one or more Auxiliary Effect Slots. The number of slots that can be created will be dependant upon the Open AL device used.

An application should check the [OpenAL](#) error state after making this call to determine if the Effect Slot was successfully created. If the function call fails then none of the requested Effect Slots are created. A good strategy for creating any [OpenAL](#) object is to use a for-loop and generate one object each loop iteration and then check for an error condition. If an error is set then the loop can be broken and the application can determine if sufficient resources are available.

Parameters

n Number of Auxiliary Effect Slots to be created.

slots Pointer addressing sufficient memory to store n Effect Slot object identifiers.

5.7.3.45 void

OpenTK.Audio.OpenAL.EffectsExtension.GenAuxiliaryEffectSlots (
int *n*, out uint *slots*)

The GenAuxiliaryEffectSlots function is used to create one or more Auxiliary Effect Slots. The number of slots that can be created will be dependant upon the Open AL device used.

An application should check the [OpenAL](#) error state after making this call to determine if the Effect Slot was successfully created. If the function call fails then none of the requested Effect Slots are created. A good strategy for creating any [OpenAL](#) object is to use a for-loop and generate one object each loop iteration and then check for an error condition. If an error is set then the loop can be broken and the application can determine if sufficient resources are available.

Parameters

n Number of Auxiliary Effect Slots to be created.

slots Pointer addressing sufficient memory to store n Effect Slot object identifiers.

5.7.3.46 int OpenTK.Audio.OpenAL.EffectsExtension.GenEffect ()

Generates a single effect object.

Returns

A handle to the generated effect object.

The GenEffects function is used to create one or more Effect objects. An Effect object stores an effect type and a set of parameter values to control that Effect. In order to use an Effect it must be attached to an Auxiliary Effect Slot object.

After creation an Effect has no type (EfxEffectType.Null), so before it can be used to store a set of parameters, the application must specify what type of effect should be stored in the object, using [Effect\(\)](#) with EfxEffecti.

5.7.3.47 void OpenTK.Audio.OpenAL.EffectsExtension.GenEffect (out uint *effect*)

Generates a single effect object.

Parameters

effect A handle to the generated effect object.

5.7.3.48 void OpenTK.Audio.OpenAL.EffectsExtension.GenEffects (int *n*, out uint *effects*)

The GenEffects function is used to create one or more Effect objects. An Effect object stores an effect type and a set of parameter values to control that Effect. In order to use an Effect it must be attached to an Auxiliary Effect Slot object.

After creation an Effect has no type (EfxEffectType.Null), so before it can be used to store a set of parameters, the application must specify what type of effect should be stored in the object, using [Effect\(\)](#) with EfxEffecti.

Parameters

n Number of Effects to be created.

effects Pointer addressing sufficient memory to store n Effect object identifiers.

5.7.3.49 void OpenTK.Audio.OpenAL.EffectsExtension.GenEffects (int *n*, out int *effects*)

The GenEffects function is used to create one or more Effect objects. An Effect object stores an effect type and a set of parameter values to control that Effect. In order to use an Effect it must be attached to an Auxiliary Effect Slot object.

After creation an Effect has no type (EfxEffectType.Null), so before it can be used to store a set of parameters, the application must specify what type of effect should be stored in the object, using [Effect\(\)](#) with EfxEffecti.

Parameters

- n* Number of Effects to be created.
- effects* Pointer addressing sufficient memory to store n Effect object identifiers.

5.7.3.50 int [] OpenTK.Audio.OpenAL.EffectsExtension.GenEffects (int *n*)

Generates one or more effect objects.

Parameters

- n* Number of Effect object identifiers to generate.

The GenEffects function is used to create one or more Effect objects. An Effect object stores an effect type and a set of parameter values to control that Effect. In order to use an Effect it must be attached to an Auxiliary Effect Slot object.

After creation an Effect has no type (EfxEffectType.Null), so before it can be used to store a set of parameters, the application must specify what type of effect should be stored in the object, using [Effect\(\)](#) with EfxEffecti.

5.7.3.51 int OpenTK.Audio.OpenAL.EffectsExtension.GenFilter ()

This function generates only one Filter.

Returns

- Storage Int32 for the new filter name/handle.

5.7.3.52 unsafe void OpenTK.Audio.OpenAL.EffectsExtension.GenFilter (out uint *filter*)

This function generates only one Filter.

Parameters

- filter* Storage UInt32 for the new filter name/handle.

5.7.3.53 void OpenTK.Audio.OpenAL.EffectsExtension.GenFilters (int *n*, out uint *filters*)

The GenFilters function is used to create one or more Filter objects. A Filter object stores a filter type and a set of parameter values to control that Filter. Filter objects can be attached to Sources as Direct Filters or Auxiliary Send Filters.

After creation a Filter has no type (`EfxFilterType.Null`), so before it can be used to store a set of parameters, the application must specify what type of filter should be stored in the object, using [Filter\(\)](#) with `EfxFilteri`.

Parameters

n Number of Filters to be created.

filters Pointer addressing sufficient memory to store *n* Filter object identifiers.

5.7.3.54 void OpenTK.Audio.OpenAL.EffectsExtension.GenFilters (int *n*, out int *filters*)

The `GenFilters` function is used to create one or more Filter objects. A Filter object stores a filter type and a set of parameter values to control that Filter. Filter objects can be attached to Sources as Direct Filters or Auxiliary Send Filters.

After creation a Filter has no type (`EfxFilterType.Null`), so before it can be used to store a set of parameters, the application must specify what type of filter should be stored in the object, using [Filter\(\)](#) with `EfxFilteri`.

Parameters

n Number of Filters to be created.

filters Pointer addressing sufficient memory to store *n* Filter object identifiers.

5.7.3.55 int [] OpenTK.Audio.OpenAL.EffectsExtension.GenFilters (int *n*)

The `GenFilters` function is used to create one or more Filter objects. A Filter object stores a filter type and a set of parameter values to control that Filter. Filter objects can be attached to Sources as Direct Filters or Auxiliary Send Filters.

After creation a Filter has no type (`EfxFilterType.Null`), so before it can be used to store a set of parameters, the application must specify what type of filter should be stored in the object, using [Filter\(\)](#) with `EfxFilteri`.

Parameters

n Number of Filters to be created.

Returns

Pointer addressing sufficient memory to store *n* Filter object identifiers.

5.7.3.56 void OpenTK.Audio.OpenAL.EffectsExtension.GetAuxiliaryEffectSlot (uint *asid*, EfxAuxiliaryf *pname*, out float *value*)

This function is used to retrieve floating properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.
pname Auxiliary Effect Slot property to retrieve.
value Address where floating-point value will be stored.

5.7.3.57 void OpenTK.Audio.OpenAL.EffectsExtension.GetAuxiliaryEffectSlot (int *asid*, EfxAuxiliaryf *pname*, out float *value*)

This function is used to retrieve floating properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.
pname Auxiliary Effect Slot property to retrieve.
value Address where floating-point value will be stored.

5.7.3.58 void OpenTK.Audio.OpenAL.EffectsExtension.GetAuxiliaryEffectSlot (uint *asid*, EfxAuxiliaryi *pname*, out int *value*)

This function is used to retrieve integer properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.
pname Auxiliary Effect Slot property to retrieve.
value Address where integer value will be stored.

5.7.3.59 void OpenTK.Audio.OpenAL.EffectsExtension.GetAuxiliaryEffectSlot (int *asid*, EfxAuxiliaryi *pname*, out int *value*)

This function is used to retrieve integer properties on Auxiliary Effect Slot objects.

Parameters

asid Auxiliary Effect Slot object identifier.
pname Auxiliary Effect Slot property to retrieve.
value Address where integer value will be stored.

5.7.3.60 void OpenTK.Audio.OpenAL.EffectsExtension.GetEffect (uint *eid*, EfxEffecti *pname*, out int *value*)

This function is used to retrieve integer properties from Effect objects.

Parameters

eid Effect object identifier.
pname Effect property to retrieve.
value Address where integer value will be stored.

5.7.3.61 void OpenTK.Audio.OpenAL.EffectsExtension.GetEffect (int *eid*, EfxEffecti *pname*, out int *value*)

This function is used to retrieve integer properties from Effect objects.

Parameters

eid Effect object identifier.
pname Effect property to retrieve.
value Address where integer value will be stored.

5.7.3.62 void OpenTK.Audio.OpenAL.EffectsExtension.GetEffect (int *eid*, EfxEffectf *pname*, out float *value*)

This function is used to retrieve floating-point properties from Effect objects.

Parameters

eid Effect object identifier.
pname Effect property to retrieve.
value Address where floating-point value will be stored.

5.7.3.63 void OpenTK.Audio.OpenAL.EffectsExtension.GetEffect (uint *eid*, EfxEffectf *pname*, out float *value*)

This function is used to retrieve floating-point properties from Effect objects.

Parameters

eid Effect object identifier.
pname Effect property to retrieve.
value Address where floating-point value will be stored.

5.7.3.64 void OpenTK.Audio.OpenAL.EffectsExtension.GetEffect (int *eid*, EfxEffect3f *param*, out Vector3 *values*)

This function is used to retrieve 3 floating-point properties from Effect objects.

Parameters

eid Effect object identifier.
param Effect property to retrieve.
values A Math.Vector3 to hold the values.

5.7.3.65 void OpenTK.Audio.OpenAL.EffectsExtension.GetEffect (uint *eid*, EfxEffect3f *param*, out Vector3 *values*)

This function is used to retrieve 3 floating-point properties from Effect objects.

Parameters

eid Effect object identifier.
param Effect property to retrieve.
values A Math.Vector3 to hold the values.

5.7.3.66 void OpenTK.Audio.OpenAL.EffectsExtension.GetFilter (int *fid*, EfxFilteri *pname*, out int *value*)

This function is used to retrieve integer properties from Filter objects.

Parameters

fid Filter object identifier.
pname Effect property to retrieve.
value Address where integer value will be stored.

5.7.3.67 void OpenTK.Audio.OpenAL.EffectsExtension.GetFilter (uint *fid*, EfxFilteri *pname*, out int *value*)

This function is used to retrieve integer properties from Filter objects.

Parameters

fid Filter object identifier.
pname Effect property to retrieve.
value Address where integer value will be stored.

5.7.3.68 void OpenTK.Audio.OpenAL.EffectsExtension.GetFilter (int *fid*, EfxFilterf *pname*, out float *value*)

This function is used to retrieve floating-point properties from Filter objects.

Parameters

fid Filter object identifier.

pname Effect property to retrieve.

value Address where floating-point value will be stored.

5.7.3.69 void OpenTK.Audio.OpenAL.EffectsExtension.GetFilter (uint *fid*, EfxFilterf *pname*, out float *value*)

This function is used to retrieve floating-point properties from Filter objects.

Parameters

fid Filter object identifier.

pname Effect property to retrieve.

value Address where floating-point value will be stored.

5.7.3.70 bool OpenTK.Audio.OpenAL.EffectsExtension.IsAuxiliaryEffectSlot (uint *slot*)

The IsAuxiliaryEffectSlot function is used to determine if an object identifier is a valid Auxiliary Effect Slot object.

Parameters

slot Effect Slot object identifier to validate.

Returns

True if the identifier is a valid Auxiliary Effect Slot, False otherwise.

5.7.3.71 bool OpenTK.Audio.OpenAL.EffectsExtension.IsAuxiliaryEffectSlot (int *slot*)

The IsAuxiliaryEffectSlot function is used to determine if an object identifier is a valid Auxiliary Effect Slot object.

Parameters

slot Effect Slot object identifier to validate.

Returns

True if the identifier is a valid Auxiliary Effect Slot, False otherwise.

5.7.3.72 bool OpenTK.Audio.OpenAL.EffectsExtension.IsEffect (uint *eid*)

The IsEffect function is used to determine if an object identifier is a valid Effect object.

Parameters

eid Effect identifier to validate.

Returns

True if the identifier is a valid Effect, False otherwise.

5.7.3.73 bool OpenTK.Audio.OpenAL.EffectsExtension.IsEffect (int *eid*)

The IsEffect function is used to determine if an object identifier is a valid Effect object.

Parameters

eid Effect identifier to validate.

Returns

True if the identifier is a valid Effect, False otherwise.

5.7.3.74 bool OpenTK.Audio.OpenAL.EffectsExtension.IsFilter (int *fid*)

The IsFilter function is used to determine if an object identifier is a valid Filter object.

Parameters

fid Effect identifier to validate.

Returns

True if the identifier is a valid Filter, False otherwise.

5.7.3.75 bool OpenTK.Audio.OpenAL.EffectsExtension.IsFilter (uint *fid*)

The IsFilter function is used to determine if an object identifier is a valid Filter object.

Parameters

fid Effect identifier to validate.

Returns

True if the identifier is a valid Filter, False otherwise.

5.7.4 Property Documentation

5.7.4.1 bool OpenTK.Audio.OpenAL.EffectsExtension.IsInitialized [get]

Returns True if the EFX Extension has been found and could be initialized.

5.8 OpenTK.Audio.OpenAL.XRamExtension Class Reference

The X-Ram Extension is provided on the top-end Sound Blaster X-Fi solutions (Sound Blaster X-Fi Fatal1ty, Sound Blaster X-Fi Elite Pro, or later). These products feature 64MB of X-Ram that can only be used for audio purposes, which can be controlled by this Extension. /summary>

Public Types

- enum [XRamStorage](#) { [Automatic](#) = 0, [Hardware](#) = 1, [Accessible](#) = 2 }

This enum is used to abstract the need of using `AL.GetEnumValue()` with the Extension. The values do NOT correspond to `AL_STORAGE_` tokens!*

Public Member Functions

- [XRamExtension](#) ()

Constructs a new [XRamExtension](#) instance.

- bool [SetBufferMode](#) (int n, ref uint buffer, [XRamStorage](#) mode)

This function is used to set the storage Mode of an array of [OpenAL](#) Buffers.

- bool [SetBufferMode](#) (int n, ref int buffer, [XRamStorage](#) mode)
This function is used to set the storage Mode of an array of [OpenAL](#) Buffers.
- [XRamStorage GetBufferMode](#) (ref uint buffer)
This function is used to retrieve the storage Mode of a single [OpenAL](#) Buffer.
- [XRamStorage GetBufferMode](#) (ref int buffer)
This function is used to retrieve the storage Mode of a single [OpenAL](#) Buffer.

Properties

- bool [IsInitialized](#) [get]
Returns True if the X-Ram Extension has been found and could be initialized.
- int [GetRamSize](#) [get]
Query total amount of X-RAM in bytes.
- int [GetRamFree](#) [get]
Query free X-RAM available in bytes.

5.8.1 Detailed Description

The X-Ram Extension is provided on the top-end Sound Blaster X-Fi solutions (Sound Blaster X-Fi Fatal1ty, Sound Blaster X-Fi Elite Pro, or later). These products feature 64MB of X-Ram that can only be used for audio purposes, which can be controlled by this Extension. /summary>

5.8.2 Member Enumeration Documentation

5.8.2.1 enum OpenTK::Audio::OpenAL::XRamExtension::XRamStorage

This enum is used to abstract the need of using `AL.GetEnumValue()` with the Extension. The values do NOT correspond to `AL_STORAGE_*` tokens!

Enumerator:

Automatic Put an Open AL Buffer into X-RAM if memory is available, otherwise use host RAM. This is the default mode.

Hardware Force an Open AL Buffer into X-RAM, good for non-streaming buffers.

Accessible Force an Open AL Buffer into 'accessible' (currently host) RAM, good for streaming buffers.

5.8.3 Constructor & Destructor Documentation

5.8.3.1 OpenTK.Audio.OpenAL.XRamExtension.XRamExtension ()

Constructs a new [XRamExtension](#) instance.

5.8.4 Member Function Documentation

5.8.4.1 XRamStorage

**OpenTK.Audio.OpenAL.XRamExtension.GetBufferMode (ref uint
buffer)**

This function is used to retrieve the storage Mode of a single [OpenAL](#) Buffer.

Parameters

buffer The handle of an [OpenAL](#) Buffer.

Returns

The current Mode of the Buffer.

5.8.4.2 XRamStorage

**OpenTK.Audio.OpenAL.XRamExtension.GetBufferMode (ref int
buffer)**

This function is used to retrieve the storage Mode of a single [OpenAL](#) Buffer.

Parameters

buffer The handle of an [OpenAL](#) Buffer.

Returns

The current Mode of the Buffer.

5.8.4.3 bool OpenTK.Audio.OpenAL.XRamExtension.SetBufferMode (int *n*, ref uint *buffer*, XRamStorage *mode*)

This function is used to set the storage Mode of an array of [OpenAL](#) Buffers.

Parameters

- n* The number of [OpenAL](#) Buffers pointed to by buffer.
- buffer* An array of [OpenAL](#) Buffer handles.
- mode* The storage mode that should be used for all the given buffers. Should be the value of one of the following enum names: `XRamStorage Automatic`, `XRamStorage Hardware`, `XRamStorage Accessible`

Returns

True if all the Buffers were successfully set to the requested storage mode, False otherwise.

5.8.4.4 bool OpenTK.Audio.OpenAL.XRamExtension.SetBufferMode (int *n*, ref int *buffer*, XRamStorage *mode*)

This function is used to set the storage Mode of an array of [OpenAL](#) Buffers.

Parameters

- n* The number of [OpenAL](#) Buffers pointed to by buffer.
- buffer* An array of [OpenAL](#) Buffer handles.
- mode* The storage mode that should be used for all the given buffers. Should be the value of one of the following enum names: `XRamStorage Automatic`, `XRamStorage Hardware`, `XRamStorage Accessible`

Returns

True if all the Buffers were successfully set to the requested storage mode, False otherwise.

5.8.5 Property Documentation**5.8.5.1 int OpenTK.Audio.OpenAL.XRamExtension.GetRamFree [get]**

Query free X-RAM available in bytes.

5.8.5.2 int OpenTK.Audio.OpenAL.XRamExtension.GetRamSize [get]

Query total amount of X-RAM in bytes.

5.8.5.3 bool OpenTK.Audio.OpenAL.XRamExtension.IsInitialized [get]

Returns True if the X-Ram Extension has been found and could be initialized.

5.9 OpenTK.AutoGeneratedAttribute Class Reference

Indicates that this function is generated automatically by a tool.

Public Member Functions

- [AutoGeneratedAttribute](#) ()
Constructs a new [AutoGeneratedAttribute](#) instance.

Public Attributes

- string [Category](#)
Specifies the category of this OpenGL function.
- string [Version](#)
Specifies the version of this OpenGL function.
- string [EntryPoint](#)
Specifies the entry point of the OpenGL function.

5.9.1 Detailed Description

Indicates that this function is generated automatically by a tool.

5.9.2 Constructor & Destructor Documentation

5.9.2.1 OpenTK.AutoGeneratedAttribute.AutoGeneratedAttribute ()

Constructs a new [AutoGeneratedAttribute](#) instance.

5.9.3 Member Data Documentation

5.9.3.1 string OpenTK.AutoGeneratedAttribute.Category

Specifies the category of this OpenGL function.

5.9.3.2 string OpenTK.AutoGeneratedAttribute.EntryPoint

Specifies the entry point of the OpenGL function.

5.9.3.3 string OpenTK.AutoGeneratedAttribute.Version

Specifies the version of this OpenGL function.

5.10 OpenTK.BezierCurve Struct Reference

Represents a bezier curve with as many points as you want.

Public Member Functions

- [BezierCurve](#) (IEnumerable< [Vector2](#) > points)
Constructs a new [BezierCurve](#).
- [BezierCurve](#) (params [Vector2](#)[] points)
Constructs a new [BezierCurve](#).
- [BezierCurve](#) (float parallel, params [Vector2](#)[] points)
Constructs a new [BezierCurve](#).
- [BezierCurve](#) (float parallel, IEnumerable< [Vector2](#) > points)
Constructs a new [BezierCurve](#).
- [Vector2 CalculatePoint](#) (float t)
Calculates the point with the specified t.
- float [CalculateLength](#) (float precision)
Calculates the length of this bezier curve.

Static Public Member Functions

- static float [CalculateLength](#) (IList< [Vector2](#) > points, float precision)
Calculates the length of the specified bezier curve.
- static float [CalculateLength](#) (IList< [Vector2](#) > points, float precision, float parallel)

Calculates the length of the specified bezier curve.

- static [Vector2 CalculatePoint](#) (IList< [Vector2](#) > points, float t)
Calculates the point on the given bezier curve with the specified t parameter.
- static [Vector2 CalculatePoint](#) (IList< [Vector2](#) > points, float t, float parallel)
Calculates the point on the given bezier curve with the specified t parameter.

Public Attributes

- float [Parallel](#)
The parallel value.

Properties

- IList< [Vector2](#) > [Points](#) [get]
Gets the points of this curve.

5.10.1 Detailed Description

Represents a bezier curve with as many points as you want.

5.10.2 Constructor & Destructor Documentation

5.10.2.1 [OpenTK.BezierCurve.BezierCurve](#) ([IEnumerable](#)< [Vector2](#) > *points*)

Constructs a new [BezierCurve](#).

Parameters

points The points.

5.10.2.2 [OpenTK.BezierCurve.BezierCurve](#) (params [Vector2](#)[] *points*)

Constructs a new [BezierCurve](#).

Parameters

points The points.

5.10.2.3 OpenTK.BezierCurve.BezierCurve (float *parallel*, params Vector2[] *points*)

Constructs a new [BezierCurve](#).

Parameters

parallel The parallel value.

points The points.

5.10.2.4 OpenTK.BezierCurve.BezierCurve (float *parallel*, IEnumerable< Vector2 > *points*)

Constructs a new [BezierCurve](#).

Parameters

parallel The parallel value.

points The points.

5.10.3 Member Function Documentation

5.10.3.1 float OpenTK.BezierCurve.CalculateLength (float *precision*)

Calculates the length of this bezier curve.

Parameters

precision The precision.

Returns

Length of curve.

The precision gets better as the *precision* value gets smaller.

5.10.3.2 static float OpenTK.BezierCurve.CalculateLength (IList< Vector2 > *points*, float *precision*, float *parallel*) [static]

Calculates the length of the specified bezier curve.

Parameters

points The points.

precision The precision value.

parallel The parallel value.

Returns

Length of curve.

The precision gets better as the *precision* value gets smaller.

The *parallel* parameter defines whether the curve should be calculated as a parallel curve to the original bezier curve. A value of 0.0f represents the original curve, 5.0f represents a curve that has always a distance of 5.0f to the original curve.

5.10.3.3 static float OpenTK.BezierCurve.CalculateLength (IList< Vector2 > *points*, float *precision*) [static]

Calculates the length of the specified bezier curve.

Parameters

points The points.

precision The precision value.

Returns

The precision gets better as the *precision* value gets smaller.

5.10.3.4 Vector2 OpenTK.BezierCurve.CalculatePoint (float *t*)

Calculates the point with the specified t.

Parameters

t The t value, between 0.0f and 1.0f.

Returns

Resulting point.

5.10.3.5 static Vector2 OpenTK.BezierCurve.CalculatePoint (IList< Vector2 > *points*, float *t*) [static]

Calculates the point on the given bezier curve with the specified t parameter.

Parameters

- points* The points.
t The t parameter, a value between 0.0f and 1.0f.

Returns

Resulting point.

5.10.3.6 static Vector2 OpenTK.BezierCurve.CalculatePoint (IList< Vector2 > *points*, float *t*, float *parallel*) [static]

Calculates the point on the given bezier curve with the specified t parameter.

Parameters

- points* The points.
t The t parameter, a value between 0.0f and 1.0f.
parallel The parallel value.

Returns

Resulting point.

The *parallel* parameter defines whether the curve should be calculated as a parallel curve to the original bezier curve. A value of 0.0f represents the original curve, 5.0f represents a curve that has always a distance of 5.0f to the original curve.

5.10.4 Member Data Documentation**5.10.4.1 float OpenTK.BezierCurve.Parallel**

The parallel value.

This value defines whether the curve should be calculated as a parallel curve to the original bezier curve. A value of 0.0f represents the original curve, 5.0f i.e. stands for a curve that has always a distance of 5.0f to the original curve at any point.

5.10.5 Property Documentation**5.10.5.1 IList<Vector2> OpenTK.BezierCurve.Points [get]**

Gets the points of this curve.

The first point and the last points represent the anchor points.

5.11 OpenTK.BezierCurveCubic Struct Reference

Represents a cubic bezier curve with two anchor and two control points.

Public Member Functions

- [BezierCurveCubic](#) ([Vector2](#) startAnchor, [Vector2](#) endAnchor, [Vector2](#) firstControlPoint, [Vector2](#) secondControlPoint)
Constructs a new [BezierCurveCubic](#).
- [BezierCurveCubic](#) (float parallel, [Vector2](#) startAnchor, [Vector2](#) endAnchor, [Vector2](#) firstControlPoint, [Vector2](#) secondControlPoint)
Constructs a new [BezierCurveCubic](#).
- [Vector2 CalculatePoint](#) (float t)
Calculates the point with the specified t.
- float [CalculateLength](#) (float precision)
Calculates the length of this bezier curve.

Public Attributes

- [Vector2 StartAnchor](#)
Start anchor point.
- [Vector2 EndAnchor](#)
End anchor point.
- [Vector2 FirstControlPoint](#)
First control point, controls the direction of the curve start.
- [Vector2 SecondControlPoint](#)
Second control point, controls the direction of the curve end.
- float [Parallel](#)
Gets or sets the parallel value.

5.11.1 Detailed Description

Represents a cubic bezier curve with two anchor and two control points.

5.11.2 Constructor & Destructor Documentation

5.11.2.1 OpenTK.BezierCurveCubic.BezierCurveCubic (Vector2 *startAnchor*, Vector2 *endAnchor*, Vector2 *firstControlPoint*, Vector2 *secondControlPoint*)

Constructs a new [BezierCurveCubic](#).

Parameters

startAnchor The start anchor point.

endAnchor The end anchor point.

firstControlPoint The first control point.

secondControlPoint The second control point.

5.11.2.2 OpenTK.BezierCurveCubic.BezierCurveCubic (float *parallel*, Vector2 *startAnchor*, Vector2 *endAnchor*, Vector2 *firstControlPoint*, Vector2 *secondControlPoint*)

Constructs a new [BezierCurveCubic](#).

Parameters

parallel The parallel value.

startAnchor The start anchor point.

endAnchor The end anchor point.

firstControlPoint The first control point.

secondControlPoint The second control point.

5.11.3 Member Function Documentation

5.11.3.1 float OpenTK.BezierCurveCubic.CalculateLength (float *precision*)

Calculates the length of this bezier curve.

Parameters

precision The precision.

Returns

Length of the curve.

The precision gets better when the *precision* value gets smaller.

5.11.3.2 Vector2 OpenTK.BezierCurveCubic.CalculatePoint (float *t*)

Calculates the point with the specified *t*.

Parameters

t The *t* value, between 0.0f and 1.0f.

Returns

Resulting point.

5.11.4 Member Data Documentation

5.11.4.1 Vector2 OpenTK.BezierCurveCubic.EndAnchor

End anchor point.

5.11.4.2 Vector2 OpenTK.BezierCurveCubic.FirstControlPoint

First control point, controls the direction of the curve start.

5.11.4.3 float OpenTK.BezierCurveCubic.Parallel

Gets or sets the parallel value.

This value defines whether the curve should be calculated as a parallel curve to the original bezier curve. A value of 0.0f represents the original curve, 5.0f i.e. stands for a curve that has always a distance of 5.f to the original curve at any point.

5.11.4.4 Vector2 OpenTK.BezierCurveCubic.SecondControlPoint

Second control point, controls the direction of the curve end.

5.11.4.5 Vector2 OpenTK.BezierCurveCubic.StartAnchor

Start anchor point.

5.12 OpenTK.BezierCurveQuadric Struct Reference

Represents a quadric bezier curve with two anchor and one control point.

Public Member Functions

- [BezierCurveQuadric](#) ([Vector2](#) startAnchor, [Vector2](#) endAnchor, [Vector2](#) controlPoint)
Constructs a new [BezierCurveQuadric](#).
- [BezierCurveQuadric](#) (float parallel, [Vector2](#) startAnchor, [Vector2](#) endAnchor, [Vector2](#) controlPoint)
Constructs a new [BezierCurveQuadric](#).
- [Vector2 CalculatePoint](#) (float t)
Calculates the point with the specified t.
- float [CalculateLength](#) (float precision)
Calculates the length of this bezier curve.

Public Attributes

- [Vector2 StartAnchor](#)
Start anchor point.
- [Vector2 EndAnchor](#)
End anchor point.
- [Vector2 ControlPoint](#)
Control point, controls the direction of both endings of the curve.
- float [Parallel](#)
The parallel value.

5.12.1 Detailed Description

Represents a quadric bezier curve with two anchor and one control point.

5.12.2 Constructor & Destructor Documentation

5.12.2.1 OpenTK.BezierCurveQuadric.BezierCurveQuadric ([Vector2](#) startAnchor, [Vector2](#) endAnchor, [Vector2](#) controlPoint)

Constructs a new [BezierCurveQuadric](#).

Parameters

startAnchor The start anchor.
endAnchor The end anchor.
controlPoint The control point.

5.12.2.2 OpenTK.BezierCurveQuadric.BezierCurveQuadric (float *parallel*, Vector2 *startAnchor*, Vector2 *endAnchor*, Vector2 *controlPoint*)

Constructs a new [BezierCurveQuadric](#).

Parameters

parallel The parallel value.
startAnchor The start anchor.
endAnchor The end anchor.
controlPoint The control point.

5.12.3 Member Function Documentation**5.12.3.1 float OpenTK.BezierCurveQuadric.CalculateLength (float *precision*)**

Calculates the length of this bezier curve.

Parameters

precision The precision.

Returns

Length of curve.

The precision gets better when the *precision* value gets smaller.

5.12.3.2 Vector2 OpenTK.BezierCurveQuadric.CalculatePoint (float *t*)

Calculates the point with the specified *t*.

Parameters

t The *t* value, between 0.0f and 1.0f.

Returns

Resulting point.

5.12.4 Member Data Documentation

5.12.4.1 Vector2 OpenTK.BezierCurveQuadric.ControlPoint

Control point, controls the direction of both endings of the curve.

5.12.4.2 Vector2 OpenTK.BezierCurveQuadric.EndAnchor

End anchor point.

5.12.4.3 float OpenTK.BezierCurveQuadric.Parallel

The parallel value.

This value defines whether the curve should be calculated as a parallel curve to the original bezier curve. A value of 0.0f represents the original curve, 5.0f i.e. stands for a curve that has always a distance of 5.f to the original curve at any point.

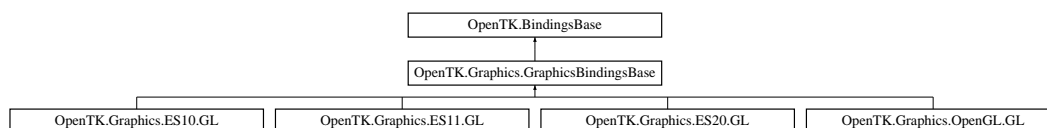
5.12.4.4 Vector2 OpenTK.BezierCurveQuadric.StartAnchor

Start anchor point.

5.13 OpenTK.BindingsBase Class Reference

Provides a common foundation for all flat API bindings and implements the extension loading interface.

Inheritance diagram for OpenTK.BindingsBase:



Public Member Functions

- [BindingsBase \(\)](#)

Constructs a new [BindingsBase](#) instance.

Protected Member Functions

- abstract IntPtr [GetAddress](#) (string funcname)

Retrieves an unmanaged function pointer to the specified function.

Protected Attributes

- readonly Type [DelegatesClass](#)

A reflection handle to the nested type that contains the function delegates.

- readonly Type [CoreClass](#)

A reflection handle to the nested type that contains core functions (i.e. not extensions).

- readonly SortedList< string, MethodInfo > [CoreFunctionMap](#) = new SortedList<string, MethodInfo>()

A mapping of core function names to MethodInfo handles.

Properties

- bool [RebuildExtensionList](#) [get, set]

Gets or sets a System.Boolean that indicates whether the list of supported extensions may have changed.

- abstract object [SyncRoot](#) [get]

Gets an object that can be used to synchronize access to the bindings implementation.

5.13.1 Detailed Description

Provides a common foundation for all flat API bindings and implements the extension loading interface.

5.13.2 Constructor & Destructor Documentation

5.13.2.1 OpenTK.BindingsBase.BindingsBase ()

Constructs a new [BindingsBase](#) instance.

5.13.3 Member Function Documentation

5.13.3.1 `abstract IntPtr OpenTK.BindingsBase.GetAddress (string funcname) [protected, pure virtual]`

Retrieves an unmanaged function pointer to the specified function.

Parameters

funcname A System.String that defines the name of the function.

Returns

A IntPtr that contains the address of funcname or IntPtr.Zero, if the function is not supported by the drivers.

Note: some drivers are known to return non-zero values for unsupported functions. Typical values include 1 and 2 - inheritors are advised to check for and ignore these values.

Implemented in [OpenTK.Graphics.GraphicsBindingsBase](#).

5.13.4 Member Data Documentation

5.13.4.1 `readonly Type OpenTK.BindingsBase.CoreClass [protected]`

A reflection handle to the nested type that contains core functions (i.e. not extensions).

5.13.4.2 `readonly SortedList<string, MethodInfo> OpenTK.BindingsBase.CoreFunctionMap = new SortedList<string, MethodInfo>() [protected]`

A mapping of core function names to MethodInfo handles.

5.13.4.3 `readonly Type OpenTK.BindingsBase.DelegatesClass [protected]`

A reflection handle to the nested type that contains the function delegates.

5.13.5 Property Documentation

5.13.5.1 `bool OpenTK.BindingsBase.RebuildExtensionList [get, set, protected]`

Gets or sets a System.Boolean that indicates whether the list of supported extensions may have changed.

5.13.5.2 abstract object `OpenTK.BindingsBase.SyncRoot` [`get`, `protected`]

Gets an object that can be used to synchronize access to the bindings implementation.

This object should be unique across bindings but consistent between bindings of the same type. For example, `ES10.GL`, `OpenGL.GL` and `CL10.CL` should all return unique objects, but all instances of `ES10.GL` should return the same object.

Reimplemented in [OpenTK.Graphics.ES10.GL](#), [OpenTK.Graphics.ES11.GL](#), [OpenTK.Graphics.ES20.GL](#), and [OpenTK.Graphics.OpenGL.GL](#).

5.14 OpenTK.Box2 Struct Reference

Defines a 2d box (rectangle).

Public Member Functions

- [Box2](#) ([Vector2](#) topLeft, [Vector2](#) bottomRight)
Constructs a new [Box2](#) with the specified dimensions.
- [Box2](#) (float left, float top, float right, float bottom)
Constructs a new [Box2](#) with the specified dimensions.
- override string [ToString](#) ()
Returns a `System.String` describing the current instance.

Static Public Member Functions

- static [Box2 FromTLRB](#) (float top, float left, float right, float bottom)
Creates a new [Box2](#) with the specified dimensions.

Public Attributes

- float [Left](#)
The left boundary of the structure.
- float [Right](#)
The right boundary of the structure.

- float [Top](#)
The top boundary of the structure.
- float [Bottom](#)
The bottom boundary of the structure.

Properties

- float [Width](#) [get]
Gets a float describing the width of the [Box2](#) structure.
- float [Height](#) [get]
Gets a float describing the height of the [Box2](#) structure.

5.14.1 Detailed Description

Defines a 2d box (rectangle).

5.14.2 Constructor & Destructor Documentation

5.14.2.1 OpenTK.Box2.Box2 ([Vector2](#) *topLeft*, [Vector2](#) *bottomRight*)

Constructs a new [Box2](#) with the specified dimensions.

Parameters

- topLeft* An [OpenTK.Vector2](#) describing the top-left corner of the [Box2](#).
bottomRight An [OpenTK.Vector2](#) describing the bottom-right corner of the [Box2](#).

5.14.2.2 OpenTK.Box2.Box2 (float *left*, float *top*, float *right*, float *bottom*)

Constructs a new [Box2](#) with the specified dimensions.

Parameters

- left* The position of the left boundary.
top The position of the top boundary.
right The position of the right boundary.
bottom The position of the bottom boundary.

5.14.3 Member Function Documentation

5.14.3.1 `static Box2 OpenTK.Box2.FromTLRB (float top, float left, float right, float bottom) [static]`

Creates a new [Box2](#) with the specified dimensions.

Parameters

- top* The position of the top boundary.
- left* The position of the left boundary.
- right* The position of the right boundary.
- bottom* The position of the bottom boundary.

Returns

A new [OpenTK.Box2](#) with the specfied dimensions.

5.14.3.2 `override string OpenTK.Box2.ToString ()`

Returns a System.String describing the current instance.

Returns

5.14.4 Member Data Documentation

5.14.4.1 `float OpenTK.Box2.Bottom`

The bottom boundary of the structure.

5.14.4.2 `float OpenTK.Box2.Left`

The left boundary of the structure.

5.14.4.3 `float OpenTK.Box2.Right`

The right boundary of the structure.

5.14.4.4 `float OpenTK.Box2.Top`

The top boundary of the structure.

5.14.5 Property Documentation

5.14.5.1 float OpenTK.Box2.Height [get]

Gets a float describing the height of the [Box2](#) structure.

5.14.5.2 float OpenTK.Box2.Width [get]

Gets a float describing the width of the [Box2](#) structure.

5.15 OpenTK.ContextExistsException Class Reference

This exception is thrown when a GraphicsContext property cannot be changed after creation.

Public Member Functions

- [ContextExistsException](#) (string message)
Constructs a new [ContextExistsException](#) instance.

Properties

- override string [Message](#) [get]
Gets a System.String explaining the cause of this exception.

5.15.1 Detailed Description

This exception is thrown when a GraphicsContext property cannot be changed after creation.

5.15.2 Constructor & Destructor Documentation

5.15.2.1 OpenTK.ContextExistsException.ContextExistsException (string message)

Constructs a new [ContextExistsException](#) instance.

Parameters

message A System.String explaining the cause of this exception.

5.15.3 Property Documentation

5.15.3.1 override string OpenTK.ContextExistsException.Message [get]

Gets a System.String explaining the cause of this exception.

5.16 OpenTK.ContextHandle Struct Reference

Represents a handle to an OpenGL or OpenAL context.

Public Member Functions

- [ContextHandle](#) (IntPtr h)
Constructs a new instance with the specified handle.
- override string [ToString](#) ()
Converts this instance to its equivalent string representation.
- override bool [Equals](#) (object obj)
Compares this instance to the specified object.
- override int [GetHashCode](#) ()
Returns the hash code for this instance.
- int [CompareTo](#) ([ContextHandle](#) other)
Compares the numerical value of this instance to the specified [ContextHandle](#) and returns a value indicating their relative order.
- bool [Equals](#) ([ContextHandle](#) other)
Compares this instance to the specified [ContextHandle](#) for equality.

Static Public Member Functions

- static operator IntPtr ([ContextHandle](#) c)
Converts the specified [ContextHandle](#) to the equivalent IntPtr.

- static `operator ContextHandle` (IntPtr p)
Converts the specified IntPtr to the equivalent `ContextHandle`.
- static bool `operator==` (`ContextHandle` left, `ContextHandle` right)
Compares two ContextHandles for equality.
- static bool `operator!=` (`ContextHandle` left, `ContextHandle` right)
Compares two ContextHandles for inequality.

Public Attributes

- IntPtr `handle`

Static Public Attributes

- static readonly `ContextHandle Zero` = new `ContextHandle`(IntPtr.Zero)
A read-only field that represents a handle that has been initialized to zero.

Properties

- IntPtr `Handle` [get]
Gets a System.IntPtr that represents the handle of this `ContextHandle`.

5.16.1 Detailed Description

Represents a handle to an OpenGL or OpenAL context.

5.16.2 Constructor & Destructor Documentation

5.16.2.1 OpenTK.ContextHandle.ContextHandle (IntPtr *h*)

Constructs a new instance with the specified handle.

Parameters

- h* A System.IntPtr containing the value for this instance.

5.16.3 Member Function Documentation

5.16.3.1 `int OpenTK.ContextHandle.CompareTo (ContextHandle other)`

Compares the numerical value of this instance to the specified [ContextHandle](#) and returns a value indicating their relative order.

Parameters

other The [ContextHandle](#) to compare to.

Returns

Less than 0, if this instance is less than other; 0 if both are equal; Greater than 0 if other is greater than this instance.

5.16.3.2 `override bool OpenTK.ContextHandle.Equals (object obj)`

Compares this instance to the specified object.

Parameters

obj The System.Object to compare to.

Returns

True if *obj* is a [ContextHandle](#) that is equal to this instance; false otherwise.

5.16.3.3 `bool OpenTK.ContextHandle.Equals (ContextHandle other)`

Compares this instance to the specified [ContextHandle](#) for equality.

Parameters

other The [ContextHandle](#) to compare to.

Returns

True if this instance is equal to other; false otherwise.

5.16.3.4 `override int OpenTK.ContextHandle.GetHashCode ()`

Returns the hash code for this instance.

Returns

A System.Int32 with the hash code of this instance.

5.16.3.5 static OpenTK.ContextHandle.operator ContextHandle (IntPtr *p*) [explicit, static]

Converts the specified IntPtr to the equivalent [ContextHandle](#).

Parameters

p The System.IntPtr to convert.

Returns

A [ContextHandle](#) equivalent to the specified IntPtr.

5.16.3.6 static OpenTK.ContextHandle.operator IntPtr (ContextHandle *c*) [explicit, static]

Converts the specified [ContextHandle](#) to the equivalent IntPtr.

Parameters

c The [ContextHandle](#) to convert.

Returns

A System.IntPtr equivalent to the specified [ContextHandle](#).

5.16.3.7 static bool OpenTK.ContextHandle.operator!= (ContextHandle *left*, ContextHandle *right*) [static]

Compares two ContextHandles for inequality.

Parameters

left The [ContextHandle](#) to compare.

right The [ContextHandle](#) to compare to.

Returns

True if left is not equal to right; false otherwise.

5.16.3.8 static bool OpenTK.ContextHandle.operator== (ContextHandle *left*, ContextHandle *right*) [static]

Compares two ContextHandles for equality.

Parameters

left The [ContextHandle](#) to compare.
right The [ContextHandle](#) to compare to.

Returns

True if left is equal to right; false otherwise.

5.16.3.9 override string OpenTK.ContextHandle.ToString ()

Converts this instance to its equivalent string representation.

Returns

A System.String that contains the string representation of this instance.

5.16.4 Member Data Documentation**5.16.4.1 readonly ContextHandle OpenTK.ContextHandle.Zero = new ContextHandle(IntPtr.Zero) [static]**

A read-only field that represents a handle that has been initialized to zero.

5.16.5 Property Documentation**5.16.5.1 IntPtr OpenTK.ContextHandle.Handle [get]**

Gets a System.IntPtr that represents the handle of this [ContextHandle](#).

5.17 OpenTK.DisplayDevice Class Reference

Defines a display device on the underlying system, and provides methods to query and change its display parameters.

Public Member Functions

- [DisplayResolution SelectResolution](#) (int width, int height, int bitsPerPixel, float refreshRate)

Selects an available resolution that matches the specified parameters.

- void [ChangeResolution](#) ([DisplayResolution](#) resolution)
Changes the resolution of the [DisplayDevice](#).
- void [ChangeResolution](#) (int width, int height, int bitsPerPixel, float refreshRate)
Changes the resolution of the [DisplayDevice](#).
- void [RestoreResolution](#) ()
Restores the original resolution of the [DisplayDevice](#).
- override string [ToString](#) ()
Returns a [System.String](#) representing this [DisplayDevice](#).

Properties

- Rectangle [Bounds](#) [get, set]
Gets the bounds of this instance in pixel coordinates..
- int [Width](#) [get]
Gets a [System.Int32](#) that contains the width of this display in pixels.
- int [Height](#) [get]
Gets a [System.Int32](#) that contains the height of this display in pixels.
- int [BitsPerPixel](#) [get, set]
Gets a [System.Int32](#) that contains number of bits per pixel of this display. Typical values include 8, 16, 24 and 32.
- float [RefreshRate](#) [get, set]
Gets a [System.Single](#) representing the vertical refresh rate of this display.
- bool [IsPrimary](#) [get, set]
Gets a [System.Boolean](#) that indicates whether this Display is the primary Display in systems with multiple Displays.
- IList< [DisplayResolution](#) > [AvailableResolutions](#) [get, set]
Gets the list of [DisplayResolution](#) objects available on this device.
- static IList< [DisplayDevice](#) > [AvailableDisplays](#) [get]
Gets the list of available [DisplayDevice](#) objects.

- static [DisplayDevice Default](#) [get]
Gets the default (primary) display of this system.

5.17.1 Detailed Description

Defines a display device on the underlying system, and provides methods to query and change its display parameters.

5.17.2 Member Function Documentation

5.17.2.1 void OpenTK.DisplayDevice.ChangeResolution ([DisplayResolution resolution](#))

Changes the resolution of the [DisplayDevice](#).

Parameters

resolution The resolution to set. [DisplayDevice.SelectResolution](#)

Exceptions

[Graphics.GraphicsModeException](#) Thrown if the requested resolution could not be set.

If the specified resolution is null, this function will restore the original [DisplayResolution](#).

5.17.2.2 void OpenTK.DisplayDevice.ChangeResolution (int *width*, int *height*, int *bitsPerPixel*, float *refreshRate*)

Changes the resolution of the [DisplayDevice](#).

Parameters

width The new width of the [DisplayDevice](#).

height The new height of the [DisplayDevice](#).

bitsPerPixel The new bits per pixel of the [DisplayDevice](#).

refreshRate The new refresh rate of the [DisplayDevice](#).

Exceptions

[Graphics.GraphicsModeException](#) Thrown if the requested resolution could not be set.

5.17.2.3 void OpenTK.DisplayDevice.RestoreResolution ()

Restores the original resolution of the [DisplayDevice](#).

Exceptions

[*Graphics.GraphicsModeException*](#) Thrown if the original resolution could not be restored.

5.17.2.4 DisplayResolution OpenTK.DisplayDevice.SelectResolution (int width, int height, int bitsPerPixel, float refreshRate)

Selects an available resolution that matches the specified parameters.

Parameters

width The width of the requested resolution in pixels.

height The height of the requested resolution in pixels.

bitsPerPixel The bits per pixel of the requested resolution.

refreshRate The refresh rate of the requested resolution in hertz.

Returns

The requested [DisplayResolution](#) or null if the parameters cannot be met.

If a matching resolution is not found, this function will retry ignoring the specified refresh rate, bits per pixel and resolution, in this order. If a matching resolution still doesn't exist, this function will return the current resolution.

A parameter set to 0 or negative numbers will not be used in the search (e.g. if refreshRate is 0, any refresh rate will be considered valid).

This function allocates memory.

5.17.2.5 override string OpenTK.DisplayDevice.ToString ()

Returns a System.String representing this [DisplayDevice](#).

Returns

A System.String representing this [DisplayDevice](#).

5.17.3 Property Documentation

5.17.3.1 **IList<DisplayDevice> OpenTK.DisplayDevice.AvailableDisplays** [static, get]

Gets the list of available [DisplayDevice](#) objects.

5.17.3.2 **IList<DisplayResolution> OpenTK.DisplayDevice.AvailableResolutions** [get, set]

Gets the list of [DisplayResolution](#) objects available on this device.

5.17.3.3 **int OpenTK.DisplayDevice.BitsPerPixel** [get, set]

Gets a System.Int32 that contains number of bits per pixel of this display. Typical values include 8, 16, 24 and 32.

5.17.3.4 **Rectangle OpenTK.DisplayDevice.Bounds** [get, set]

Gets the bounds of this instance in pixel coordinates..

5.17.3.5 **DisplayDevice OpenTK.DisplayDevice.Default** [static, get]

Gets the default (primary) display of this system.

5.17.3.6 **int OpenTK.DisplayDevice.Height** [get]

Gets a System.Int32 that contains the height of this display in pixels.

5.17.3.7 **bool OpenTK.DisplayDevice.IsPrimary** [get, set]

Gets a System.Boolean that indicates whether this Display is the primary Display in systems with multiple Displays.

5.17.3.8 **float OpenTK.DisplayDevice.RefreshRate** [get, set]

Gets a System.Single representing the vertical refresh rate of this display.

5.17.3.9 int OpenTK.DisplayDevice.Width [get]

Gets a System.Int32 that contains the width of this display in pixels.

5.18 OpenTK.DisplayResolution Class Reference

Contains information regarding a monitor's display resolution.

Public Member Functions

- override string [ToString](#) ()
Returns a System.String representing this [DisplayResolution](#).
- override bool [Equals](#) (object obj)
Determines whether the specified resolutions are equal.
- override int [GetHashCode](#) ()
Returns a unique hash representing this resolution.

Static Public Member Functions

- static bool [operator==](#) ([DisplayResolution](#) left, [DisplayResolution](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([DisplayResolution](#) left, [DisplayResolution](#) right)
Compares two instances for inequality.

Properties

- Rectangle [Bounds](#) [get]
Gets a System.Drawing.Rectangle that contains the bounds of this display device.
- int [Width](#) [get, set]
Gets a System.Int32 that contains the width of this display in pixels.
- int [Height](#) [get, set]
Gets a System.Int32 that contains the height of this display in pixels.

- int [BitsPerPixel](#) [get, set]
Gets a System.Int32 that contains number of bits per pixel of this display. Typical values include 8, 16, 24 and 32.
- float [RefreshRate](#) [get, set]
Gets a System.Single representing the vertical refresh rate of this display.

5.18.1 Detailed Description

Contains information regarding a monitor's display resolution.

5.18.2 Member Function Documentation

5.18.2.1 override bool OpenTK.DisplayResolution.Equals (object *obj*)

Determines whether the specified resolutions are equal.

Parameters

obj The System.Object to check against.

Returns

True if the System.Object is an equal [DisplayResolution](#); false otherwise.

5.18.2.2 override int OpenTK.DisplayResolution.GetHashCode ()

Returns a unique hash representing this resolution.

Returns

A System.Int32 that may serve as a hash code for this resolution.

5.18.2.3 static bool OpenTK.DisplayResolution.operator!= (DisplayResolution *left*, DisplayResolution *right*) [static]

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.18.2.4 `static bool OpenTK.DisplayResolution.operator==(DisplayResolution left, DisplayResolution right) [static]`

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.18.2.5 `override string OpenTK.DisplayResolution.ToString ()`

Returns a System.String representing this [DisplayResolution](#).

Returns

A System.String representing this [DisplayResolution](#).

5.18.3 Property Documentation

5.18.3.1 `int OpenTK.DisplayResolution.BitsPerPixel [get, set]`

Gets a System.Int32 that contains number of bits per pixel of this display. Typical values include 8, 16, 24 and 32.

5.18.3.2 `Rectangle OpenTK.DisplayResolution.Bounds [get]`

Gets a System.Drawing.Rectangle that contains the bounds of this display device.

5.18.3.3 `int OpenTK.DisplayResolution.Height [get, set]`

Gets a System.Int32 that contains the height of this display in pixels.

5.18.3.4 float OpenTK.DisplayResolution.RefreshRate [get, set]

Gets a System.Single representing the vertical refresh rate of this display.

5.18.3.5 int OpenTK.DisplayResolution.Width [get, set]

Gets a System.Int32 that contains the width of this display in pixels.

5.19 OpenTK.FrameEventArgs Class Reference

Defines the arguments for frame events. A [FrameEventArgs](#) instance is only valid for the duration of the relevant event; do not store references to [FrameEventArgs](#) outside this event.

Public Member Functions

- [FrameEventArgs](#) ()

Constructs a new [FrameEventArgs](#) instance.

- [FrameEventArgs](#) (double elapsed)

Constructs a new [FrameEventArgs](#) instance.

Properties

- double [Time](#) [get, set]

Gets a System.Double that indicates how many seconds of time elapsed since the previous event.

5.19.1 Detailed Description

Defines the arguments for frame events. A [FrameEventArgs](#) instance is only valid for the duration of the relevant event; do not store references to [FrameEventArgs](#) outside this event.

5.19.2 Constructor & Destructor Documentation

5.19.2.1 OpenTK.FrameEventArgs.FrameEventArgs ()

Constructs a new [FrameEventArgs](#) instance.

5.19.2.2 OpenTK.FrameEventArgs.FrameEventArgs (double *elapsed*)

Constructs a new [FrameEventArgs](#) instance.

Parameters

elapsed The amount of time that has elapsed since the previous event, in seconds.

5.19.3 Property Documentation

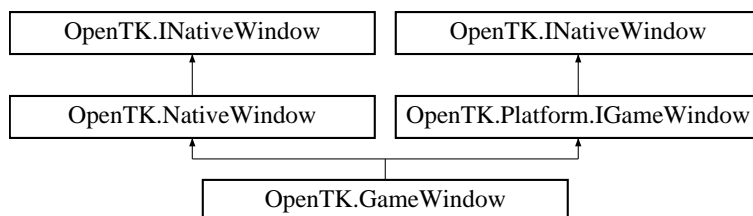
5.19.3.1 double OpenTK.FrameEventArgs.Time [get, set]

Gets a System.Double that indicates how many seconds of time elapsed since the previous event.

5.20 OpenTK.GameWindow Class Reference

The [GameWindow](#) class contains cross-platform methods to create and render on an OpenGL window, handle input and load resources.

Inheritance diagram for OpenTK.GameWindow:



Public Member Functions

- [GameWindow](#) ()

Constructs a new [GameWindow](#) with sensible default attributes.

- [GameWindow](#) (int width, int height)
Constructs a new [GameWindow](#) with the specified attributes.
- [GameWindow](#) (int width, int height, [GraphicsMode](#) mode)
Constructs a new [GameWindow](#) with the specified attributes.
- [GameWindow](#) (int width, int height, [GraphicsMode](#) mode, string title)
Constructs a new [GameWindow](#) with the specified attributes.
- [GameWindow](#) (int width, int height, [GraphicsMode](#) mode, string title, [GameWindowFlags](#) options)
Constructs a new [GameWindow](#) with the specified attributes.
- [GameWindow](#) (int width, int height, [GraphicsMode](#) mode, string title, [GameWindowFlags](#) options, [DisplayDevice](#) device)
Constructs a new [GameWindow](#) with the specified attributes.
- [GameWindow](#) (int width, int height, [GraphicsMode](#) mode, string title, [GameWindowFlags](#) options, [DisplayDevice](#) device, int major, int minor, [GraphicsContextFlags](#) flags)
Constructs a new [GameWindow](#) with the specified attributes.
- [GameWindow](#) (int width, int height, [GraphicsMode](#) mode, string title, [GameWindowFlags](#) options, [DisplayDevice](#) device, int major, int minor, [GraphicsContextFlags](#) flags, [IGraphicsContext](#) sharedContext)
Constructs a new [GameWindow](#) with the specified attributes.
- override void [Dispose](#) ()
Disposes of the [GameWindow](#), releasing all resources consumed by it.
- virtual void [Exit](#) ()
Closes the [GameWindow](#). Equivalent to [NativeWindow.Close](#) method.
- void [MakeCurrent](#) ()
Makes the [GraphicsContext](#) current on the calling thread.
- void [Run](#) ()
Enters the game loop of the [GameWindow](#) using the maximum update rate.
- void [Run](#) (double updateRate)
Enters the game loop of the [GameWindow](#) using the specified update rate. maximum possible render frequency.

- void [Run](#) (double updates_per_second, double frames_per_second)
Enters the game loop of the [GameWindow](#) updating and rendering at the specified frequency.
- void [SwapBuffers](#) ()
Swaps the front and back buffer, presenting the rendered scene to the user.

Protected Member Functions

- override void [OnClosing](#) (System.ComponentModel.CancelEventArgs e)
Called when the [NativeWindow](#) is about to close.
- virtual void [OnLoad](#) (EventArgs e)
Called after an OpenGL context has been established, but before entering the main loop.
- virtual void [OnUnload](#) (EventArgs e)
Called after [GameWindow.Exit](#) was called, but before destroying the OpenGL context.
- virtual void [Dispose](#) (bool manual)
Override to add custom cleanup logic.
- virtual void [OnRenderFrame](#) ([FrameEventArgs](#) e)
Called when the frame is rendered.
- virtual void [OnUpdateFrame](#) ([FrameEventArgs](#) e)
Called when the frame is updated.
- virtual void [OnWindowInfoChanged](#) (EventArgs e)
Called when the WindowInfo for this [GameWindow](#) has changed.
- override void [OnResize](#) (EventArgs e)

Properties

- [IGraphicsContext Context](#) [get]
Returns the opengl IGraphicsContext associated with the current [GameWindow](#).
- bool [IsExiting](#) [get]

Gets a value indicating whether the shutdown sequence has been initiated for this window, by calling [GameWindow.Exit\(\)](#) or hitting the 'close' button. If this property is true, it is no longer safe to use any [OpenTK.Input](#) or [OpenTK.Graphics.OpenGL](#) functions or properties.

- [IList< JoystickDevice > Joysticks](#) [get]
Gets a readonly IList containing all available [OpenTK.Input.JoystickDevices](#).
- [KeyboardDevice Keyboard](#) [get]
Gets the primary Keyboard device, or null if no Keyboard exists.
- [MouseDevice Mouse](#) [get]
Gets the primary Mouse device, or null if no Mouse exists.
- double [RenderFrequency](#) [get]
Gets a double representing the actual frequency of [RenderFrame](#) events, in hertz (i.e. fps or frames per second).
- double [RenderPeriod](#) [get]
Gets a double representing the period of [RenderFrame](#) events, in seconds.
- double [RenderTime](#) [get, set]
Gets a double representing the time spent in the [RenderFrame](#) function, in seconds.
- double [TargetRenderFrequency](#) [get, set]
Gets or sets a double representing the target render frequency, in hertz.
- double [TargetRenderPeriod](#) [get, set]
Gets or sets a double representing the target render period, in seconds.
- double [TargetUpdateFrequency](#) [get, set]
Gets or sets a double representing the target update frequency, in hertz.
- double [TargetUpdatePeriod](#) [get, set]
Gets or sets a double representing the target update period, in seconds.
- double [UpdateFrequency](#) [get]
Gets a double representing the frequency of [UpdateFrame](#) events, in hertz.
- double [UpdatePeriod](#) [get]
Gets a double representing the period of [UpdateFrame](#) events, in seconds.
- double [UpdateTime](#) [get]

Gets a double representing the time spent in the UpdateFrame function, in seconds.

- VSyncMode [VSync](#) [get, set]
Gets or sets the VSyncMode.
- override WindowState [WindowState](#) [get, set]
Gets or states the state of the [NativeWindow](#).

Events

- EventHandler< EventArgs > [Load](#)
Occurs before the window is displayed for the first time.
- EventHandler< [FrameEventArgs](#) > [RenderFrame](#)
Occurs when it is time to render a frame.
- EventHandler< EventArgs > [Unload](#)
Occurs before the window is destroyed.
- EventHandler< [FrameEventArgs](#) > [UpdateFrame](#)
Occurs when it is time to update a frame.

5.20.1 Detailed Description

The [GameWindow](#) class contains cross-platform methods to create and render on an OpenGL window, handle input and load resources. [GameWindow](#) contains several events you can hook or override to add your custom logic:

- OnLoad: Occurs after creating the OpenGL context, but before entering the main loop. Override to load resources.
- OnUnload: Occurs after exiting the main loop, but before deleting the OpenGL context. Override to unload resources.
- OnResize: Occurs whenever [GameWindow](#) is resized. You should update the OpenGL Viewport and Projection Matrix here.
- OnUpdateFrame: Occurs at the specified logic update rate. Override to add your game logic.
- OnRenderFrame: Occurs at the specified frame render rate. Override to add your rendering code.

Call the [Run\(\)](#) method to start the application's main loop. [Run\(double, double\)](#) takes two parameters that specify the logic update rate, and the render update rate.

5.20.2 Constructor & Destructor Documentation

5.20.2.1 `OpenTK.GameWindow.GameWindow ()`

Constructs a new [GameWindow](#) with sensible default attributes.

5.20.2.2 `OpenTK.GameWindow.GameWindow (int width, int height)`

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

5.20.2.3 `OpenTK.GameWindow.GameWindow (int width, int height, GraphicsMode mode)`

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GameWindow](#).

5.20.2.4 `OpenTK.GameWindow.GameWindow (int width, int height, GraphicsMode mode, string title)`

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GameWindow](#).

title The title of the [GameWindow](#).

5.20.2.5 OpenTK.GameWindow.GameWindow (int *width*, int *height*, GraphicsMode *mode*, string *title*, GameWindowFlags *options*)

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GameWindow](#).

title The title of the [GameWindow](#).

options [GameWindow](#) options regarding window appearance and behavior.

5.20.2.6 OpenTK.GameWindow.GameWindow (int *width*, int *height*, GraphicsMode *mode*, string *title*, GameWindowFlags *options*, DisplayDevice *device*)

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GameWindow](#).

title The title of the [GameWindow](#).

options [GameWindow](#) options regarding window appearance and behavior.

device The [OpenTK.Graphics.DisplayDevice](#) to construct the [GameWindow](#) in.

5.20.2.7 OpenTK.GameWindow.GameWindow (int *width*, int *height*, GraphicsMode *mode*, string *title*, GameWindowFlags *options*, DisplayDevice *device*, int *major*, int *minor*, GraphicsContextFlags *flags*)

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GameWindow](#).

title The title of the [GameWindow](#).

options [GameWindow](#) options regarding window appearance and behavior.

device The `OpenTK.Graphics.DisplayDevice` to construct the [GameWindow](#) in.

major The major version for the OpenGL `GraphicsContext`.

minor The minor version for the OpenGL `GraphicsContext`.

flags The `GraphicsContextFlags` version for the OpenGL `GraphicsContext`.

5.20.2.8 `OpenTK.GameWindow.GameWindow (int width, int height, GraphicsMode mode, string title, GameWindowFlags options, DisplayDevice device, int major, int minor, GraphicsContextFlags flags, IGraphicsContext sharedContext)`

Constructs a new [GameWindow](#) with the specified attributes.

Parameters

width The width of the [GameWindow](#) in pixels.

height The height of the [GameWindow](#) in pixels.

mode The `OpenTK.Graphics.GraphicsMode` of the [GameWindow](#).

title The title of the [GameWindow](#).

options [GameWindow](#) options regarding window appearance and behavior.

device The `OpenTK.Graphics.DisplayDevice` to construct the [GameWindow](#) in.

major The major version for the OpenGL `GraphicsContext`.

minor The minor version for the OpenGL `GraphicsContext`.

flags The `GraphicsContextFlags` version for the OpenGL `GraphicsContext`.

sharedContext An `IGraphicsContext` to share resources with.

5.20.3 Member Function Documentation

5.20.3.1 `override void OpenTK.GameWindow.Dispose ()`

Disposes of the [GameWindow](#), releasing all resources consumed by it.

5.20.3.2 `virtual void OpenTK.GameWindow.Dispose (bool manual)` [protected, virtual]

Override to add custom cleanup logic.

Parameters

manual True, if this method was called by the application; false if this was called by the finalizer thread.

5.20.3.3 virtual void OpenTK.GameWindow.Exit () [virtual]

Closes the [GameWindow](#). Equivalent to [NativeWindow.Close](#) method.

Override if you are not using [GameWindow.Run\(\)](#).

If you override this method, place a call to `base.Exit()`, to ensure proper OpenTK shutdown.

5.20.3.4 void OpenTK.GameWindow.MakeCurrent ()

Makes the `GraphicsContext` current on the calling thread.

Implements [OpenTK.Platform.IGameWindow](#).

5.20.3.5 override void OpenTK.GameWindow.OnClosing (System.ComponentModel.CancelEventArgs e) [protected]

Called when the [NativeWindow](#) is about to close.

Parameters

- e* The `System.ComponentModel.CancelEventArgs` for this event. Set `e.Cancel` to true in order to stop the [GameWindow](#) from closing.

5.20.3.6 virtual void OpenTK.GameWindow.OnLoad (EventArgs e) [protected, virtual]

Called after an OpenGL context has been established, but before entering the main loop.

Parameters

- e* Not used.

5.20.3.7 virtual void OpenTK.GameWindow.OnRenderFrame (FrameEventArgs e) [protected, virtual]

Called when the frame is rendered.

Parameters

- e* Contains information necessary for frame rendering.

Subscribe to the [RenderFrame](#) event instead of overriding this method.

5.20.3.8 virtual void OpenTK.GameWindow.OnUnload (EventArgs *e*) [protected, virtual]

Called after [GameWindow.Exit](#) was called, but before destroying the OpenGL context.

Parameters

e Not used.

5.20.3.9 virtual void OpenTK.GameWindow.OnUpdateFrame (FrameEventArgs *e*) [protected, virtual]

Called when the frame is updated.

Parameters

e Contains information necessary for frame updating.

Subscribe to the [UpdateFrame](#) event instead of overriding this method.

5.20.3.10 virtual void OpenTK.GameWindow.OnWindowInfoChanged (EventArgs *e*) [protected, virtual]

Called when the WindowInfo for this [GameWindow](#) has changed.

Parameters

e Not used.

5.20.3.11 void OpenTK.GameWindow.Run (double *updates_per_second*, double *frames_per_second*)

Enters the game loop of the [GameWindow](#) updating and rendering at the specified frequency.

When overriding the default game loop you should call [ProcessEvents\(\)](#) to ensure that your [GameWindow](#) responds to operating system events.

Once [ProcessEvents\(\)](#) returns, it is time to call update and render the next frame.

Parameters

updates_per_second The frequency of UpdateFrame events.

frames_per_second The frequency of RenderFrame events.

5.20.3.12 void OpenTK.GameWindow.Run ()

Enters the game loop of the [GameWindow](#) using the maximum update rate.

See also

[Run\(double\)](#)

Implements [OpenTK.Platform.IGameWindow](#).

5.20.3.13 void OpenTK.GameWindow.Run (double *updateRate*)

Enters the game loop of the [GameWindow](#) using the specified update rate. maximum possible render frequency.

Implements [OpenTK.Platform.IGameWindow](#).

5.20.3.14 void OpenTK.GameWindow.SwapBuffers ()

Swaps the front and back buffer, presenting the rendered scene to the user.

Implements [OpenTK.Platform.IGameWindow](#).

5.20.4 Property Documentation**5.20.4.1 IGraphicsContext OpenTK.GameWindow.Context [get]**

Returns the opengl IGraphicsContext associated with the current [GameWindow](#).

5.20.4.2 bool OpenTK.GameWindow.IsExiting [get]

Gets a value indicating whether the shutdown sequence has been initiated for this window, by calling [GameWindow.Exit\(\)](#) or hitting the 'close' button. If this property is true, it is no longer safe to use any [OpenTK.Input](#) or [OpenTK.Graphics.OpenGL](#) functions or properties.

5.20.4.3 IList<JoystickDevice> OpenTK.GameWindow.Joysticks [get]

Gets a readonly IList containing all available OpenTK.Input.JoystickDevices.

5.20.4.4 KeyboardDevice OpenTK.GameWindow.Keyboard [get]

Gets the primary Keyboard device, or null if no Keyboard exists.

5.20.4.5 MouseDevice OpenTK.GameWindow.Mouse [get]

Gets the primary Mouse device, or null if no Mouse exists.

5.20.4.6 double OpenTK.GameWindow.RenderFrequency [get]

Gets a double representing the actual frequency of RenderFrame events, in hertz (i.e. fps or frames per second).

5.20.4.7 double OpenTK.GameWindow.RenderPeriod [get]

Gets a double representing the period of RenderFrame events, in seconds.

5.20.4.8 double OpenTK.GameWindow.RenderTime [get, set]

Gets a double representing the time spent in the RenderFrame function, in seconds.

5.20.4.9 double OpenTK.GameWindow.TargetRenderFrequency [get, set]

Gets or sets a double representing the target render frequency, in hertz.

A value of 0.0 indicates that RenderFrame events are generated at the maximum possible frequency (i.e. only limited by the hardware's capabilities).

Values lower than 1.0Hz are clamped to 1.0Hz. Values higher than 200.0Hz are clamped to 200.0Hz.

5.20.4.10 double OpenTK.GameWindow.TargetRenderPeriod [get, set]

Gets or sets a double representing the target render period, in seconds.

A value of 0.0 indicates that RenderFrame events are generated at the maximum possible frequency (i.e. only limited by the hardware's capabilities).

Values lower than 0.005 seconds (200Hz) are clamped to 0.0. Values higher than 1.0 seconds (1Hz) are clamped to 1.0.

5.20.4.11 double OpenTK.GameWindow.TargetUpdateFrequency [get, set]

Gets or sets a double representing the target update frequency, in hertz.

A value of 0.0 indicates that UpdateFrame events are generated at the maximum possible frequency (i.e. only limited by the hardware's capabilities).

Values lower than 1.0Hz are clamped to 1.0Hz. Values higher than 200.0Hz are clamped to 200.0Hz.

5.20.4.12 double OpenTK.GameWindow.TargetUpdatePeriod [get, set]

Gets or sets a double representing the target update period, in seconds.

A value of 0.0 indicates that UpdateFrame events are generated at the maximum possible frequency (i.e. only limited by the hardware's capabilities).

Values lower than 0.005 seconds (200Hz) are clamped to 0.0. Values higher than 1.0 seconds (1Hz) are clamped to 1.0.

5.20.4.13 double OpenTK.GameWindow.UpdateFrequency [get]

Gets a double representing the frequency of UpdateFrame events, in hertz.

5.20.4.14 double OpenTK.GameWindow.UpdatePeriod [get]

Gets a double representing the period of UpdateFrame events, in seconds.

5.20.4.15 double OpenTK.GameWindow.UpdateTime [get]

Gets a double representing the time spent in the UpdateFrame function, in seconds.

5.20.4.16 VSyncMode OpenTK.GameWindow.VSync [get, set]

Gets or sets the VSyncMode.

5.20.4.17 override WindowState OpenTK.GameWindow.WindowState [get, set]

Gets or states the state of the [NativeWindow](#).

Implements [OpenTK.INativeWindow](#).

5.20.5 Event Documentation

5.20.5.1 EventHandler<EventArgs> OpenTK.GameWindow.Load

Occurs before the window is displayed for the first time.

Implements [OpenTK.Platform.IGameWindow](#).

5.20.5.2 EventHandler<FrameEventArgs> OpenTK.GameWindow.RenderFrame

Occurs when it is time to render a frame.

Implements [OpenTK.Platform.IGameWindow](#).

5.20.5.3 EventHandler<EventArgs> OpenTK.GameWindow.Unload

Occurs before the window is destroyed.

Implements [OpenTK.Platform.IGameWindow](#).

5.20.5.4 EventHandler<FrameEventArgs> OpenTK.GameWindow.UpdateFrame

Occurs when it is time to update a frame.

Implements [OpenTK.Platform.IGameWindow](#).

5.21 OpenTK.GLControl Class Reference

Defines a UserControl with OpenGL rendering capabilities.

Public Member Functions

- [GLControl](#) ()
Constructs a new [GLControl](#).
- [GLControl](#) (GraphicsMode mode)
Constructs a new [GLControl](#) with the specified GraphicsMode.
- [GLControl](#) (GraphicsMode mode, int major, int minor, GraphicsContextFlags flags)
Constructs a new [GLControl](#) with the specified GraphicsMode.

- void [SwapBuffers](#) ()
Swaps the front and back buffers, presenting the rendered scene to the screen.
- void [MakeCurrent](#) ()
Makes the underlying this [GLControl](#) current in the calling thread. All OpenGL commands issued are hereafter interpreted by this [GLControl](#).
- Bitmap [GrabScreenshot](#) ()
Grabs a screenshot of the frontbuffer contents.

Protected Member Functions

- override void [OnHandleCreated](#) (EventArgs e)
Raises the HandleCreated event.
- override void [OnHandleDestroyed](#) (EventArgs e)
Raises the HandleDestroyed event.
- override void [OnPaint](#) (PaintEventArgs e)
Raises the System.Windows.Forms.Control.Paint event.
- override void [OnResize](#) (EventArgs e)
Raises the Resize event. Note: this method may be called before the OpenGL context is ready. Check that IsHandleCreated is true before using any OpenGL methods.
- override void [OnParentChanged](#) (EventArgs e)
Raises the ParentChanged event.
- override void [Dispose](#) (bool disposing)
Clean up any resources being used.

Properties

- bool [IsIdle](#) [get]
Gets a value indicating whether the current thread contains pending system messages.
- [IGraphicsContext Context](#) [get, set]
Gets an interface to the underlying GraphicsContext used by this [GLControl](#).

- float [AspectRatio](#) [get]
Gets the aspect ratio of this [GLControl](#).
- bool [VSync](#) [get, set]
Gets or sets a value indicating whether vsync is active for this [GLControl](#).
- [GraphicsMode GraphicsMode](#) [get]
Gets the GraphicsMode of the GraphicsContext attached to this [GLControl](#).
- [IWindowInfo WindowInfo](#) [get]
Gets the [OpenTK.Platform.IWindowInfo](#) for this instance.

5.21.1 Detailed Description

Defines a UserControl with OpenGL rendering capabilities.

5.21.2 Constructor & Destructor Documentation

5.21.2.1 [OpenTK.GLControl.GLControl](#) ()

Constructs a new [GLControl](#).

5.21.2.2 [OpenTK.GLControl.GLControl](#) ([GraphicsMode mode](#))

Constructs a new [GLControl](#) with the specified GraphicsMode.

Parameters

mode The [OpenTK.Graphics.GraphicsMode](#) of the control.

5.21.2.3 [OpenTK.GLControl.GLControl](#) ([GraphicsMode mode](#), int *major*, int *minor*, [GraphicsContextFlags flags](#))

Constructs a new [GLControl](#) with the specified GraphicsMode.

Parameters

mode The [OpenTK.Graphics.GraphicsMode](#) of the control.

major The major version for the OpenGL GraphicsContext.

minor The minor version for the OpenGL GraphicsContext.

flags The GraphicsContextFlags for the OpenGL GraphicsContext.

5.21.3 Member Function Documentation

5.21.3.1 override void OpenTK.GLControl.Dispose (bool *disposing*) [protected]

Clean up any resources being used.

Parameters

disposing true if managed resources should be disposed; otherwise, false.

5.21.3.2 Bitmap OpenTK.GLControl.GrabScreenshot ()

Grabs a screenshot of the frontbuffer contents.

Returns

A System.Drawing.Bitmap, containing the contents of the frontbuffer.

Exceptions

[OpenTK.Graphics.GraphicsContextException](#) Occurs when no [OpenTK.Graphics.GraphicsContext](#) is current in the calling thread.

5.21.3.3 void OpenTK.GLControl.MakeCurrent ()

Makes the underlying this [GLControl](#) current in the calling thread. All OpenGL commands issued are hereafter interpreted by this [GLControl](#).

5.21.3.4 override void OpenTK.GLControl.OnHandleCreated (EventArgs *e*) [protected]

Raises the HandleCreated event.

Parameters

e Not used.

5.21.3.5 override void OpenTK.GLControl.OnHandleDestroyed (EventArgs *e*) [protected]

Raises the HandleDestroyed event.

Parameters

e Not used.

**5.21.3.6 override void OpenTK.GLControl.OnPaint (PaintEventArgs *e*)
[protected]**

Raises the System.Windows.Forms.Control.Paint event.

Parameters

e A System.Windows.Forms.PaintEventArgs that contains the event data.

**5.21.3.7 override void OpenTK.GLControl.OnParentChanged (EventArgs *e*)
[protected]**

Raises the ParentChanged event.

Parameters

e A System.EventArgs that contains the event data.

**5.21.3.8 override void OpenTK.GLControl.OnResize (EventArgs *e*)
[protected]**

Raises the Resize event. Note: this method may be called before the OpenGL context is ready. Check that IsHandleCreated is true before using any OpenGL methods.

Parameters

e A System.EventArgs that contains the event data.

5.21.3.9 void OpenTK.GLControl.SwapBuffers ()

Swaps the front and back buffers, presenting the rendered scene to the screen.

5.21.4 Property Documentation**5.21.4.1 float OpenTK.GLControl.AspectRatio [get]**

Gets the aspect ratio of this [GLControl](#).

5.21.4.2 IGraphicsContext OpenTK.GLControl.Context [get, set]

Gets an interface to the underlying GraphicsContext used by this [GLControl](#).

5.21.4.3 GraphicsMode OpenTK.GLControl.GraphicsMode [get]

Gets the GraphicsMode of the GraphicsContext attached to this [GLControl](#).
To change the GraphicsMode, you must destroy and recreate the [GLControl](#).

5.21.4.4 bool OpenTK.GLControl.IsIdle [get]

Gets a value indicating whether the current thread contains pending system messages.

5.21.4.5 bool OpenTK.GLControl.VSync [get, set]

Gets or sets a value indicating whether vsync is active for this [GLControl](#).

5.21.4.6 IWindowInfo OpenTK.GLControl.WindowInfo [get]

Gets the [OpenTK.Platform.IWindowInfo](#) for this instance.

5.22 OpenTK.Graphics.Color4 Struct Reference

Represents a color with 4 floating-point components (R, G, B, A).

Public Member Functions

- [Color4](#) (float r, float g, float b, float a)
Constructs a new [Color4](#) structure from the specified components.
- [Color4](#) (byte r, byte g, byte b, byte a)
Constructs a new [Color4](#) structure from the specified components.
- [Color4](#) (System.Drawing.Color color)
Constructs a new [Color4](#) structure from the specified System.Drawing.Color.
- int [ToArgb](#) ()
Converts this color to an integer representation with 8 bits per channel.

- override bool [Equals](#) (object obj)
Compares whether this [Color4](#) structure is equal to the specified object.
- override int [GetHashCode](#) ()
Calculates the hash code for this [Color4](#) structure.
- override string [ToString](#) ()
Creates a System.String that describes this [Color4](#) structure.
- bool [Equals](#) ([Color4](#) other)
Compares whether this [Color4](#) structure is equal to the specified [Color4](#).

Static Public Member Functions

- static bool [operator==](#) ([Color4](#) left, [Color4](#) right)
Compares the specified [Color4](#) structures for equality.
- static bool [operator!=](#) ([Color4](#) left, [Color4](#) right)
Compares the specified [Color4](#) structures for inequality.
- static implicit [operator Color4](#) (System.Drawing.Color color)
Converts the specified System.Drawing.Color to a [Color4](#) structure.
- static [operator System.Drawing.Color](#) ([Color4](#) color)
Converts the specified [Color4](#) to a System.Drawing.Color structure.

Public Attributes

- float [R](#)
The red component of this [Color4](#) structure.
- float [G](#)
The green component of this [Color4](#) structure.
- float [B](#)
The blue component of this [Color4](#) structure.
- float [A](#)
The alpha component of this [Color4](#) structure.

Properties

- static [Color4 Transparent](#) [get]
Gets the system color with $(R, G, B, A) = (255, 255, 255, 0)$.
- static [Color4 AliceBlue](#) [get]
Gets the system color with $(R, G, B, A) = (240, 248, 255, 255)$.
- static [Color4 AntiqueWhite](#) [get]
Gets the system color with $(R, G, B, A) = (250, 235, 215, 255)$.
- static [Color4 Aqua](#) [get]
Gets the system color with $(R, G, B, A) = (0, 255, 255, 255)$.
- static [Color4 Aquamarine](#) [get]
Gets the system color with $(R, G, B, A) = (127, 255, 212, 255)$.
- static [Color4 Azure](#) [get]
Gets the system color with $(R, G, B, A) = (240, 255, 255, 255)$.
- static [Color4 Beige](#) [get]
Gets the system color with $(R, G, B, A) = (245, 245, 220, 255)$.
- static [Color4 Bisque](#) [get]
Gets the system color with $(R, G, B, A) = (255, 228, 196, 255)$.
- static [Color4 Black](#) [get]
Gets the system color with $(R, G, B, A) = (0, 0, 0, 255)$.
- static [Color4 BlanchedAlmond](#) [get]
Gets the system color with $(R, G, B, A) = (255, 235, 205, 255)$.
- static [Color4 Blue](#) [get]
Gets the system color with $(R, G, B, A) = (0, 0, 255, 255)$.
- static [Color4 BlueViolet](#) [get]
Gets the system color with $(R, G, B, A) = (138, 43, 226, 255)$.
- static [Color4 Brown](#) [get]
Gets the system color with $(R, G, B, A) = (165, 42, 42, 255)$.
- static [Color4 BurlyWood](#) [get]
Gets the system color with $(R, G, B, A) = (222, 184, 135, 255)$.

- static [Color4 CadetBlue](#) [get]
Gets the system color with (R, G, B, A) = (95, 158, 160, 255).
- static [Color4 Chartreuse](#) [get]
Gets the system color with (R, G, B, A) = (127, 255, 0, 255).
- static [Color4 Chocolate](#) [get]
Gets the system color with (R, G, B, A) = (210, 105, 30, 255).
- static [Color4 Coral](#) [get]
Gets the system color with (R, G, B, A) = (255, 127, 80, 255).
- static [Color4 CornflowerBlue](#) [get]
Gets the system color with (R, G, B, A) = (100, 149, 237, 255).
- static [Color4 Cornsilk](#) [get]
Gets the system color with (R, G, B, A) = (255, 248, 220, 255).
- static [Color4 Crimson](#) [get]
Gets the system color with (R, G, B, A) = (220, 20, 60, 255).
- static [Color4 Cyan](#) [get]
Gets the system color with (R, G, B, A) = (0, 255, 255, 255).
- static [Color4 DarkBlue](#) [get]
Gets the system color with (R, G, B, A) = (0, 0, 139, 255).
- static [Color4 DarkCyan](#) [get]
Gets the system color with (R, G, B, A) = (0, 139, 139, 255).
- static [Color4 DarkGoldenrod](#) [get]
Gets the system color with (R, G, B, A) = (184, 134, 11, 255).
- static [Color4 DarkGray](#) [get]
Gets the system color with (R, G, B, A) = (169, 169, 169, 255).
- static [Color4 DarkGreen](#) [get]
Gets the system color with (R, G, B, A) = (0, 100, 0, 255).
- static [Color4 DarkKhaki](#) [get]
Gets the system color with (R, G, B, A) = (189, 183, 107, 255).

- static [Color4 DarkMagenta](#) [get]
Gets the system color with $(R, G, B, A) = (139, 0, 139, 255)$.
- static [Color4 DarkOliveGreen](#) [get]
Gets the system color with $(R, G, B, A) = (85, 107, 47, 255)$.
- static [Color4 DarkOrange](#) [get]
Gets the system color with $(R, G, B, A) = (255, 140, 0, 255)$.
- static [Color4 DarkOrchid](#) [get]
Gets the system color with $(R, G, B, A) = (153, 50, 204, 255)$.
- static [Color4 DarkRed](#) [get]
Gets the system color with $(R, G, B, A) = (139, 0, 0, 255)$.
- static [Color4 DarkSalmon](#) [get]
Gets the system color with $(R, G, B, A) = (233, 150, 122, 255)$.
- static [Color4 DarkSeaGreen](#) [get]
Gets the system color with $(R, G, B, A) = (143, 188, 139, 255)$.
- static [Color4 DarkSlateBlue](#) [get]
Gets the system color with $(R, G, B, A) = (72, 61, 139, 255)$.
- static [Color4 DarkSlateGray](#) [get]
Gets the system color with $(R, G, B, A) = (47, 79, 79, 255)$.
- static [Color4 DarkTurquoise](#) [get]
Gets the system color with $(R, G, B, A) = (0, 206, 209, 255)$.
- static [Color4 DarkViolet](#) [get]
Gets the system color with $(R, G, B, A) = (148, 0, 211, 255)$.
- static [Color4 DeepPink](#) [get]
Gets the system color with $(R, G, B, A) = (255, 20, 147, 255)$.
- static [Color4 DeepSkyBlue](#) [get]
Gets the system color with $(R, G, B, A) = (0, 191, 255, 255)$.
- static [Color4 DimGray](#) [get]
Gets the system color with $(R, G, B, A) = (105, 105, 105, 255)$.

- static [Color4 DodgerBlue](#) [get]
Gets the system color with (R, G, B, A) = (30, 144, 255, 255).
- static [Color4 Firebrick](#) [get]
Gets the system color with (R, G, B, A) = (178, 34, 34, 255).
- static [Color4 FloralWhite](#) [get]
Gets the system color with (R, G, B, A) = (255, 250, 240, 255).
- static [Color4 ForestGreen](#) [get]
Gets the system color with (R, G, B, A) = (34, 139, 34, 255).
- static [Color4 Fuchsia](#) [get]
Gets the system color with (R, G, B, A) = (255, 0, 255, 255).
- static [Color4 Gainsboro](#) [get]
Gets the system color with (R, G, B, A) = (220, 220, 220, 255).
- static [Color4 GhostWhite](#) [get]
Gets the system color with (R, G, B, A) = (248, 248, 255, 255).
- static [Color4 Gold](#) [get]
Gets the system color with (R, G, B, A) = (255, 215, 0, 255).
- static [Color4 Goldenrod](#) [get]
Gets the system color with (R, G, B, A) = (218, 165, 32, 255).
- static [Color4 Gray](#) [get]
Gets the system color with (R, G, B, A) = (128, 128, 128, 255).
- static [Color4 Green](#) [get]
Gets the system color with (R, G, B, A) = (0, 128, 0, 255).
- static [Color4 GreenYellow](#) [get]
Gets the system color with (R, G, B, A) = (173, 255, 47, 255).
- static [Color4 Honeydew](#) [get]
Gets the system color with (R, G, B, A) = (240, 255, 240, 255).
- static [Color4 HotPink](#) [get]
Gets the system color with (R, G, B, A) = (255, 105, 180, 255).

- static [Color4 IndianRed](#) [get]
Gets the system color with (R, G, B, A) = (205, 92, 92, 255).
- static [Color4 Indigo](#) [get]
Gets the system color with (R, G, B, A) = (75, 0, 130, 255).
- static [Color4 Ivory](#) [get]
Gets the system color with (R, G, B, A) = (255, 255, 240, 255).
- static [Color4 Khaki](#) [get]
Gets the system color with (R, G, B, A) = (240, 230, 140, 255).
- static [Color4 Lavender](#) [get]
Gets the system color with (R, G, B, A) = (230, 230, 250, 255).
- static [Color4 LavenderBlush](#) [get]
Gets the system color with (R, G, B, A) = (255, 240, 245, 255).
- static [Color4 LawnGreen](#) [get]
Gets the system color with (R, G, B, A) = (124, 252, 0, 255).
- static [Color4 LemonChiffon](#) [get]
Gets the system color with (R, G, B, A) = (255, 250, 205, 255).
- static [Color4 LightBlue](#) [get]
Gets the system color with (R, G, B, A) = (173, 216, 230, 255).
- static [Color4 LightCoral](#) [get]
Gets the system color with (R, G, B, A) = (240, 128, 128, 255).
- static [Color4 LightCyan](#) [get]
Gets the system color with (R, G, B, A) = (224, 255, 255, 255).
- static [Color4 LightGoldenrodYellow](#) [get]
Gets the system color with (R, G, B, A) = (250, 250, 210, 255).
- static [Color4 LightGreen](#) [get]
Gets the system color with (R, G, B, A) = (144, 238, 144, 255).
- static [Color4 LightGray](#) [get]
Gets the system color with (R, G, B, A) = (211, 211, 211, 255).

- static [Color4 LightPink](#) [get]
Gets the system color with (R, G, B, A) = (255, 182, 193, 255).
- static [Color4 LightSalmon](#) [get]
Gets the system color with (R, G, B, A) = (255, 160, 122, 255).
- static [Color4 LightSeaGreen](#) [get]
Gets the system color with (R, G, B, A) = (32, 178, 170, 255).
- static [Color4 LightSkyBlue](#) [get]
Gets the system color with (R, G, B, A) = (135, 206, 250, 255).
- static [Color4 LightSlateGray](#) [get]
Gets the system color with (R, G, B, A) = (119, 136, 153, 255).
- static [Color4 LightSteelBlue](#) [get]
Gets the system color with (R, G, B, A) = (176, 196, 222, 255).
- static [Color4 LightYellow](#) [get]
Gets the system color with (R, G, B, A) = (255, 255, 224, 255).
- static [Color4 Lime](#) [get]
Gets the system color with (R, G, B, A) = (0, 255, 0, 255).
- static [Color4 LimeGreen](#) [get]
Gets the system color with (R, G, B, A) = (50, 205, 50, 255).
- static [Color4 Linen](#) [get]
Gets the system color with (R, G, B, A) = (250, 240, 230, 255).
- static [Color4 Magenta](#) [get]
Gets the system color with (R, G, B, A) = (255, 0, 255, 255).
- static [Color4 Maroon](#) [get]
Gets the system color with (R, G, B, A) = (128, 0, 0, 255).
- static [Color4 MediumAquamarine](#) [get]
Gets the system color with (R, G, B, A) = (102, 205, 170, 255).
- static [Color4 MediumBlue](#) [get]
Gets the system color with (R, G, B, A) = (0, 0, 205, 255).

- static [Color4 MediumOrchid](#) [get]
Gets the system color with (R, G, B, A) = (186, 85, 211, 255).
- static [Color4 MediumPurple](#) [get]
Gets the system color with (R, G, B, A) = (147, 112, 219, 255).
- static [Color4 MediumSeaGreen](#) [get]
Gets the system color with (R, G, B, A) = (60, 179, 113, 255).
- static [Color4 MediumSlateBlue](#) [get]
Gets the system color with (R, G, B, A) = (123, 104, 238, 255).
- static [Color4 MediumSpringGreen](#) [get]
Gets the system color with (R, G, B, A) = (0, 250, 154, 255).
- static [Color4 MediumTurquoise](#) [get]
Gets the system color with (R, G, B, A) = (72, 209, 204, 255).
- static [Color4 MediumVioletRed](#) [get]
Gets the system color with (R, G, B, A) = (199, 21, 133, 255).
- static [Color4 MidnightBlue](#) [get]
Gets the system color with (R, G, B, A) = (25, 25, 112, 255).
- static [Color4 MintCream](#) [get]
Gets the system color with (R, G, B, A) = (245, 255, 250, 255).
- static [Color4 MistyRose](#) [get]
Gets the system color with (R, G, B, A) = (255, 228, 225, 255).
- static [Color4 Moccasin](#) [get]
Gets the system color with (R, G, B, A) = (255, 228, 181, 255).
- static [Color4 NavajoWhite](#) [get]
Gets the system color with (R, G, B, A) = (255, 222, 173, 255).
- static [Color4 Navy](#) [get]
Gets the system color with (R, G, B, A) = (0, 0, 128, 255).
- static [Color4 OldLace](#) [get]
Gets the system color with (R, G, B, A) = (253, 245, 230, 255).

- static [Color4 Olive](#) [get]
Gets the system color with (R, G, B, A) = (128, 128, 0, 255).
- static [Color4 OliveDrab](#) [get]
Gets the system color with (R, G, B, A) = (107, 142, 35, 255).
- static [Color4 Orange](#) [get]
Gets the system color with (R, G, B, A) = (255, 165, 0, 255).
- static [Color4 OrangeRed](#) [get]
Gets the system color with (R, G, B, A) = (255, 69, 0, 255).
- static [Color4 Orchid](#) [get]
Gets the system color with (R, G, B, A) = (218, 112, 214, 255).
- static [Color4 PaleGoldenrod](#) [get]
Gets the system color with (R, G, B, A) = (238, 232, 170, 255).
- static [Color4 PaleGreen](#) [get]
Gets the system color with (R, G, B, A) = (152, 251, 152, 255).
- static [Color4 PaleTurquoise](#) [get]
Gets the system color with (R, G, B, A) = (175, 238, 238, 255).
- static [Color4 PaleVioletRed](#) [get]
Gets the system color with (R, G, B, A) = (219, 112, 147, 255).
- static [Color4 PapayaWhip](#) [get]
Gets the system color with (R, G, B, A) = (255, 239, 213, 255).
- static [Color4 PeachPuff](#) [get]
Gets the system color with (R, G, B, A) = (255, 218, 185, 255).
- static [Color4 Peru](#) [get]
Gets the system color with (R, G, B, A) = (205, 133, 63, 255).
- static [Color4 Pink](#) [get]
Gets the system color with (R, G, B, A) = (255, 192, 203, 255).
- static [Color4 Plum](#) [get]
Gets the system color with (R, G, B, A) = (221, 160, 221, 255).

- static [Color4 PowderBlue](#) [get]
Gets the system color with (R, G, B, A) = (176, 224, 230, 255).
- static [Color4 Purple](#) [get]
Gets the system color with (R, G, B, A) = (128, 0, 128, 255).
- static [Color4 Red](#) [get]
Gets the system color with (R, G, B, A) = (255, 0, 0, 255).
- static [Color4 RosyBrown](#) [get]
Gets the system color with (R, G, B, A) = (188, 143, 143, 255).
- static [Color4 RoyalBlue](#) [get]
Gets the system color with (R, G, B, A) = (65, 105, 225, 255).
- static [Color4 SaddleBrown](#) [get]
Gets the system color with (R, G, B, A) = (139, 69, 19, 255).
- static [Color4 Salmon](#) [get]
Gets the system color with (R, G, B, A) = (250, 128, 114, 255).
- static [Color4 SandyBrown](#) [get]
Gets the system color with (R, G, B, A) = (244, 164, 96, 255).
- static [Color4 SeaGreen](#) [get]
Gets the system color with (R, G, B, A) = (46, 139, 87, 255).
- static [Color4 SeaShell](#) [get]
Gets the system color with (R, G, B, A) = (255, 245, 238, 255).
- static [Color4 Sienna](#) [get]
Gets the system color with (R, G, B, A) = (160, 82, 45, 255).
- static [Color4 Silver](#) [get]
Gets the system color with (R, G, B, A) = (192, 192, 192, 255).
- static [Color4 SkyBlue](#) [get]
Gets the system color with (R, G, B, A) = (135, 206, 235, 255).
- static [Color4 SlateBlue](#) [get]
Gets the system color with (R, G, B, A) = (106, 90, 205, 255).

- static [Color4 SlateGray](#) [get]
Gets the system color with (R, G, B, A) = (112, 128, 144, 255).
- static [Color4 Snow](#) [get]
Gets the system color with (R, G, B, A) = (255, 250, 250, 255).
- static [Color4 SpringGreen](#) [get]
Gets the system color with (R, G, B, A) = (0, 255, 127, 255).
- static [Color4 SteelBlue](#) [get]
Gets the system color with (R, G, B, A) = (70, 130, 180, 255).
- static [Color4 Tan](#) [get]
Gets the system color with (R, G, B, A) = (210, 180, 140, 255).
- static [Color4 Teal](#) [get]
Gets the system color with (R, G, B, A) = (0, 128, 128, 255).
- static [Color4 Thistle](#) [get]
Gets the system color with (R, G, B, A) = (216, 191, 216, 255).
- static [Color4 Tomato](#) [get]
Gets the system color with (R, G, B, A) = (255, 99, 71, 255).
- static [Color4 Turquoise](#) [get]
Gets the system color with (R, G, B, A) = (64, 224, 208, 255).
- static [Color4 Violet](#) [get]
Gets the system color with (R, G, B, A) = (238, 130, 238, 255).
- static [Color4 Wheat](#) [get]
Gets the system color with (R, G, B, A) = (245, 222, 179, 255).
- static [Color4 White](#) [get]
Gets the system color with (R, G, B, A) = (255, 255, 255, 255).
- static [Color4 WhiteSmoke](#) [get]
Gets the system color with (R, G, B, A) = (245, 245, 245, 255).
- static [Color4 Yellow](#) [get]
Gets the system color with (R, G, B, A) = (255, 255, 0, 255).

- static [Color4 YellowGreen](#) [get]

Gets the system color with (R, G, B, A) = (154, 205, 50, 255).

5.22.1 Detailed Description

Represents a color with 4 floating-point components (R, G, B, A).

5.22.2 Constructor & Destructor Documentation

5.22.2.1 OpenTK.Graphics.Color4.Color4 (float *r*, float *g*, float *b*, float *a*)

Constructs a new [Color4](#) structure from the specified components.

Parameters

- r* The red component of the new [Color4](#) structure.
- g* The green component of the new [Color4](#) structure.
- b* The blue component of the new [Color4](#) structure.
- a* The alpha component of the new [Color4](#) structure.

5.22.2.2 OpenTK.Graphics.Color4.Color4 (byte *r*, byte *g*, byte *b*, byte *a*)

Constructs a new [Color4](#) structure from the specified components.

Parameters

- r* The red component of the new [Color4](#) structure.
- g* The green component of the new [Color4](#) structure.
- b* The blue component of the new [Color4](#) structure.
- a* The alpha component of the new [Color4](#) structure.

5.22.2.3 OpenTK.Graphics.Color4.Color4 (System.Drawing.Color *color*)

Constructs a new [Color4](#) structure from the specified System.Drawing.Color.

Parameters

- color* The System.Drawing.Color containing the component values.

5.22.3 Member Function Documentation

5.22.3.1 override bool OpenTK.Graphics.Color4.Equals (object *obj*)

Compares whether this [Color4](#) structure is equal to the specified object.

Parameters

obj An object to compare to.

Returns

True if *obj* is a [Color4](#) structure with the same components as this [Color4](#); false otherwise.

5.22.3.2 bool OpenTK.Graphics.Color4.Equals (Color4 *other*)

Compares whether this [Color4](#) structure is equal to the specified [Color4](#).

Parameters

other The [Color4](#) structure to compare to.

Returns

True if both [Color4](#) structures contain the same components; false otherwise.

5.22.3.3 override int OpenTK.Graphics.Color4.GetHashCode ()

Calculates the hash code for this [Color4](#) structure.

Returns

A System.Int32 containing the hashcode of this [Color4](#) structure.

5.22.3.4 static implicit OpenTK.Graphics.Color4.operator Color4 (System.Drawing.Color *color*) [static]

Converts the specified System.Drawing.Color to a [Color4](#) structure.

Parameters

color The System.Drawing.Color to convert.

Returns

A new [Color4](#) structure containing the converted components.

5.22.3.5 static OpenTK.Graphics.Color4.operator System.Drawing.Color (Color4 *color*) [explicit, static]

Converts the specified [Color4](#) to a System.Drawing.Color structure.

Parameters

color The [Color4](#) to convert.

Returns

A new System.Drawing.Color structure containing the converted components.

5.22.3.6 static bool OpenTK.Graphics.Color4.operator!= (Color4 *left*, Color4 *right*) [static]

Compares the specified [Color4](#) structures for inequality.

Parameters

left The left-hand side of the comparison.

right The right-hand side of the comparison.

Returns

True if left is not equal to right; false otherwise.

5.22.3.7 static bool OpenTK.Graphics.Color4.operator== (Color4 *left*, Color4 *right*) [static]

Compares the specified [Color4](#) structures for equality.

Parameters

left The left-hand side of the comparison.

right The right-hand side of the comparison.

Returns

True if left is equal to right; false otherwise.

5.22.3.8 int OpenTK.Graphics.Color4.ToArgb ()

Converts this color to an integer representation with 8 bits per channel.

Returns

A System.Int32 that represents this instance.

This method is intended only for compatibility with System.Drawing. It compresses the color into 8 bits per channel, which means color information is lost.

5.22.3.9 override string OpenTK.Graphics.Color4.ToString ()

Creates a System.String that describes this [Color4](#) structure.

Returns

A System.String that describes this [Color4](#) structure.

5.22.4 Member Data Documentation

5.22.4.1 float OpenTK.Graphics.Color4.A

The alpha component of this [Color4](#) structure.

5.22.4.2 float OpenTK.Graphics.Color4.B

The blue component of this [Color4](#) structure.

5.22.4.3 float OpenTK.Graphics.Color4.G

The green component of this [Color4](#) structure.

5.22.4.4 float OpenTK.Graphics.Color4.R

The red component of this [Color4](#) structure.

5.22.5 Property Documentation

5.22.5.1 Color4 OpenTK.Graphics.Color4.AliceBlue [static, get]

Gets the system color with (R, G, B, A) = (240, 248, 255, 255).

5.22.5.2 Color4 OpenTK.Graphics.Color4.AntiqueWhite [static, get]

Gets the system color with (R, G, B, A) = (250, 235, 215, 255).

5.22.5.3 Color4 OpenTK.Graphics.Color4.Aqua [static, get]

Gets the system color with (R, G, B, A) = (0, 255, 255, 255).

5.22.5.4 Color4 OpenTK.Graphics.Color4.Aquamarine [static, get]

Gets the system color with (R, G, B, A) = (127, 255, 212, 255).

5.22.5.5 Color4 OpenTK.Graphics.Color4.Azure [static, get]

Gets the system color with (R, G, B, A) = (240, 255, 255, 255).

5.22.5.6 Color4 OpenTK.Graphics.Color4.Beige [static, get]

Gets the system color with (R, G, B, A) = (245, 245, 220, 255).

5.22.5.7 Color4 OpenTK.Graphics.Color4.Bisque [static, get]

Gets the system color with (R, G, B, A) = (255, 228, 196, 255).

5.22.5.8 Color4 OpenTK.Graphics.Color4.Black [static, get]

Gets the system color with (R, G, B, A) = (0, 0, 0, 255).

5.22.5.9 Color4 OpenTK.Graphics.Color4.BlanchedAlmond [static, get]

Gets the system color with (R, G, B, A) = (255, 235, 205, 255).

5.22.5.10 Color4 OpenTK.Graphics.Color4.Blue [static, get]

Gets the system color with (R, G, B, A) = (0, 0, 255, 255).

5.22.5.11 Color4 OpenTK.Graphics.Color4.BlueViolet [static, get]

Gets the system color with (R, G, B, A) = (138, 43, 226, 255).

5.22.5.12 Color4 OpenTK.Graphics.Color4.Brown [static, get]

Gets the system color with (R, G, B, A) = (165, 42, 42, 255).

5.22.5.13 Color4 OpenTK.Graphics.Color4.BurlyWood [static, get]

Gets the system color with (R, G, B, A) = (222, 184, 135, 255).

5.22.5.14 Color4 OpenTK.Graphics.Color4.CadetBlue [static, get]

Gets the system color with (R, G, B, A) = (95, 158, 160, 255).

5.22.5.15 Color4 OpenTK.Graphics.Color4.Chartreuse [static, get]

Gets the system color with (R, G, B, A) = (127, 255, 0, 255).

5.22.5.16 Color4 OpenTK.Graphics.Color4.Chocolate [static, get]

Gets the system color with (R, G, B, A) = (210, 105, 30, 255).

5.22.5.17 Color4 OpenTK.Graphics.Color4.Coral [static, get]

Gets the system color with (R, G, B, A) = (255, 127, 80, 255).

5.22.5.18 Color4 OpenTK.Graphics.Color4.CornflowerBlue [static, get]

Gets the system color with (R, G, B, A) = (100, 149, 237, 255).

5.22.5.19 Color4 OpenTK.Graphics.Color4.Cornsilk [static, get]

Gets the system color with (R, G, B, A) = (255, 248, 220, 255).

5.22.5.20 Color4 OpenTK.Graphics.Color4.Crimson [static, get]

Gets the system color with (R, G, B, A) = (220, 20, 60, 255).

5.22.5.21 Color4 OpenTK.Graphics.Color4.Cyan [static, get]

Gets the system color with (R, G, B, A) = (0, 255, 255, 255).

5.22.5.22 Color4 OpenTK.Graphics.Color4.DarkBlue [static, get]

Gets the system color with (R, G, B, A) = (0, 0, 139, 255).

5.22.5.23 Color4 OpenTK.Graphics.Color4.DarkCyan [static, get]

Gets the system color with (R, G, B, A) = (0, 139, 139, 255).

5.22.5.24 Color4 OpenTK.Graphics.Color4.DarkGoldenrod [static, get]

Gets the system color with (R, G, B, A) = (184, 134, 11, 255).

5.22.5.25 Color4 OpenTK.Graphics.Color4.DarkGray [static, get]

Gets the system color with (R, G, B, A) = (169, 169, 169, 255).

5.22.5.26 Color4 OpenTK.Graphics.Color4.DarkGreen [static, get]

Gets the system color with (R, G, B, A) = (0, 100, 0, 255).

5.22.5.27 Color4 OpenTK.Graphics.Color4.DarkKhaki [static, get]

Gets the system color with (R, G, B, A) = (189, 183, 107, 255).

5.22.5.28 Color4 OpenTK.Graphics.Color4.DarkMagenta [static, get]

Gets the system color with (R, G, B, A) = (139, 0, 139, 255).

5.22.5.29 Color4 OpenTK.Graphics.Color4.DarkOliveGreen [static, get]

Gets the system color with (R, G, B, A) = (85, 107, 47, 255).

5.22.5.30 Color4 OpenTK.Graphics.Color4.DarkOrange [static, get]

Gets the system color with (R, G, B, A) = (255, 140, 0, 255).

5.22.5.31 Color4 OpenTK.Graphics.Color4.DarkOrchid [static, get]

Gets the system color with (R, G, B, A) = (153, 50, 204, 255).

5.22.5.32 Color4 OpenTK.Graphics.Color4.DarkRed [static, get]

Gets the system color with (R, G, B, A) = (139, 0, 0, 255).

5.22.5.33 Color4 OpenTK.Graphics.Color4.DarkSalmon [static, get]

Gets the system color with (R, G, B, A) = (233, 150, 122, 255).

5.22.5.34 Color4 OpenTK.Graphics.Color4.DarkSeaGreen [static, get]

Gets the system color with (R, G, B, A) = (143, 188, 139, 255).

5.22.5.35 Color4 OpenTK.Graphics.Color4.DarkSlateBlue [static, get]

Gets the system color with (R, G, B, A) = (72, 61, 139, 255).

5.22.5.36 Color4 OpenTK.Graphics.Color4.DarkSlateGray [static, get]

Gets the system color with (R, G, B, A) = (47, 79, 79, 255).

5.22.5.37 Color4 OpenTK.Graphics.Color4.DarkTurquoise [static, get]

Gets the system color with (R, G, B, A) = (0, 206, 209, 255).

5.22.5.38 Color4 OpenTK.Graphics.Color4.DarkViolet [static, get]

Gets the system color with (R, G, B, A) = (148, 0, 211, 255).

5.22.5.39 Color4 OpenTK.Graphics.Color4.DeepPink [static, get]

Gets the system color with (R, G, B, A) = (255, 20, 147, 255).

5.22.5.40 Color4 OpenTK.Graphics.Color4.DeepSkyBlue [static, get]

Gets the system color with (R, G, B, A) = (0, 191, 255, 255).

5.22.5.41 Color4 OpenTK.Graphics.Color4.DimGray [static, get]

Gets the system color with (R, G, B, A) = (105, 105, 105, 255).

5.22.5.42 Color4 OpenTK.Graphics.Color4.DodgerBlue [static, get]

Gets the system color with (R, G, B, A) = (30, 144, 255, 255).

5.22.5.43 Color4 OpenTK.Graphics.Color4.Firebrick [static, get]

Gets the system color with (R, G, B, A) = (178, 34, 34, 255).

5.22.5.44 Color4 OpenTK.Graphics.Color4.FloralWhite [static, get]

Gets the system color with (R, G, B, A) = (255, 250, 240, 255).

5.22.5.45 Color4 OpenTK.Graphics.Color4.ForestGreen [static, get]

Gets the system color with (R, G, B, A) = (34, 139, 34, 255).

5.22.5.46 Color4 OpenTK.Graphics.Color4.Fuchsia [static, get]

Gets the system color with (R, G, B, A) = (255, 0, 255, 255).

5.22.5.47 Color4 OpenTK.Graphics.Color4.Gainsboro [static, get]

Gets the system color with (R, G, B, A) = (220, 220, 220, 255).

5.22.5.48 Color4 OpenTK.Graphics.Color4.GhostWhite [static, get]

Gets the system color with (R, G, B, A) = (248, 248, 255, 255).

5.22.5.49 Color4 OpenTK.Graphics.Color4.Gold [static, get]

Gets the system color with (R, G, B, A) = (255, 215, 0, 255).

5.22.5.50 Color4 OpenTK.Graphics.Color4.Goldenrod [static, get]

Gets the system color with (R, G, B, A) = (218, 165, 32, 255).

5.22.5.51 Color4 OpenTK.Graphics.Color4.Gray [static, get]

Gets the system color with (R, G, B, A) = (128, 128, 128, 255).

5.22.5.52 Color4 OpenTK.Graphics.Color4.Green [static, get]

Gets the system color with (R, G, B, A) = (0, 128, 0, 255).

5.22.5.53 Color4 OpenTK.Graphics.Color4.GreenYellow [static, get]

Gets the system color with (R, G, B, A) = (173, 255, 47, 255).

5.22.5.54 Color4 OpenTK.Graphics.Color4.Honeydew [static, get]

Gets the system color with (R, G, B, A) = (240, 255, 240, 255).

5.22.5.55 Color4 OpenTK.Graphics.Color4.HotPink [static, get]

Gets the system color with (R, G, B, A) = (255, 105, 180, 255).

5.22.5.56 Color4 OpenTK.Graphics.Color4.IndianRed [static, get]

Gets the system color with (R, G, B, A) = (205, 92, 92, 255).

5.22.5.57 Color4 OpenTK.Graphics.Color4.Indigo [static, get]

Gets the system color with (R, G, B, A) = (75, 0, 130, 255).

5.22.5.58 Color4 OpenTK.Graphics.Color4.Ivory [static, get]

Gets the system color with (R, G, B, A) = (255, 255, 240, 255).

5.22.5.59 Color4 OpenTK.Graphics.Color4.Khaki [static, get]

Gets the system color with (R, G, B, A) = (240, 230, 140, 255).

5.22.5.60 Color4 OpenTK.Graphics.Color4.Lavender [static, get]

Gets the system color with (R, G, B, A) = (230, 230, 250, 255).

5.22.5.61 Color4 OpenTK.Graphics.Color4.LavenderBlush [static, get]

Gets the system color with (R, G, B, A) = (255, 240, 245, 255).

5.22.5.62 Color4 OpenTK.Graphics.Color4.LawnGreen [static, get]

Gets the system color with (R, G, B, A) = (124, 252, 0, 255).

5.22.5.63 Color4 OpenTK.Graphics.Color4.LemonChiffon [static, get]

Gets the system color with (R, G, B, A) = (255, 250, 205, 255).

5.22.5.64 Color4 OpenTK.Graphics.Color4.LightBlue [static, get]

Gets the system color with (R, G, B, A) = (173, 216, 230, 255).

5.22.5.65 Color4 OpenTK.Graphics.Color4.LightCoral [static, get]

Gets the system color with (R, G, B, A) = (240, 128, 128, 255).

5.22.5.66 Color4 OpenTK.Graphics.Color4.LightCyan [static, get]

Gets the system color with (R, G, B, A) = (224, 255, 255, 255).

5.22.5.67 Color4 OpenTK.Graphics.Color4.LightGoldenrodYellow [static, get]

Gets the system color with (R, G, B, A) = (250, 250, 210, 255).

5.22.5.68 Color4 OpenTK.Graphics.Color4.LightGray [static, get]

Gets the system color with (R, G, B, A) = (211, 211, 211, 255).

5.22.5.69 Color4 OpenTK.Graphics.Color4.LightGreen [static, get]

Gets the system color with (R, G, B, A) = (144, 238, 144, 255).

5.22.5.70 Color4 OpenTK.Graphics.Color4.LightPink [static, get]

Gets the system color with (R, G, B, A) = (255, 182, 193, 255).

5.22.5.71 Color4 OpenTK.Graphics.Color4.LightSalmon [static, get]

Gets the system color with (R, G, B, A) = (255, 160, 122, 255).

5.22.5.72 Color4 OpenTK.Graphics.Color4.LightSeaGreen [static, get]

Gets the system color with (R, G, B, A) = (32, 178, 170, 255).

5.22.5.73 Color4 OpenTK.Graphics.Color4.LightSkyBlue [static, get]

Gets the system color with (R, G, B, A) = (135, 206, 250, 255).

5.22.5.74 Color4 OpenTK.Graphics.Color4.LightSlateGray [static, get]

Gets the system color with (R, G, B, A) = (119, 136, 153, 255).

5.22.5.75 Color4 OpenTK.Graphics.Color4.LightSteelBlue [static, get]

Gets the system color with (R, G, B, A) = (176, 196, 222, 255).

5.22.5.76 Color4 OpenTK.Graphics.Color4.LightYellow [static, get]

Gets the system color with (R, G, B, A) = (255, 255, 224, 255).

5.22.5.77 Color4 OpenTK.Graphics.Color4.Lime [static, get]

Gets the system color with (R, G, B, A) = (0, 255, 0, 255).

5.22.5.78 Color4 OpenTK.Graphics.Color4.LimeGreen [static, get]

Gets the system color with (R, G, B, A) = (50, 205, 50, 255).

5.22.5.79 Color4 OpenTK.Graphics.Color4.Linen [static, get]

Gets the system color with (R, G, B, A) = (250, 240, 230, 255).

5.22.5.80 Color4 OpenTK.Graphics.Color4.Magenta [static, get]

Gets the system color with (R, G, B, A) = (255, 0, 255, 255).

5.22.5.81 Color4 OpenTK.Graphics.Color4.Maroon [static, get]

Gets the system color with (R, G, B, A) = (128, 0, 0, 255).

5.22.5.82 Color4 OpenTK.Graphics.Color4.MediumAquamarine [static, get]

Gets the system color with (R, G, B, A) = (102, 205, 170, 255).

5.22.5.83 Color4 OpenTK.Graphics.Color4.MediumBlue [static, get]

Gets the system color with (R, G, B, A) = (0, 0, 205, 255).

5.22.5.84 Color4 OpenTK.Graphics.Color4.MediumOrchid [static, get]

Gets the system color with (R, G, B, A) = (186, 85, 211, 255).

5.22.5.85 Color4 OpenTK.Graphics.Color4.MediumPurple [static, get]

Gets the system color with (R, G, B, A) = (147, 112, 219, 255).

5.22.5.86 Color4 OpenTK.Graphics.Color4.MediumSeaGreen [static, get]

Gets the system color with (R, G, B, A) = (60, 179, 113, 255).

5.22.5.87 Color4 OpenTK.Graphics.Color4.MediumSlateBlue [static, get]

Gets the system color with (R, G, B, A) = (123, 104, 238, 255).

5.22.5.88 Color4 OpenTK.Graphics.Color4.MediumSpringGreen [static, get]

Gets the system color with (R, G, B, A) = (0, 250, 154, 255).

5.22.5.89 Color4 OpenTK.Graphics.Color4.MediumTurquoise [static, get]

Gets the system color with (R, G, B, A) = (72, 209, 204, 255).

5.22.5.90 Color4 OpenTK.Graphics.Color4.MediumVioletRed [static, get]

Gets the system color with (R, G, B, A) = (199, 21, 133, 255).

5.22.5.91 Color4 OpenTK.Graphics.Color4.MidnightBlue [static, get]

Gets the system color with (R, G, B, A) = (25, 25, 112, 255).

5.22.5.92 Color4 OpenTK.Graphics.Color4.MintCream [static, get]

Gets the system color with (R, G, B, A) = (245, 255, 250, 255).

5.22.5.93 Color4 OpenTK.Graphics.Color4.MistyRose [static, get]

Gets the system color with (R, G, B, A) = (255, 228, 225, 255).

5.22.5.94 Color4 OpenTK.Graphics.Color4.Moccasin [static, get]

Gets the system color with (R, G, B, A) = (255, 228, 181, 255).

5.22.5.95 Color4 OpenTK.Graphics.Color4.NavajoWhite [static, get]

Gets the system color with (R, G, B, A) = (255, 222, 173, 255).

5.22.5.96 Color4 OpenTK.Graphics.Color4.Navy [static, get]

Gets the system color with (R, G, B, A) = (0, 0, 128, 255).

5.22.5.97 Color4 OpenTK.Graphics.Color4.OldLace [static, get]

Gets the system color with (R, G, B, A) = (253, 245, 230, 255).

5.22.5.98 Color4 OpenTK.Graphics.Color4.Olive [static, get]

Gets the system color with (R, G, B, A) = (128, 128, 0, 255).

5.22.5.99 Color4 OpenTK.Graphics.Color4.OliveDrab [static, get]

Gets the system color with (R, G, B, A) = (107, 142, 35, 255).

5.22.5.100 Color4 OpenTK.Graphics.Color4.Orange [static, get]

Gets the system color with (R, G, B, A) = (255, 165, 0, 255).

5.22.5.101 Color4 OpenTK.Graphics.Color4.OrangeRed [static, get]

Gets the system color with (R, G, B, A) = (255, 69, 0, 255).

5.22.5.102 Color4 OpenTK.Graphics.Color4.Orchid [static, get]

Gets the system color with (R, G, B, A) = (218, 112, 214, 255).

5.22.5.103 Color4 OpenTK.Graphics.Color4.PaleGoldenrod [static, get]

Gets the system color with (R, G, B, A) = (238, 232, 170, 255).

5.22.5.104 Color4 OpenTK.Graphics.Color4.PaleGreen [static, get]

Gets the system color with (R, G, B, A) = (152, 251, 152, 255).

5.22.5.105 Color4 OpenTK.Graphics.Color4.PaleTurquoise [static, get]

Gets the system color with (R, G, B, A) = (175, 238, 238, 255).

5.22.5.106 Color4 OpenTK.Graphics.Color4.PaleVioletRed [static, get]

Gets the system color with (R, G, B, A) = (219, 112, 147, 255).

5.22.5.107 Color4 OpenTK.Graphics.Color4.PapayaWhip [static, get]

Gets the system color with (R, G, B, A) = (255, 239, 213, 255).

5.22.5.108 Color4 OpenTK.Graphics.Color4.PeachPuff [static, get]

Gets the system color with (R, G, B, A) = (255, 218, 185, 255).

5.22.5.109 Color4 OpenTK.Graphics.Color4.Peru [static, get]

Gets the system color with (R, G, B, A) = (205, 133, 63, 255).

5.22.5.110 Color4 OpenTK.Graphics.Color4.Pink [static, get]

Gets the system color with (R, G, B, A) = (255, 192, 203, 255).

5.22.5.111 Color4 OpenTK.Graphics.Color4.Plum [static, get]

Gets the system color with (R, G, B, A) = (221, 160, 221, 255).

5.22.5.112 Color4 OpenTK.Graphics.Color4.PowderBlue [static, get]

Gets the system color with (R, G, B, A) = (176, 224, 230, 255).

5.22.5.113 Color4 OpenTK.Graphics.Color4.Purple [static, get]

Gets the system color with (R, G, B, A) = (128, 0, 128, 255).

5.22.5.114 Color4 OpenTK.Graphics.Color4.Red [static, get]

Gets the system color with (R, G, B, A) = (255, 0, 0, 255).

5.22.5.115 Color4 OpenTK.Graphics.Color4.RosyBrown [static, get]

Gets the system color with (R, G, B, A) = (188, 143, 143, 255).

5.22.5.116 Color4 OpenTK.Graphics.Color4.RoyalBlue [static, get]

Gets the system color with (R, G, B, A) = (65, 105, 225, 255).

5.22.5.117 Color4 OpenTK.Graphics.Color4.SaddleBrown [static, get]

Gets the system color with (R, G, B, A) = (139, 69, 19, 255).

5.22.5.118 Color4 OpenTK.Graphics.Color4.Salmon [static, get]

Gets the system color with (R, G, B, A) = (250, 128, 114, 255).

5.22.5.119 Color4 OpenTK.Graphics.Color4.SandyBrown [static, get]

Gets the system color with (R, G, B, A) = (244, 164, 96, 255).

5.22.5.120 Color4 OpenTK.Graphics.Color4.SeaGreen [static, get]

Gets the system color with (R, G, B, A) = (46, 139, 87, 255).

5.22.5.121 Color4 OpenTK.Graphics.Color4.SeaShell [static, get]

Gets the system color with (R, G, B, A) = (255, 245, 238, 255).

5.22.5.122 Color4 OpenTK.Graphics.Color4.Sienna [static, get]

Gets the system color with (R, G, B, A) = (160, 82, 45, 255).

5.22.5.123 Color4 OpenTK.Graphics.Color4.Silver [static, get]

Gets the system color with (R, G, B, A) = (192, 192, 192, 255).

5.22.5.124 Color4 OpenTK.Graphics.Color4.SkyBlue [static, get]

Gets the system color with (R, G, B, A) = (135, 206, 235, 255).

5.22.5.125 Color4 OpenTK.Graphics.Color4.SlateBlue [static, get]

Gets the system color with (R, G, B, A) = (106, 90, 205, 255).

5.22.5.126 Color4 OpenTK.Graphics.Color4.SlateGray [static, get]

Gets the system color with (R, G, B, A) = (112, 128, 144, 255).

5.22.5.127 Color4 OpenTK.Graphics.Color4.Snow [static, get]

Gets the system color with (R, G, B, A) = (255, 250, 250, 255).

5.22.5.128 Color4 OpenTK.Graphics.Color4.SpringGreen [static, get]

Gets the system color with (R, G, B, A) = (0, 255, 127, 255).

5.22.5.129 Color4 OpenTK.Graphics.Color4.SteelBlue [static, get]

Gets the system color with (R, G, B, A) = (70, 130, 180, 255).

5.22.5.130 Color4 OpenTK.Graphics.Color4.Tan [static, get]

Gets the system color with (R, G, B, A) = (210, 180, 140, 255).

5.22.5.131 Color4 OpenTK.Graphics.Color4.Teal [static, get]

Gets the system color with (R, G, B, A) = (0, 128, 128, 255).

5.22.5.132 Color4 OpenTK.Graphics.Color4.Thistle [static, get]

Gets the system color with (R, G, B, A) = (216, 191, 216, 255).

5.22.5.133 Color4 OpenTK.Graphics.Color4.Tomato [static, get]

Gets the system color with (R, G, B, A) = (255, 99, 71, 255).

5.22.5.134 Color4 OpenTK.Graphics.Color4.Transparent [static, get]

Gets the system color with (R, G, B, A) = (255, 255, 255, 0).

5.22.5.135 Color4 OpenTK.Graphics.Color4.Turquoise [static, get]

Gets the system color with (R, G, B, A) = (64, 224, 208, 255).

5.22.5.136 Color4 OpenTK.Graphics.Color4.Violet [static, get]

Gets the system color with (R, G, B, A) = (238, 130, 238, 255).

5.22.5.137 Color4 OpenTK.Graphics.Color4.Wheat [static, get]

Gets the system color with (R, G, B, A) = (245, 222, 179, 255).

5.22.5.138 Color4 OpenTK.Graphics.Color4.White [static, get]

Gets the system color with (R, G, B, A) = (255, 255, 255, 255).

5.22.5.139 Color4 OpenTK.Graphics.Color4.WhiteSmoke [static, get]

Gets the system color with (R, G, B, A) = (245, 245, 245, 255).

5.22.5.140 Color4 OpenTK.Graphics.Color4.Yellow [static, get]

Gets the system color with (R, G, B, A) = (255, 255, 0, 255).

5.22.5.141 Color4 OpenTK.Graphics.Color4.YellowGreen [static, get]

Gets the system color with (R, G, B, A) = (154, 205, 50, 255).

5.23 OpenTK.Graphics.ColorFormat Struct Reference

Defines the [ColorFormat](#) component of a [GraphicsMode](#).

Public Member Functions

- [ColorFormat](#) (int bpp)
Constructs a new [ColorFormat](#) with the specified aggregate bits per pixel.
- [ColorFormat](#) (int red, int green, int blue, int alpha)
Constructs a new [ColorFormat](#) with the specified bits per pixel for the Red, Green, Blue and Alpha color channels.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- override int [GetHashCode](#) ()
Returns the hash code for this instance.
- override string [ToString](#) ()

Returns a System.String that describes this instance.

Static Public Member Functions

- static implicit [operator ColorFormat](#) (int bpp)
Converts the specified bpp into a new [ColorFormat](#).
- static bool [operator==](#) ([ColorFormat](#) left, [ColorFormat](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([ColorFormat](#) left, [ColorFormat](#) right)
Compares two instances for inequality.

Public Attributes

- byte **red**
- byte **green**
- byte **blue**
- byte **alpha**
- bool **isIndexed**
- int **bitsPerPixel**

Properties

- int [Red](#) [get, set]
Gets the bits per pixel for the Red channel.
- int [Green](#) [get, set]
Gets the bits per pixel for the Green channel.
- int [Blue](#) [get, set]
Gets the bits per pixel for the Blue channel.
- int [Alpha](#) [get, set]
Gets the bits per pixel for the Alpha channel.
- bool [IsIndexed](#) [get, set]
Gets a System.Boolean indicating whether this [ColorFormat](#) is indexed.

- int [BitsPerPixel](#) [get, set]
Gets the sum of Red, Green, Blue and Alpha bits per pixel.

5.23.1 Detailed Description

Defines the [ColorFormat](#) component of a [GraphicsMode](#). A [ColorFormat](#) contains Red, Green, Blue and Alpha components that describe the allocated bits per pixel for the corresponding color.

5.23.2 Constructor & Destructor Documentation

5.23.2.1 OpenTK.Graphics.ColorFormat.ColorFormat (int *bpp*)

Constructs a new [ColorFormat](#) with the specified aggregate bits per pixel.

Parameters

bpp The bits per pixel sum for the Red, Green, Blue and Alpha color channels.

5.23.2.2 OpenTK.Graphics.ColorFormat.ColorFormat (int *red*, int *green*, int *blue*, int *alpha*)

Constructs a new [ColorFormat](#) with the specified bits per pixel for the Red, Green, Blue and Alpha color channels.

Parameters

red Bits per pixel for the Red color channel.

green Bits per pixel for the Green color channel.

blue Bits per pixel for the Blue color channel.

alpha Bits per pixel for the Alpha color channel.

5.23.3 Member Function Documentation

5.23.3.1 override bool OpenTK.Graphics.ColorFormat.Equals (object *obj*)

Indicates whether this instance and a specified object are equal.

Parameters

obj Another object to compare to.

Returns

True if this instance is equal to obj; false otherwise.

5.23.3.2 override int OpenTK.Graphics.ColorFormat.GetHashCode ()

Returns the hash code for this instance.

Returns

A System.Int32 with the hash code of this instance.

5.23.3.3 static implicit OpenTK.Graphics.ColorFormat.operator ColorFormat (int *bpp*) [static]

Converts the specified bpp into a new [ColorFormat](#).

Parameters

bpp The bits per pixel to convert.

Returns

A [ColorFormat](#) with the specified bits per pixel.

5.23.3.4 static bool OpenTK.Graphics.ColorFormat.operator!= (ColorFormat *left*, ColorFormat *right*) [static]

Compares two instances for inequality.

Parameters

left The left operand.

right The right operand.

Returns

True if both instances are not equal; false otherwise.

5.23.3.5 static bool OpenTK.Graphics.ColorFormat.operator==(ColorFormat *left*, ColorFormat *right*) [static]

Compares two instances for equality.

Parameters

left The left operand.

right The right operand.

Returns

True if both instances are equal; false otherwise.

5.23.3.6 override string OpenTK.Graphics.ColorFormat.ToString ()

Returns a System.String that describes this instance.

Returns

A System.String that describes this instance.

5.23.4 Property Documentation

5.23.4.1 int OpenTK.Graphics.ColorFormat.Alpha [get, set]

Gets the bits per pixel for the Alpha channel.

5.23.4.2 int OpenTK.Graphics.ColorFormat.BitsPerPixel [get, set]

Gets the sum of Red, Green, Blue and Alpha bits per pixel.

5.23.4.3 int OpenTK.Graphics.ColorFormat.Blue [get, set]

Gets the bits per pixel for the Blue channel.

5.23.4.4 int OpenTK.Graphics.ColorFormat.Green [get, set]

Gets the bits per pixel for the Green channel.

5.23.4.5 bool OpenTK.Graphics.ColorFormat.IsIndexed [get, set]

Gets a System.Boolean indicating whether this [ColorFormat](#) is indexed.

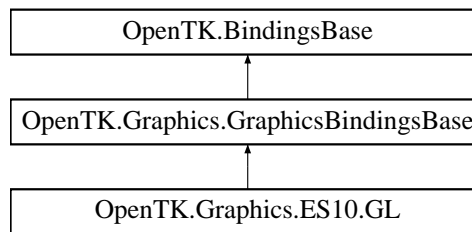
5.23.4.6 int OpenTK.Graphics.ColorFormat.Red [get, set]

Gets the bits per pixel for the Red channel.

5.24 OpenTK.Graphics.ES10.GL Class Reference

Provides access to [OpenGL](#) ES 1.0 methods.

Inheritance diagram for OpenTK.Graphics.ES10.GL:



Static Public Member Functions

- static void [ActiveTexture](#) (OpenTK.Graphics.ES10.All texture)
Select active texture unit.
- static void [AlphaFunc](#) (OpenTK.Graphics.ES10.All func, Single @ref)
Specify the alpha test function.
- static void **AlphaFuncx** (OpenTK.Graphics.ES10.All func, int @ref)
- static void [BindTexture](#) (OpenTK.Graphics.ES10.All target, Int32 texture)
Bind a named texture to a texturing target.
- static void [BindTexture](#) (OpenTK.Graphics.ES10.All target, UInt32 texture)
Bind a named texture to a texturing target.
- static void [BlendFunc](#) (OpenTK.Graphics.ES10.All sfactor, OpenTK.Graphics.ES10.All dfactor)
Specify pixel arithmetic.
- static void [Clear](#) (Int32 mask)
Clear buffers to preset values.
- static void [Clear](#) (UInt32 mask)

Clear buffers to preset values.

- static void **ClearColor** (Single red, Single green, Single blue, Single alpha)
Specify clear values for the color buffers.
- static void **ClearColorx** (int red, int green, int blue, int alpha)
- static void **ClearDepth** (Single depth)
Specify the clear value for the depth buffer.
- static void **ClearDepthx** (int depth)
- static void **ClearStencil** (Int32 s)
Specify the clear value for the stencil buffer.
- static void **ClientActiveTexture** (OpenTK.Graphics.ES10.All texture)
Select active texture unit.
- static void **Color4** (Single red, Single green, Single blue, Single alpha)
Set the current color.
- static void **Color4x** (int red, int green, int blue, int alpha)
- static void **ColorMask** (bool red, bool green, bool blue, bool alpha)
Enable and disable writing of frame buffer color components.
- static void **ColorPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)
Define an array of colors.
- static void **ColorPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of colors.
- static void **ColorPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of colors.
- static void **ColorPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)
Define an array of colors.
- static void **ColorPointer** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer)
Define an array of colors.

- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] ref T7 data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[,] data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[,] data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[] data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D](#) (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, Int32 imageSize,[InAttribute, OutAttribute] ref T8 data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, Int32 imageSize,[InAttribute, OutAttribute] T8[,] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, Int32 imageSize,[InAttribute, OutAttribute] T8[,] data)
Specify a two-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, Int32 imageSize, [InAttribute, OutAttribute] T8[] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, Int32 imageSize, IntPtr data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CopyTexImage2D](#) (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 x, Int32 y, Int32 width, Int32 height, Int32 border)
Copy pixels into a 2D texture image.
- static void [CopyTexSubImage2D](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height)
Copy a two-dimensional texture subimage.
- static void [CullFace](#) (OpenTK.Graphics.ES10.All mode)
Specify whether front- or back-facing facets can be culled.
- static unsafe void [DeleteTextures](#) (Int32 n, Int32 *textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, Int32[] textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, ref Int32 textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, ref UInt32 textures)
Delete named textures.
- static unsafe void [DeleteTextures](#) (Int32 n, UInt32 *textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, UInt32[] textures)
Delete named textures.
- static void [DepthFunc](#) (OpenTK.Graphics.ES10.All func)
Specify the value used for depth buffer comparisons.

- static void [DepthMask](#) (bool flag)
Enable or disable writing into the depth buffer.
- static void [DepthRange](#) (Single zNear, Single zFar)
Specify mapping of depth values from normalized device coordinates to window coordinates.
- static void **DepthRangex** (int zNear, int zFar)
- static void **Disable** (OpenTK.Graphics.ES10.All cap)
- static void **DisableClientState** (OpenTK.Graphics.ES10.All array)
- static void [DrawArrays](#) (OpenTK.Graphics.ES10.All mode, Int32 first, Int32 count)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] ref T3 indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T3[,] indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T3[,] indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T3[] indices)
Render primitives from array data.
- static void [DrawElements](#) (OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type, IntPtr indices)
Render primitives from array data.
- static void [Enable](#) (OpenTK.Graphics.ES10.All cap)
Enable or disable server-side GL capabilities.
- static void [EnableClientState](#) (OpenTK.Graphics.ES10.All array)

Enable or disable client-side capability.

- static void **Finish** ()
*Block until all **GL** execution is complete.*
- static void **Flush** ()
*Force execution of **GL** commands in finite time.*
- static void **Fog** (OpenTK.Graphics.ES10.All pname, Single param)
Specify fog parameters.
- static unsafe void **Fog** (OpenTK.Graphics.ES10.All pname, Single *@params)
Specify fog parameters.
- static void **Fog** (OpenTK.Graphics.ES10.All pname, Single[] @params)
Specify fog parameters.
- static void **Fogx** (OpenTK.Graphics.ES10.All pname, int param)
- static unsafe void **Fogx** (OpenTK.Graphics.ES10.All pname, int *@params)
- static void **Fogx** (OpenTK.Graphics.ES10.All pname, int[] @params)
- static void **FrontFace** (OpenTK.Graphics.ES10.All mode)
Define front- and back-facing polygons.
- static void **Frustum** (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar)
Multiply the current matrix by a perspective matrix.
- static void **Frustumx** (int left, int right, int bottom, int top, int zNear, int zFar)
- static unsafe void **GenTextures** (Int32 n, Int32 *textures)
Generate texture names.
- static void **GenTextures** (Int32 n, Int32[] textures)
Generate texture names.
- static void **GenTextures** (Int32 n, ref Int32 textures)
Generate texture names.
- static void **GenTextures** (Int32 n, ref UInt32 textures)
Generate texture names.
- static unsafe void **GenTextures** (Int32 n, UInt32 *textures)
Generate texture names.

- static void [GenTextures](#) (Int32 n, UInt32[] textures)
Generate texture names.
- static OpenTK.Graphics.ES10.All [GetError](#) ()
Return error information.
- static unsafe void **GetInteger** (OpenTK.Graphics.ES10.All pname, Int32 *@params)
- static void **GetInteger** (OpenTK.Graphics.ES10.All pname, Int32[] @params)
- static void **GetInteger** (OpenTK.Graphics.ES10.All pname, ref Int32 @params)
- static unsafe System.String [GetString](#) (OpenTK.Graphics.ES10.All name)
Return a string describing the current [GL](#) connection.
- static void [Hint](#) (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All mode)
Specify implementation-specific hints.
- static void [Light](#) (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, Single param)
Set light source parameters.
- static unsafe void [Light](#) (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, Single *@params)
Set light source parameters.
- static void [Light](#) (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, Single[] @params)
Set light source parameters.
- static void [LightModel](#) (OpenTK.Graphics.ES10.All pname, Single param)
Set the lighting model parameters.
- static unsafe void [LightModel](#) (OpenTK.Graphics.ES10.All pname, Single *@params)
Set the lighting model parameters.
- static void [LightModel](#) (OpenTK.Graphics.ES10.All pname, Single[] @params)
Set the lighting model parameters.
- static void **LightModelx** (OpenTK.Graphics.ES10.All pname, int param)

- static unsafe void **LightModelx** (OpenTK.Graphics.ES10.All pname, int *@params)
- static void **LightModelx** (OpenTK.Graphics.ES10.All pname, int[] @params)
- static void **Lightx** (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, int param)
- static unsafe void **Lightx** (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, int *@params)
- static void **Lightx** (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, int[] @params)
- static void **LineWidth** (Single width)
Specify the width of rasterized lines.
- static void **LineWidthx** (int width)
- static void **LoadIdentity** ()
Replace the current matrix with the identity matrix.
- static void **LoadMatrix** (ref Single m)
Replace the current matrix with the specified matrix.
- static unsafe void **LoadMatrix** (Single *m)
Replace the current matrix with the specified matrix.
- static void **LoadMatrix** (Single[] m)
Replace the current matrix with the specified matrix.
- static unsafe void **LoadMatrixx** (int *m)
- static void **LoadMatrixx** (int[] m)
- static void **LoadMatrixx** (ref int m)
- static void **LogicOp** (OpenTK.Graphics.ES10.All opcode)
Specify a logical pixel operation for color index rendering.
- static void **Material** (OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All pname, Single param)
Specify material parameters for the lighting model.
- static unsafe void **Material** (OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All pname, Single *@params)
Specify material parameters for the lighting model.
- static void **Material** (OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All pname, Single[] @params)
Specify material parameters for the lighting model.

- static void **Materialx** (OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All pname, int param)
- static unsafe void **Materialx** (OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All pname, int *@params)
- static void **Materialx** (OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All pname, int[] @params)
- static void **MatrixMode** (OpenTK.Graphics.ES10.All mode)

Specify which matrix is the current matrix.

- static void **MultiTexCoord4** (OpenTK.Graphics.ES10.All target, Single s, Single t, Single r, Single q)

Set the current texture coordinates.

- static void **MultiTexCoord4x** (OpenTK.Graphics.ES10.All target, int s, int t, int r, int q)
- static void **MultMatrix** (ref Single m)

Multiply the current matrix with the specified matrix.

- static unsafe void **MultMatrix** (Single *m)

Multiply the current matrix with the specified matrix.

- static void **MultMatrix** (Single[] m)

Multiply the current matrix with the specified matrix.

- static unsafe void **MultMatrixx** (int *m)
- static void **MultMatrixx** (int[] m)
- static void **MultMatrixx** (ref int m)
- static void **Normal3** (Single nx, Single ny, Single nz)

Set the current normal vector.

- static void **Normal3x** (int nx, int ny, int nz)
- static void **NormalPointer**< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] ref T2 pointer)

Define an array of normals.

- static void **NormalPointer**< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T2[,], pointer)

Define an array of normals.

- static void **NormalPointer**< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T2[,], pointer)

Define an array of normals.

- static void [NormalPointer](#)< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T2[] pointer)
Define an array of normals.
- static void [NormalPointer](#) (OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer)
Define an array of normals.
- static void [Ortho](#) (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar)
Multiply the current matrix with an orthographic matrix.
- static void **Orthox** (int left, int right, int bottom, int top, int zNear, int zFar)
- static void [PixelStore](#) (OpenTK.Graphics.ES10.All pname, Int32 param)
Set pixel storage modes.
- static void [PointSize](#) (Single size)
Specify the diameter of rasterized points.
- static void **PointSizex** (int size)
- static void [PolygonOffset](#) (Single factor, Single units)
Set the scale and units used to calculate depth values.
- static void **PolygonOffsetx** (int factor, int units)
- static void **PopMatrix** ()
- static void [PushMatrix](#) ()
Push and pop the current matrix stack.
- static void [ReadPixels](#)< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] ref T6 pixels)
Read a block of pixels from the frame buffer.
- static void [ReadPixels](#)< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T6[,], pixels)
Read a block of pixels from the frame buffer.
- static void [ReadPixels](#)< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T6[,], pixels)
Read a block of pixels from the frame buffer.

- static void [ReadPixels< T6 >](#) (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T6[] pixels)
Read a block of pixels from the frame buffer.
- static void [ReadPixels](#) (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, IntPtr pixels)
Read a block of pixels from the frame buffer.
- static void [Rotate](#) (Single angle, Single x, Single y, Single z)
Multiply the current matrix by a rotation matrix.
- static void **Rotatex** (int angle, int x, int y, int z)
- static void [SampleCoverage](#) (Single value, bool invert)
Specify multisample coverage parameters.
- static void **SampleCoveragex** (int value, bool invert)
- static void [Scale](#) (Single x, Single y, Single z)
Multiply the current matrix by a general scaling matrix.
- static void **Scalex** (int x, int y, int z)
- static void [Scissor](#) (Int32 x, Int32 y, Int32 width, Int32 height)
Define the scissor box.
- static void [ShadeModel](#) (OpenTK.Graphics.ES10.All mode)
Select flat or smooth shading.
- static void [StencilFunc](#) (OpenTK.Graphics.ES10.All func, Int32 @ref, Int32 mask)
Set front and back function and reference value for stencil testing.
- static void [StencilFunc](#) (OpenTK.Graphics.ES10.All func, Int32 @ref, UInt32 mask)
Set front and back function and reference value for stencil testing.
- static void [StencilMask](#) (Int32 mask)
Control the front and back writing of individual bits in the stencil planes.
- static void [StencilMask](#) (UInt32 mask)
Control the front and back writing of individual bits in the stencil planes.

- static void **StencilOp** (OpenTK.Graphics.ES10.All fail, OpenTK.Graphics.ES10.All zfail, OpenTK.Graphics.ES10.All zpass)
Set front and back stencil test actions.
- static void **TexCoordPointer< T3 >** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)
Define an array of texture coordinates.
- static void **TexCoordPointer< T3 >** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of texture coordinates.
- static void **TexCoordPointer< T3 >** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of texture coordinates.
- static void **TexCoordPointer< T3 >** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)
Define an array of texture coordinates.
- static void **TexCoordPointer** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer)
Define an array of texture coordinates.
- static void **TexEnv** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, Single param)
Set texture environment parameters.
- static unsafe void **TexEnv** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, Single *@params)
Set texture environment parameters.
- static void **TexEnv** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, Single[]@params)
Set texture environment parameters.
- static void **TexEnvx** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, int param)
- static unsafe void **TexEnvx** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, int *@params)
- static void **TexEnvx** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, int[]@params)

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#) (OpenTK.Graphics.ES10.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, IntPtr pixels)

Specify a two-dimensional texture image.

- static void [TexParameter](#) (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, Single param)

Set texture parameters.

- static void **TexParameterx** (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, int param)

- static void [TexSubImage2D](#)< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D](#)< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32

height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture subimage.

- static void **TexSubImage2D**< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture subimage.

- static void **TexSubImage2D**< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type,[InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture subimage.

- static void **TexSubImage2D** (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, IntPtr pixels)

Specify a two-dimensional texture subimage.

- static void **Translate** (Single x, Single y, Single z)

Multiply the current matrix by a translation matrix.

- static void **Translatex** (int x, int y, int z)
- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)

Define an array of vertex data.

- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)

Define an array of vertex data.

- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)

Define an array of vertex data.

- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)

Define an array of vertex data.

- static void **VertexPointer** (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer)

Define an array of vertex data.

- static void [Viewport](#) (Int32 x, Int32 y, Int32 width, Int32 height)

Set the viewport.

Properties

- override object [SyncRoot](#) [get]

Returns a synchronization token unique for the [GL](#) class.

5.24.1 Detailed Description

Provides access to [OpenGL](#) ES 1.0 methods.

5.24.2 Member Function Documentation

5.24.2.1 static void [OpenTK.Graphics.ES10.GL.ActiveTexture](#) ([OpenTK.Graphics.ES10.All texture](#)) [static]

Select active texture unit.

Parameters

texture Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. texture must be one of GL_TEXTURE, where i ranges from 0 to the larger of (GL_MAX_TEXTURE_COORDS - 1) and (GL_MAX_COMBINED_TEXTURE_IMAGE_UNITS - 1). The initial value is GL_TEXTURE0.

5.24.2.2 static void [OpenTK.Graphics.ES10.GL.AlphaFunc](#) ([OpenTK.Graphics.ES10.All func](#), [Single @ ref](#)) [static]

Specify the alpha test function.

Parameters

func Specifies the alpha comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_ALWAYS.

ref Specifies the reference value that incoming alpha values are compared to. This value is clamped to the range [0,1], where 0 represents the lowest possible alpha value and 1 the highest possible value. The initial reference value is 0.

5.24.2.3 static void OpenTK.Graphics.ES10.GL.BindTexture (
OpenTK.Graphics.ES10.All *target*, Int32 *texture*) [static]

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.24.2.4 static void OpenTK.Graphics.ES10.GL.BindTexture (
OpenTK.Graphics.ES10.All *target*, UInt32 *texture*) [static]

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.24.2.5 static void OpenTK.Graphics.ES10.GL.BlendFunc (
OpenTK.Graphics.ES10.All *sfactor*, OpenTK.Graphics.ES10.All
***dfactor*) [static]**

Specify pixel arithmetic.

Parameters

sfactor Specifies how the red, green, blue, and alpha source blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA,

GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dfactor Specifies how the red, green, blue, and alpha destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

5.24.2.6 **static void OpenTK.Graphics.ES10.GL.Clear (UInt32 *mask*) [static]**

Clear buffers to preset values.

Parameters

mask Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, GL_ACCUM_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

5.24.2.7 **static void OpenTK.Graphics.ES10.GL.Clear (Int32 *mask*) [static]**

Clear buffers to preset values.

Parameters

mask Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, GL_ACCUM_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

5.24.2.8 **static void OpenTK.Graphics.ES10.GL.ClearColor (Single *red*, Single *green*, Single *blue*, Single *alpha*) [static]**

Specify clear values for the color buffers.

Parameters

red Specify the red, green, blue, and alpha values used when the color buffers are cleared. The initial values are all 0.

**5.24.2.9 static void OpenTK.Graphics.ES10.GL.ClearDepth (Single *depth*)
[static]**

Specify the clear value for the depth buffer.

Parameters

depth Specifies the depth value used when the depth buffer is cleared. The initial value is 1.

**5.24.2.10 static void OpenTK.Graphics.ES10.GL.ClearStencil (Int32 *s*)
[static]**

Specify the clear value for the stencil buffer.

Parameters

s Specifies the index used when the stencil buffer is cleared. The initial value is 0.

5.24.2.11 static void OpenTK.Graphics.ES10.GL.ClientActiveTexture (OpenTK.Graphics.ES10.All *texture*) [static]

Select active texture unit.

Parameters

texture Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. *texture* must be one of GL_TEXTURE*i*, where *i* ranges from 0 to the value of GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value. The initial value is GL_TEXTURE0.

5.24.2.12 static void OpenTK.Graphics.ES10.GL.Color4 (Single *red*, Single *green*, Single *blue*, Single *alpha*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.24.2.13 `static void OpenTK.Graphics.ES10.GL.ColorMask (bool red, bool green, bool blue, bool alpha) [static]`

Enable and disable writing of frame buffer color components.

Parameters

red Specify whether red, green, blue, and alpha can or cannot be written into the frame buffer. The initial values are all GL_TRUE, indicating that the color components can be written.

5.24.2.14 `static void OpenTK.Graphics.ES10.GL.ColorPointer (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer) [static]`

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

5.24.2.15 `static void OpenTK.Graphics.ES10.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer) [static]`

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.24.2.16 `static void OpenTK.Graphics.ES10.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.24.2.17 `static void OpenTK.Graphics.ES10.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.24.2.18 `static void OpenTK.Graphics.ES10.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer) [static]`

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.24.2.19 `static void OpenTK.Graphics.ES10.GL.CompressedTexImage2D (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

```
5.24.2.20 static void OpenTK.Graphics.ES10.GL.CompressedTexImage2D<
T7 > ( OpenTK.Graphics.ES10.All target, Int32 level,
OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32
height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
ref T7 data ) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

```
5.24.2.21 static void OpenTK.Graphics.ES10.GL.CompressedTexImage2D<
    T7 > ( OpenTK.Graphics.ES10.All target, Int32 level,
    OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32
    height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
    T7 data[,]) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

```
5.24.2.22 static void OpenTK.Graphics.ES10.GL.CompressedTexImage2D<
    T7 > ( OpenTK.Graphics.ES10.All target, Int32 level,
    OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32
    height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
    T7 data[,]) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

5.24.2.23 `static void OpenTK.Graphics.ES10.GL.CompressedTexImage2D< T7 > (OpenTK.Graphics.ES10.All target, Int32 level, OpenTK.Graphics.ES10.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T7[] data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

5.24.2.24 static void OpenTK.Graphics.ES10.GL.CompressedTexSubImage2D
(OpenTK.Graphics.ES10.All *target*, Int32 *level*, Int32
***xoffset*, Int32 *yoffset*, Int32 *width*, Int32 *height*,**
OpenTK.Graphics.ES10.All *format*, Int32 *imageSize*, IntPtr *data*)
[static]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.24.2.25 static void
OpenTK.Graphics.ES10.GL.CompressedTexSubImage2D< T8 > (
OpenTK.Graphics.ES10.All *target*, Int32 *level*, Int32 *xoffset*, Int32
***yoffset*, Int32 *width*, Int32 *height*, OpenTK.Graphics.ES10.All**
***format*, Int32 *imageSize*, [InAttribute, OutAttribute] ref T8 *data*)**
[static]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.24.2.26 static void

```
OpenTK.Graphics.ES10.GL.CompressedTexSubImage2D< T8 > (
    OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32
    yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All
    format, Int32 imageSize, [InAttribute, OutAttribute] T8 data[,], )
[static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

```
5.24.2.27 static void
OpenTK.Graphics.ES10.GL.CompressedTexSubImage2D< T8 > (
    OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32
    yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All
    format, Int32 imageSize, [InAttribute, OutAttribute] T8 data[, ] )
[static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.24.2.28 static void
OpenTK.Graphics.ES10.GL.CompressedTexSubImage2D< T8 > (
 OpenTK.Graphics.ES10.All *target*, Int32 *level*, Int32 *xoffset*, Int32
yoffset, Int32 *width*, Int32 *height*, OpenTK.Graphics.ES10.All
format, Int32 *imageSize*, [InAttribute, OutAttribute] T8[] *data*)
 [static]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.24.2.29 static void OpenTK.Graphics.ES10.GL.CopyTexImage2D
 (OpenTK.Graphics.ES10.All *target*, Int32 *level*,
 OpenTK.Graphics.ES10.All *internalformat*, Int32 *x*, Int32 *y*, Int32
width, Int32 *height*, Int32 *border*) [static]

Copy pixels into a 2D texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the internal format of the texture. Must be one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_RGB, GL_R3_G3_B2, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture image. Must be 0 or $2^{\sup n} + 2$ (border) for some integer.

height Specifies the height of the texture image. Must be 0 or $2^{\sup m} + 2$ (border) for some integer.

border Specifies the width of the border. Must be either 0 or 1.

5.24.2.30 static void OpenTK.Graphics.ES10.GL.CopyTexSubImage2D (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height) [static]

Copy a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

5.24.2.31 `static void OpenTK.Graphics.ES10.GL.CullFace (OpenTK.Graphics.ES10.All mode) [static]`

Specify whether front- or back-facing facets can be culled.

Parameters

mode Specifies whether front- or back-facing facets are candidates for culling. Symbolic constants `GL_FRONT`, `GL_BACK`, and `GL_FRONT_AND_BACK` are accepted. The initial value is `GL_BACK`.

5.24.2.32 `static unsafe void OpenTK.Graphics.ES10.GL.DeleteTextures (Int32 n, Int32 * textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.24.2.33 `static void OpenTK.Graphics.ES10.GL.DeleteTextures (Int32 n, Int32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.24.2.34 `static void OpenTK.Graphics.ES10.GL.DeleteTextures (Int32 n, ref Int32 textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.24.2.35 `static void OpenTK.Graphics.ES10.GL.DeleteTextures (Int32 n, ref UInt32 textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.24.2.36 `static unsafe void OpenTK.Graphics.ES10.GL.DeleteTextures (Int32 n, UInt32 * textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.24.2.37 `static void OpenTK.Graphics.ES10.GL.DeleteTextures (Int32 n, UInt32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.24.2.38 static void OpenTK.Graphics.ES10.GL.DepthFunc (
OpenTK.Graphics.ES10.All *func*) [static]

Specify the value used for depth buffer comparisons.

Parameters

func Specifies the depth comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_LESS.

5.24.2.39 static void OpenTK.Graphics.ES10.GL.DepthMask (bool *flag*)
[static]

Enable or disable writing into the depth buffer.

Parameters

flag Specifies whether the depth buffer is enabled for writing. If flag is GL_FALSE, depth buffer writing is disabled. Otherwise, it is enabled. Initially, depth buffer writing is enabled.

5.24.2.40 static void OpenTK.Graphics.ES10.GL.DepthRange (Single *zNear*,
Single *zFar*) [static]

Specify mapping of depth values from normalized device coordinates to window coordinates.

Parameters

nearVal Specifies the mapping of the near clipping plane to window coordinates. The initial value is 0.

farVal Specifies the mapping of the far clipping plane to window coordinates. The initial value is 1.

5.24.2.41 static void OpenTK.Graphics.ES10.GL.DrawArrays (
OpenTK.Graphics.ES10.All *mode*, Int32 *first*, Int32 *count*)
[static]

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Specifies the starting index in the enabled arrays.

count Specifies the number of indices to be rendered.

5.24.2.42 static void OpenTK.Graphics.ES10.GL.DrawElements
(OpenTK.Graphics.ES10.All mode, Int32 count,
OpenTK.Graphics.ES10.All type, IntPtr indices) [static]

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

5.24.2.43 static void OpenTK.Graphics.ES10.GL.DrawElements<
T3 > (OpenTK.Graphics.ES10.All mode, Int32 count,
OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] ref T3
indices) [static]

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.24.2.44 `static void OpenTK.Graphics.ES10.GL.DrawElements<T3>(OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T3 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.24.2.45 `static void OpenTK.Graphics.ES10.GL.DrawElements<T3>(OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T3 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.24.2.46 `static void OpenTK.Graphics.ES10.GL.DrawElements< T3 > (OpenTK.Graphics.ES10.All mode, Int32 count, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T3[] indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.24.2.47 `static void OpenTK.Graphics.ES10.GL.Enable (OpenTK.Graphics.ES10.All cap) [static]`

Enable or disable server-side GL capabilities.

Parameters

cap Specifies a symbolic constant indicating a GL capability.

5.24.2.48 static void OpenTK.Graphics.ES10.GL.EnableClientState (OpenTK.Graphics.ES10.All *array*) [static]

Enable or disable client-side capability.

Parameters

cap Specifies the capability to enable. Symbolic constants GL_COLOR_ARRAY, GL_EDGE_FLAG_ARRAY, GL_FOG_COORD_ARRAY, GL_INDEX_ARRAY, GL_NORMAL_ARRAY, GL_SECONDARY_COLOR_ARRAY, GL_TEXTURE_COORD_ARRAY, and GL_VERTEX_ARRAY are accepted.

5.24.2.49 static void OpenTK.Graphics.ES10.GL.Finish () [static]

Block until all GL execution is complete.

5.24.2.50 static void OpenTK.Graphics.ES10.GL.Flush () [static]

Force execution of GL commands in finite time.

5.24.2.51 static void OpenTK.Graphics.ES10.GL.Fog (OpenTK.Graphics.ES10.All *pname*, Single *param*) [static]

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that *pname* will be set to.

5.24.2.52 static unsafe void OpenTK.Graphics.ES10.GL.Fog (OpenTK.Graphics.ES10.All *pname*, Single *@ *params*) [static]

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.24.2.53 `static void OpenTK.Graphics.ES10.GL.Fog (`
`OpenTK.Graphics.ES10.All pname, Single @[] params)`
`[static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.24.2.54 `static void OpenTK.Graphics.ES10.GL.FrontFace (`
`OpenTK.Graphics.ES10.All mode) [static]`

Define front- and back-facing polygons.

Parameters

mode Specifies the orientation of front-facing polygons. GL_CW and GL_CCW are accepted. The initial value is GL_CCW.

5.24.2.55 `static void OpenTK.Graphics.ES10.GL.Frustum (Single left, Single`
`right, Single bottom, Single top, Single zNear, Single zFar)`
`[static]`

Multiply the current matrix by a perspective matrix.

Parameters

left Specify the coordinates for the left and right vertical clipping planes.

bottom Specify the coordinates for the bottom and top horizontal clipping planes.

nearVal Specify the distances to the near and far depth clipping planes. Both distances must be positive.

5.24.2.56 `static unsafe void OpenTK.Graphics.ES10.GL.GenTextures (Int32 n, Int32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.24.2.57 `static void OpenTK.Graphics.ES10.GL.GenTextures (Int32 n, Int32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.24.2.58 `static void OpenTK.Graphics.ES10.GL.GenTextures (Int32 n, ref Int32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.24.2.59 `static void OpenTK.Graphics.ES10.GL.GenTextures (Int32 n, ref UInt32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.24.2.60 `static unsafe void OpenTK.Graphics.ES10.GL.GenTextures (Int32 n, UInt32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.24.2.61 `static void OpenTK.Graphics.ES10.GL.GenTextures (Int32 n, UInt32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.24.2.62 `static OpenTK.Graphics.ES10.All
OpenTK.Graphics.ES10.GL.GetError ()
[static]`

Return error information.

5.24.2.63 `static unsafe System.String OpenTK.Graphics.ES10.GL.GetString (OpenTK.Graphics.ES10.All name) [static]`

Return a string describing the current [GL](#) connection.

Parameters

name Specifies a symbolic constant, one of GL_VENDOR, GL_RENDERER, GL_VERSION, GL_SHADING_LANGUAGE_VERSION, or GL_EXTENSIONS.

5.24.2.64 `static void OpenTK.Graphics.ES10.GL.Hint (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All mode) [static]`

Specify implementation-specific hints.

Parameters

target Specifies a symbolic constant indicating the behavior to be controlled. GL_FOG_HINT, GL_GENERATE_MIPMAP_HINT, GL_LINE_SMOOTH_HINT, GL_PERSPECTIVE_CORRECTION_HINT, GL_POINT_SMOOTH_HINT, GL_POLYGON_SMOOTH_HINT, GL_TEXTURE_COMPRESSION_HINT, and GL_FRAGMENT_SHADER_DERIVATIVE_HINT are accepted.

mode Specifies a symbolic constant indicating the desired behavior. GL_FASTEST, GL_NICEST, and GL_DONT_CARE are accepted.

5.24.2.65 `static void OpenTK.Graphics.ES10.GL.Light (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, Single param) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where _i ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

5.24.2.66 `static unsafe void OpenTK.Graphics.ES10.GL.Light (OpenTK.Graphics.ES10.All light, OpenTK.Graphics.ES10.All pname, Single *@ params) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where _i ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

5.24.2.67 static void OpenTK.Graphics.ES10.GL.Light (
OpenTK.Graphics.ES10.All *light*, OpenTK.Graphics.ES10.All
***pname*, Single @[] *params*) [static]**

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where i ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

5.24.2.68 static void OpenTK.Graphics.ES10.GL.LightModel (
OpenTK.Graphics.ES10.All *pname*, Single *param*) [static]

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.24.2.69 static unsafe void OpenTK.Graphics.ES10.GL.LightModel (
OpenTK.Graphics.ES10.All *pname*, Single *@ *params*)
[static]

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.24.2.70 static void OpenTK.Graphics.ES10.GL.LightModel (OpenTK.Graphics.ES10.All *pname*, Single @[] *params*) [static]

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.24.2.71 static void OpenTK.Graphics.ES10.GL.LineWidth (Single *width*) [static]

Specify the width of rasterized lines.

Parameters

width Specifies the width of rasterized lines. The initial value is 1.

5.24.2.72 static void OpenTK.Graphics.ES10.GL.LoadIdentity () [static]

Replace the current matrix with the identity matrix.

5.24.2.73 static unsafe void OpenTK.Graphics.ES10.GL.LoadMatrix (Single * *m*) [static]

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

**5.24.2.74 static void OpenTK.Graphics.ES10.GL.LoadMatrix (ref Single *m*)
[static]**

Replace the current matrix with the specified matrix.

Parameters

- m* Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

**5.24.2.75 static void OpenTK.Graphics.ES10.GL.LoadMatrix (Single[] *m*)
[static]**

Replace the current matrix with the specified matrix.

Parameters

- m* Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.24.2.76 static void OpenTK.Graphics.ES10.GL.LogicOp (OpenTK.Graphics.ES10.All *opcode*) [static]

Specify a logical pixel operation for color index rendering.

Parameters

- opcode* Specifies a symbolic constant that selects a logical operation. The following symbols are accepted: GL_CLEAR, GL_SET, GL_COPY, GL_COPY_INVERTED, GL_NOOP, GL_INVERT, GL_AND, GL_NAND, GL_OR, GL_NOR, GL_XOR, GL_EQUIV, GL_AND_REVERSE, GL_AND_INVERTED, GL_OR_REVERSE, and GL_OR_INVERTED. The initial value is GL_COPY.

5.24.2.77 static void OpenTK.Graphics.ES10.GL.Material (OpenTK.Graphics.ES10.All *face*, OpenTK.Graphics.ES10.All *pname*, Single @[] *params*) [static]

Specify material parameters for the lighting model.

Parameters

- face* Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.24.2.78 `static void OpenTK.Graphics.ES10.GL.Material (`
`OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All`
`pname, Single param) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.24.2.79 `static unsafe void OpenTK.Graphics.ES10.GL.Material (`
`OpenTK.Graphics.ES10.All face, OpenTK.Graphics.ES10.All`
`pname, Single *@ params) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.24.2.80 `static void OpenTK.Graphics.ES10.GL.MatrixMode (`
`OpenTK.Graphics.ES10.All mode) [static]`

Specify which matrix is the current matrix.

Parameters

mode Specifies which matrix stack is the target for subsequent matrix operations. Three values are accepted: GL_MODELVIEW, GL_PROJECTION, and GL_TEXTURE. The initial value is GL_MODELVIEW. Additionally, if the ARB_imaging extension is supported, GL_COLOR is also accepted.

5.24.2.81 `static void OpenTK.Graphics.ES10.GL.MultiTexCoord4 (OpenTK.Graphics.ES10.All target, Single s, Single t, Single r, Single q) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE*i*, where *i* ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify *s*, *t*, *r*, and *q* texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.24.2.82 `static unsafe void OpenTK.Graphics.ES10.GL.MultMatrix (Single * m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

- m* Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.24.2.83 `static void OpenTK.Graphics.ES10.GL.MultMatrix (ref Single m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

- m* Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.24.2.84 `static void OpenTK.Graphics.ES10.GL.MultMatrix (Single[] m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

- m* Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.24.2.85 `static void OpenTK.Graphics.ES10.GL.Normal3 (Single nx, Single ny, Single nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.24.2.86 `static void OpenTK.Graphics.ES10.GL.NormalPointer (OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants `GL_BYTE`, `GL_SHORT`, `GL_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

5.24.2.87 `static void OpenTK.Graphics.ES10.GL.NormalPointer< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,,]) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants `GL_BYTE`, `GL_SHORT`, `GL_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : struct

5.24.2.88 `static void OpenTK.Graphics.ES10.GL.NormalPointer< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : struct

5.24.2.89 `static void OpenTK.Graphics.ES10.GL.NormalPointer< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : struct

5.24.2.90 `static void OpenTK.Graphics.ES10.GL.NormalPointer< T2 > (OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.24.2.91 `static void OpenTK.Graphics.ES10.GL.Ortho (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar) [static]`

Multiply the current matrix with an orthographic matrix.

Parameters

left Specify the coordinates for the left and right vertical clipping planes.

bottom Specify the coordinates for the bottom and top horizontal clipping planes.

nearVal Specify the distances to the nearer and farther depth clipping planes. These values are negative if the plane is to be behind the viewer.

5.24.2.92 `static void OpenTK.Graphics.ES10.GL.PixelStore (OpenTK.Graphics.ES10.All pname, Int32 param) [static]`

Set pixel storage modes.

Parameters

pname Specifies the symbolic name of the parameter to be set. Six values affect the packing of pixel data into memory: GL_PACK_SWAP_BYTES,

GL_PACK_LSB_FIRST, GL_PACK_ROW_LENGTH, GL_PACK_IMAGE_HEIGHT, GL_PACK_SKIP_PIXELS, GL_PACK_SKIP_ROWS, GL_PACK_SKIP_IMAGES, and GL_PACK_ALIGNMENT. Six more affect the unpacking of pixel data from memory: GL_UNPACK_SWAP_BYTES, GL_UNPACK_LSB_FIRST, GL_UNPACK_ROW_LENGTH, GL_UNPACK_IMAGE_HEIGHT, GL_UNPACK_SKIP_PIXELS, GL_UNPACK_SKIP_ROWS, GL_UNPACK_SKIP_IMAGES, and GL_UNPACK_ALIGNMENT.

param Specifies the value that pname is set to.

5.24.2.93 static void OpenTK.Graphics.ES10.GL.PointSize (Single *size*) [static]

Specify the diameter of rasterized points.

Parameters

size Specifies the diameter of rasterized points. The initial value is 1.

5.24.2.94 static void OpenTK.Graphics.ES10.GL.PolygonOffset (Single *factor*, Single *units*) [static]

Set the scale and units used to calculate depth values.

Parameters

factor Specifies a scale factor that is used to create a variable depth offset for each polygon. The initial value is 0.

units Is multiplied by an implementation-specific value to create a constant depth offset. The initial value is 0.

5.24.2.95 static void OpenTK.Graphics.ES10.GL.PushMatrix () [static]

Push and pop the current matrix stack.

5.24.2.96 static void OpenTK.Graphics.ES10.GL.ReadPixels (Int32 *x*, Int32 *y*, Int32 *width*, Int32 *height*, OpenTK.Graphics.ES10.All *format*, OpenTK.Graphics.ES10.All *type*, IntPtr *pixels*) [static]

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. *width* and *height* of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: `GL_COLOR_INDEX`, `GL_STENCIL_INDEX`, `GL_DEPTH_COMPONENT`, `GL_RED`, `GL_GREEN`, `GL_BLUE`, `GL_ALPHA`, `GL_RGB`, `GL_BGR`, `GL_RGBA`, `GL_BGRA`, `GL_LUMINANCE`, and `GL_LUMINANCE_ALPHA`.

type Specifies the data type of the pixel data. Must be one of `GL_UNSIGNED_BYTE`, `GL_BYTE`, `GL_BITMAP`, `GL_UNSIGNED_SHORT`, `GL_SHORT`, `GL_UNSIGNED_INT`, `GL_INT`, `GL_FLOAT`, `GL_UNSIGNED_BYTE_3_2`, `GL_UNSIGNED_BYTE_2_3_3_REV`, `GL_UNSIGNED_SHORT_5_6_5`, `GL_UNSIGNED_SHORT_5_6_5_REV`, `GL_UNSIGNED_SHORT_4_4_4_4`, `GL_UNSIGNED_SHORT_4_4_4_4_REV`, `GL_UNSIGNED_SHORT_5_5_5_1`, `GL_UNSIGNED_SHORT_1_5_5_5_REV`, `GL_UNSIGNED_INT_8_8_8_8`, `GL_UNSIGNED_INT_8_8_8_8_REV`, `GL_UNSIGNED_INT_10_10_10_2`, or `GL_UNSIGNED_INT_2_10_10_10_REV`.

data Returns the pixel data.

```
5.24.2.97 static void OpenTK.Graphics.ES10.GL.ReadPixels< T6 > ( Int32 x,
Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All
format, OpenTK.Graphics.ES10.All type, [InAttribute,
OutAttribute] T6[] pixels ) [static]
```

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. *width* and *height* of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: `GL_COLOR_INDEX`, `GL_STENCIL_INDEX`, `GL_DEPTH_COMPONENT`, `GL_RED`, `GL_GREEN`, `GL_BLUE`, `GL_ALPHA`, `GL_RGB`, `GL_BGR`, `GL_RGBA`, `GL_BGRA`, `GL_LUMINANCE`, and `GL_LUMINANCE_ALPHA`.

type Specifies the data type of the pixel data. Must be one of `GL_UNSIGNED_BYTE`, `GL_BYTE`, `GL_BITMAP`, `GL_UNSIGNED_SHORT`, `GL_SHORT`,

GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : *struct*

5.24.2.98 `static void OpenTK.Graphics.ES10.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T6 pixels[,]) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints*T6 : struct*

5.24.2.99 `static void OpenTK.Graphics.ES10.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T6 pixels[,]) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints*T6 : struct*

5.24.2.100 `static void OpenTK.Graphics.ES10.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] ref T6 pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

- x*** Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.
- width*** Specify the dimensions of the pixel rectangle. *width* and *height* of one correspond to a single pixel.
- format*** Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.
- type*** Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.
- data*** Returns the pixel data.

Type Constraints

T6 : *struct*

5.24.2.101 static void OpenTK.Graphics.ES10.GL.Rotate (Single *angle*, Single *x*, Single *y*, Single *z*) [static]

Multiply the current matrix by a rotation matrix.

Parameters

- angle*** Specifies the angle of rotation, in degrees.
- x*** Specify the x, y, and z coordinates of a vector, respectively.

5.24.2.102 static void OpenTK.Graphics.ES10.GL.SampleCoverage (Single *value*, bool *invert*) [static]

Specify multisample coverage parameters.

Parameters

value Specify a single floating-point sample coverage value. The value is clamped to the range [0,1]. The initial value is 1.0.

invert Specify a single boolean value representing if the coverage masks should be inverted. GL_TRUE and GL_FALSE are accepted. The initial value is GL_FALSE.

5.24.2.103 `static void OpenTK.Graphics.ES10.GL.Scale (Single x, Single y, Single z) [static]`

Multiply the current matrix by a general scaling matrix.

Parameters

x Specify scale factors along the x, y, and z axes, respectively.

5.24.2.104 `static void OpenTK.Graphics.ES10.GL.Scissor (Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Define the scissor box.

Parameters

x Specify the lower left corner of the scissor box. Initially (0, 0).

width Specify the width and height of the scissor box. When a GL context is first attached to a window, width and height are set to the dimensions of that window.

5.24.2.105 `static void OpenTK.Graphics.ES10.GL.ShadeModel (OpenTK.Graphics.ES10.All mode) [static]`

Select flat or smooth shading.

Parameters

mode Specifies a symbolic value representing a shading technique. Accepted values are GL_FLAT and GL_SMOOTH. The initial value is GL_SMOOTH.

5.24.2.106 `static void OpenTK.Graphics.ES10.GL.StencilFunc (`
`OpenTK.Graphics.ES10.All func, Int32 @ ref, Int32 mask)`
`[static]`

Set front and back function and reference value for stencil testing.

Parameters

- func* Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.
- ref* Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where n is the number of bitplanes in the stencil buffer. The initial value is 0.
- mask* Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.24.2.107 `static void OpenTK.Graphics.ES10.GL.StencilFunc (`
`OpenTK.Graphics.ES10.All func, Int32 @ ref, UInt32 mask)`
`[static]`

Set front and back function and reference value for stencil testing.

Parameters

- func* Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.
- ref* Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where n is the number of bitplanes in the stencil buffer. The initial value is 0.
- mask* Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.24.2.108 `static void OpenTK.Graphics.ES10.GL.StencilMask (UInt32 mask`
`) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

- mask* Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.24.2.109 static void OpenTK.Graphics.ES10.GL.StencilMask (Int32 *mask*) [static]

Control the front and back writing of individual bits in the stencil planes.

Parameters

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.24.2.110 static void OpenTK.Graphics.ES10.GL.StencilOp (OpenTK.Graphics.ES10.All *fail*, OpenTK.Graphics.ES10.All *zfail*, OpenTK.Graphics.ES10.All *zpass*) [static]

Set front and back stencil test actions.

Parameters

sfail Specifies the action to take when the stencil test fails. Eight symbolic constants are accepted: GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR, GL_INCR_WRAP, GL_DECR, GL_DECR_WRAP, and GL_INVERT. The initial value is GL_KEEP.

dpfail Specifies the stencil action when the stencil test passes, but the depth test fails. dpfail accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

dppass Specifies the stencil action when both the stencil test and the depth test pass, or when the stencil test passes and either there is no depth buffer or depth testing is not enabled. dppass accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

5.24.2.111 static void OpenTK.Graphics.ES10.GL.TexCoordPointer (Int32 *size*, OpenTK.Graphics.ES10.All *type*, Int32 *stride*, IntPtr *pointer*) [static]

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

5.24.2.112 `static void OpenTK.Graphics.ES10.GL.TexCoordPointer< T3 >
(Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
[InAttribute, OutAttribute] ref T3 pointer) [static]`

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.24.2.113 `static void OpenTK.Graphics.ES10.GL.TexCoordPointer< T3 >
(Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
[InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.24.2.114 `static void OpenTK.Graphics.ES10.GL.TexCoordPointer< T3 >
(Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
[InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.24.2.115 `static void OpenTK.Graphics.ES10.GL.TexCoordPointer< T3 >
(Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
[InAttribute, OutAttribute] T3[] pointer) [static]`

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.24.2.116 static unsafe void OpenTK.Graphics.ES10.GL.TexEnv (
OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All
pname, Single *@ params) [static]

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.24.2.117 `static void OpenTK.Graphics.ES10.GL.TexEnv (`
`OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All`
`pname, Single @[] params) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.24.2.118 `static void OpenTK.Graphics.ES10.GL.TexEnv (`
`OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All`
`pname, Single param) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA,

GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.24.2.119 static void OpenTK.Graphics.ES10.GL.TextureImage2D
 (**OpenTK.Graphics.ES10.All target**, **Int32 level**, **Int32 internalformat**, **Int32 width**, **Int32 height**, **Int32 border**, **OpenTK.Graphics.ES10.All format**, **OpenTK.Graphics.ES10.All type**, **IntPtr pixels**) [**static**]

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12,

GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n} + 2$ (border) for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m} + 2$ (border) for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

```
5.24.2.120 static void OpenTK.Graphics.ES10.GL.TextureImage2D< T8 >
( OpenTK.Graphics.ES10.All target, Int32 level, Int32
internalformat, Int32 width, Int32 height, Int32 border,
OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All
type, [InAttribute, OutAttribute] T8[] pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL-

TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer *n*. All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{m+2} (border) for some integer *m*. All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_4_4_4_4_REV,

5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

5.24.2.121 `static void OpenTK.Graphics.ES10.GL.TextureImage2D< T8 >
(OpenTK.Graphics.ES10.All target, Int32 level, Int32
internalformat, Int32 width, Int32 height, Int32 border,
OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All
type, [InAttribute, OutAttribute] T8 pixels[,]) [static]`

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5,

GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2}$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m + 2}$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.24.2.122 static void OpenTK.Graphics.ES10.GL.Texture2D< T8 >
( OpenTK.Graphics.ES10.All target, Int32 level, Int32
internalformat, Int32 width, Int32 height, Int32 border,
OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All
type, [InAttribute, OutAttribute] ref T8 pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n} + 2$ (border) for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup m} + 2$ (border) for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_

UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

5.24.2.123 `static void OpenTK.Graphics.ES10.GL.TextureImage2D< T8 >`
`(OpenTK.Graphics.ES10.All target, Int32 level, Int32`
internalformat, *Int32 width*, *Int32 height*, *Int32 border*,
`OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All`
type, [*InAttribute*, *OutAttribute*] *T8 pixels[,]*) [**static**]

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_

LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGB12, GL_RGB16, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.24.2.124 static void OpenTK.Graphics.ES10.GL.TextureParameter (OpenTK.Graphics.ES10.All target, OpenTK.Graphics.ES10.All pname, Single param) [static]

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. pname can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of pname.

5.24.2.125 static void OpenTK.Graphics.ES10.GL.TexSubImage2D (
OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32
yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All
format, OpenTK.Graphics.ES10.All type, IntPtr pixels)
[static]

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_

3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.24.2.126 `static void OpenTK.Graphics.ES10.GL.TexSubImage2D<T8> (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] ref T8 pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

5.24.2.127 `static void OpenTK.Graphics.ES10.GL.TexSubImage2D<T8> (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T8 pixels[,]) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.24.2.128 `static void OpenTK.Graphics.ES10.GL.TexSubImage2D< T8 > (OpenTK.Graphics.ES10.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES10.All format, OpenTK.Graphics.ES10.All type, [InAttribute, OutAttribute] T8[] pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.24.2.129 static void OpenTK.Graphics.ES10.GL.TextureSubImage2D< T8 > (OpenTK.Graphics.ES10.All *target*, Int32 *level*, Int32 *xoffset*, Int32 *yoffset*, Int32 *width*, Int32 *height*, OpenTK.Graphics.ES10.All *format*, OpenTK.Graphics.ES10.All *type*, [InAttribute, OutAttribute] T8 *pixels*[,,]) [static]

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.24.2.130 `static void OpenTK.Graphics.ES10.GL.Translate (Single x, Single y, Single z) [static]`

Multiply the current matrix by a translation matrix.

Parameters

x Specify the x, y, and z coordinates of a translation vector.

5.24.2.131 `static void OpenTK.Graphics.ES10.GL.VertexPointer (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, IntPtr pointer) [static]`

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

5.24.2.132 `static void OpenTK.Graphics.ES10.GL.VertexPointer< T3 > (Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer) [static]`

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.24.2.133 static void OpenTK.Graphics.ES10.GL.VertexPointer< T3 >
( Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
  [InAttribute, OutAttribute] ref T3 pointer ) [static]
```

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.24.2.134 static void OpenTK.Graphics.ES10.GL.VertexPointer< T3 >
( Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
  [InAttribute, OutAttribute] T3 pointer[, ] ) [static]
```

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.24.2.135 static void OpenTK.Graphics.ES10.GL.VertexPointer< T3 >
( Int32 size, OpenTK.Graphics.ES10.All type, Int32 stride,
  [InAttribute, OutAttribute] T3 pointer[, ] ) [static]
```

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.24.2.136 static void OpenTK.Graphics.ES10.GL.Viewport ( Int32 x, Int32
  y, Int32 width, Int32 height ) [static]
```

Set the viewport.

Parameters

x Specify the lower left corner of the viewport rectangle, in pixels. The initial value is (0,0).

width Specify the width and height of the viewport. When a [GL](#) context is first attached to a window, width and height are set to the dimensions of that window.

5.24.3 Property Documentation

5.24.3.1 override object OpenTK.Graphics.ES10.GL.SyncRoot [get, protected]

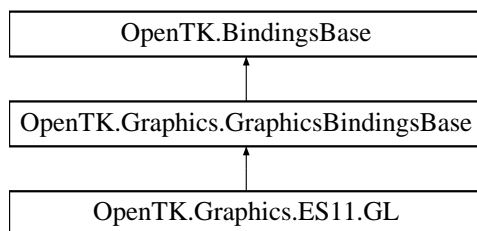
Returns a synchronization token unique for the [GL](#) class.

Reimplemented from [OpenTK.BindingsBase](#).

5.25 OpenTK.Graphics.ES11.GL Class Reference

Provides access to [OpenGL](#) ES 1.1 methods.

Inheritance diagram for OpenTK.Graphics.ES11.GL:



Static Public Member Functions

- static void [ActiveTexture](#) (OpenTK.Graphics.ES11.All texture)
Select active texture unit.
- static void [AlphaFunc](#) (OpenTK.Graphics.ES11.All func, Single @ref)
Specify the alpha test function.
- static void **AlphaFuncx** (OpenTK.Graphics.ES11.All func, int @ref)
- static void [BindBuffer](#) (OpenTK.Graphics.ES11.All target, Int32 buffer)
Bind a named buffer object.
- static void [BindBuffer](#) (OpenTK.Graphics.ES11.All target, UInt32 buffer)
Bind a named buffer object.

- static void [BindTexture](#) (OpenTK.Graphics.ES11.All target, Int32 texture)
Bind a named texture to a texturing target.
- static void [BindTexture](#) (OpenTK.Graphics.ES11.All target, UInt32 texture)
Bind a named texture to a texturing target.
- static void [BlendFunc](#) (OpenTK.Graphics.ES11.All sfactor, OpenTK.Graphics.ES11.All dfactor)
Specify pixel arithmetic.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES11.All target, IntPtr size,[InAttribute, OutAttribute] ref T2 data, OpenTK.Graphics.ES11.All usage)
Creates and initializes a buffer object's data store.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES11.All target, IntPtr size,[InAttribute, OutAttribute] T2[,] data, OpenTK.Graphics.ES11.All usage)
Creates and initializes a buffer object's data store.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES11.All target, IntPtr size,[InAttribute, OutAttribute] T2[] data, OpenTK.Graphics.ES11.All usage)
Creates and initializes a buffer object's data store.
- static void [BufferData](#) (OpenTK.Graphics.ES11.All target, IntPtr size, IntPtr data, OpenTK.Graphics.ES11.All usage)
Creates and initializes a buffer object's data store.
- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] ref T3 data)
Updates a subset of a buffer object's data store.
- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[,] data)
Updates a subset of a buffer object's data store.
- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[] data)

Updates a subset of a buffer object's data store.

- static void **BufferSubData**< T3 > (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[] data)

Updates a subset of a buffer object's data store.

- static void **BufferSubData** (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size, IntPtr data)

Updates a subset of a buffer object's data store.

- static void **Clear** (Int32 mask)

Clear buffers to preset values.

- static void **Clear** (UInt32 mask)

Clear buffers to preset values.

- static void **ClearColor** (Single red, Single green, Single blue, Single alpha)

Specify clear values for the color buffers.

- static void **ClearColorx** (int red, int green, int blue, int alpha)

- static void **ClearDepth** (Single depth)

Specify the clear value for the depth buffer.

- static void **ClearDepthx** (int depth)

- static void **ClearStencil** (Int32 s)

Specify the clear value for the stencil buffer.

- static void **ClientActiveTexture** (OpenTK.Graphics.ES11.All texture)

Select active texture unit.

- static void **ClipPlane** (OpenTK.Graphics.ES11.All plane, ref Single equation)

Specify a plane against which all geometry is clipped.

- static unsafe void **ClipPlane** (OpenTK.Graphics.ES11.All plane, Single *equation)

Specify a plane against which all geometry is clipped.

- static void **ClipPlane** (OpenTK.Graphics.ES11.All plane, Single[] equation)

Specify a plane against which all geometry is clipped.

- static void **ClipPlanefIMG** (OpenTK.Graphics.ES11.All p, ref Single eqn)

- static unsafe void **ClipPlanefIMG** (OpenTK.Graphics.ES11.All p, Single *eqn)

- static void **ClipPlanefIMG** (OpenTK.Graphics.ES11.All p, Single[] eqn)
- static unsafe void **ClipPlanex** (OpenTK.Graphics.ES11.All plane, int *equation)

- static void **ClipPlanex** (OpenTK.Graphics.ES11.All plane, int[] equation)
- static void **ClipPlanex** (OpenTK.Graphics.ES11.All plane, ref int equation)
- static unsafe void **ClipPlanexIMG** (OpenTK.Graphics.ES11.All p, int *eqn)
- static void **ClipPlanexIMG** (OpenTK.Graphics.ES11.All p, int[] eqn)
- static void **ClipPlanexIMG** (OpenTK.Graphics.ES11.All p, ref int eqn)
- static void **Color4** (Single red, Single green, Single blue, Single alpha)
Set the current color.

- static void **Color4** (Byte red, Byte green, Byte blue, Byte alpha)
Set the current color.

- static void **Color4x** (int red, int green, int blue, int alpha)
- static void **ColorMask** (bool red, bool green, bool blue, bool alpha)
Enable and disable writing of frame buffer color components.

- static void **ColorPointer< T3 >** (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)
Define an array of colors.

- static void **ColorPointer< T3 >** (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of colors.

- static void **ColorPointer< T3 >** (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of colors.

- static void **ColorPointer< T3 >** (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)
Define an array of colors.

- static void **ColorPointer** (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer)
Define an array of colors.

- static void **CompressedTexImage2D< T7 >** (OpenTK.Graphics.ES11.All target, Int32 level, OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] ref T7 data)
Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[,] data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[,] data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D< T7 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[] data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexImage2D](#) (OpenTK.Graphics.ES11.All target, Int32 level, OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data)
Specify a two-dimensional texture image in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, Int32 imageSize,[InAttribute, OutAttribute] ref T8 data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, Int32 imageSize,[InAttribute, OutAttribute] T8[,] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, Int32 imageSize,[InAttribute, OutAttribute] T8[] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, Int32 imageSize,[InAttribute, OutAttribute] T8[] data)
Specify a two-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage2D](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, Int32 imageSize, IntPtr data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CopyTexImage2D](#) (OpenTK.Graphics.ES11.All target, Int32 level, OpenTK.Graphics.ES11.All internalformat, Int32 x, Int32 y, Int32 width, Int32 height, Int32 border)
Copy pixels into a 2D texture image.
- static void [CopyTexSubImage2D](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height)
Copy a two-dimensional texture subimage.
- static void [CullFace](#) (OpenTK.Graphics.ES11.All mode)
Specify whether front- or back-facing facets can be culled.
- static unsafe void [DeleteBuffers](#) (Int32 n, Int32 *buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, Int32[] buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, ref Int32 buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, ref UInt32 buffers)
Delete named buffer objects.
- static unsafe void [DeleteBuffers](#) (Int32 n, UInt32 *buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, UInt32[] buffers)
Delete named buffer objects.
- static unsafe void [DeleteTextures](#) (Int32 n, Int32 *textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, Int32[] textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, ref Int32 textures)
Delete named textures.

- static void [DeleteTextures](#) (Int32 n, ref UInt32 textures)
Delete named textures.
- static unsafe void [DeleteTextures](#) (Int32 n, UInt32 *textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, UInt32[] textures)
Delete named textures.
- static void [DepthFunc](#) (OpenTK.Graphics.ES11.All func)
Specify the value used for depth buffer comparisons.
- static void [DepthMask](#) (bool flag)
Enable or disable writing into the depth buffer.
- static void [DepthRange](#) (Single zNear, Single zFar)
Specify mapping of depth values from normalized device coordinates to window coordinates.
- static void **DepthRangex** (int zNear, int zFar)
- static void **Disable** (OpenTK.Graphics.ES11.All cap)
- static void **DisableClientState** (OpenTK.Graphics.ES11.All array)
- static void **DisableDriverControlQCOM** (Int32 driverControl)
- static void **DisableDriverControlQCOM** (UInt32 driverControl)
- static void [DrawArrays](#) (OpenTK.Graphics.ES11.All mode, Int32 first, Int32 count)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] ref T3 indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T3[,], indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T3[,], indices)
Render primitives from array data.

- static void [DrawElements](#)< T3 > (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T3[] indices)
Render primitives from array data.
- static void [DrawElements](#) (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type, IntPtr indices)
Render primitives from array data.
- static void [Enable](#) (OpenTK.Graphics.ES11.All cap)
Enable or disable server-side [GL](#) capabilities.
- static void [EnableClientState](#) (OpenTK.Graphics.ES11.All array)
Enable or disable client-side capability.
- static void **EnableDriverControlQCOM** (Int32 driverControl)
- static void **EnableDriverControlQCOM** (UInt32 driverControl)
- static void [Finish](#) ()
Block until all [GL](#) execution is complete.
- static void [Flush](#) ()
Force execution of [GL](#) commands in finite time.
- static void [Fog](#) (OpenTK.Graphics.ES11.All pname, Single param)
Specify fog parameters.
- static unsafe void [Fog](#) (OpenTK.Graphics.ES11.All pname, Single *@params)
Specify fog parameters.
- static void [Fog](#) (OpenTK.Graphics.ES11.All pname, Single[] @params)
Specify fog parameters.
- static void **Fogx** (OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **Fogx** (OpenTK.Graphics.ES11.All pname, int *@params)
- static void **Fogx** (OpenTK.Graphics.ES11.All pname, int[] @params)
- static void [FrontFace](#) (OpenTK.Graphics.ES11.All mode)
Define front- and back-facing polygons.
- static void [Frustum](#) (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar)
Multiply the current matrix by a perspective matrix.

- static void **Frustumx** (int left, int right, int bottom, int top, int zNear, int zFar)
- static unsafe void **GenBuffers** (Int32 n, Int32 *buffers)
Generate buffer object names.
- static void **GenBuffers** (Int32 n, Int32[] buffers)
Generate buffer object names.
- static void **GenBuffers** (Int32 n, ref Int32 buffers)
Generate buffer object names.
- static void **GenBuffers** (Int32 n, ref UInt32 buffers)
Generate buffer object names.
- static unsafe void **GenBuffers** (Int32 n, UInt32 *buffers)
Generate buffer object names.
- static void **GenBuffers** (Int32 n, UInt32[] buffers)
Generate buffer object names.
- static unsafe void **GenTextures** (Int32 n, Int32 *textures)
Generate texture names.
- static void **GenTextures** (Int32 n, Int32[] textures)
Generate texture names.
- static void **GenTextures** (Int32 n, ref Int32 textures)
Generate texture names.
- static void **GenTextures** (Int32 n, ref UInt32 textures)
Generate texture names.
- static unsafe void **GenTextures** (Int32 n, UInt32 *textures)
Generate texture names.
- static void **GenTextures** (Int32 n, UInt32[] textures)
Generate texture names.
- static unsafe void **GetBoolean** (OpenTK.Graphics.ES11.All pname, bool *@params)
- static void **GetBoolean** (OpenTK.Graphics.ES11.All pname, bool[]@params)
- static void **GetBoolean** (OpenTK.Graphics.ES11.All pname, ref bool @params)

- static unsafe void [GetBufferParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 *@params)
Return parameters of a buffer object.
- static void [GetBufferParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32[]@params)
Return parameters of a buffer object.
- static void [GetBufferParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, ref Int32 @params)
Return parameters of a buffer object.
- static void [GetClipPlane](#) (OpenTK.Graphics.ES11.All pname, ref Single eqn)
Return the coefficients of the specified clipping plane.
- static unsafe void [GetClipPlane](#) (OpenTK.Graphics.ES11.All pname, Single *eqn)
Return the coefficients of the specified clipping plane.
- static void [GetClipPlane](#) (OpenTK.Graphics.ES11.All pname, Single[] eqn)
Return the coefficients of the specified clipping plane.
- static unsafe void **GetClipPlanex** (OpenTK.Graphics.ES11.All pname, int *eqn)
- static void **GetClipPlanex** (OpenTK.Graphics.ES11.All pname, int[] eqn)
- static void **GetClipPlanex** (OpenTK.Graphics.ES11.All pname, ref int eqn)
- static unsafe void **GetDriverControlsQCOM** (Int32 *num, Int32 size, Int32 *driverControls)
- static unsafe void **GetDriverControlsQCOM** (Int32 *num, Int32 size, UInt32 *driverControls)
- static void **GetDriverControlsQCOM** (Int32[] num, Int32 size, Int32[] driverControls)
- static void **GetDriverControlsQCOM** (Int32[] num, Int32 size, UInt32[] driverControls)
- static void **GetDriverControlsQCOM** (ref Int32 num, Int32 size, ref Int32 driverControls)
- static void **GetDriverControlsQCOM** (ref Int32 num, Int32 size, ref UInt32 driverControls)
- static unsafe void **GetDriverControlStringQCOM** (Int32 driverControl, Int32 bufSize, Int32 *length, String driverControlString)
- static void **GetDriverControlStringQCOM** (Int32 driverControl, Int32 bufSize, Int32[] length, String driverControlString)
- static void **GetDriverControlStringQCOM** (Int32 driverControl, Int32 bufSize, ref Int32 length, String driverControlString)

- static unsafe void **GetDriverControlStringQCOM** (UInt32 driverControl, Int32 bufSize, Int32 *length, String driverControlString)
- static void **GetDriverControlStringQCOM** (UInt32 driverControl, Int32 bufSize, Int32[] length, String driverControlString)
- static void **GetDriverControlStringQCOM** (UInt32 driverControl, Int32 bufSize, ref Int32 length, String driverControlString)
- static OpenTK.Graphics.ES11.All **GetError** ()

Return error information.

- static unsafe void **GetFixed** (OpenTK.Graphics.ES11.All pname, int *@params)
- static void **GetFixed** (OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **GetFixed** (OpenTK.Graphics.ES11.All pname, ref int @params)
- static void **GetFloat** (OpenTK.Graphics.ES11.All pname, ref Single @params)
- static unsafe void **GetFloat** (OpenTK.Graphics.ES11.All pname, Single *@params)
- static void **GetFloat** (OpenTK.Graphics.ES11.All pname, Single[] @params)
- static unsafe void **GetInteger** (OpenTK.Graphics.ES11.All pname, Int32 *@params)
- static void **GetInteger** (OpenTK.Graphics.ES11.All pname, Int32[] @params)
- static void **GetInteger** (OpenTK.Graphics.ES11.All pname, ref Int32 @params)

- static void **GetLight** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, ref Single @params)

Return light source parameter values.

- static unsafe void **GetLight** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single *@params)

Return light source parameter values.

- static void **GetLight** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single[] @params)

Return light source parameter values.

- static unsafe void **GetLightx** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, int *@params)
- static void **GetLightx** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **GetLightx** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, ref int @params)
- static void **GetMaterial** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, ref Single @params)

Return material parameters.

- static unsafe void [GetMaterial](#) (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single *@params)
Return material parameters.
- static void [GetMaterial](#) (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single[] @params)
Return material parameters.
- static unsafe void **GetMaterialx** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, int *@params)
- static void **GetMaterialx** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **GetMaterialx** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, ref int @params)
- static void [GetPointer< T1 >](#) (OpenTK.Graphics.ES11.All pname,[InAttribute, OutAttribute] ref T1 @params)
Return the address of the specified pointer.
- static void [GetPointer< T1 >](#) (OpenTK.Graphics.ES11.All pname,[InAttribute, OutAttribute] T1[,,@params)
Return the address of the specified pointer.
- static void [GetPointer< T1 >](#) (OpenTK.Graphics.ES11.All pname,[InAttribute, OutAttribute] T1[,,@params)
Return the address of the specified pointer.
- static void [GetPointer< T1 >](#) (OpenTK.Graphics.ES11.All pname,[InAttribute, OutAttribute] T1[] @params)
Return the address of the specified pointer.
- static void [GetPointer](#) (OpenTK.Graphics.ES11.All pname, IntPtr @params)
Return the address of the specified pointer.
- static unsafe System.String [GetString](#) (OpenTK.Graphics.ES11.All name)
Return a string describing the current [GL](#) connection.
- static void [GetTexEnv](#) (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, ref Single @params)
Return texture environment parameters.
- static unsafe void [GetTexEnv](#) (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Single *@params)

Return texture environment parameters.

- static void [GetTexEnv](#) (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Single[] @params)

Return texture environment parameters.

- static unsafe void [GetTexEnv](#) (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Int32 * @params)

Return texture environment parameters.

- static void [GetTexEnv](#) (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Int32[] @params)

Return texture environment parameters.

- static void [GetTexEnv](#) (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, ref Int32 @params)

Return texture environment parameters.

- static unsafe void **GetTexEnvx** (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, int * @params)

- static void **GetTexEnvx** (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, int[] @params)

- static void **GetTexEnvx** (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, ref int @params)

- static void [GetTexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, ref Single @params)

Return texture parameter values.

- static unsafe void [GetTexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single * @params)

Return texture parameter values.

- static void [GetTexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single[] @params)

Return texture parameter values.

- static unsafe void [GetTexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 * @params)

Return texture parameter values.

- static void [GetTexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32[] @params)

Return texture parameter values.

- static void [GetTexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, ref Int32 @params)
Return texture parameter values.
- static unsafe void **GetTexParameterx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int *@params)
- static void **GetTexParameterx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **GetTexParameterx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, ref int @params)
- static void [Hint](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All mode)
Specify implementation-specific hints.
- static bool [IsBuffer](#) (Int32 buffer)
Determine if a name corresponds to a buffer object.
- static bool [IsBuffer](#) (UInt32 buffer)
Determine if a name corresponds to a buffer object.
- static bool [IsEnabled](#) (OpenTK.Graphics.ES11.All cap)
Test whether a capability is enabled.
- static bool [IsTexture](#) (Int32 texture)
Determine if a name corresponds to a texture.
- static bool [IsTexture](#) (UInt32 texture)
Determine if a name corresponds to a texture.
- static void [Light](#) (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single param)
Set light source parameters.
- static unsafe void [Light](#) (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single *@params)
Set light source parameters.
- static void [Light](#) (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single[] @params)
Set light source parameters.
- static void [LightModel](#) (OpenTK.Graphics.ES11.All pname, Single param)

Set the lighting model parameters.

- static unsafe void **LightModel** (OpenTK.Graphics.ES11.All pname, Single *@params)

Set the lighting model parameters.

- static void **LightModel** (OpenTK.Graphics.ES11.All pname, Single[] @params)

Set the lighting model parameters.

- static void **LightModelx** (OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **LightModelx** (OpenTK.Graphics.ES11.All pname, int *@params)
- static void **LightModelx** (OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **Lightx** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **Lightx** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, int *@params)
- static void **Lightx** (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **LineWidth** (Single width)

Specify the width of rasterized lines.

- static void **LineWidthx** (int width)
- static void **LoadIdentity** ()

Replace the current matrix with the identity matrix.

- static void **LoadMatrix** (ref Single m)

Replace the current matrix with the specified matrix.

- static unsafe void **LoadMatrix** (Single *m)

Replace the current matrix with the specified matrix.

- static void **LoadMatrix** (Single[] m)

Replace the current matrix with the specified matrix.

- static unsafe void **LoadMatrixx** (int *m)
- static void **LoadMatrixx** (int[] m)
- static void **LoadMatrixx** (ref int m)
- static void **LogicOp** (OpenTK.Graphics.ES11.All opcode)

Specify a logical pixel operation for color index rendering.

- static void [Material](#) (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single param)
Specify material parameters for the lighting model.
- static unsafe void [Material](#) (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single *@params)
Specify material parameters for the lighting model.
- static void [Material](#) (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single[] @params)
Specify material parameters for the lighting model.
- static void **Materialx** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **Materialx** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, int *@params)
- static void **Materialx** (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void [MatrixMode](#) (OpenTK.Graphics.ES11.All mode)
Specify which matrix is the current matrix.
- static void [MultiTexCoord4](#) (OpenTK.Graphics.ES11.All target, Single s, Single t, Single r, Single q)
Set the current texture coordinates.
- static void **MultiTexCoord4x** (OpenTK.Graphics.ES11.All target, int s, int t, int r, int q)
- static void [MultMatrix](#) (ref Single m)
Multiply the current matrix with the specified matrix.
- static unsafe void [MultMatrix](#) (Single *m)
Multiply the current matrix with the specified matrix.
- static void [MultMatrix](#) (Single[] m)
Multiply the current matrix with the specified matrix.
- static unsafe void **MultMatrixx** (int *m)
- static void **MultMatrixx** (int[] m)
- static void **MultMatrixx** (ref int m)
- static void [Normal3](#) (Single nx, Single ny, Single nz)
Set the current normal vector.
- static void **Normal3x** (int nx, int ny, int nz)

- static void [NormalPointer](#)< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] ref T2 pointer)
Define an array of normals.
- static void [NormalPointer](#)< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T2[,], pointer)
Define an array of normals.
- static void [NormalPointer](#)< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T2[,], pointer)
Define an array of normals.
- static void [NormalPointer](#)< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T2[] pointer)
Define an array of normals.
- static void [NormalPointer](#) (OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer)
Define an array of normals.
- static void [Ortho](#) (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar)
Multiply the current matrix with an orthographic matrix.
- static void **Orthox** (int left, int right, int bottom, int top, int zNear, int zFar)
- static void [PixelStore](#) (OpenTK.Graphics.ES11.All pname, Int32 param)
Set pixel storage modes.
- static void [PointParameter](#) (OpenTK.Graphics.ES11.All pname, Single param)
Specify point parameters.
- static unsafe void [PointParameter](#) (OpenTK.Graphics.ES11.All pname, Single *@params)
Specify point parameters.
- static void [PointParameter](#) (OpenTK.Graphics.ES11.All pname, Single[] *@params)
Specify point parameters.
- static void **PointParameterx** (OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **PointParameterx** (OpenTK.Graphics.ES11.All pname, int *@params)

- static void **PointParameterx** (OpenTK.Graphics.ES11.All pname, int[] @params)
- static void **PointSize** (Single size)
Specify the diameter of rasterized points.
- static void **PointSizex** (int size)
- static void **PolygonOffset** (Single factor, Single units)
Set the scale and units used to calculate depth values.
- static void **PolygonOffsetx** (int factor, int units)
- static void **PopMatrix** ()
- static void **PushMatrix** ()
Push and pop the current matrix stack.
- static void **ReadPixels**< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] ref T6 pixels)
Read a block of pixels from the frame buffer.
- static void **ReadPixels**< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T6[,] pixels)
Read a block of pixels from the frame buffer.
- static void **ReadPixels**< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T6[,] pixels)
Read a block of pixels from the frame buffer.
- static void **ReadPixels**< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T6[] pixels)
Read a block of pixels from the frame buffer.
- static void **ReadPixels** (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, IntPtr pixels)
Read a block of pixels from the frame buffer.
- static void **Rotate** (Single angle, Single x, Single y, Single z)
Multiply the current matrix by a rotation matrix.
- static void **Rotatex** (int angle, int x, int y, int z)

- static void [SampleCoverage](#) (Single value, bool invert)
Specify multisample coverage parameters.
- static void **SampleCoveragex** (int value, bool invert)
- static void [Scale](#) (Single x, Single y, Single z)
Multiply the current matrix by a general scaling matrix.
- static void **Scalex** (int x, int y, int z)
- static void [Scissor](#) (Int32 x, Int32 y, Int32 width, Int32 height)
Define the scissor box.
- static void [ShadeModel](#) (OpenTK.Graphics.ES11.All mode)
Select flat or smooth shading.
- static void [StencilFunc](#) (OpenTK.Graphics.ES11.All func, Int32 @ref, Int32 mask)
Set front and back function and reference value for stencil testing.
- static void [StencilFunc](#) (OpenTK.Graphics.ES11.All func, Int32 @ref, UInt32 mask)
Set front and back function and reference value for stencil testing.
- static void [StencilMask](#) (Int32 mask)
Control the front and back writing of individual bits in the stencil planes.
- static void [StencilMask](#) (UInt32 mask)
Control the front and back writing of individual bits in the stencil planes.
- static void [StencilOp](#) (OpenTK.Graphics.ES11.All fail, OpenTK.Graphics.ES11.All zfail, OpenTK.Graphics.ES11.All zpass)
Set front and back stencil test actions.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[,], pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)
Define an array of texture coordinates.

- static void [TexCoordPointer](#)< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer](#) (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer)
Define an array of texture coordinates.
- static void [TexEnv](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single param)
Set texture environment parameters.
- static unsafe void [TexEnv](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single *@params)
Set texture environment parameters.
- static void [TexEnv](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single[] @params)
Set texture environment parameters.
- static void [TexEnv](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 param)
Set texture environment parameters.
- static unsafe void [TexEnv](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 *@params)
Set texture environment parameters.
- static void [TexEnv](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32[] @params)
Set texture environment parameters.
- static void **TexEnvx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **TexEnvx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int *@params)
- static void **TexEnvx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES11.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES11.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES11.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#)< T8 > (OpenTK.Graphics.ES11.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type,[InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, IntPtr pixels)

Specify a two-dimensional texture image.

- static void [TexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single param)

Set texture parameters.

- static unsafe void [TexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single *@params)

Set texture parameters.

- static void [TexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single[]@params)

Set texture parameters.

- static void [TexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 param)

Set texture parameters.

- static unsafe void [TexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 *@params)

Set texture parameters.

- static void [TexParameter](#) (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32[] @params)

Set texture parameters.

- static void **TexParameterx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int param)
- static unsafe void **TexParameterx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int * @params)
- static void **TexParameterx** (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, int[] @params)
- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D](#) (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, IntPtr pixels)

Specify a two-dimensional texture subimage.

- static void [Translate](#) (Single x, Single y, Single z)

Multiply the current matrix by a translation matrix.

- static void **Translate**(int x, int y, int z)
- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)
Define an array of vertex data.
- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[,], pointer)
Define an array of vertex data.
- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[,], pointer)
Define an array of vertex data.
- static void **VertexPointer**< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)
Define an array of vertex data.
- static void **VertexPointer** (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer)
Define an array of vertex data.
- static void **Viewport** (Int32 x, Int32 y, Int32 width, Int32 height)
Set the viewport.

Properties

- override object **SyncRoot** [get]
Returns a synchronization token unique for the [GL](#) class.

5.25.1 Detailed Description

Provides access to [OpenGL](#) ES 1.1 methods.

5.25.2 Member Function Documentation

5.25.2.1 static void OpenTK.Graphics.ES11.GL.ActiveTexture (OpenTK.Graphics.ES11.All texture) [static]

Select active texture unit.

Parameters

texture Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. texture must be one of GL_TEXTURE*i*, where *i* ranges from 0 to the larger of (GL_MAX_TEXTURE_COORDS - 1) and (GL_MAX_COMBINED_TEXTURE_IMAGE_UNITS - 1). The initial value is GL_TEXTURE0.

5.25.2.2 static void OpenTK.Graphics.ES11.GL.AlphaFunc (OpenTK.Graphics.ES11.All *func*, Single @ *ref*) [static]

Specify the alpha test function.

Parameters

func Specifies the alpha comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_ALWAYS.

ref Specifies the reference value that incoming alpha values are compared to. This value is clamped to the range [0,1], where 0 represents the lowest possible alpha value and 1 the highest possible value. The initial reference value is 0.

5.25.2.3 static void OpenTK.Graphics.ES11.GL.BindBuffer (OpenTK.Graphics.ES11.All *target*, Int32 *buffer*) [static]

Bind a named buffer object.

Parameters

target Specifies the target to which the buffer object is bound. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

buffer Specifies the name of a buffer object.

5.25.2.4 static void OpenTK.Graphics.ES11.GL.BindBuffer (OpenTK.Graphics.ES11.All *target*, UInt32 *buffer*) [static]

Bind a named buffer object.

Parameters

target Specifies the target to which the buffer object is bound. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

buffer Specifies the name of a buffer object.

5.25.2.5 `static void OpenTK.Graphics.ES11.GL.BindTexture (`
`OpenTK.Graphics.ES11.All target, Int32 texture) [static]`

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.25.2.6 `static void OpenTK.Graphics.ES11.GL.BindTexture (`
`OpenTK.Graphics.ES11.All target, UInt32 texture) [static]`

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.25.2.7 `static void OpenTK.Graphics.ES11.GL.BlendFunc (`
`OpenTK.Graphics.ES11.All sfactor, OpenTK.Graphics.ES11.All`
`dfactor) [static]`

Specify pixel arithmetic.

Parameters

sfactor Specifies how the red, green, blue, and alpha source blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dfactor Specifies how the red, green, blue, and alpha destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

5.25.2.8 static void OpenTK.Graphics.ES11.GL.BufferData (
OpenTK.Graphics.ES11.All target, IntPtr size, IntPtr data,
OpenTK.Graphics.ES11.All usage) [static]

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

5.25.2.9 static void OpenTK.Graphics.ES11.GL.BufferData< T2 > (
OpenTK.Graphics.ES11.All target, IntPtr size, [InAttribute,
OutAttribute] ref T2 data, OpenTK.Graphics.ES11.All usage)
[static]

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct

```
5.25.2.10 static void OpenTK.Graphics.ES11.GL.BufferData< T2 > (
    OpenTK.Graphics.ES11.All target, IntPtr size, [InAttribute,
    OutAttribute] T2 data[,], OpenTK.Graphics.ES11.All usage )
    [static]
```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct

```
5.25.2.11 static void OpenTK.Graphics.ES11.GL.BufferData< T2 > (
    OpenTK.Graphics.ES11.All target, IntPtr size, [InAttribute,
    OutAttribute] T2 data[,], OpenTK.Graphics.ES11.All usage )
    [static]
```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct

```
5.25.2.12 static void OpenTK.Graphics.ES11.GL.BufferData< T2 > (
    OpenTK.Graphics.ES11.All target, IntPtr size, [InAttribute,
    OutAttribute] T2[] data, OpenTK.Graphics.ES11.All usage )
    [static]
```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct

5.25.2.13 `static void OpenTK.Graphics.ES11.GL.BufferSubData (`
`OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size, IntPtr`
`data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

5.25.2.14 `static void OpenTK.Graphics.ES11.GL.BufferSubData< T3 > (`
`OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size,`
`[InAttribute, OutAttribute] T3 data[,]) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.25.2.15 `static void OpenTK.Graphics.ES11.GL.BufferSubData< T3 > (`
`OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size,`
`[InAttribute, OutAttribute] T3[] data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.25.2.16 `static void OpenTK.Graphics.ES11.GL.BufferSubData< T3 > (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] ref T3 data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.25.2.17 `static void OpenTK.Graphics.ES11.GL.BufferSubData< T3 > (OpenTK.Graphics.ES11.All target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] T3 data[,]) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.25.2.18 `static void OpenTK.Graphics.ES11.GL.Clear (Int32 mask)`
[static]

Clear buffers to preset values.

Parameters

mask Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, GL_ACCUM_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

5.25.2.19 `static void OpenTK.Graphics.ES11.GL.Clear (UInt32 mask)`
[static]

Clear buffers to preset values.

Parameters

mask Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, GL_ACCUM_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

5.25.2.20 `static void OpenTK.Graphics.ES11.GL.ClearColor (Single red, Single green, Single blue, Single alpha)` [static]

Specify clear values for the color buffers.

Parameters

red Specify the red, green, blue, and alpha values used when the color buffers are cleared. The initial values are all 0.

**5.25.2.21 static void OpenTK.Graphics.ES11.GL.ClearDepth (Single *depth*)
[static]**

Specify the clear value for the depth buffer.

Parameters

depth Specifies the depth value used when the depth buffer is cleared. The initial value is 1.

**5.25.2.22 static void OpenTK.Graphics.ES11.GL.ClearStencil (Int32 *s*)
[static]**

Specify the clear value for the stencil buffer.

Parameters

s Specifies the index used when the stencil buffer is cleared. The initial value is 0.

5.25.2.23 static void OpenTK.Graphics.ES11.GL.ClientActiveTexture (OpenTK.Graphics.ES11.All *texture*) [static]

Select active texture unit.

Parameters

texture Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. *texture* must be one of GL_TEXTURE*i*, where *i* ranges from 0 to the value of GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value. The initial value is GL_TEXTURE0.

**5.25.2.24 static void OpenTK.Graphics.ES11.GL.ClipPlane (OpenTK.Graphics.ES11.All *plane*, ref Single *equation*)
[static]**

Specify a plane against which all geometry is clipped.

Parameters

plane Specifies which clipping plane is being positioned. Symbolic names of the form GL_CLIP_PLANE*i*, where *i* is an integer between 0 and GL_MAX_CLIP_PLANES - 1, are accepted.

equation Specifies the address of an array of four double-precision floating-point values. These values are interpreted as a plane equation.

5.25.2.25 `static unsafe void OpenTK.Graphics.ES11.GL.ClipPlane (OpenTK.Graphics.ES11.All plane, Single * equation) [static]`

Specify a plane against which all geometry is clipped.

Parameters

plane Specifies which clipping plane is being positioned. Symbolic names of the form GL_CLIP_PLANE*i*, where *i* is an integer between 0 and GL_MAX_CLIP_PLANES - 1, are accepted.

equation Specifies the address of an array of four double-precision floating-point values. These values are interpreted as a plane equation.

5.25.2.26 `static void OpenTK.Graphics.ES11.GL.ClipPlane (OpenTK.Graphics.ES11.All plane, Single[] equation) [static]`

Specify a plane against which all geometry is clipped.

Parameters

plane Specifies which clipping plane is being positioned. Symbolic names of the form GL_CLIP_PLANE*i*, where *i* is an integer between 0 and GL_MAX_CLIP_PLANES - 1, are accepted.

equation Specifies the address of an array of four double-precision floating-point values. These values are interpreted as a plane equation.

5.25.2.27 `static void OpenTK.Graphics.ES11.GL.Color4 (Single red, Single green, Single blue, Single alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.25.2.28 `static void OpenTK.Graphics.ES11.GL.Color4 (Byte red, Byte green, Byte blue, Byte alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.25.2.29 static void OpenTK.Graphics.ES11.GL.ColorMask (bool *red*, bool *green*, bool *blue*, bool *alpha*) [static]

Enable and disable writing of frame buffer color components.

Parameters

red Specify whether red, green, blue, and alpha can or cannot be written into the frame buffer. The initial values are all GL_TRUE, indicating that the color components can be written.

5.25.2.30 static void OpenTK.Graphics.ES11.GL.ColorPointer (Int32 *size*, OpenTK.Graphics.ES11.All *type*, Int32 *stride*, IntPtr *pointer*) [static]

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

5.25.2.31 static void OpenTK.Graphics.ES11.GL.ColorPointer< T3 > (Int32 *size*, OpenTK.Graphics.ES11.All *type*, Int32 *stride*, [InAttribute, OutAttribute] ref T3 *pointer*) [static]

Define an array of colors.

Parameters

- size* Specifies the number of components per color. Must be 3 or 4. The initial value is 4.
- type* Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.32 `static void OpenTK.Graphics.ES11.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of colors.

Parameters

- size* Specifies the number of components per color. Must be 3 or 4. The initial value is 4.
- type* Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.33 `static void OpenTK.Graphics.ES11.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of colors.

Parameters

- size* Specifies the number of components per color. Must be 3 or 4. The initial value is 4.
- type* Specifies the data type of each color component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.34 `static void OpenTK.Graphics.ES11.GL.ColorPointer< T3 > (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer) [static]`

Define an array of colors.

Parameters

- size* Specifies the number of components per color. Must be 3 or 4. The initial value is 4.
- type* Specifies the data type of each color component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.35 `static void OpenTK.Graphics.ES11.GL.CompressedTexImage2D
(OpenTK.Graphics.ES11.All target, Int32 level,
OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32
height, Int32 border, Int32 imageSize, IntPtr data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n + 2} (border)$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n + 2} (border)$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.25.2.36 `static void OpenTK.Graphics.ES11.GL.CompressedTexImage2D<
T7 > (OpenTK.Graphics.ES11.All target, Int32 level,
OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32
height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
ref T7 data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

```
5.25.2.37 static void OpenTK.Graphics.ES11.GL.CompressedTexImage2D<
T7 > ( OpenTK.Graphics.ES11.All target, Int32 level,
OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32
height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
T7 data[, ] ) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y,

GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2}$ (border) for some integer *n*. All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2}$ (border) for some integer *n*. All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

```
5.25.2.38 static void OpenTK.Graphics.ES11.GL.CompressedTexImage2D<
T7 > ( OpenTK.Graphics.ES11.All target, Int32 level,
OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32
height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
T7 data[,]) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

```
5.25.2.39 static void OpenTK.Graphics.ES11.GL.CompressedTexImage2D<
    T7 > ( OpenTK.Graphics.ES11.All target, Int32 level,
    OpenTK.Graphics.ES11.All internalformat, Int32 width, Int32
    height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute]
    T7[] data ) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

5.25.2.40 static void OpenTK.Graphics.ES11.GL.CompressedTexSubImage2D
 (**OpenTK.Graphics.ES11.All target**, **Int32 level**, **Int32 xoffset**, **Int32 yoffset**, **Int32 width**, **Int32 height**,
OpenTK.Graphics.ES11.All format, **Int32 imageSize**, **IntPtr data**)
[static]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.25.2.41 static void

```
OpenTK.Graphics.E511.GL.CompressedTexSubImage2D< T8 > (
    OpenTK.Graphics.E511.All target, Int32 level, Int32 xoffset, Int32
    yoffset, Int32 width, Int32 height, OpenTK.Graphics.E511.All
    format, Int32 imageSize, [InAttribute, OutAttribute] ref T8 data )
[static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.25.2.42 static void

```
OpenTK.Graphics.E511.GL.CompressedTexSubImage2D< T8 > (
    OpenTK.Graphics.E511.All target, Int32 level, Int32 xoffset, Int32
    yoffset, Int32 width, Int32 height, OpenTK.Graphics.E511.All
    format, Int32 imageSize, [InAttribute, OutAttribute] T8 data[,])
[static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

```
5.25.2.43 static void
OpenTK.Graphics.ES11.GL.CompressedTexSubImage2D< T8 > (
    OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32
    yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All
    format, Int32 imageSize, [InAttribute, OutAttribute] T8 data[, ] )
    [static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.25.2.44 static void

```
OpenTK.Graphics.ES11.GL.CompressedTexSubImage2D< T8 > (
    OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32
    yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All
    format, Int32 imageSize, [InAttribute, OutAttribute] T8[] data )
[static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.25.2.45 `static void OpenTK.Graphics.ES11.GL.CopyTexImage2D`
 (`OpenTK.Graphics.ES11.All target`, `Int32 level`,
`OpenTK.Graphics.ES11.All internalformat`, `Int32 x`, `Int32 y`, `Int32`
`width`, `Int32 height`, `Int32 border`) [`static`]

Copy pixels into a 2D texture image.

Parameters

target Specifies the target texture. Must be `GL_TEXTURE_2D`, `GL_TEXTURE_CUBE_MAP_POSITIVE_X`, `GL_TEXTURE_CUBE_MAP_NEGATIVE_X`, `GL_TEXTURE_CUBE_MAP_POSITIVE_Y`, `GL_TEXTURE_CUBE_MAP_NEGATIVE_Y`, `GL_TEXTURE_CUBE_MAP_POSITIVE_Z`, or `GL_TEXTURE_CUBE_MAP_NEGATIVE_Z`.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the internal format of the texture. Must be one of the following symbolic constants: `GL_ALPHA`, `GL_ALPHA4`, `GL_ALPHA8`, `GL_ALPHA12`, `GL_ALPHA16`, `GL_COMPRESSED_ALPHA`, `GL_COMPRESSED_LUMINANCE`, `GL_COMPRESSED_LUMINANCE_ALPHA`, `GL_COMPRESSED_INTENSITY`, `GL_COMPRESSED_RGB`, `GL_COMPRESSED_RGBA`, `GL_DEPTH_COMPONENT`, `GL_DEPTH_COMPONENT16`, `GL_DEPTH_COMPONENT24`, `GL_DEPTH_COMPONENT32`, `GL_LUMINANCE`, `GL_LUMINANCE4`, `GL_LUMINANCE8`, `GL_LUMINANCE12`, `GL_LUMINANCE16`, `GL_LUMINANCE_ALPHA`, `GL_LUMINANCE4_ALPHA4`, `GL_LUMINANCE6_ALPHA2`, `GL_LUMINANCE8_ALPHA8`, `GL_LUMINANCE12_ALPHA4`, `GL_LUMINANCE12_ALPHA12`, `GL_LUMINANCE16_ALPHA16`, `GL_INTENSITY`, `GL_INTENSITY4`, `GL_INTENSITY8`, `GL_INTENSITY12`, `GL_INTENSITY16`, `GL_RGB`, `GL_R3_G3_B2`, `GL_RGB4`, `GL_RGB5`, `GL_RGB8`, `GL_RGB10`, `GL_RGB12`, `GL_RGB16`, `GL_RGBA`, `GL_RGBA2`, `GL_RGBA4`, `GL_RGB5_A1`, `GL_RGBA8`, `GL_RGB10_A2`, `GL_RGBA12`, `GL_RGBA16`, `GL_SLUMINANCE`, `GL_SLUMINANCE8`, `GL_SLUMINANCE_ALPHA`, `GL_SLUMINANCE8_ALPHA8`, `GL_SRGB`, `GL_SRGB8`, `GL_SRGB_ALPHA`, or `GL_SRGB8_ALPHA8`.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture image. Must be $0 \text{ or } 2^{\sup n} + 2$ (*border*) for some integer .

height Specifies the height of the texture image. Must be $0 \text{ or } 2^{\sup m} + 2$ (*border*) for some integer .

border Specifies the width of the border. Must be either 0 or 1.

5.25.2.46 `static void OpenTK.Graphics.ES11.GL.CopyTexSubImage2D (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Copy a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

5.25.2.47 `static void OpenTK.Graphics.ES11.GL.CullFace (OpenTK.Graphics.ES11.All mode) [static]`

Specify whether front- or back-facing facets can be culled.

Parameters

mode Specifies whether front- or back-facing facets are candidates for culling. Symbolic constants GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK are accepted. The initial value is GL_BACK.

5.25.2.48 `static void OpenTK.Graphics.ES11.GL.DeleteBuffers (Int32 n, ref UInt32 buffers) [static]`

Delete named buffer objects.

Parameters

n Specifies the number of buffer objects to be deleted.

buffers Specifies an array of buffer objects to be deleted.

5.25.2.49 `static unsafe void OpenTK.Graphics.ES11.GL.DeleteBuffers (Int32 n, UInt32 * buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
- buffers* Specifies an array of buffer objects to be deleted.

5.25.2.50 `static void OpenTK.Graphics.ES11.GL.DeleteBuffers (Int32 n, UInt32[] buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
- buffers* Specifies an array of buffer objects to be deleted.

5.25.2.51 `static unsafe void OpenTK.Graphics.ES11.GL.DeleteBuffers (Int32 n, Int32 * buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
- buffers* Specifies an array of buffer objects to be deleted.

5.25.2.52 `static void OpenTK.Graphics.ES11.GL.DeleteBuffers (Int32 n, Int32[] buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
- buffers* Specifies an array of buffer objects to be deleted.

5.25.2.53 `static void OpenTK.Graphics.ES11.GL.DeleteBuffers (Int32 n, ref Int32 buffers) [static]`

Delete named buffer objects.

Parameters

n Specifies the number of buffer objects to be deleted.

buffers Specifies an array of buffer objects to be deleted.

5.25.2.54 `static unsafe void OpenTK.Graphics.ES11.GL.DeleteTextures (Int32 n, Int32 * textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.25.2.55 `static void OpenTK.Graphics.ES11.GL.DeleteTextures (Int32 n, Int32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.25.2.56 `static void OpenTK.Graphics.ES11.GL.DeleteTextures (Int32 n, ref Int32 textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.25.2.57 `static void OpenTK.Graphics.ES11.GL.DeleteTextures (Int32 n, ref UInt32 textures) [static]`

Delete named textures.

Parameters

- n* Specifies the number of textures to be deleted.
- textures* Specifies an array of textures to be deleted.

5.25.2.58 `static unsafe void OpenTK.Graphics.ES11.GL.DeleteTextures (Int32 n, UInt32 * textures) [static]`

Delete named textures.

Parameters

- n* Specifies the number of textures to be deleted.
- textures* Specifies an array of textures to be deleted.

5.25.2.59 `static void OpenTK.Graphics.ES11.GL.DeleteTextures (Int32 n, UInt32[] textures) [static]`

Delete named textures.

Parameters

- n* Specifies the number of textures to be deleted.
- textures* Specifies an array of textures to be deleted.

5.25.2.60 `static void OpenTK.Graphics.ES11.GL.DepthFunc (OpenTK.Graphics.ES11.All func) [static]`

Specify the value used for depth buffer comparisons.

Parameters

- func* Specifies the depth comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_LESS.

5.25.2.61 static void OpenTK.Graphics.ES11.GL.DepthMask (bool *flag*) [static]

Enable or disable writing into the depth buffer.

Parameters

flag Specifies whether the depth buffer is enabled for writing. If *flag* is GL_FALSE, depth buffer writing is disabled. Otherwise, it is enabled. Initially, depth buffer writing is enabled.

5.25.2.62 static void OpenTK.Graphics.ES11.GL.DepthRange (Single *zNear*, Single *zFar*) [static]

Specify mapping of depth values from normalized device coordinates to window coordinates.

Parameters

nearVal Specifies the mapping of the near clipping plane to window coordinates. The initial value is 0.

farVal Specifies the mapping of the far clipping plane to window coordinates. The initial value is 1.

5.25.2.63 static void OpenTK.Graphics.ES11.GL.DrawArrays (OpenTK.Graphics.ES11.All *mode*, Int32 *first*, Int32 *count*) [static]

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Specifies the starting index in the enabled arrays.

count Specifies the number of indices to be rendered.

5.25.2.64 `static void OpenTK.Graphics.ES11.GL.DrawElements
(OpenTK.Graphics.ES11.All mode, Int32 count,
OpenTK.Graphics.ES11.All type, IntPtr indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

5.25.2.65 `static void OpenTK.Graphics.ES11.GL.DrawElements<
T3 > (OpenTK.Graphics.ES11.All mode, Int32 count,
OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] ref T3
indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.25.2.66 `static void OpenTK.Graphics.ES11.GL.DrawElements< T3 > (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T3 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.25.2.67 `static void OpenTK.Graphics.ES11.GL.DrawElements< T3 > (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T3 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.25.2.68 `static void OpenTK.Graphics.ES11.GL.DrawElements< T3 > (OpenTK.Graphics.ES11.All mode, Int32 count, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T3[] indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants `GL_POINTS`, `GL_LINE_STRIP`, `GL_LINE_LOOP`, `GL_LINES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`, `GL_TRIANGLES`, `GL_QUAD_STRIP`, `GL_QUADS`, and `GL_POLYGON` are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of `GL_UNSIGNED_BYTE`, `GL_UNSIGNED_SHORT`, or `GL_UNSIGNED_INT`.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.25.2.69 `static void OpenTK.Graphics.ES11.GL.Enable (OpenTK.Graphics.ES11.All cap) [static]`

Enable or disable server-side [GL](#) capabilities.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.25.2.70 `static void OpenTK.Graphics.ES11.GL.EnableClientState (OpenTK.Graphics.ES11.All array) [static]`

Enable or disable client-side capability.

Parameters

cap Specifies the capability to enable. Symbolic constants `GL_COLOR_ARRAY`, `GL_EDGE_FLAG_ARRAY`, `GL_FOG_COORD_ARRAY`, `GL_INDEX_ARRAY`, `GL_NORMAL_ARRAY`, `GL_SECONDARY_COLOR_ARRAY`, `GL_TEXTURE_COORD_ARRAY`, and `GL_VERTEX_ARRAY` are accepted.

5.25.2.71 static void OpenTK.Graphics.ES11.GL.Finish () [static]

Block until all [GL](#) execution is complete.

5.25.2.72 static void OpenTK.Graphics.ES11.GL.Flush () [static]

Force execution of [GL](#) commands in finite time.

5.25.2.73 static void OpenTK.Graphics.ES11.GL.Fog (OpenTK.Graphics.ES11.All pname, Single param) [static]

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.25.2.74 static unsafe void OpenTK.Graphics.ES11.GL.Fog (OpenTK.Graphics.ES11.All pname, Single *@ params) [static]

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.25.2.75 static void OpenTK.Graphics.ES11.GL.Fog (OpenTK.Graphics.ES11.All pname, Single @[] params) [static]

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.25.2.76 `static void OpenTK.Graphics.ES11.GL.FrontFace (OpenTK.Graphics.ES11.All mode) [static]`

Define front- and back-facing polygons.

Parameters

mode Specifies the orientation of front-facing polygons. GL_CW and GL_CCW are accepted. The initial value is GL_CCW.

5.25.2.77 `static void OpenTK.Graphics.ES11.GL.Frustum (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar) [static]`

Multiply the current matrix by a perspective matrix.

Parameters

left Specify the coordinates for the left and right vertical clipping planes.

bottom Specify the coordinates for the bottom and top horizontal clipping planes.

nearVal Specify the distances to the near and far depth clipping planes. Both distances must be positive.

5.25.2.78 `static unsafe void OpenTK.Graphics.ES11.GL.GenBuffers (Int32 n, Int32 * buffers) [static]`

Generate buffer object names.

Parameters

n Specifies the number of buffer object names to be generated.

buffers Specifies an array in which the generated buffer object names are stored.

5.25.2.79 `static void OpenTK.Graphics.ES11.GL.GenBuffers (Int32 n, Int32[] buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.25.2.80 `static void OpenTK.Graphics.ES11.GL.GenBuffers (Int32 n, ref Int32 buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.25.2.81 `static void OpenTK.Graphics.ES11.GL.GenBuffers (Int32 n, ref UInt32 buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.25.2.82 `static unsafe void OpenTK.Graphics.ES11.GL.GenBuffers (Int32 n, UInt32 * buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.25.2.83 `static void OpenTK.Graphics.ES11.GL.GenBuffers (Int32 n, UInt32[] buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.25.2.84 `static unsafe void OpenTK.Graphics.ES11.GL.GenTextures (Int32 n, Int32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.25.2.85 `static void OpenTK.Graphics.ES11.GL.GenTextures (Int32 n, Int32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.25.2.86 `static void OpenTK.Graphics.ES11.GL.GenTextures (Int32 n, ref Int32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.25.2.87 `static void OpenTK.Graphics.ES11.GL.GenTextures (Int32 n, ref UInt32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.25.2.88 `static unsafe void OpenTK.Graphics.ES11.GL.GenTextures (Int32 n, UInt32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.25.2.89 `static void OpenTK.Graphics.ES11.GL.GenTextures (Int32 n, UInt32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.25.2.90 `static unsafe void OpenTK.Graphics.ES11.GL.GetBufferParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 *@ params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.25.2.91 `static void OpenTK.Graphics.ES11.GL.GetBufferParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 @[] params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.25.2.92 `static void OpenTK.Graphics.ES11.GL.GetBufferParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, ref Int32 @ params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.25.2.93 `static void OpenTK.Graphics.ES11.GL.GetClipPlane (OpenTK.Graphics.ES11.All pname, ref Single eqn) [static]`

Return the coefficients of the specified clipping plane.

Parameters

plane Specifies a clipping plane. The number of clipping planes depends on the implementation, but at least six clipping planes are supported. They are identified by symbolic names of the form GL_CLIP_PLANE where i ranges from 0 to the value of GL_MAX_CLIP_PLANES - 1.

equation Returns four double-precision values that are the coefficients of the plane equation of plane in eye coordinates. The initial value is (0, 0, 0, 0).

5.25.2.94 `static unsafe void OpenTK.Graphics.ES11.GL.GetClipPlane (`
`OpenTK.Graphics.ES11.All pname, Single * eqn) [static]`

Return the coefficients of the specified clipping plane.

Parameters

plane Specifies a clipping plane. The number of clipping planes depends on the implementation, but at least six clipping planes are supported. They are identified by symbolic names of the form GL_CLIP_PLANE where i ranges from 0 to the value of GL_MAX_CLIP_PLANES - 1.

equation Returns four double-precision values that are the coefficients of the plane equation of plane in eye coordinates. The initial value is (0, 0, 0, 0).

5.25.2.95 `static void OpenTK.Graphics.ES11.GL.GetClipPlane (`
`OpenTK.Graphics.ES11.All pname, Single[] eqn) [static]`

Return the coefficients of the specified clipping plane.

Parameters

plane Specifies a clipping plane. The number of clipping planes depends on the implementation, but at least six clipping planes are supported. They are identified by symbolic names of the form GL_CLIP_PLANE where i ranges from 0 to the value of GL_MAX_CLIP_PLANES - 1.

equation Returns four double-precision values that are the coefficients of the plane equation of plane in eye coordinates. The initial value is (0, 0, 0, 0).

5.25.2.96 `static OpenTK.Graphics.ES11.All`
`OpenTK.Graphics.ES11.GL.GetError ()`
`[static]`

Return error information.

5.25.2.97 `static void OpenTK.Graphics.ES11.GL.GetLight (`
`OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All`
`pname, ref Single @ params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by

symbolic names of the form `GL_LIGHT` where ranges from 0 to the value of `GL_MAX_LIGHTS - 1`.

pname Specifies a light source parameter for light. Accepted symbolic names are `GL_AMBIENT`, `GL_DIFFUSE`, `GL_SPECULAR`, `GL_POSITION`, `GL_SPOT_DIRECTION`, `GL_SPOT_EXPONENT`, `GL_SPOT_CUTOFF`, `GL_CONSTANT_ATTENUATION`, `GL_LINEAR_ATTENUATION`, and `GL_QUADRATIC_ATTENUATION`.

params Returns the requested data.

5.25.2.98 `static unsafe void OpenTK.Graphics.ES11.GL.GetLight (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single *@ params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form `GL_LIGHT` where ranges from 0 to the value of `GL_MAX_LIGHTS - 1`.

pname Specifies a light source parameter for light. Accepted symbolic names are `GL_AMBIENT`, `GL_DIFFUSE`, `GL_SPECULAR`, `GL_POSITION`, `GL_SPOT_DIRECTION`, `GL_SPOT_EXPONENT`, `GL_SPOT_CUTOFF`, `GL_CONSTANT_ATTENUATION`, `GL_LINEAR_ATTENUATION`, and `GL_QUADRATIC_ATTENUATION`.

params Returns the requested data.

5.25.2.99 `static void OpenTK.Graphics.ES11.GL.GetLight (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single @[] params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form `GL_LIGHT` where ranges from 0 to the value of `GL_MAX_LIGHTS - 1`.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.25.2.100 static void OpenTK.Graphics.ES11.GL.GetMaterial (OpenTK.Graphics.ES11.All *face*, OpenTK.Graphics.ES11.All *pname*, ref Single @ *params*) [static]

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

5.25.2.101 static unsafe void OpenTK.Graphics.ES11.GL.GetMaterial (OpenTK.Graphics.ES11.All *face*, OpenTK.Graphics.ES11.All *pname*, Single *@ *params*) [static]

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

5.25.2.102 static void OpenTK.Graphics.ES11.GL.GetMaterial (OpenTK.Graphics.ES11.All *face*, OpenTK.Graphics.ES11.All *pname*, Single @[] *params*) [static]

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

5.25.2.103 static void OpenTK.Graphics.ES11.GL.GetPointer (
OpenTK.Graphics.ES11.All pname, IntPtr @ params)
[static]

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by pname.

5.25.2.104 static void OpenTK.Graphics.ES11.GL.GetPointer< T1 > (
OpenTK.Graphics.ES11.All pname, [InAttribute, OutAttribute] ref
T1 @ params) [static]

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by pname.

Type Constraints*T1 : struct*

5.25.2.105 `static void OpenTK.Graphics.ES11.GL.GetPointer< T1 > (OpenTK.Graphics.ES11.All pname, [InAttribute, OutAttribute] T1 @ params[,]) [static]`

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

Type Constraints*T1 : struct*

5.25.2.106 `static void OpenTK.Graphics.ES11.GL.GetPointer< T1 > (OpenTK.Graphics.ES11.All pname, [InAttribute, OutAttribute] T1 @ params[,]) [static]`

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

Type Constraints*T1 : struct*

5.25.2.107 `static void OpenTK.Graphics.ES11.GL.GetPointer< T1 > (OpenTK.Graphics.ES11.All pname, [InAttribute, OutAttribute] T1 @[] params) [static]`

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by pname.

Type Constraints

T1 : *struct*

5.25.2.108 `static unsafe System.String OpenTK.Graphics.ES11.GL.GetString (OpenTK.Graphics.ES11.All name) [static]`

Return a string describing the current GL connection.

Parameters

name Specifies a symbolic constant, one of GL_VENDOR, GL_RENDERER, GL_VERSION, GL_SHADING_LANGUAGE_VERSION, or GL_EXTENSIONS.

5.25.2.109 `static void OpenTK.Graphics.ES11.GL.GetTexEnv (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, ref Int32 @ params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.25.2.110 static unsafe void OpenTK.Graphics.ES11.GL.GetTexEnv (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Single *@ params) [static]

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.25.2.111 static void OpenTK.Graphics.ES11.GL.GetTexEnv (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, ref Single @ params) [static]

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.25.2.112 static void OpenTK.Graphics.ES11.GL.GetTexEnv (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Single @[] params) [static]

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.25.2.113 static unsafe void OpenTK.Graphics.ES11.GL.GetTexEnv (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Int32 *@ params) [static]

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.25.2.114 static void OpenTK.Graphics.ES11.GL.GetTexEnv (OpenTK.Graphics.ES11.All env, OpenTK.Graphics.ES11.All pname, Int32 @[] params) [static]

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.25.2.115 static void OpenTK.Graphics.ES11.GL.GetTexParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, ref Single @ params) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.25.2.116 static unsafe void OpenTK.Graphics.ES11.GL.GetTexParameter (OpenTK.Graphics.ES11.All *target*, OpenTK.Graphics.ES11.All *pname*, Single *@ *params*) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.25.2.117 static void OpenTK.Graphics.ES11.GL.GetTexParameter (OpenTK.Graphics.ES11.All *target*, OpenTK.Graphics.ES11.All *pname*, Single @[] *params*) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.25.2.118 static unsafe void OpenTK.Graphics.ES11.GL.GetTexParameter (OpenTK.Graphics.ES11.All *target*, OpenTK.Graphics.ES11.All *pname*, Int32 *@ *params*) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.25.2.119 static void OpenTK.Graphics.ES11.GL.GetTexParameter (OpenTK.Graphics.ES11.All *target*, OpenTK.Graphics.ES11.All *pname*, Int32 @[] *params*) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.25.2.120 static void OpenTK.Graphics.ES11.GL.GetTexParameter (
OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All
pname, ref Int32 @ params) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.25.2.121 static void OpenTK.Graphics.ES11.GL.Hint (
OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All
mode) [static]

Specify implementation-specific hints.

Parameters

target Specifies a symbolic constant indicating the behavior to be controlled. GL_FOG_HINT, GL_GENERATE_MIPMAP_HINT, GL_LINE_SMOOTH_HINT, GL_PERSPECTIVE_CORRECTION_HINT,

GL_POINT_SMOOTH_HINT, GL_POLYGON_SMOOTH_HINT, GL_TEXTURE_COMPRESSION_HINT, and GL_FRAGMENT_SHADER_DERIVATIVE_HINT are accepted.

mode Specifies a symbolic constant indicating the desired behavior. GL_FASTEST, GL_NICEST, and GL_DONT_CARE are accepted.

5.25.2.122 static bool OpenTK.Graphics.ES11.GL.IsBuffer (Int32 *buffer*) [static]

Determine if a name corresponds to a buffer object.

Parameters

buffer Specifies a value that may be the name of a buffer object.

5.25.2.123 static bool OpenTK.Graphics.ES11.GL.IsBuffer (UInt32 *buffer*) [static]

Determine if a name corresponds to a buffer object.

Parameters

buffer Specifies a value that may be the name of a buffer object.

5.25.2.124 static bool OpenTK.Graphics.ES11.GL.IsEnabled (OpenTK.Graphics.ES11.All *cap*) [static]

Test whether a capability is enabled.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.25.2.125 static bool OpenTK.Graphics.ES11.GL.IsTexture (Int32 *texture*) [static]

Determine if a name corresponds to a texture.

Parameters

texture Specifies a value that may be the name of a texture.

5.25.2.126 `static bool OpenTK.Graphics.ES11.GL.IsTexture (UInt32 texture) [static]`

Determine if a name corresponds to a texture.

Parameters

texture Specifies a value that may be the name of a texture.

5.25.2.127 `static void OpenTK.Graphics.ES11.GL.Light (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single param) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form `GL_LIGHTi`, where *i* ranges from 0 to the value of `GL_MAX_LIGHTS - 1`.

pname Specifies a single-valued light source parameter for light. `GL_SPOT_EXPONENT`, `GL_SPOT_CUTOFF`, `GL_CONSTANT_ATTENUATION`, `GL_LINEAR_ATTENUATION`, and `GL_QUADRATIC_ATTENUATION` are accepted.

param Specifies the value that parameter *pname* of light source *light* will be set to.

5.25.2.128 `static void OpenTK.Graphics.ES11.GL.Light (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single @[] params) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form `GL_LIGHTi`, where *i* ranges from 0 to the value of `GL_MAX_LIGHTS - 1`.

pname Specifies a single-valued light source parameter for light. `GL_SPOT_EXPONENT`, `GL_SPOT_CUTOFF`, `GL_CONSTANT_ATTENUATION`, `GL_LINEAR_ATTENUATION`, and `GL_QUADRATIC_ATTENUATION` are accepted.

param Specifies the value that parameter pname of light source light will be set to.

5.25.2.129 `static unsafe void OpenTK.Graphics.ES11.GL.Light (OpenTK.Graphics.ES11.All light, OpenTK.Graphics.ES11.All pname, Single *@ params) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where i ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

5.25.2.130 `static void OpenTK.Graphics.ES11.GL.LightModel (OpenTK.Graphics.ES11.All pname, Single param) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.25.2.131 `static unsafe void OpenTK.Graphics.ES11.GL.LightModel (OpenTK.Graphics.ES11.All pname, Single *@ params) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.25.2.132 static void OpenTK.Graphics.ES11.GL.LightModel (
OpenTK.Graphics.ES11.All *pname*, Single @[] *params*)
[static]

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.25.2.133 static void OpenTK.Graphics.ES11.GL.LineWidth (Single *width*)
[static]

Specify the width of rasterized lines.

Parameters

width Specifies the width of rasterized lines. The initial value is 1.

5.25.2.134 static void OpenTK.Graphics.ES11.GL.LoadIdentity ()
[static]

Replace the current matrix with the identity matrix.

5.25.2.135 static void OpenTK.Graphics.ES11.GL.LoadMatrix (ref Single *m*
) [static]

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.25.2.136 `static unsafe void OpenTK.Graphics.ES11.GL.LoadMatrix (Single * m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.25.2.137 `static void OpenTK.Graphics.ES11.GL.LoadMatrix (Single[] m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.25.2.138 `static void OpenTK.Graphics.ES11.GL.LogicOp (OpenTK.Graphics.ES11.All opcode) [static]`

Specify a logical pixel operation for color index rendering.

Parameters

opcode Specifies a symbolic constant that selects a logical operation. The following symbols are accepted: GL_CLEAR, GL_SET, GL_COPY, GL_COPY_INVERTED, GL_NOOP, GL_INVERT, GL_AND, GL_NAND, GL_OR, GL_NOR, GL_XOR, GL_EQUIV, GL_AND_REVERSE, GL_AND_INVERTED, GL_OR_REVERSE, and GL_OR_INVERTED. The initial value is GL_COPY.

5.25.2.139 `static void OpenTK.Graphics.ES11.GL.Material (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single @[] params) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.25.2.140 `static unsafe void OpenTK.Graphics.ES11.GL.Material (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single *@ params) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.25.2.141 `static void OpenTK.Graphics.ES11.GL.Material (OpenTK.Graphics.ES11.All face, OpenTK.Graphics.ES11.All pname, Single param) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.25.2.142 `static void OpenTK.Graphics.ES11.GL.MatrixMode (OpenTK.Graphics.ES11.All mode) [static]`

Specify which matrix is the current matrix.

Parameters

mode Specifies which matrix stack is the target for subsequent matrix operations. Three values are accepted: GL_MODELVIEW, GL_PROJECTION, and GL_TEXTURE. The initial value is GL_MODELVIEW. Additionally, if the ARB_imaging extension is supported, GL_COLOR is also accepted.

5.25.2.143 `static void OpenTK.Graphics.ES11.GL.MultiTexCoord4 (OpenTK.Graphics.ES11.All target, Single s, Single t, Single r, Single q) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE*i*, where *i* ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify *s*, *t*, *r*, and *q* texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.25.2.144 `static unsafe void OpenTK.Graphics.ES11.GL.MultMatrix (Single * m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

- m* Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.25.2.145 `static void OpenTK.Graphics.ES11.GL.MultMatrix (Single[] m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

- m* Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.25.2.146 `static void OpenTK.Graphics.ES11.GL.MultMatrix (ref Single m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

- m* Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.25.2.147 `static void OpenTK.Graphics.ES11.GL.Normal3 (Single nx,
Single ny, Single nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.25.2.148 `static void OpenTK.Graphics.ES11.GL.NormalPointer (OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

5.25.2.149 `static void OpenTK.Graphics.ES11.GL.NormalPointer< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints*T2 : struct*

5.25.2.150 `static void OpenTK.Graphics.ES11.GL.NormalPointer< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints*T2 : struct*

5.25.2.151 `static void OpenTK.Graphics.ES11.GL.NormalPointer< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints*T2 : struct*

5.25.2.152 `static void OpenTK.Graphics.ES11.GL.NormalPointer< T2 > (OpenTK.Graphics.ES11.All type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.25.2.153 `static void OpenTK.Graphics.ES11.GL.Ortho (Single left, Single right, Single bottom, Single top, Single zNear, Single zFar) [static]`

Multiply the current matrix with an orthographic matrix.

Parameters

left Specify the coordinates for the left and right vertical clipping planes.

bottom Specify the coordinates for the bottom and top horizontal clipping planes.

nearVal Specify the distances to the nearer and farther depth clipping planes. These values are negative if the plane is to be behind the viewer.

5.25.2.154 `static void OpenTK.Graphics.ES11.GL.PixelStore (OpenTK.Graphics.ES11.All pname, Int32 param) [static]`

Set pixel storage modes.

Parameters

pname Specifies the symbolic name of the parameter to be set. Six values affect the packing of pixel data into memory: GL_PACK_SWAP_BYTES,

GL_PACK_LSB_FIRST, GL_PACK_ROW_LENGTH, GL_PACK_IMAGE_HEIGHT, GL_PACK_SKIP_PIXELS, GL_PACK_SKIP_ROWS, GL_PACK_SKIP_IMAGES, and GL_PACK_ALIGNMENT. Six more affect the unpacking of pixel data from memory: GL_UNPACK_SWAP_BYTES, GL_UNPACK_LSB_FIRST, GL_UNPACK_ROW_LENGTH, GL_UNPACK_IMAGE_HEIGHT, GL_UNPACK_SKIP_PIXELS, GL_UNPACK_SKIP_ROWS, GL_UNPACK_SKIP_IMAGES, and GL_UNPACK_ALIGNMENT.

param Specifies the value that pname is set to.

5.25.2.155 `static void OpenTK.Graphics.ES11.GL.PointParameter (OpenTK.Graphics.ES11.All pname, Single param) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.25.2.156 `static unsafe void OpenTK.Graphics.ES11.GL.PointParameter (OpenTK.Graphics.ES11.All pname, Single *@ params) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.25.2.157 `static void OpenTK.Graphics.ES11.GL.PointParameter (OpenTK.Graphics.ES11.All pname, Single @[] params) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.25.2.158 `static void OpenTK.Graphics.ES11.GL.PointSize (Single size) [static]`

Specify the diameter of rasterized points.

Parameters

size Specifies the diameter of rasterized points. The initial value is 1.

5.25.2.159 `static void OpenTK.Graphics.ES11.GL.PolygonOffset (Single factor, Single units) [static]`

Set the scale and units used to calculate depth values.

Parameters

factor Specifies a scale factor that is used to create a variable depth offset for each polygon. The initial value is 0.

units Is multiplied by an implementation-specific value to create a constant depth offset. The initial value is 0.

5.25.2.160 `static void OpenTK.Graphics.ES11.GL.PushMatrix () [static]`

Push and pop the current matrix stack.

5.25.2.161 `static void OpenTK.Graphics.ES11.GL.ReadPixels (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, IntPtr pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

5.25.2.162 `static void OpenTK.Graphics.ES11.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T6[] pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_

SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : *struct*

5.25.2.163 `static void OpenTK.Graphics.ES11.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T6 pixels[,J]) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : *struct*

5.25.2.164 `static void OpenTK.Graphics.ES11.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] ref T6 pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : *struct*

5.25.2.165 `static void OpenTK.Graphics.ES11.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T6 pixels[,]) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : struct

5.25.2.166 static void OpenTK.Graphics.ES11.GL.Rotate (Single angle, Single x, Single y, Single z) [static]

Multiply the current matrix by a rotation matrix.

Parameters

angle Specifies the angle of rotation, in degrees.

x Specify the x, y, and z coordinates of a vector, respectively.

5.25.2.167 static void OpenTK.Graphics.ES11.GL.SampleCoverage (Single value, bool invert) [static]

Specify multisample coverage parameters.

Parameters

value Specify a single floating-point sample coverage value. The value is clamped to the range [0,1]. The initial value is 1.0.

invert Specify a single boolean value representing if the coverage masks should be inverted. GL_TRUE and GL_FALSE are accepted. The initial value is GL_FALSE.

5.25.2.168 `static void OpenTK.Graphics.ES11.GL.Scale (Single x, Single y, Single z) [static]`

Multiply the current matrix by a general scaling matrix.

Parameters

x Specify scale factors along the x, y, and z axes, respectively.

5.25.2.169 `static void OpenTK.Graphics.ES11.GL.Scissor (Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Define the scissor box.

Parameters

x Specify the lower left corner of the scissor box. Initially (0, 0).

width Specify the width and height of the scissor box. When a GL context is first attached to a window, width and height are set to the dimensions of that window.

5.25.2.170 `static void OpenTK.Graphics.ES11.GL.ShadeModel (OpenTK.Graphics.ES11.All mode) [static]`

Select flat or smooth shading.

Parameters

mode Specifies a symbolic value representing a shading technique. Accepted values are GL_FLAT and GL_SMOOTH. The initial value is GL_SMOOTH.

5.25.2.171 `static void OpenTK.Graphics.ES11.GL.StencilFunc (OpenTK.Graphics.ES11.All func, Int32 @ ref, UInt32 mask) [static]`

Set front and back function and reference value for stencil testing.

Parameters

- func* Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.
- ref* Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\text{sup } n - 1}]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.
- mask* Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.25.2.172 `static void OpenTK.Graphics.ES11.GL.StencilFunc (OpenTK.Graphics.ES11.All func, Int32 @ ref, Int32 mask) [static]`

Set front and back function and reference value for stencil testing.

Parameters

- func* Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.
- ref* Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\text{sup } n - 1}]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.
- mask* Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.25.2.173 `static void OpenTK.Graphics.ES11.GL.StencilMask (Int32 mask) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

- mask* Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.25.2.174 `static void OpenTK.Graphics.ES11.GL.StencilMask (UInt32 mask) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.25.2.175 `static void OpenTK.Graphics.ES11.GL.StencilOp (OpenTK.Graphics.ES11.All fail, OpenTK.Graphics.ES11.All zfail, OpenTK.Graphics.ES11.All zpass) [static]`

Set front and back stencil test actions.

Parameters

sfail Specifies the action to take when the stencil test fails. Eight symbolic constants are accepted: GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR, GL_INCR_WRAP, GL_DECR, GL_DECR_WRAP, and GL_INVERT. The initial value is GL_KEEP.

dpfail Specifies the stencil action when the stencil test passes, but the depth test fails. dpfail accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

dppass Specifies the stencil action when both the stencil test and the depth test pass, or when the stencil test passes and either there is no depth buffer or depth testing is not enabled. dppass accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

5.25.2.176 `static void OpenTK.Graphics.ES11.GL.TexCoordPointer (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer) [static]`

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

5.25.2.177 `static void OpenTK.Graphics.ES11.GL.TexCoordPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of texture coordinates.

Parameters

- size* Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.178 `static void OpenTK.Graphics.ES11.GL.TexCoordPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] ref T3 pointer) [static]`

Define an array of texture coordinates.

Parameters

- size* Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.179 `static void OpenTK.Graphics.ES11.GL.TexCoordPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] T3[] pointer) [static]`

Define an array of texture coordinates.

Parameters

- size* Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.180 `static void OpenTK.Graphics.ES11.GL.TexCoordPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of texture coordinates.

Parameters

- size* Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.181 `static void OpenTK.Graphics.ES11.GL.TextureEnv (`
`OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All`
`pname, Single param) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.25.2.182 `static void OpenTK.Graphics.ES11.GL.TextureEnv (`
`OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All`
`pname, Single @[] params) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA,

GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.25.2.183 `static unsafe void OpenTK.Graphics.ES11.GL.TexEnv (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single*@ params) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.25.2.184 `static void OpenTK.Graphics.ES11.GL.TexEnv (`
`OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All`
`pname, Int32 param) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.25.2.185 `static unsafe void OpenTK.Graphics.ES11.GL.TexEnv (`
`OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All`
`pname, Int32 *@ params) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA,

GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.25.2.186 static void OpenTK.Graphics.ES11.GL.TexEnv (
OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All
pname, Int32 @[] params) [static]

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.25.2.187 static void OpenTK.Graphics.ES11.GL.Texture2D
 (**OpenTK.Graphics.ES11.All** *target*, **Int32** *level*, **Int32** *internalFormat*, **Int32** *width*, **Int32** *height*, **Int32** *border*, **OpenTK.Graphics.ES11.All** *format*, **OpenTK.Graphics.ES11.All** *type*, **IntPtr** *pixels*) [**static**]

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \text{ (border)}$ for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m + 2} \text{ (border)}$ for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.25.2.188 `static void OpenTK.Graphics.ES11.GL.TextureImage2D< T8 >
(OpenTK.Graphics.ES11.All target, Int32 level, Int32
internalformat, Int32 width, Int32 height, Int32 border,
OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
type, [InAttribute, OutAttribute] T8 pixels[,]) [static]`

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_

LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup m + 2}$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

```

5.25.2.189 static void OpenTK.Graphics.ES11.GL.Texture2D< T8 >
( OpenTK.Graphics.ES11.All target, Int32 level, Int32
  internalformat, Int32 width, Int32 height, Int32 border,
  OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
  type, [InAttribute, OutAttribute] T8 pixels[, ] ) [static]

```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

```
5.25.2.190 static void OpenTK.Graphics.ES11.GL.Texture2D< T8 >
( OpenTK.Graphics.ES11.All target, Int32 level, Int32
internalformat, Int32 width, Int32 height, Int32 border,
OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
type, [InAttribute, OutAttribute] ref T8 pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE,

GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.25.2.191 `static void OpenTK.Graphics.ES11.GL.Texture2D< T8 >
 (OpenTK.Graphics.ES11.All target, Int32 level, Int32
internalformat, Int32 width, Int32 height, Int32 border,
 OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
type, [InAttribute, OutAttribute] T8[] pixels) [static]`

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n} + 2$ (*border*) for some integer *n*. All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m} + 2$ (*border*) for some integer *m*. All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.25.2.192 static void OpenTK.Graphics.ES11.GL.TextureParameter (OpenTK.Graphics.ES11.All *target*, OpenTK.Graphics.ES11.All *pname*, Single @[] *params*) [static]

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.25.2.193 `static void OpenTK.Graphics.ES11.GL.TextureParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 param) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. pname can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of pname.

5.25.2.194 `static void OpenTK.Graphics.ES11.GL.TextureParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Int32 @[] params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. pname can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of pname.

5.25.2.195 `static void OpenTK.Graphics.ES11.GL.TextureParameter (`
`OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All`
`pname, Single param) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.25.2.196 `static unsafe void OpenTK.Graphics.ES11.GL.TextureParameter (`
`OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All`
`pname, Int32 *@ params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.25.2.197 `static unsafe void OpenTK.Graphics.ES11.GL.TextureParameter (OpenTK.Graphics.ES11.All target, OpenTK.Graphics.ES11.All pname, Single *@ params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. pname can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of pname.

5.25.2.198 `static void OpenTK.Graphics.ES11.GL.TextureSubImage2D (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, IntPtr pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.25.2.199 `static void OpenTK.Graphics.ES11.GL.TexSubImage2D<T8> (OpenTK.Graphics.ES11.All target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All type, [InAttribute, OutAttribute] T8[] pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

```
5.25.2.200 static void OpenTK.Graphics.ES11.GL.TexSubImage2D<
T8 > ( OpenTK.Graphics.ES11.All target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
type, [InAttribute, OutAttribute] T8 pixels[, ] ) [static]
```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

```
5.25.2.201 static void OpenTK.Graphics.ES11.GL.TexSubImage2D<
            T8 > ( OpenTK.Graphics.ES11.All target, Int32 level,
                  Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
                  OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
                  type, [InAttribute, OutAttribute] ref T8 pixels ) [static]
```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

```
5.25.2.202 static void OpenTK.Graphics.ES11.GL.TexSubImage2D<
T8 > ( OpenTK.Graphics.ES11.All target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES11.All format, OpenTK.Graphics.ES11.All
type, [InAttribute, OutAttribute] T8 pixels[,] ) [static]
```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.25.2.203 static void OpenTK.Graphics.ES11.GL.Translate (Single x, Single y, Single z) [static]

Multiply the current matrix by a translation matrix.

Parameters

x Specify the x, y, and z coordinates of a translation vector.

5.25.2.204 static void OpenTK.Graphics.ES11.GL.VertexPointer (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride, IntPtr pointer) [static]

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

5.25.2.205 `static void OpenTK.Graphics.ES11.GL.VertexPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] T3[] pointer) [static]`

Define an array of vertex data.

Parameters

- size* Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.206 `static void OpenTK.Graphics.ES11.GL.VertexPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of vertex data.

Parameters

- size* Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

5.25.2.207 `static void OpenTK.Graphics.ES11.GL.VertexPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of vertex data.

Parameters

- size* Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.25.2.208 `static void OpenTK.Graphics.ES11.GL.VertexPointer< T3 >
 (Int32 size, OpenTK.Graphics.ES11.All type, Int32 stride,
 [InAttribute, OutAttribute] ref T3 pointer) [static]`

Define an array of vertex data.

Parameters

- size* Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.25.2.209 `static void OpenTK.Graphics.ES11.GL.Viewport (Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Set the viewport.

Parameters

- x** Specify the lower left corner of the viewport rectangle, in pixels. The initial value is (0,0).
- width** Specify the width and height of the viewport. When a [GL](#) context is first attached to a window, width and height are set to the dimensions of that window.

5.25.3 Property Documentation

5.25.3.1 `override object OpenTK.Graphics.ES11.GL.SyncRoot [get, protected]`

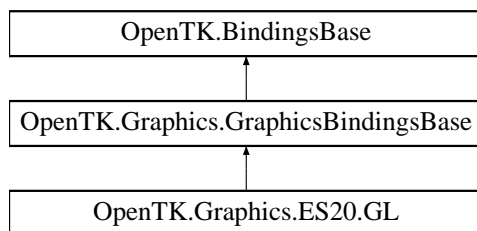
Returns a synchronization token unique for the [GL](#) class.

Reimplemented from [OpenTK.BindingsBase](#).

5.26 OpenTK.Graphics.ES20.GL Class Reference

Provides access to [OpenGL](#) ES 2.0 methods.

Inheritance diagram for OpenTK.Graphics.ES20.GL:



Static Public Member Functions

- static void [ActiveTexture](#) (OpenTK.Graphics.ES20.TextureUnit texture)
Select active texture unit.
- static void [AttachShader](#) (Int32 program, Int32 shader)

Attaches a shader object to a program object.

- static void [AttachShader](#) (UInt32 program, UInt32 shader)
Attaches a shader object to a program object.
- static void [BindAttribLocation](#) (Int32 program, Int32 index, String name)
Associates a generic vertex attribute index with a named attribute variable.
- static void [BindAttribLocation](#) (UInt32 program, UInt32 index, String name)
Associates a generic vertex attribute index with a named attribute variable.
- static void [BindBuffer](#) (OpenTK.Graphics.ES20.BufferTarget target, Int32 buffer)
Bind a named buffer object.
- static void [BindBuffer](#) (OpenTK.Graphics.ES20.BufferTarget target, UInt32 buffer)
Bind a named buffer object.
- static void **BindFramebuffer** (OpenTK.Graphics.ES20.FramebufferTarget target, Int32 framebuffer)
- static void **BindFramebuffer** (OpenTK.Graphics.ES20.FramebufferTarget target, UInt32 framebuffer)
- static void **BindRenderbuffer** (OpenTK.Graphics.ES20.RenderbufferTarget target, Int32 renderbuffer)
- static void **BindRenderbuffer** (OpenTK.Graphics.ES20.RenderbufferTarget target, UInt32 renderbuffer)
- static void [BindTexture](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 texture)
Bind a named texture to a texturing target.
- static void [BindTexture](#) (OpenTK.Graphics.ES20.TextureTarget target, UInt32 texture)
Bind a named texture to a texturing target.
- static void [BlendColor](#) (Single red, Single green, Single blue, Single alpha)
Set the blend color.
- static void [BlendEquation](#) (OpenTK.Graphics.ES20.BlendEquationMode mode)
Specify the equation used for both the RGB blend equation and the Alpha blend equation.

- static void [BlendEquationSeparate](#) (OpenTK.Graphics.ES20.BlendEquationMode modeRGB, OpenTK.Graphics.ES20.BlendEquationMode modeAlpha)
Set the RGB blend equation and the alpha blend equation separately.
- static void [BlendFunc](#) (OpenTK.Graphics.ES20.BlendingFactorSrc sfactor, OpenTK.Graphics.ES20.BlendingFactorDest dfactor)
Specify pixel arithmetic.
- static void [BlendFuncSeparate](#) (OpenTK.Graphics.ES20.BlendingFactorSrc srcRGB, OpenTK.Graphics.ES20.BlendingFactorDest dstRGB, OpenTK.Graphics.ES20.BlendingFactorSrc srcAlpha, OpenTK.Graphics.ES20.BlendingFactorDest dstAlpha)
Specify pixel arithmetic for RGB and alpha components separately.
- static void [BufferData](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, IntPtr data, OpenTK.Graphics.ES20.BufferUsage usage)
Creates and initializes a buffer object's data store.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] T2[] data, OpenTK.Graphics.ES20.BufferUsage usage)
Creates and initializes a buffer object's data store.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] T2[,] data, OpenTK.Graphics.ES20.BufferUsage usage)
Creates and initializes a buffer object's data store.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] T2[,.] data, OpenTK.Graphics.ES20.BufferUsage usage)
Creates and initializes a buffer object's data store.
- static void [BufferData< T2 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] ref T2 data, OpenTK.Graphics.ES20.BufferUsage usage)
Creates and initializes a buffer object's data store.
- static void [BufferSubData](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, IntPtr data)
Updates a subset of a buffer object's data store.
- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] T3[] data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[,] data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[,] data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData< T3 >](#) (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] ref T3 data)

Updates a subset of a buffer object's data store.

- static OpenTK.Graphics.ES20.FramebufferErrorCode **CheckFramebufferStatus** (OpenTK.Graphics.ES20.FramebufferTarget target)
- static void [Clear](#) (OpenTK.Graphics.ES20.ClearBufferMask mask)

Clear buffers to preset values.

- static void [ClearColor](#) (Single red, Single green, Single blue, Single alpha)

Specify clear values for the color buffers.

- static void [ClearDepth](#) (Single depth)

Specify the clear value for the depth buffer.

- static void [ClearStencil](#) (Int32 s)

Specify the clear value for the stencil buffer.

- static void [ColorMask](#) (bool red, bool green, bool blue, bool alpha)

Enable and disable writing of frame buffer color components.

- static void [CompileShader](#) (Int32 shader)

Compiles a shader object.

- static void [CompileShader](#) (UInt32 shader)

Compiles a shader object.

- static void [CompressedTexImage2D](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level,
OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[]
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level,
OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[]
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level,
OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[,]
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level,
OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] ref T7
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexSubImage2D](#) (OpenTK.Graphics.ES20.TextureTarget
target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize, IntPtr data)

Specify a two-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset,
Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat
format, Int32 imageSize,[InAttribute, OutAttribute] T8[] data)

Specify a two-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset,
Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat
format, Int32 imageSize,[InAttribute, OutAttribute] T8[,] data)

Specify a two-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T8[,.] data)

Specify a two-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] ref T8 data)

Specify a two-dimensional texture subimage in a compressed format.

- static void [CopyTexImage2D](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width, Int32 height, Int32 border)

Copy pixels into a 2D texture image.

- static void [CopyTexSubImage2D](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height)

Copy a two-dimensional texture subimage.

- static Int32 [CreateProgram](#) ()

Creates a program object.

- static Int32 [CreateShader](#) (OpenTK.Graphics.ES20.ShaderType type)

Creates a shader object.

- static void [CullFace](#) (OpenTK.Graphics.ES20.CullFaceMode mode)

Specify whether front- or back-facing facets can be culled.

- static void [DeleteBuffers](#) (Int32 n, Int32[] buffers)

Delete named buffer objects.

- static void [DeleteBuffers](#) (Int32 n, ref Int32 buffers)

Delete named buffer objects.

- static unsafe void [DeleteBuffers](#) (Int32 n, Int32 *buffers)

Delete named buffer objects.

- static void [DeleteBuffers](#) (Int32 n, UInt32[] buffers)

Delete named buffer objects.

- static void [DeleteBuffers](#) (Int32 n, ref UInt32 buffers)
Delete named buffer objects.
- static unsafe void [DeleteBuffers](#) (Int32 n, UInt32 *buffers)
Delete named buffer objects.
- static void **DeleteFramebuffers** (Int32 n, Int32[] framebuffers)
- static void **DeleteFramebuffers** (Int32 n, ref Int32 framebuffers)
- static unsafe void **DeleteFramebuffers** (Int32 n, Int32 *framebuffers)
- static void **DeleteFramebuffers** (Int32 n, UInt32[] framebuffers)
- static void **DeleteFramebuffers** (Int32 n, ref UInt32 framebuffers)
- static unsafe void **DeleteFramebuffers** (Int32 n, UInt32 *framebuffers)
- static void [DeleteProgram](#) (Int32 program)
Deletes a program object.
- static void [DeleteProgram](#) (UInt32 program)
Deletes a program object.
- static void **DeleteRenderbuffers** (Int32 n, Int32[] renderbuffers)
- static void **DeleteRenderbuffers** (Int32 n, ref Int32 renderbuffers)
- static unsafe void **DeleteRenderbuffers** (Int32 n, Int32 *renderbuffers)
- static void **DeleteRenderbuffers** (Int32 n, UInt32[] renderbuffers)
- static void **DeleteRenderbuffers** (Int32 n, ref UInt32 renderbuffers)
- static unsafe void **DeleteRenderbuffers** (Int32 n, UInt32 *renderbuffers)
- static void [DeleteShader](#) (Int32 shader)
Deletes a shader object.
- static void [DeleteShader](#) (UInt32 shader)
Deletes a shader object.
- static void [DeleteTextures](#) (Int32 n, Int32[] textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, ref Int32 textures)
Delete named textures.
- static unsafe void [DeleteTextures](#) (Int32 n, Int32 *textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, UInt32[] textures)
Delete named textures.

- static void [DeleteTextures](#) (Int32 n, ref UInt32 textures)
Delete named textures.
- static unsafe void [DeleteTextures](#) (Int32 n, UInt32 *textures)
Delete named textures.
- static void [DepthFunc](#) (OpenTK.Graphics.ES20.DepthFunction func)
Specify the value used for depth buffer comparisons.
- static void [DepthMask](#) (bool flag)
Enable or disable writing into the depth buffer.
- static void [DepthRange](#) (Single zNear, Single zFar)
Specify mapping of depth values from normalized device coordinates to window coordinates.
- static void [DetachShader](#) (Int32 program, Int32 shader)
Detaches a shader object from a program object to which it is attached.
- static void [DetachShader](#) (UInt32 program, UInt32 shader)
Detaches a shader object from a program object to which it is attached.
- static void **Disable** (OpenTK.Graphics.ES20.EnableCap cap)
- static void **DisableDriverControlQCOM** (Int32 driverControl)
- static void **DisableDriverControlQCOM** (UInt32 driverControl)
- static void **DisableVertexAttribArray** (Int32 index)
- static void **DisableVertexAttribArray** (UInt32 index)
- static void [DrawArrays](#) (OpenTK.Graphics.ES20.BeginMode mode, Int32 first, Int32 count)
Render primitives from array data.
- static void [DrawElements](#) (OpenTK.Graphics.ES20.BeginMode mode, Int32 count, OpenTK.Graphics.ES20.DrawElementsType type, IntPtr indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES20.BeginMode mode, Int32 count, OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute, OutAttribute] T3[] indices)
Render primitives from array data.
- static void [DrawElements< T3 >](#) (OpenTK.Graphics.ES20.BeginMode mode, Int32 count, OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute, OutAttribute] T3[,] indices)

Render primitives from array data.

- static void **DrawElements**< T3 > (OpenTK.Graphics.ES20.BeginMode mode, Int32 count, OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute, OutAttribute] T3[,], indices)

Render primitives from array data.

- static void **DrawElements**< T3 > (OpenTK.Graphics.ES20.BeginMode mode, Int32 count, OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute, OutAttribute] ref T3 indices)

Render primitives from array data.

- static void **Enable** (OpenTK.Graphics.ES20.EnableCap cap)

Enable or disable server-side GL capabilities.

- static void **EnableDriverControlQCOM** (Int32 driverControl)
- static void **EnableDriverControlQCOM** (UInt32 driverControl)
- static void **EnableVertexAttribArray** (Int32 index)

Enable or disable a generic vertex attribute array.

- static void **EnableVertexAttribArray** (UInt32 index)

Enable or disable a generic vertex attribute array.

- static void **Finish** ()

Block until all GL execution is complete.

- static void **Flush** ()

Force execution of GL commands in finite time.

- static void **FramebufferRenderbuffer** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.RenderbufferTarget renderbuffertarget, Int32 renderbuffer)
- static void **FramebufferRenderbuffer** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.RenderbufferTarget renderbuffertarget, UInt32 renderbuffer)
- static void **FramebufferTexture2D** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.TextureTarget textarget, Int32 texture, Int32 level)
- static void **FramebufferTexture2D** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.TextureTarget textarget, UInt32 texture, Int32 level)

- static void [FrontFace](#) (OpenTK.Graphics.ES20.FrontFaceDirection mode)
Define front- and back-facing polygons.
- static void [GenBuffers](#) (Int32 n,[OutAttribute] Int32[] buffers)
Generate buffer object names.
- static void [GenBuffers](#) (Int32 n,[OutAttribute] out Int32 buffers)
Generate buffer object names.
- static unsafe void [GenBuffers](#) (Int32 n,[OutAttribute] Int32 *buffers)
Generate buffer object names.
- static void [GenBuffers](#) (Int32 n,[OutAttribute] UInt32[] buffers)
Generate buffer object names.
- static void [GenBuffers](#) (Int32 n,[OutAttribute] out UInt32 buffers)
Generate buffer object names.
- static unsafe void [GenBuffers](#) (Int32 n,[OutAttribute] UInt32 *buffers)
Generate buffer object names.
- static void **GenerateMipmap** (OpenTK.Graphics.ES20.TextureTarget target)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] Int32[] framebuffers)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] out Int32 framebuffers)
- static unsafe void **GenFramebuffers** (Int32 n,[OutAttribute] Int32 *framebuffers)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] UInt32[] framebuffers)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] out UInt32 framebuffers)
- static unsafe void **GenFramebuffers** (Int32 n,[OutAttribute] UInt32 *framebuffers)
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] Int32[] renderbuffers)
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] out Int32 renderbuffers)
- static unsafe void **GenRenderbuffers** (Int32 n,[OutAttribute] Int32 *renderbuffers)
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] UInt32[] renderbuffers)
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] out UInt32 renderbuffers)
- static unsafe void **GenRenderbuffers** (Int32 n,[OutAttribute] UInt32 *renderbuffers)
- static void [GenTextures](#) (Int32 n,[OutAttribute] Int32[] textures)
Generate texture names.
- static void [GenTextures](#) (Int32 n,[OutAttribute] out Int32 textures)

Generate texture names.

- static unsafe void [GenTextures](#) (Int32 n,[OutAttribute] Int32 *textures)

Generate texture names.

- static void [GenTextures](#) (Int32 n,[OutAttribute] UInt32[] textures)

Generate texture names.

- static void [GenTextures](#) (Int32 n,[OutAttribute] out UInt32 textures)

Generate texture names.

- static unsafe void [GenTextures](#) (Int32 n,[OutAttribute] UInt32 *textures)

Generate texture names.

- static void [GetActiveAttrib](#) (Int32 program, Int32 index, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] Int32[] size,[OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType[] type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static void [GetActiveAttrib](#) (Int32 program, Int32 index, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.ES20.ActiveAttribType type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static unsafe void [GetActiveAttrib](#) (Int32 program, Int32 index, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType *type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static void [GetActiveAttrib](#) (UInt32 program, UInt32 index, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] Int32[] size,[OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType[] type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static void [GetActiveAttrib](#) (UInt32 program, UInt32 index, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] out Int32

size,[OutAttribute] out OpenTK.Graphics.ES20.ActiveAttribType
type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static unsafe void [GetActiveAttrib](#) (UInt32 program, UInt32 index, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType *type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static void [GetActiveUniform](#) (Int32 program, Int32 index, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] Int32[] size,[OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType[] type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static void [GetActiveUniform](#) (Int32 program, Int32 index, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.ES20.ActiveUniformType type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static unsafe void [GetActiveUniform](#) (Int32 program, Int32 index, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType *type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static void [GetActiveUniform](#) (UInt32 program, UInt32 index, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] Int32[] size,[OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType[] type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static void [GetActiveUniform](#) (UInt32 program, UInt32 index, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.ES20.ActiveUniformType type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static unsafe void **GetActiveUniform** (UInt32 program, UInt32 index, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType *type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static void **GetAttachedShaders** (Int32 program, Int32 maxcount,[OutAttribute] Int32[] count,[OutAttribute] Int32[] shaders)

Returns the handles of the shader objects attached to a program object.

- static void **GetAttachedShaders** (Int32 program, Int32 maxcount,[OutAttribute] out Int32 count,[OutAttribute] out Int32 shaders)

Returns the handles of the shader objects attached to a program object.

- static unsafe void **GetAttachedShaders** (Int32 program, Int32 maxcount,[OutAttribute] Int32 *count,[OutAttribute] Int32 *shaders)

Returns the handles of the shader objects attached to a program object.

- static void **GetAttachedShaders** (UInt32 program, Int32 maxcount,[OutAttribute] Int32[] count,[OutAttribute] UInt32[] shaders)

Returns the handles of the shader objects attached to a program object.

- static void **GetAttachedShaders** (UInt32 program, Int32 maxcount,[OutAttribute] out Int32 count,[OutAttribute] out UInt32 shaders)

Returns the handles of the shader objects attached to a program object.

- static unsafe void **GetAttachedShaders** (UInt32 program, Int32 maxcount,[OutAttribute] Int32 *count,[OutAttribute] UInt32 *shaders)

Returns the handles of the shader objects attached to a program object.

- static int **GetAttribLocation** (Int32 program, String name)

Returns the location of an attribute variable.

- static int **GetAttribLocation** (UInt32 program, String name)

Returns the location of an attribute variable.

- static void **GetBoolean** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] bool[] @params)

- static void **GetBoolean** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] out bool @params)

- static unsafe void **GetBoolean** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] bool *@params)
- static void **GetBufferParameter** (OpenTK.Graphics.ES20.BufferTarget target, OpenTK.Graphics.ES20.BufferParameterName pname,[OutAttribute] Int32[] @params)

Return parameters of a buffer object.

- static void **GetBufferParameter** (OpenTK.Graphics.ES20.BufferTarget target, OpenTK.Graphics.ES20.BufferParameterName pname,[OutAttribute] out Int32 @params)

Return parameters of a buffer object.

- static unsafe void **GetBufferParameter** (OpenTK.Graphics.ES20.BufferTarget target, OpenTK.Graphics.ES20.BufferParameterName pname,[OutAttribute] Int32 *@params)

Return parameters of a buffer object.

- static void **GetDriverControlsQCOM** ([OutAttribute] Int32[] num, Int32 size,[OutAttribute] Int32[] driverControls)
- static void **GetDriverControlsQCOM** ([OutAttribute] Int32[] num, Int32 size,[OutAttribute] UInt32[] driverControls)
- static void **GetDriverControlsQCOM** ([OutAttribute] out Int32 num, Int32 size,[OutAttribute] out Int32 driverControls)
- static void **GetDriverControlsQCOM** ([OutAttribute] out Int32 num, Int32 size,[OutAttribute] out UInt32 driverControls)
- static unsafe void **GetDriverControlsQCOM** ([OutAttribute] Int32 *num, Int32 size,[OutAttribute] Int32 *driverControls)
- static unsafe void **GetDriverControlsQCOM** ([OutAttribute] Int32 *num, Int32 size,[OutAttribute] UInt32 *driverControls)
- static void **GetDriverControlStringQCOM** (Int32 driverControl, Int32 bufSize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder driverControlString)
- static void **GetDriverControlStringQCOM** (Int32 driverControl, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder driverControlString)
- static unsafe void **GetDriverControlStringQCOM** (Int32 driverControl, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder driverControlString)
- static void **GetDriverControlStringQCOM** (UInt32 driverControl, Int32 bufSize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder driverControlString)
- static void **GetDriverControlStringQCOM** (UInt32 driverControl, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder driverControlString)

- static unsafe void **GetDriverControlStringQCOM** (UInt32 driverControl, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder driverControlString)
- static OpenTK.Graphics.ES20.ErrorCode **GetError** ()

Return error information.

- static void **GetFloat** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] Single[] @params)
- static void **GetFloat** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] out Single @params)
- static unsafe void **GetFloat** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] Single *@params)
- static void **GetFramebufferAttachmentParameter** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.FramebufferParameterName pname,[OutAttribute] Int32[] @params)
- static void **GetFramebufferAttachmentParameter** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.FramebufferParameterName pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetFramebufferAttachmentParameter** (OpenTK.Graphics.ES20.FramebufferTarget target, OpenTK.Graphics.ES20.FramebufferSlot attachment, OpenTK.Graphics.ES20.FramebufferParameterName pname,[OutAttribute] Int32 *@params)
- static void **GetInteger** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] Int32[] @params)
- static void **GetInteger** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetInteger** (OpenTK.Graphics.ES20.GetPName pname,[OutAttribute] Int32 *@params)
- static void **GetProgramInfoLog** (Int32 program, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder infolog)

Returns the information log for a program object.

- static void **GetProgramInfoLog** (Int32 program, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infolog)

Returns the information log for a program object.

- static unsafe void **GetProgramInfoLog** (Int32 program, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infolog)

Returns the information log for a program object.

- static void [GetProgramInfoLog](#) (UInt32 program, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder infolog)
Returns the information log for a program object.
- static void [GetProgramInfoLog](#) (UInt32 program, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infolog)
Returns the information log for a program object.
- static unsafe void [GetProgramInfoLog](#) (UInt32 program, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infolog)
Returns the information log for a program object.
- static void [GetProgram](#) (Int32 program, OpenTK.Graphics.ES20.ProgramParameter pname,[OutAttribute] Int32[]@params)
Returns a parameter from a program object.
- static void [GetProgram](#) (Int32 program, OpenTK.Graphics.ES20.ProgramParameter pname,[OutAttribute] out Int32 @params)
Returns a parameter from a program object.
- static unsafe void [GetProgram](#) (Int32 program, OpenTK.Graphics.ES20.ProgramParameter pname,[OutAttribute] Int32 *@params)
Returns a parameter from a program object.
- static void [GetProgram](#) (UInt32 program, OpenTK.Graphics.ES20.ProgramParameter pname,[OutAttribute] Int32[]@params)
Returns a parameter from a program object.
- static void [GetProgram](#) (UInt32 program, OpenTK.Graphics.ES20.ProgramParameter pname,[OutAttribute] out Int32 @params)
Returns a parameter from a program object.
- static unsafe void [GetProgram](#) (UInt32 program, OpenTK.Graphics.ES20.ProgramParameter pname,[OutAttribute] Int32 *@params)
Returns a parameter from a program object.
- static void **GetRenderbufferParameter** (OpenTK.Graphics.ES20.RenderbufferTarget target, OpenTK.Graphics.ES20.RenderbufferParameterName pname,[OutAttribute] Int32[]@params)

- static void **GetRenderbufferParameter** (OpenTK.Graphics.ES20.RenderbufferTarget target, OpenTK.Graphics.ES20.RenderbufferParameterName pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetRenderbufferParameter** (OpenTK.Graphics.ES20.RenderbufferTarget target, OpenTK.Graphics.ES20.RenderbufferParameterName pname,[OutAttribute] Int32 *@params)
- static void **GetShaderInfoLog** (Int32 shader, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder infolog)
Returns the information log for a shader object.
- static void **GetShaderInfoLog** (Int32 shader, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infolog)
Returns the information log for a shader object.
- static unsafe void **GetShaderInfoLog** (Int32 shader, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infolog)
Returns the information log for a shader object.
- static void **GetShaderInfoLog** (UInt32 shader, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder infolog)
Returns the information log for a shader object.
- static void **GetShaderInfoLog** (UInt32 shader, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infolog)
Returns the information log for a shader object.
- static unsafe void **GetShaderInfoLog** (UInt32 shader, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infolog)
Returns the information log for a shader object.
- static void **GetShader** (Int32 shader, OpenTK.Graphics.ES20.ShaderParameter pname,[OutAttribute] Int32[] @params)
Returns a parameter from a shader object.
- static void **GetShader** (Int32 shader, OpenTK.Graphics.ES20.ShaderParameter pname,[OutAttribute] out Int32 @params)
Returns a parameter from a shader object.
- static unsafe void **GetShader** (Int32 shader, OpenTK.Graphics.ES20.ShaderParameter pname,[OutAttribute] Int32 *@params)
Returns a parameter from a shader object.

- static void [GetShader](#) (UInt32 shader, OpenTK.Graphics.ES20.ShaderParameter pname,[OutAttribute] Int32[]@params)
Returns a parameter from a shader object.
- static void [GetShader](#) (UInt32 shader, OpenTK.Graphics.ES20.ShaderParameter pname,[OutAttribute] out Int32 @params)
Returns a parameter from a shader object.
- static unsafe void [GetShader](#) (UInt32 shader, OpenTK.Graphics.ES20.ShaderParameter pname,[OutAttribute] Int32 *@params)
Returns a parameter from a shader object.
- static void **GetShaderPrecisionFormat** (OpenTK.Graphics.ES20.ShaderType shadertype, OpenTK.Graphics.ES20.ShaderPrecision precisiontype,[OutAttribute] Int32[] range,[OutAttribute] Int32[] precision)
- static void **GetShaderPrecisionFormat** (OpenTK.Graphics.ES20.ShaderType shadertype, OpenTK.Graphics.ES20.ShaderPrecision precisiontype,[OutAttribute] out Int32 range,[OutAttribute] out Int32 precision)
- static unsafe void **GetShaderPrecisionFormat** (OpenTK.Graphics.ES20.ShaderType shadertype, OpenTK.Graphics.ES20.ShaderPrecision precisiontype,[OutAttribute] Int32 *range,[OutAttribute] Int32 *precision)
- static void [GetShaderSource](#) (Int32 shader, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static void [GetShaderSource](#) (Int32 shader, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static unsafe void [GetShaderSource](#) (Int32 shader, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static void [GetShaderSource](#) (UInt32 shader, Int32 bufsize,[OutAttribute] Int32[] length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static void [GetShaderSource](#) (UInt32 shader, Int32 bufsize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.

- static unsafe void [GetShaderSource](#) (UInt32 shader, Int32 bufsize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static unsafe System.String [GetString](#) (OpenTK.Graphics.ES20.StringName name)
Return a string describing the current [GL](#) connection.
- static void [GetTexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname,[OutAttribute] Single[] @params)
Return texture parameter values.
- static void [GetTexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname,[OutAttribute] out Single[] @params)
Return texture parameter values.
- static unsafe void [GetTexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname,[OutAttribute] Single[] @params)
Return texture parameter values.
- static void [GetTexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname,[OutAttribute] Int32[] @params)
Return texture parameter values.
- static void [GetTexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname,[OutAttribute] out Int32[] @params)
Return texture parameter values.
- static unsafe void [GetTexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname,[OutAttribute] Int32[] @params)
Return texture parameter values.
- static void [GetUniform](#) (Int32 program, Int32 location,[OutAttribute] Single[] @params)
Returns the value of a uniform variable.
- static void [GetUniform](#) (Int32 program, Int32 location,[OutAttribute] out Single[] @params)

Returns the value of a uniform variable.

- static unsafe void [GetUniform](#) (Int32 program, Int32 location,[OutAttribute] Single *@params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] Single[] @params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] out Single @params)

Returns the value of a uniform variable.

- static unsafe void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] Single *@params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (Int32 program, Int32 location,[OutAttribute] Int32[] @params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (Int32 program, Int32 location,[OutAttribute] out Int32 @params)

Returns the value of a uniform variable.

- static unsafe void [GetUniform](#) (Int32 program, Int32 location,[OutAttribute] Int32 *@params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] Int32[] @params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] out Int32 @params)

Returns the value of a uniform variable.

- static unsafe void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] Int32 *@params)

Returns the value of a uniform variable.

- static int [GetUniformLocation](#) (Int32 program, String name)

Returns the location of a uniform variable.

- static int [GetUniformLocation](#) (UInt32 program, String name)
Returns the location of a uniform variable.
- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Single[] @params)
Return a generic vertex attribute parameter.
- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] out Single @params)
Return a generic vertex attribute parameter.
- static unsafe void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Single * @params)
Return a generic vertex attribute parameter.
- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Single[] @params)
Return a generic vertex attribute parameter.
- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] out Single @params)
Return a generic vertex attribute parameter.
- static unsafe void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Single * @params)
Return a generic vertex attribute parameter.
- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Int32[] @params)
Return a generic vertex attribute parameter.
- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] out Int32 @params)
Return a generic vertex attribute parameter.
- static unsafe void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Int32 * @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Int32[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] out Int32 @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname,[OutAttribute] Int32 *@params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttribPointer](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[OutAttribute] IntPtr pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] ref T2 pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#) (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[OutAttribute] IntPtr pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[, ,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] ref T2 pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [Hint](#) (OpenTK.Graphics.ES20.HintTarget target, OpenTK.Graphics.ES20.HintMode mode)

Specify implementation-specific hints.

- static bool [IsBuffer](#) (Int32 buffer)

Determine if a name corresponds to a buffer object.

- static bool [IsBuffer](#) (UInt32 buffer)

Determine if a name corresponds to a buffer object.

- static bool [IsEnabled](#) (OpenTK.Graphics.ES20.EnableCap cap)

Test whether a capability is enabled.

- static bool [IsFramebuffer](#) (Int32 framebuffer)
- static bool [IsFramebuffer](#) (UInt32 framebuffer)
- static bool [IsProgram](#) (Int32 program)

Determines if a name corresponds to a program object.

- static bool [IsProgram](#) (UInt32 program)

Determines if a name corresponds to a program object.

- static bool **IsRenderbuffer** (Int32 renderbuffer)
- static bool **IsRenderbuffer** (UInt32 renderbuffer)
- static bool **IsShader** (Int32 shader)
Determines if a name corresponds to a shader object.
- static bool **IsShader** (UInt32 shader)
Determines if a name corresponds to a shader object.
- static bool **IsTexture** (Int32 texture)
Determine if a name corresponds to a texture.
- static bool **IsTexture** (UInt32 texture)
Determine if a name corresponds to a texture.
- static void **LineWidth** (Single width)
Specify the width of rasterized lines.
- static void **LinkProgram** (Int32 program)
Links a program object.
- static void **LinkProgram** (UInt32 program)
Links a program object.
- static void **PixelStore** (OpenTK.Graphics.ES20.PixelStoreParameter pname, Int32 param)
Set pixel storage modes.
- static void **PolygonOffset** (Single factor, Single units)
Set the scale and units used to calculate depth values.
- static void **ReadPixels** (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, IntPtr pixels)
Read a block of pixels from the frame buffer.
- static void **ReadPixels< T6 >** (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T6[] pixels)
Read a block of pixels from the frame buffer.

- static void **ReadPixels**< **T6** > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type,[InAttribute, OutAttribute] T6[, pixels])

Read a block of pixels from the frame buffer.

- static void **ReadPixels**< **T6** > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type,[InAttribute, OutAttribute] T6[, pixels])

Read a block of pixels from the frame buffer.

- static void **ReadPixels**< **T6** > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type,[InAttribute, OutAttribute] ref T6 pixels)

Read a block of pixels from the frame buffer.

- static void **ReleaseShaderCompiler** ()
- static void **RenderbufferStorage** (OpenTK.Graphics.ES20.RenderbufferTarget target, OpenTK.Graphics.ES20.RenderbufferInternalFormat internalformat, Int32 width, Int32 height)
- static void **SampleCoverage** (Single value, bool invert)

Specify multisample coverage parameters.

- static void **Scissor** (Int32 x, Int32 y, Int32 width, Int32 height)

Define the scissor box.

- static void **ShaderBinary** (Int32 n, Int32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, IntPtr binary, Int32 length)
- static void **ShaderBinary**< **T3** > (Int32 n, Int32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[] binary, Int32 length)
- static void **ShaderBinary**< **T3** > (Int32 n, Int32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< **T3** > (Int32 n, Int32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< **T3** > (Int32 n, Int32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] ref T3 binary, Int32 length)

- static void **ShaderBinary** (Int32 n, ref Int32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, IntPtr binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref Int32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref Int32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref Int32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref Int32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] ref T3 binary, Int32 length)
- static unsafe void **ShaderBinary** (Int32 n, Int32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, IntPtr binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, Int32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[] binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, Int32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, Int32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, Int32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] ref T3 binary, Int32 length)
- static void **ShaderBinary** (Int32 n, UInt32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, IntPtr binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, UInt32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, UInt32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, UInt32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, UInt32[] shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat,[InAttribute, OutAttribute] ref T3 binary, Int32 length)

- static void **ShaderBinary** (Int32 n, ref UInt32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, IntPtr binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref UInt32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] T3[] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref UInt32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref UInt32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static void **ShaderBinary**< T3 > (Int32 n, ref UInt32 shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] ref T3 binary, Int32 length)
- static unsafe void **ShaderBinary** (Int32 n, UInt32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, IntPtr binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, UInt32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] T3[] binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, UInt32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, UInt32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] T3[,] binary, Int32 length)
- static unsafe void **ShaderBinary**< T3 > (Int32 n, UInt32 *shaders, OpenTK.Graphics.ES20.ShaderBinaryFormat binaryformat, [InAttribute, OutAttribute] ref T3 binary, Int32 length)
- static void **ShaderSource** (Int32 shader, Int32 count, String[]@string, Int32[] length)

Replaces the source code in a shader object.
- static void **ShaderSource** (Int32 shader, Int32 count, String[]@string, ref Int32 length)

Replaces the source code in a shader object.
- static unsafe void **ShaderSource** (Int32 shader, Int32 count, String[]@string, Int32 *length)

Replaces the source code in a shader object.
- static void **ShaderSource** (UInt32 shader, Int32 count, String[]@string, Int32[] length)

Replaces the source code in a shader object.

- static void [ShaderSource](#) (UInt32 shader, Int32 count, String[]@string, ref Int32 length)

Replaces the source code in a shader object.

- static unsafe void [ShaderSource](#) (UInt32 shader, Int32 count, String[]@string, Int32 *length)

Replaces the source code in a shader object.

- static void [StencilFunc](#) (OpenTK.Graphics.ES20.StencilFunction func, Int32 @ref, Int32 mask)

Set front and back function and reference value for stencil testing.

- static void [StencilFunc](#) (OpenTK.Graphics.ES20.StencilFunction func, Int32 @ref, UInt32 mask)

Set front and back function and reference value for stencil testing.

- static void [StencilFuncSeparate](#) (OpenTK.Graphics.ES20.CullFaceMode face, OpenTK.Graphics.ES20.StencilFunction func, Int32 @ref, Int32 mask)

Set front and/or back function and reference value for stencil testing.

- static void [StencilFuncSeparate](#) (OpenTK.Graphics.ES20.CullFaceMode face, OpenTK.Graphics.ES20.StencilFunction func, Int32 @ref, UInt32 mask)

Set front and/or back function and reference value for stencil testing.

- static void [StencilMask](#) (Int32 mask)

Control the front and back writing of individual bits in the stencil planes.

- static void [StencilMask](#) (UInt32 mask)

Control the front and back writing of individual bits in the stencil planes.

- static void [StencilMaskSeparate](#) (OpenTK.Graphics.ES20.CullFaceMode face, Int32 mask)

Control the front and/or back writing of individual bits in the stencil planes.

- static void [StencilMaskSeparate](#) (OpenTK.Graphics.ES20.CullFaceMode face, UInt32 mask)

Control the front and/or back writing of individual bits in the stencil planes.

- static void [StencilOp](#) (OpenTK.Graphics.ES20.StencilOp fail, OpenTK.Graphics.ES20.StencilOp zfail, OpenTK.Graphics.ES20.StencilOp zpass)

Set front and back stencil test actions.

- static void [StencilOpSeparate](#) (OpenTK.Graphics.ES20.CullFaceMode face, OpenTK.Graphics.ES20.StencilOp fail, OpenTK.Graphics.ES20.StencilOp zfail, OpenTK.Graphics.ES20.StencilOp zpass)

Set front and/or back stencil test actions.

- static void [TexImage2D](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, IntPtr pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8[,], pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture image.

- static void [TexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.TextureParameterName pname, Single param)

Set texture parameters.

- static void [TexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.TextureParameterName pname, Single[] @params)
Set texture parameters.
- static unsafe void [TexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.TextureParameterName pname, Single * @params)
Set texture parameters.
- static void [TexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.TextureParameterName pname, Int32 param)
Set texture parameters.
- static void [TexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.TextureParameterName pname, Int32[] @params)
Set texture parameters.
- static unsafe void [TexParameter](#) (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.TextureParameterName pname, Int32 * @params)
Set texture parameters.
- static void [TexSubImage2D](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, IntPtr pixels)
Specify a two-dimensional texture subimage.
- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8[] pixels)
Specify a two-dimensional texture subimage.
- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8[,] pixels)
Specify a two-dimensional texture subimage.
- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format,

OpenTK.Graphics.ES20.PixelType type,[InAttribute, OutAttribute] T8[,
pixels)

Specify a two-dimensional texture subimage.

- static void **TexSubImage2D**< T8 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type,[InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture subimage.

- static void **Uniform1** (Int32 location, Single x)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, Single[] v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, ref Single v)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform1** (Int32 location, Int32 count, Single *v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 x)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, Int32[] v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, ref Int32 v)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform1** (Int32 location, Int32 count, Int32 *v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Single x, Single y)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Int32 count, Single[] v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Int32 count, ref Single v)
Specify the value of a uniform variable for the current program object.

- static unsafe void [Uniform2](#) (Int32 location, Int32 count, Single *v)
Specify the value of a uniform variable for the current program object.
- static void [Uniform2](#) (Int32 location, Int32 x, Int32 y)
Specify the value of a uniform variable for the current program object.
- static void [Uniform2](#) (Int32 location, Int32 count, Int32[] v)
Specify the value of a uniform variable for the current program object.
- static unsafe void [Uniform2](#) (Int32 location, Int32 count, Int32 *v)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Single x, Single y, Single z)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, Single[] v)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, ref Single v)
Specify the value of a uniform variable for the current program object.
- static unsafe void [Uniform3](#) (Int32 location, Int32 count, Single *v)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 x, Int32 y, Int32 z)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, Int32[] v)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, ref Int32 v)
Specify the value of a uniform variable for the current program object.
- static unsafe void [Uniform3](#) (Int32 location, Int32 count, Int32 *v)
Specify the value of a uniform variable for the current program object.
- static void [Uniform4](#) (Int32 location, Single x, Single y, Single z, Single w)
Specify the value of a uniform variable for the current program object.
- static void [Uniform4](#) (Int32 location, Int32 count, Single[] v)
Specify the value of a uniform variable for the current program object.

- static void **Uniform4** (Int32 location, Int32 count, ref Single v)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform4** (Int32 location, Int32 count, Single *v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 x, Int32 y, Int32 z, Int32 w)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, Int32[] v)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, ref Int32 v)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform4** (Int32 location, Int32 count, Int32 *v)
Specify the value of a uniform variable for the current program object.
- static void **UniformMatrix2** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix2** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix2** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix3** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix3** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix3** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix4** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix4** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix4** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UseProgram** (Int32 program)
Installs a program object as part of current rendering state.
- static void **UseProgram** (UInt32 program)
Installs a program object as part of current rendering state.

- static void [ValidateProgram](#) (Int32 program)
Validates a program object.
- static void [ValidateProgram](#) (UInt32 program)
Validates a program object.
- static void [VertexAttrib1](#) (Int32 indx, Single x)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (UInt32 indx, Single x)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (Int32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (Int32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (UInt32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (UInt32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 indx, Single x, Single y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 indx, Single x, Single y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 indx, ref Single values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (Int32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 indx, Single[] values)
Specifies the value of a generic vertex attribute.

- static void [VertexAttrib2](#) (UInt32 indx, ref Single values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (UInt32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 indx, Single x, Single y, Single z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 indx, Single x, Single y, Single z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 indx, ref Single values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (Int32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 indx, ref Single values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (UInt32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 indx, Single x, Single y, Single z, Single w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 indx, Single x, Single y, Single z, Single w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 indx, ref Single values)
Specifies the value of a generic vertex attribute.

- static unsafe void [VertexAttrib4](#) (Int32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 indx, Single[] values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 indx, ref Single values)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 indx, Single *values)
Specifies the value of a generic vertex attribute.
- static void [VertexAttribPointer](#) (Int32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride, IntPtr ptr)
Define an array of generic vertex attribute data.
- static void [VertexAttribPointer< T5 >](#) (Int32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[] ptr)
Define an array of generic vertex attribute data.
- static void [VertexAttribPointer< T5 >](#) (Int32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,] ptr)
Define an array of generic vertex attribute data.
- static void [VertexAttribPointer< T5 >](#) (Int32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,.] ptr)
Define an array of generic vertex attribute data.
- static void [VertexAttribPointer< T5 >](#) (Int32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride,[InAttribute, OutAttribute] ref T5 ptr)
Define an array of generic vertex attribute data.
- static void [VertexAttribPointer](#) (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride, IntPtr ptr)
Define an array of generic vertex attribute data.

- static void [VertexAttribPointer](#)< [T5](#) > (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[] ptr)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer](#)< [T5](#) > (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,] ptr)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer](#)< [T5](#) > (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,], ptr)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer](#)< [T5](#) > (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] ref T5 ptr)

Define an array of generic vertex attribute data.

- static void [Viewport](#) (Int32 x, Int32 y, Int32 width, Int32 height)

Set the viewport.

- static void **ClearColor** (System.Drawing.Color color)
- static void **ClearColor** ([Color4](#) color)
- static void **BlendColor** (System.Drawing.Color color)
- static void **BlendColor** ([Color4](#) color)
- static void **Uniform2** (int location, ref [Vector2](#) vector)
- static void **Uniform3** (int location, ref [Vector3](#) vector)
- static void **Uniform4** (int location, ref [Vector4](#) vector)
- static void **Uniform2** (int location, [Vector2](#) vector)
- static void **Uniform3** (int location, [Vector3](#) vector)
- static void **Uniform4** (int location, [Vector4](#) vector)
- static void **Uniform4** (int location, [Color4](#) color)
- static void **Uniform4** (int location, [Quaternion](#) quaternion)
- static void **UniformMatrix4** (int location, bool transpose, ref [Matrix4](#) matrix)
- static string **GetActiveAttrib** (int program, int index, out int size, out ActiveAttribType type)
- static string **GetActiveUniform** (int program, int uniformIndex, out int size, out ActiveUniformType type)
- static void **ShaderSource** (Int32 shader, System.String @string)
- static string **GetShaderInfoLog** (Int32 shader)
- static void **GetShaderInfoLog** (Int32 shader, out string info)

- static string **GetProgramInfoLog** (Int32 program)
- static void **GetProgramInfoLog** (Int32 program, out string info)
- static void **VertexAttrib2** (Int32 index, ref [Vector2](#) v)
- static void **VertexAttrib3** (Int32 index, ref [Vector3](#) v)
- static void **VertexAttrib4** (Int32 index, ref [Vector4](#) v)
- static void **VertexAttrib2** (Int32 index, [Vector2](#) v)
- static void **VertexAttrib3** (Int32 index, [Vector3](#) v)
- static void **VertexAttrib4** (Int32 index, [Vector4](#) v)
- static void **VertexAttribPointer** (int index, int size, VertexAttribPointerType type, bool normalized, int stride, int offset)
- static void **VertexAttribPointer** (uint index, int size, VertexAttribPointerType type, bool normalized, int stride, int offset)
- static void **DrawElements** (BeginMode mode, int count, DrawElementsType type, int offset)
- static int **GenTexture** ()
- static void **DeleteTexture** (int id)
- static void **GetFloat** (GetPName pname, out [Vector2](#) vector)
- static void **GetFloat** (GetPName pname, out [Vector3](#) vector)
- static void **GetFloat** (GetPName pname, out [Vector4](#) vector)
- static void **GetFloat** (GetPName pname, out [Matrix4](#) matrix)
- static void **Viewport** (System.Drawing.Size size)
- static void **Viewport** (System.Drawing.Point location, System.Drawing.Size size)
- static void **Viewport** (System.Drawing.Rectangle rectangle)

Properties

- override object [SyncRoot](#) [get]
Returns a synchronization token unique for the [GL](#) class.

5.26.1 Detailed Description

Provides access to [OpenGL](#) ES 2.0 methods.

5.26.2 Member Function Documentation

5.26.2.1 static void OpenTK.Graphics.ES20.GL.ActiveTexture (OpenTK.Graphics.ES20.TextureUnit texture) [static]

Select active texture unit.

Parameters

texture Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. texture must be one of GL_TEXTURE*i*, where *i* ranges from 0 to the larger of (GL_MAX_TEXTURE_COORDS - 1) and (GL_MAX_COMBINED_TEXTURE_IMAGE_UNITS - 1). The initial value is GL_TEXTURE0.

5.26.2.2 static void OpenTK.Graphics.ES20.GL.AttachShader (Int32 program, Int32 shader) [static]

Attaches a shader object to a program object.

Parameters

program Specifies the program object to which a shader object will be attached.

shader Specifies the shader object that is to be attached.

5.26.2.3 static void OpenTK.Graphics.ES20.GL.AttachShader (UInt32 program, UInt32 shader) [static]

Attaches a shader object to a program object.

Parameters

program Specifies the program object to which a shader object will be attached.

shader Specifies the shader object that is to be attached.

5.26.2.4 static void OpenTK.Graphics.ES20.GL.BindAttribLocation (Int32 program, Int32 index, String name) [static]

Associates a generic vertex attribute index with a named attribute variable.

Parameters

program Specifies the handle of the program object in which the association is to be made.

index Specifies the index of the generic vertex attribute to be bound.

name Specifies a null terminated string containing the name of the vertex shader attribute variable to which index is to be bound.

5.26.2.5 static void OpenTK.Graphics.ES20.GL.BindAttribLocation (UInt32 *program*, UInt32 *index*, String *name*) [static]

Associates a generic vertex attribute index with a named attribute variable.

Parameters

program Specifies the handle of the program object in which the association is to be made.

index Specifies the index of the generic vertex attribute to be bound.

name Specifies a null terminated string containing the name of the vertex shader attribute variable to which index is to be bound.

5.26.2.6 static void OpenTK.Graphics.ES20.GL.BindBuffer (OpenTK.Graphics.ES20.BufferTarget *target*, Int32 *buffer*) [static]

Bind a named buffer object.

Parameters

target Specifies the target to which the buffer object is bound. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

buffer Specifies the name of a buffer object.

5.26.2.7 static void OpenTK.Graphics.ES20.GL.BindBuffer (OpenTK.Graphics.ES20.BufferTarget *target*, UInt32 *buffer*) [static]

Bind a named buffer object.

Parameters

target Specifies the target to which the buffer object is bound. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

buffer Specifies the name of a buffer object.

5.26.2.8 `static void OpenTK.Graphics.ES20.GL.BindTexture (`
`OpenTK.Graphics.ES20.TextureTarget target, Int32 texture)`
`[static]`

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.26.2.9 `static void OpenTK.Graphics.ES20.GL.BindTexture (`
`OpenTK.Graphics.ES20.TextureTarget target, UInt32 texture)`
`[static]`

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.26.2.10 `static void OpenTK.Graphics.ES20.GL.BlendColor (Single red,`
`Single green, Single blue, Single alpha) [static]`

Set the blend color.

Parameters

red specify the components of GL_BLEND_COLOR

5.26.2.11 `static void OpenTK.Graphics.ES20.GL.BlendEquation (`
`OpenTK.Graphics.ES20.BlendEquationMode mode) [static]`

Specify the equation used for both the RGB blend equation and the Alpha blend equation.

Parameters

mode specifies how source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.26.2.12 static void OpenTK.Graphics.ES20.GL.BlendEquationSeparate (OpenTK.Graphics.ES20.BlendEquationMode *modeRGB*, OpenTK.Graphics.ES20.BlendEquationMode *modeAlpha*) [static]

Set the RGB blend equation and the alpha blend equation separately.

Parameters

modeRGB specifies the RGB blend equation, how the red, green, and blue components of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

modeAlpha specifies the alpha blend equation, how the alpha component of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.26.2.13 static void OpenTK.Graphics.ES20.GL.BlendFunc (OpenTK.Graphics.ES20.BlendingFactorSrc *sfactor*, OpenTK.Graphics.ES20.BlendingFactorDest *dfactor*) [static]

Specify pixel arithmetic.

Parameters

sfactor Specifies how the red, green, blue, and alpha source blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dfactor Specifies how the red, green, blue, and alpha destination blending factors are computed. The following symbolic constants are accepted:

GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

```
5.26.2.14 static void OpenTK.Graphics.ES20.GL.BlendFuncSeparate
( OpenTK.Graphics.ES20.BlendingFactorSrc srcRGB,
  OpenTK.Graphics.ES20.BlendingFactorDest dstRGB,
  OpenTK.Graphics.ES20.BlendingFactorSrc srcAlpha,
  OpenTK.Graphics.ES20.BlendingFactorDest dstAlpha )
[static]
```

Specify pixel arithmetic for RGB and alpha components separately.

Parameters

srcRGB Specifies how the red, green, and blue blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dstRGB Specifies how the red, green, and blue destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

srcAlpha Specified how the alpha source blending factor is computed. The same symbolic constants are accepted as for srcRGB. The initial value is GL_ONE.

dstAlpha Specified how the alpha destination blending factor is computed. The same symbolic constants are accepted as for dstRGB. The initial value is GL_ZERO.

5.26.2.15 `static void OpenTK.Graphics.ES20.GL.BufferData (`
`OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, IntPtr`
`data, OpenTK.Graphics.ES20.BufferUsage usage) [static]`

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

5.26.2.16 `static void OpenTK.Graphics.ES20.GL.BufferData<`
`T2 > (OpenTK.Graphics.ES20.BufferTarget target,`
`IntPtr size, [InAttribute, OutAttribute] T2[] data,`
`OpenTK.Graphics.ES20.BufferUsage usage) [static]`

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct


```

5.26.2.17 static void OpenTK.Graphics.ES20.GL.BufferData<
           T2 > ( OpenTK.Graphics.ES20.BufferTarget target,
                 IntPtr size, [InAttribute, OutAttribute] T2 data[,],
                 OpenTK.Graphics.ES20.BufferUsage usage ) [static]

```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct

```

5.26.2.18 static void OpenTK.Graphics.ES20.GL.BufferData<
           T2 > ( OpenTK.Graphics.ES20.BufferTarget target,
                 IntPtr size, [InAttribute, OutAttribute] T2 data[,],
                 OpenTK.Graphics.ES20.BufferUsage usage ) [static]

```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints*T2 : struct*

5.26.2.19 `static void OpenTK.Graphics.ES20.GL.BufferData< T2 > (OpenTK.Graphics.ES20.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] ref T2 data, OpenTK.Graphics.ES20.BufferUsage usage) [static]`

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints*T2 : struct*

5.26.2.20 `static void OpenTK.Graphics.ES20.GL.BufferSubData (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, IntPtr data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

5.26.2.21 `static void OpenTK.Graphics.ES20.GL.BufferSubData< T3 > (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] T3[] data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.26.2.22 `static void OpenTK.Graphics.ES20.GL.BufferSubData< T3 > (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] T3 data[,]) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.26.2.23 `static void OpenTK.Graphics.ES20.GL.BufferSubData< T3 > (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] T3 data[,]) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.26.2.24 `static void OpenTK.Graphics.ES20.GL.BufferSubData< T3 > (OpenTK.Graphics.ES20.BufferTarget target, IntPtr offset, IntPtr size, [InAttribute, OutAttribute] ref T3 data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.26.2.25 `static void OpenTK.Graphics.ES20.GL.Clear (OpenTK.Graphics.ES20.ClearBufferMask mask) [static]`

Clear buffers to preset values.

Parameters

mask Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, GL_ACCUM_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

5.26.2.26 `static void OpenTK.Graphics.ES20.GL.ClearColor (Single red, Single green, Single blue, Single alpha) [static]`

Specify clear values for the color buffers.

Parameters

red Specify the red, green, blue, and alpha values used when the color buffers are cleared. The initial values are all 0.

5.26.2.27 `static void OpenTK.Graphics.ES20.GL.ClearDepth (Single depth) [static]`

Specify the clear value for the depth buffer.

Parameters

depth Specifies the depth value used when the depth buffer is cleared. The initial value is 1.

5.26.2.28 `static void OpenTK.Graphics.ES20.GL.ClearStencil (Int32 s) [static]`

Specify the clear value for the stencil buffer.

Parameters

s Specifies the index used when the stencil buffer is cleared. The initial value is 0.

5.26.2.29 `static void OpenTK.Graphics.ES20.GL.ColorMask (bool red, bool green, bool blue, bool alpha) [static]`

Enable and disable writing of frame buffer color components.

Parameters

red Specify whether red, green, blue, and alpha can or cannot be written into the frame buffer. The initial values are all GL_TRUE, indicating that the color components can be written.

5.26.2.30 `static void OpenTK.Graphics.ES20.GL.CompileShader (Int32 shader) [static]`

Compiles a shader object.

Parameters

shader Specifies the shader object to be compiled.

5.26.2.31 `static void OpenTK.Graphics.ES20.GL.CompileShader (UInt32 shader) [static]`

Compiles a shader object.

Parameters

shader Specifies the shader object to be compiled.

5.26.2.32 `static void OpenTK.Graphics.ES20.GL.CompressedTexImage2D (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.26.2.33 `static void OpenTK.Graphics.ES20.GL.CompressedTexImage2D< T7 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T7[] data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (border) for some integer . All implementations support 2D texture

images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

5.26.2.34 `static void OpenTK.Graphics.ES20.GL.CompressedTexImage2D< T7 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T7 data[,]) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (*border*) for some integer . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

5.26.2.35 `static void OpenTK.Graphics.ES20.GL.CompressedTexImage2D< T7 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T7 data[,],) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (*border*) for some integer . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2}$ (*border*) for some integer . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

5.26.2.36 `static void OpenTK.Graphics.ES20.GL.CompressedTexImage2D< T7 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] ref T7 data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n} + 2 + (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n} + 2 + (\text{border})$ for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

5.26.2.37 `static void OpenTK.Graphics.ES20.GL.CompressedTexSubImage2D
(OpenTK.Graphics.ES20.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize,
IntPtr data) [static]`

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address *data*.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by *data*.

data Specifies a pointer to the compressed image data in memory.

5.26.2.38 `static void
OpenTK.Graphics.ES20.GL.CompressedTexSubImage2D<
T8 > (OpenTK.Graphics.ES20.TextureTarget target, Int32
level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize,
[InAttribute, OutAttribute] T8[] data) [static]`

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.26.2.39 static void

```
OpenTK.Graphics.ES20.GL.CompressedTexSubImage2D<
T8 > ( OpenTK.Graphics.ES20.TextureTarget target, Int32
level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize,
[InAttribute, OutAttribute] T8 data[, ] ) [static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.26.2.40 static void

```
OpenTK.Graphics.ES20.GL.CompressedTexSubImage2D<
T8 > ( OpenTK.Graphics.ES20.TextureTarget target, Int32
level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.ES20.PixelFormat format, Int32 imageSize,
[InAttribute, OutAttribute] T8 data[,], ) [static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.26.2.41 static void
OpenTK.Graphics.ES20.GL.CompressedTexSubImage2D<
T8 > (**OpenTK.Graphics.ES20.TextureTarget** *target*, **Int32**
level, **Int32** *xoffset*, **Int32** *yoffset*, **Int32** *width*, **Int32** *height*,
OpenTK.Graphics.ES20.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] ref T8 data) [**static**]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.26.2.42 static void **OpenTK.Graphics.ES20.GL.CopyTexImage2D** (
OpenTK.Graphics.ES20.TextureTarget *target*, **Int32** *level*,
OpenTK.Graphics.ES20.PixelInternalFormat *internalformat*, **Int32**
x, **Int32** *y*, **Int32** *width*, **Int32** *height*, **Int32** *border*) [**static**]

Copy pixels into a 2D texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the internal format of the texture. Must be one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_RGB, GL_R3_G3_B2, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA8, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture image. Must be 0 or 2^{n+2} (border) for some integer n .

height Specifies the height of the texture image. Must be 0 or 2^{m+2} (border) for some integer m .

border Specifies the width of the border. Must be either 0 or 1 .

5.26.2.43 `static void OpenTK.Graphics.ES20.GL.CopyTexSubImage2D (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Copy a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

5.26.2.44 `static Int32 OpenTK.Graphics.ES20.GL.CreateProgram ()`
[static]

Creates a program object.

5.26.2.45 `static Int32 OpenTK.Graphics.ES20.GL.CreateShader (`
`OpenTK.Graphics.ES20.ShaderType type) [static]`

Creates a shader object.

Parameters

shaderType Specifies the type of shader to be created. Must be either GL_VERTEX_SHADER or GL_FRAGMENT_SHADER.

5.26.2.46 `static void OpenTK.Graphics.ES20.GL.CullFace (`
`OpenTK.Graphics.ES20.CullFaceMode mode) [static]`

Specify whether front- or back-facing facets can be culled.

Parameters

mode Specifies whether front- or back-facing facets are candidates for culling. Symbolic constants GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK are accepted. The initial value is GL_BACK.

5.26.2.47 `static void OpenTK.Graphics.ES20.GL.DeleteBuffers (Int32 n,`
`Int32[] buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.26.2.48 `static void OpenTK.Graphics.ES20.GL.DeleteBuffers (Int32 n, ref Int32 buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.26.2.49 `static unsafe void OpenTK.Graphics.ES20.GL.DeleteBuffers (Int32 n, Int32 * buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.26.2.50 `static void OpenTK.Graphics.ES20.GL.DeleteBuffers (Int32 n, UInt32[] buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.26.2.51 `static void OpenTK.Graphics.ES20.GL.DeleteBuffers (Int32 n, ref UInt32 buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.26.2.52 `static unsafe void OpenTK.Graphics.ES20.GL.DeleteBuffers (Int32 n, UInt32 * buffers) [static]`

Delete named buffer objects.

Parameters

n Specifies the number of buffer objects to be deleted.

buffers Specifies an array of buffer objects to be deleted.

5.26.2.53 `static void OpenTK.Graphics.ES20.GL.DeleteProgram (Int32 program) [static]`

Deletes a program object.

Parameters

program Specifies the program object to be deleted.

5.26.2.54 `static void OpenTK.Graphics.ES20.GL.DeleteProgram (UInt32 program) [static]`

Deletes a program object.

Parameters

program Specifies the program object to be deleted.

5.26.2.55 `static void OpenTK.Graphics.ES20.GL.DeleteShader (Int32 shader) [static]`

Deletes a shader object.

Parameters

shader Specifies the shader object to be deleted.

5.26.2.56 `static void OpenTK.Graphics.ES20.GL.DeleteShader (UInt32 shader) [static]`

Deletes a shader object.

Parameters

shader Specifies the shader object to be deleted.

5.26.2.57 `static void OpenTK.Graphics.ES20.GL.DeleteTextures (Int32 n, Int32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.26.2.58 `static void OpenTK.Graphics.ES20.GL.DeleteTextures (Int32 n, ref Int32 textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.26.2.59 `static unsafe void OpenTK.Graphics.ES20.GL.DeleteTextures (Int32 n, Int32 * textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.26.2.60 `static void OpenTK.Graphics.ES20.GL.DeleteTextures (Int32 n, UInt32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.26.2.61 static void OpenTK.Graphics.ES20.GL.DeleteTextures (Int32 *n*, ref UInt32 *textures*) [static]

Delete named textures.

Parameters

- n* Specifies the number of textures to be deleted.
- textures* Specifies an array of textures to be deleted.

5.26.2.62 static unsafe void OpenTK.Graphics.ES20.GL.DeleteTextures (Int32 *n*, UInt32 * *textures*) [static]

Delete named textures.

Parameters

- n* Specifies the number of textures to be deleted.
- textures* Specifies an array of textures to be deleted.

5.26.2.63 static void OpenTK.Graphics.ES20.GL.DepthFunc (OpenTK.Graphics.ES20.DepthFunction *func*) [static]

Specify the value used for depth buffer comparisons.

Parameters

- func* Specifies the depth comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_LESS.

5.26.2.64 static void OpenTK.Graphics.ES20.GL.DepthMask (bool *flag*) [static]

Enable or disable writing into the depth buffer.

Parameters

- flag* Specifies whether the depth buffer is enabled for writing. If flag is GL_FALSE, depth buffer writing is disabled. Otherwise, it is enabled. Initially, depth buffer writing is enabled.

5.26.2.65 static void OpenTK.Graphics.ES20.GL.DepthRange (Single *zNear*, Single *zFar*) [static]

Specify mapping of depth values from normalized device coordinates to window coordinates.

Parameters

nearVal Specifies the mapping of the near clipping plane to window coordinates. The initial value is 0.

farVal Specifies the mapping of the far clipping plane to window coordinates. The initial value is 1.

5.26.2.66 static void OpenTK.Graphics.ES20.GL.DetachShader (Int32 *program*, Int32 *shader*) [static]

Detaches a shader object from a program object to which it is attached.

Parameters

program Specifies the program object from which to detach the shader object.

shader Specifies the shader object to be detached.

5.26.2.67 static void OpenTK.Graphics.ES20.GL.DetachShader (UInt32 *program*, UInt32 *shader*) [static]

Detaches a shader object from a program object to which it is attached.

Parameters

program Specifies the program object from which to detach the shader object.

shader Specifies the shader object to be detached.

5.26.2.68 static void OpenTK.Graphics.ES20.GL.DrawArrays (OpenTK.Graphics.ES20.BeginMode *mode*, Int32 *first*, Int32 *count*) [static]

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Specifies the starting index in the enabled arrays.

count Specifies the number of indices to be rendered.

5.26.2.69 `static void OpenTK.Graphics.ES20.GL.DrawElements (`
`OpenTK.Graphics.ES20.BeginMode mode, Int32 count,`
`OpenTK.Graphics.ES20.DrawElementsType type, IntPtr indices)`
`[static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

5.26.2.70 `static void OpenTK.Graphics.ES20.GL.DrawElements< T3 >`
`(OpenTK.Graphics.ES20.BeginMode mode, Int32 count,`
`OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute,`
`OutAttribute] T3[] indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.26.2.71 `static void OpenTK.Graphics.ES20.GL.DrawElements< T3 >
(OpenTK.Graphics.ES20.BeginMode mode, Int32 count,
OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute,
OutAttribute] T3 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.26.2.72 `static void OpenTK.Graphics.ES20.GL.DrawElements< T3 >
(OpenTK.Graphics.ES20.BeginMode mode, Int32 count,
OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute,
OutAttribute] T3 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.26.2.73 `static void OpenTK.Graphics.ES20.GL.DrawElements< T3 >`
`(OpenTK.Graphics.ES20.BeginMode mode, Int32 count,`
`OpenTK.Graphics.ES20.DrawElementsType type, [InAttribute,`
`OutAttribute] ref T3 indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

5.26.2.74 `static void OpenTK.Graphics.ES20.GL.Enable (`
`OpenTK.Graphics.ES20.EnableCap cap) [static]`

Enable or disable server-side [GL](#) capabilities.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.26.2.75 `static void OpenTK.Graphics.ES20.GL.EnableVertexAttribArray (`
`Int32 index) [static]`

Enable or disable a generic vertex attribute array.

Parameters

index Specifies the index of the generic vertex attribute to be enabled or disabled.

5.26.2.76 `static void OpenTK.Graphics.ES20.GL.EnableVertexAttribArray (UInt32 index) [static]`

Enable or disable a generic vertex attribute array.

Parameters

index Specifies the index of the generic vertex attribute to be enabled or disabled.

5.26.2.77 `static void OpenTK.Graphics.ES20.GL.Finish () [static]`

Block until all GL execution is complete.

5.26.2.78 `static void OpenTK.Graphics.ES20.GL.Flush () [static]`

Force execution of GL commands in finite time.

5.26.2.79 `static void OpenTK.Graphics.ES20.GL.FrontFace (OpenTK.Graphics.ES20.FrontFaceDirection mode) [static]`

Define front- and back-facing polygons.

Parameters

mode Specifies the orientation of front-facing polygons. GL_CW and GL_CCW are accepted. The initial value is GL_CCW.

5.26.2.80 `static void OpenTK.Graphics.ES20.GL.GenBuffers (Int32 n, [OutAttribute] Int32[] buffers) [static]`

Generate buffer object names.

Parameters

n Specifies the number of buffer object names to be generated.

buffers Specifies an array in which the generated buffer object names are stored.

5.26.2.81 `static void OpenTK.Graphics.ES20.GL.GenBuffers (Int32 n, [OutAttribute] out Int32 buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.26.2.82 `static unsafe void OpenTK.Graphics.ES20.GL.GenBuffers (Int32 n,
[OutAttribute] Int32 * buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.26.2.83 `static void OpenTK.Graphics.ES20.GL.GenBuffers (Int32 n,
[OutAttribute] UInt32[] buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.26.2.84 `static void OpenTK.Graphics.ES20.GL.GenBuffers (Int32 n,
[OutAttribute] out UInt32 buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.26.2.85 `static unsafe void OpenTK.Graphics.ES20.GL.GenBuffers (Int32 n,
[OutAttribute] UInt32 * buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.26.2.86 `static void OpenTK.Graphics.ES20.GL.GenTextures (Int32 n,
[OutAttribute] out Int32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.26.2.87 `static unsafe void OpenTK.Graphics.ES20.GL.GenTextures (Int32
n, [OutAttribute] Int32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.26.2.88 `static void OpenTK.Graphics.ES20.GL.GenTextures (Int32 n,
[OutAttribute] UInt32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.26.2.89 `static void OpenTK.Graphics.ES20.GL.GenTextures (Int32 n,
[OutAttribute] out UInt32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.26.2.90 `static unsafe void OpenTK.Graphics.ES20.GL.GenTextures (Int32 n, [OutAttribute] UInt32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.26.2.91 `static void OpenTK.Graphics.ES20.GL.GenTextures (Int32 n, [OutAttribute] Int32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.26.2.92 `static void OpenTK.Graphics.ES20.GL.GetActiveAttrib (Int32 program, Int32 index, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] Int32[] size, [OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType[] type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

program Specifies the program object to be queried.

index Specifies the index of the attribute variable to be queried.

bufSize Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.

length Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.

size Returns the size of the attribute variable.

type Returns the data type of the attribute variable.

name Returns a null terminated string containing the name of the attribute variable.

5.26.2.93 `static void OpenTK.Graphics.ES20.GL.GetActiveAttrib (Int32 program, Int32 index, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.ES20.ActiveAttribType type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.26.2.94 `static unsafe void OpenTK.Graphics.ES20.GL.GetActiveAttrib (Int32 program, Int32 index, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] Int32 * size, [OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType * type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.26.2.95 `static void OpenTK.Graphics.ES20.GL.GetActiveAttrib (UInt32 program, UInt32 index, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] Int32[] size, [OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType[] type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.26.2.96 `static void OpenTK.Graphics.ES20.GL.GetActiveAttrib (UInt32 program, UInt32 index, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.ES20.ActiveAttribType type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.26.2.97 `static unsafe void OpenTK.Graphics.ES20.GL.GetActiveAttrib (UInt32 program, UInt32 index, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] Int32 * size, [OutAttribute] OpenTK.Graphics.ES20.ActiveAttribType * type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.26.2.98 `static void OpenTK.Graphics.ES20.GL.GetActiveUniform (Int32 program, Int32 index, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] Int32[] size, [OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType[] type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.26.2.99 `static void OpenTK.Graphics.ES20.GL.GetActiveUniform (Int32 program, Int32 index, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.ES20.ActiveUniformType type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.26.2.100 `static unsafe void OpenTK.Graphics.ES20.GL.GetActiveUniform (Int32 program, Int32 index, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] Int32 * size, [OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType * type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.26.2.101 `static void OpenTK.Graphics.ES20.GL.GetActiveUniform (UInt32 program, UInt32 index, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] Int32[] size, [OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType[] type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.26.2.102 `static void OpenTK.Graphics.ES20.GL.GetActiveUniform (UInt32 program, UInt32 index, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.ES20.ActiveUniformType type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.26.2.103 `static unsafe void OpenTK.Graphics.ES20.GL.GetActiveUniform (UInt32 program, UInt32 index, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] Int32 * size, [OutAttribute] OpenTK.Graphics.ES20.ActiveUniformType * type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.26.2.104 `static void OpenTK.Graphics.ES20.GL.GetAttachedShaders (Int32 program, Int32 maxcount, [OutAttribute] Int32[] count, [OutAttribute] Int32[] shaders) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

- program* Specifies the program object to be queried.
- maxCount* Specifies the size of the array for storing the returned object names.
- count* Returns the number of names actually returned in objects.
- shaders* Specifies an array that is used to return the names of attached shader objects.

5.26.2.105 `static void OpenTK.Graphics.ES20.GL.GetAttachedShaders (Int32 program, Int32 maxcount, [OutAttribute] out Int32 count, [OutAttribute] out Int32 shaders) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

- program* Specifies the program object to be queried.
- maxCount* Specifies the size of the array for storing the returned object names.
- count* Returns the number of names actually returned in objects.
- shaders* Specifies an array that is used to return the names of attached shader objects.

5.26.2.106 `static unsafe void OpenTK.Graphics.ES20.GL.GetAttachedShaders (Int32 program, Int32 maxcount, [OutAttribute] Int32 * count, [OutAttribute] Int32 * shaders) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

- program* Specifies the program object to be queried.
- maxCount* Specifies the size of the array for storing the returned object names.
- count* Returns the number of names actually returned in objects.
- shaders* Specifies an array that is used to return the names of attached shader objects.

5.26.2.107 `static void OpenTK.Graphics.ES20.GL.GetAttachedShaders (UInt32 program, Int32 maxcount, [OutAttribute] Int32[] count, [OutAttribute] UInt32[] shaders) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

- program* Specifies the program object to be queried.
- maxCount* Specifies the size of the array for storing the returned object names.
- count* Returns the number of names actually returned in objects.
- shaders* Specifies an array that is used to return the names of attached shader objects.

5.26.2.108 `static void OpenTK.Graphics.ES20.GL.GetAttachedShaders (UInt32 program, Int32 maxcount, [OutAttribute] out Int32 count, [OutAttribute] out UInt32 shaders) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

- program* Specifies the program object to be queried.
- maxCount* Specifies the size of the array for storing the returned object names.
- count* Returns the number of names actually returned in objects.
- shaders* Specifies an array that is used to return the names of attached shader objects.

5.26.2.109 `static unsafe void OpenTK.Graphics.ES20.GL.GetAttachedShaders (UInt32 program, Int32 maxcount, [OutAttribute] Int32 * count, [OutAttribute] UInt32 * shaders) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

- program* Specifies the program object to be queried.
- maxCount* Specifies the size of the array for storing the returned object names.
- count* Returns the number of names actually returned in objects.
- shaders* Specifies an array that is used to return the names of attached shader objects.

5.26.2.110 `static int OpenTK.Graphics.ES20.GL.GetAttribLocation (Int32 program, String name) [static]`

Returns the location of an attribute variable.

Parameters

- program* Specifies the program object to be queried.
- name* Points to a null terminated string containing the name of the attribute variable whose location is to be queried.

5.26.2.111 `static int OpenTK.Graphics.ES20.GL.GetAttribLocation (UInt32 program, String name) [static]`

Returns the location of an attribute variable.

Parameters

- program* Specifies the program object to be queried.
- name* Points to a null terminated string containing the name of the attribute variable whose location is to be queried.

5.26.2.112 `static void OpenTK.Graphics.ES20.GL.GetBufferParameter
(OpenTK.Graphics.ES20.BufferTarget target,
OpenTK.Graphics.ES20.BufferParameterName pname,
[OutAttribute] Int32 @[] params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.26.2.113 `static void OpenTK.Graphics.ES20.GL.GetBufferParameter
(OpenTK.Graphics.ES20.BufferTarget target,
OpenTK.Graphics.ES20.BufferParameterName pname,
[OutAttribute] out Int32 @ params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.26.2.114 `static unsafe void OpenTK.Graphics.ES20.GL.GetBufferParameter
(OpenTK.Graphics.ES20.BufferTarget target,
OpenTK.Graphics.ES20.BufferParameterName pname,
[OutAttribute] Int32 *@ params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.26.2.115 `static OpenTK.Graphics.ES20.ErrorCode
OpenTK.Graphics.ES20.GL.GetError () [static]`

Return error information.

5.26.2.116 `static void OpenTK.Graphics.ES20.GL.GetProgram (Int32
program, OpenTK.Graphics.ES20.ProgramParameter pname,
[OutAttribute] Int32 @[] params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.26.2.117 `static void OpenTK.Graphics.ES20.GL.GetProgram (Int32
program, OpenTK.Graphics.ES20.ProgramParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.26.2.118 static unsafe void OpenTK.Graphics.ES20.GL.GetProgram (Int32 program, OpenTK.Graphics.ES20.ProgramParameter pname, [OutAttribute] Int32 *@ params) [static]

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.26.2.119 static void OpenTK.Graphics.ES20.GL.GetProgram (UInt32 program, OpenTK.Graphics.ES20.ProgramParameter pname, [OutAttribute] Int32 @[] params) [static]

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.26.2.120 `static void OpenTK.Graphics.ES20.GL.GetProgram (UInt32
program, OpenTK.Graphics.ES20.ProgramParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.26.2.121 `static unsafe void OpenTK.Graphics.ES20.GL.GetProgram (UInt32 program, OpenTK.Graphics.ES20.ProgramParameter pname, [OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.26.2.122 `static void OpenTK.Graphics.ES20.GL.GetProgramInfoLog (Int32 program, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] StringBuilder infolog) [static]`

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.26.2.123 `static void OpenTK.Graphics.ES20.GL.GetProgramInfoLog (Int32 program, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.26.2.124 `static unsafe void OpenTK.Graphics.ES20.GL.GetProgramInfoLog (Int32 program, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.26.2.125 `static void OpenTK.Graphics.ES20.GL.GetProgramInfoLog (UInt32 program, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

- program* Specifies the program object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.126 `static void OpenTK.Graphics.ES20.GL.GetProgramInfoLog (UInt32 program, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

- program* Specifies the program object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.127 `static unsafe void OpenTK.Graphics.ES20.GL.GetProgramInfoLog (UInt32 program, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

- program* Specifies the program object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.128 `static void OpenTK.Graphics.ES20.GL.GetShader (Int32 shader,
OpenTK.Graphics.ES20.ShaderParameter pname, [OutAttribute]
Int32 @[] params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.26.2.129 `static void OpenTK.Graphics.ES20.GL.GetShader (Int32 shader,
OpenTK.Graphics.ES20.ShaderParameter pname, [OutAttribute]
out Int32 @ params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.26.2.130 `static unsafe void OpenTK.Graphics.ES20.GL.GetShader (Int32
shader, OpenTK.Graphics.ES20.ShaderParameter pname,
[OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.26.2.131 `static void OpenTK.Graphics.ES20.GL.GetShader (UInt32 shader,
OpenTK.Graphics.ES20.ShaderParameter pname, [OutAttribute]
Int32 @[] params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.26.2.132 `static void OpenTK.Graphics.ES20.GL.GetShader (UInt32 shader,
OpenTK.Graphics.ES20.ShaderParameter pname, [OutAttribute]
out Int32 @ params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.26.2.133 `static unsafe void OpenTK.Graphics.ES20.GL.GetShader (UInt32
shader, OpenTK.Graphics.ES20.ShaderParameter pname,
[OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.26.2.134 `static void OpenTK.Graphics.ES20.GL.GetShaderInfoLog (Int32 shader, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.135 `static void OpenTK.Graphics.ES20.GL.GetShaderInfoLog (Int32 shader, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.136 `static unsafe void OpenTK.Graphics.ES20.GL.GetShaderInfoLog (Int32 shader, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.137 `static void OpenTK.Graphics.ES20.GL.GetShaderInfoLog (UInt32 shader, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.138 `static void OpenTK.Graphics.ES20.GL.GetShaderInfoLog (UInt32 shader, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.139 `static unsafe void OpenTK.Graphics.ES20.GL.GetShaderInfoLog (UInt32 shader, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.26.2.140 `static void OpenTK.Graphics.ES20.GL.GetShaderSource (Int32 shader, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.26.2.141 `static void OpenTK.Graphics.ES20.GL.GetShaderSource (Int32 shader, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.26.2.142 `static unsafe void OpenTK.Graphics.ES20.GL.GetShaderSource (Int32 shader, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.26.2.143 `static void OpenTK.Graphics.ES20.GL.GetShaderSource (UInt32 shader, Int32 bufsize, [OutAttribute] Int32[] length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.26.2.144 `static void OpenTK.Graphics.ES20.GL.GetShaderSource (UInt32 shader, Int32 bufsize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.26.2.145 `static unsafe void OpenTK.Graphics.ES20.GL.GetShaderSource (UInt32 shader, Int32 bufsize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.26.2.146 `static unsafe System.String OpenTK.Graphics.ES20.GL.GetString (OpenTK.Graphics.ES20.StringName name) [static]`

Return a string describing the current [GL](#) connection.

Parameters

name Specifies a symbolic constant, one of GL_VENDOR, GL_RENDERER, GL_VERSION, GL_SHADING_LANGUAGE_VERSION, or GL_EXTENSIONS.

5.26.2.147 `static void OpenTK.Graphics.ES20.GL.GetTexParameter (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname, [OutAttribute] Single @[] params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.26.2.148 `static void OpenTK.Graphics.ES20.GL.GetTexParameter (OpenTK.Graphics.ES20.TextureTarget target, OpenTK.Graphics.ES20.GetTextureParameter pname, [OutAttribute] out Single @ params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.26.2.149 static unsafe void OpenTK.Graphics.ES20.GL.GetTexParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.GetTextureParameter pname,
[OutAttribute] Single *@ params) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.26.2.150 `static void OpenTK.Graphics.ES20.GL.GetTexParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.GetTextureParameter pname,
[OutAttribute] Int32 @[] params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.26.2.151 `static void OpenTK.Graphics.ES20.GL.GetTexParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.GetTextureParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.26.2.152 `static unsafe void OpenTK.Graphics.ES20.GL.GetTexParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.GetTextureParameter pname,
[OutAttribute] Int32 *@ params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.26.2.153 `static void OpenTK.Graphics.ES20.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] Single @[] params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.154 `static void OpenTK.Graphics.ES20.GL.GetUniform (UInt32
program, Int32 location, [OutAttribute] Single @[] params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.155 `static void OpenTK.Graphics.ES20.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] out Int32 @ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.156 `static void OpenTK.Graphics.ES20.GL.GetUniform (UInt32
program, Int32 location, [OutAttribute] Int32 @[] params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.157 `static void OpenTK.Graphics.ES20.GL.GetUniform (UInt32
program, Int32 location, [OutAttribute] out Int32 @ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.158 `static unsafe void OpenTK.Graphics.ES20.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] Int32 *@ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.159 `static void OpenTK.Graphics.ES20.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] out Single @ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.160 `static unsafe void OpenTK.Graphics.ES20.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] Single *@ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.26.2.161 `static void OpenTK.Graphics.ES20.GL.GetUniform (UInt32
program, Int32 location, [OutAttribute] out Single @ params)
[static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.26.2.162 `static unsafe void OpenTK.Graphics.ES20.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] Single *@ params)`
`[static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.26.2.163 `static void OpenTK.Graphics.ES20.GL.GetUniform (Int32 program, Int32 location, [OutAttribute] Int32 @[] params)`
`[static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.26.2.164 `static unsafe void OpenTK.Graphics.ES20.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] Int32 *@ params)`
`[static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.26.2.165 `static int OpenTK.Graphics.ES20.GL.GetUniformLocation (Int32 program, String name) [static]`

Returns the location of a uniform variable.

Parameters

program Specifies the program object to be queried.

name Points to a null terminated string containing the name of the uniform variable whose location is to be queried.

5.26.2.166 `static int OpenTK.Graphics.ES20.GL.GetUniformLocation (UInt32 program, String name) [static]`

Returns the location of a uniform variable.

Parameters

program Specifies the program object to be queried.

name Points to a null terminated string containing the name of the uniform variable whose location is to be queried.

5.26.2.167 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] out Single @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.168 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Int32 @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.169 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Single @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.170 `static unsafe void OpenTK.Graphics.ES20.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Single *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.171 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Single @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.172 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] out Single @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.173 `static unsafe void OpenTK.Graphics.ES20.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Single *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.174 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Int32 @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.175 `static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] out Int32 @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.176 static unsafe void OpenTK.Graphics.ES20.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Int32 *@ params) [static]

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.177 static void OpenTK.Graphics.ES20.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] out Int32 @ params) [static]

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.178 `static unsafe void OpenTK.Graphics.ES20.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.ES20.VertexAttribParameter pname, [OutAttribute] Int32 *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.26.2.179 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [OutAttribute] IntPtr pointer) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

5.26.2.180 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [OutAttribute] IntPtr pointer) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

5.26.2.181 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2> (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] T2[] pointer) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.182 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2> (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.183 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2> (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.184 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2 > (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.185 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2 > (UInt32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] ref T2 pointer) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.186 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2> (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] T2[] pointer) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.187 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2> (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.188 `static void OpenTK.Graphics.ES20.GL.GetVertexAttribPointer<T2> (Int32 index, OpenTK.Graphics.ES20.VertexAttribPointerParameter pname, [InAttribute, OutAttribute] ref T2 pointer) [static]`

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

5.26.2.189 `static void OpenTK.Graphics.ES20.GL.Hint
(OpenTK.Graphics.ES20.HintTarget target,
OpenTK.Graphics.ES20.HintMode mode) [static]`

Specify implementation-specific hints.

Parameters

- target* Specifies a symbolic constant indicating the behavior to be controlled. GL_FOG_HINT, GL_GENERATE_MIPMAP_HINT, GL_LINE_SMOOTH_HINT, GL_PERSPECTIVE_CORRECTION_HINT, GL_POINT_SMOOTH_HINT, GL_POLYGON_SMOOTH_HINT, GL_TEXTURE_COMPRESSION_HINT, and GL_FRAGMENT_SHADER_DERIVATIVE_HINT are accepted.
- mode* Specifies a symbolic constant indicating the desired behavior. GL_FASTEST, GL_NICEST, and GL_DONT_CARE are accepted.

5.26.2.190 `static bool OpenTK.Graphics.ES20.GL.IsBuffer (Int32 buffer)
[static]`

Determine if a name corresponds to a buffer object.

Parameters

- buffer* Specifies a value that may be the name of a buffer object.

5.26.2.191 `static bool OpenTK.Graphics.ES20.GL.IsBuffer (UInt32 buffer)
[static]`

Determine if a name corresponds to a buffer object.

Parameters

- buffer* Specifies a value that may be the name of a buffer object.

5.26.2.192 `static bool OpenTK.Graphics.ES20.GL.IsEnabled (OpenTK.Graphics.ES20.EnableCap cap) [static]`

Test whether a capability is enabled.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.26.2.193 `static bool OpenTK.Graphics.ES20.GL.IsProgram (Int32 program) [static]`

Determines if a name corresponds to a program object.

Parameters

program Specifies a potential program object.

5.26.2.194 `static bool OpenTK.Graphics.ES20.GL.IsProgram (UInt32 program) [static]`

Determines if a name corresponds to a program object.

Parameters

program Specifies a potential program object.

5.26.2.195 `static bool OpenTK.Graphics.ES20.GL.IsShader (Int32 shader) [static]`

Determines if a name corresponds to a shader object.

Parameters

shader Specifies a potential shader object.

5.26.2.196 `static bool OpenTK.Graphics.ES20.GL.IsShader (UInt32 shader) [static]`

Determines if a name corresponds to a shader object.

Parameters

shader Specifies a potential shader object.

5.26.2.197 `static bool OpenTK.Graphics.ES20.GL.IsTexture (Int32 texture)`
`[static]`

Determine if a name corresponds to a texture.

Parameters

texture Specifies a value that may be the name of a texture.

5.26.2.198 `static bool OpenTK.Graphics.ES20.GL.IsTexture (UInt32 texture`
`) [static]`

Determine if a name corresponds to a texture.

Parameters

texture Specifies a value that may be the name of a texture.

5.26.2.199 `static void OpenTK.Graphics.ES20.GL.LineWidth (Single width)`
`[static]`

Specify the width of rasterized lines.

Parameters

width Specifies the width of rasterized lines. The initial value is 1.

5.26.2.200 `static void OpenTK.Graphics.ES20.GL.LinkProgram (UInt32`
`program) [static]`

Links a program object.

Parameters

program Specifies the handle of the program object to be linked.

5.26.2.201 `static void OpenTK.Graphics.ES20.GL.LinkProgram (Int32`
`program) [static]`

Links a program object.

Parameters

program Specifies the handle of the program object to be linked.

5.26.2.202 `static void OpenTK.Graphics.ES20.GL.PixelStore (`
`OpenTK.Graphics.ES20.PixelStoreParameter pname, Int32 param`
`) [static]`

Set pixel storage modes.

Parameters

pname Specifies the symbolic name of the parameter to be set. Six values affect the packing of pixel data into memory: GL_PACK_SWAP_BYTES, GL_PACK_LSB_FIRST, GL_PACK_ROW_LENGTH, GL_PACK_IMAGE_HEIGHT, GL_PACK_SKIP_PIXELS, GL_PACK_SKIP_ROWS, GL_PACK_SKIP_IMAGES, and GL_PACK_ALIGNMENT. Six more affect the unpacking of pixel data from memory: GL_UNPACK_SWAP_BYTES, GL_UNPACK_LSB_FIRST, GL_UNPACK_ROW_LENGTH, GL_UNPACK_IMAGE_HEIGHT, GL_UNPACK_SKIP_PIXELS, GL_UNPACK_SKIP_ROWS, GL_UNPACK_SKIP_IMAGES, and GL_UNPACK_ALIGNMENT.

param Specifies the value that pname is set to.

5.26.2.203 `static void OpenTK.Graphics.ES20.GL.PolygonOffset (Single`
`factor, Single units) [static]`

Set the scale and units used to calculate depth values.

Parameters

factor Specifies a scale factor that is used to create a variable depth offset for each polygon. The initial value is 0.

units Is multiplied by an implementation-specific value to create a constant depth offset. The initial value is 0.

5.26.2.204 `static void OpenTK.Graphics.ES20.GL.ReadPixels (Int32 x, Int32`
`y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat`
`format, OpenTK.Graphics.ES20.PixelType type, IntPtr pixels)`
`[static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

```
5.26.2.205 static void OpenTK.Graphics.ES20.GL.ReadPixels<
T6 > ( Int32 x, Int32 y, Int32 width, Int32
height, OpenTK.Graphics.ES20.PixelFormat format,
OpenTK.Graphics.ES20.PixelType type, [InAttribute,
OutAttribute] T6[] pixels ) [static]
```

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : struct

5.26.2.206 `static void OpenTK.Graphics.ES20.GL.ReadPixels<T6> (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T6 pixels[,]) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : struct

5.26.2.207 `static void OpenTK.Graphics.ES20.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T6 pixels[,]) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : *struct*

5.26.2.208 `static void OpenTK.Graphics.ES20.GL.ReadPixels< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] ref T6 pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

- x** Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.
- width** Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.
- format** Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.
- type** Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.
- data** Returns the pixel data.

Type Constraints

T6 : struct

5.26.2.209 `static void OpenTK.Graphics.ES20.GL.SampleCoverage (Single value, bool invert) [static]`

Specify multisample coverage parameters.

Parameters

- value** Specify a single floating-point sample coverage value. The value is clamped to the range [0,1]. The initial value is 1.0.
- invert** Specify a single boolean value representing if the coverage masks should be inverted. GL_TRUE and GL_FALSE are accepted. The initial value is GL_FALSE.

5.26.2.210 `static void OpenTK.Graphics.ES20.GL.Scissor (Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Define the scissor box.

Parameters

- x* Specify the lower left corner of the scissor box. Initially (0, 0).
- width* Specify the width and height of the scissor box. When a [GL](#) context is first attached to a window, width and height are set to the dimensions of that window.

5.26.2.211 `static unsafe void OpenTK.Graphics.ES20.GL.ShaderSource (UInt32 shader, Int32 count, String @[] string, Int32 * length) [static]`

Replaces the source code in a shader object.

Parameters

- shader* Specifies the handle of the shader object whose source code is to be replaced.
- count* Specifies the number of elements in the string and length arrays.
- string* Specifies an array of pointers to strings containing the source code to be loaded into the shader.
- length* Specifies an array of string lengths.

5.26.2.212 `static unsafe void OpenTK.Graphics.ES20.GL.ShaderSource (Int32 shader, Int32 count, String @[] string, Int32 * length) [static]`

Replaces the source code in a shader object.

Parameters

- shader* Specifies the handle of the shader object whose source code is to be replaced.
- count* Specifies the number of elements in the string and length arrays.
- string* Specifies an array of pointers to strings containing the source code to be loaded into the shader.
- length* Specifies an array of string lengths.

5.26.2.213 `static void OpenTK.Graphics.ES20.GL.ShaderSource (UInt32 shader, Int32 count, String @[] string, Int32[] length) [static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.26.2.214 `static void OpenTK.Graphics.ES20.GL.ShaderSource (UInt32
shader, Int32 count, String @[] string, ref Int32 length)
[static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.26.2.215 `static void OpenTK.Graphics.ES20.GL.ShaderSource (Int32
shader, Int32 count, String @[] string, ref Int32 length)
[static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.26.2.216 `static void OpenTK.Graphics.ES20.GL.ShaderSource (Int32
shader, Int32 count, String @[] string, Int32[] length)
[static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.26.2.217 `static void OpenTK.Graphics.ES20.GL.StencilFunc (
OpenTK.Graphics.ES20.StencilFunction func, Int32 @ ref,
UInt32 mask) [static]`

Set front and back function and reference value for stencil testing.

Parameters

func Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.

ref Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n} - 1]$, where n is the number of bitplanes in the stencil buffer. The initial value is 0.

mask Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.26.2.218 `static void OpenTK.Graphics.ES20.GL.StencilFunc (
OpenTK.Graphics.ES20.StencilFunction func, Int32 @ ref, Int32
mask) [static]`

Set front and back function and reference value for stencil testing.

Parameters

func Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.

ref Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.

mask Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.26.2.219 `static void OpenTK.Graphics.ES20.GL.StencilFuncSeparate (OpenTK.Graphics.ES20.CullFaceMode face, OpenTK.Graphics.ES20.StencilFunction func, Int32 @ ref, Int32 mask) [static]`

Set front and/or back function and reference value for stencil testing.

Parameters

face Specifies whether front and/or back stencil state is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

func Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.

ref Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.

mask Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.26.2.220 `static void OpenTK.Graphics.ES20.GL.StencilFuncSeparate (OpenTK.Graphics.ES20.CullFaceMode face, OpenTK.Graphics.ES20.StencilFunction func, Int32 @ ref, UInt32 mask) [static]`

Set front and/or back function and reference value for stencil testing.

Parameters

face Specifies whether front and/or back stencil state is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

func Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.

ref Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\text{sup } n} - 1]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.

mask Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.26.2.221 `static void OpenTK.Graphics.ES20.GL.StencilMask (UInt32 mask) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.26.2.222 `static void OpenTK.Graphics.ES20.GL.StencilMask (Int32 mask) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.26.2.223 `static void OpenTK.Graphics.ES20.GL.StencilMaskSeparate (OpenTK.Graphics.ES20.CullFaceMode face, Int32 mask) [static]`

Control the front and/or back writing of individual bits in the stencil planes.

Parameters

face Specifies whether the front and/or back stencil writemask is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.26.2.224 `static void OpenTK.Graphics.ES20.GL.StencilMaskSeparate
(OpenTK.Graphics.ES20.CullFaceMode face, UInt32 mask)
[static]`

Control the front and/or back writing of individual bits in the stencil planes.

Parameters

face Specifies whether the front and/or back stencil writemask is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.26.2.225 `static void OpenTK.Graphics.ES20.GL.StencilOp
(OpenTK.Graphics.ES20.StencilOp fail,
OpenTK.Graphics.ES20.StencilOp zfail,
OpenTK.Graphics.ES20.StencilOp zpass) [static]`

Set front and back stencil test actions.

Parameters

sfail Specifies the action to take when the stencil test fails. Eight symbolic constants are accepted: GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR, GL_INCR_WRAP, GL_DECR, GL_DECR_WRAP, and GL_INVERT. The initial value is GL_KEEP.

dpfail Specifies the stencil action when the stencil test passes, but the depth test fails. dpfail accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

dppass Specifies the stencil action when both the stencil test and the depth test pass, or when the stencil test passes and either there is no depth buffer or depth testing is not enabled. dppass accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

5.26.2.226 `static void OpenTK.Graphics.ES20.GL.StencilOpSeparate
(OpenTK.Graphics.ES20.CullFaceMode
face, OpenTK.Graphics.ES20.StencilOp fail,
OpenTK.Graphics.ES20.StencilOp zfail,
OpenTK.Graphics.ES20.StencilOp zpass) [static]`

Set front and/or back stencil test actions.

Parameters

- face* Specifies whether front and/or back stencil state is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.
- sfail* Specifies the action to take when the stencil test fails. Eight symbolic constants are accepted: GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR, GL_INCR_WRAP, GL_DECR, GL_DECR_WRAP, and GL_INVERT. The initial value is GL_KEEP.
- dpfail* Specifies the stencil action when the stencil test passes, but the depth test fails. dpfail accepts the same symbolic constants as sfail. The initial value is GL_KEEP.
- dppass* Specifies the stencil action when both the stencil test and the depth test pass, or when the stencil test passes and either there is no depth buffer or depth testing is not enabled. dppass accepts the same symbolic constants as sfail. The initial value is GL_KEEP.

5.26.2.227 static void OpenTK.Graphics.ES20.GL.TextureImage2D
(OpenTK.Graphics.ES20.TextureTarget *target*, Int32
***level*, OpenTK.Graphics.ES20.PixelInternalFormat**
***internalformat*, Int32 *width*, Int32 *height*, Int32**
***border*, OpenTK.Graphics.ES20.PixelFormat *format*,**
OpenTK.Graphics.ES20.PixelType *type*, IntPtr *pixels*)
[static]

Specify a two-dimensional texture image.

Parameters

- target* Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.
- level* Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
- internalFormat* Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA,

GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGB12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.26.2.228 `static void OpenTK.Graphics.ES20.GL.TextureImage2D< T8 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8 pixels[,J]) [static]`

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer n .

$\text{sup } m + 2 (\text{border})$ for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.26.2.229 static void OpenTK.Graphics.ES20.GL.TextureImage2D<
    T8 > ( OpenTK.Graphics.ES20.TextureTarget target,
    Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32
    border, OpenTK.Graphics.ES20.PixelFormat format,
    OpenTK.Graphics.ES20.PixelType type, [InAttribute,
    OutAttribute] ref T8 pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalFormat Specifies the number of color components in the texture.

Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n} + 2 \cdot (\text{border})$ for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m} + 2 \cdot (\text{border})$ for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.26.2.230 static void OpenTK.Graphics.ES20.GL.TextureImage2D<
            T8 > ( OpenTK.Graphics.ES20.TextureTarget target,
                  Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat
                  internalformat, Int32 width, Int32 height, Int32
                  border, OpenTK.Graphics.ES20.PixelFormat format,
                  OpenTK.Graphics.ES20.PixelType type, [InAttribute,
                  OutAttribute] T8 pixels[,]) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{m+2} (border) for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.26.2.231 static void OpenTK.Graphics.ES20.GL.TextureImage2D<
    T8 > ( OpenTK.Graphics.ES20.TextureTarget target,
    Int32 level, OpenTK.Graphics.ES20.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32
    border, OpenTK.Graphics.ES20.PixelFormat format,
    OpenTK.Graphics.ES20.PixelType type, [InAttribute,
    OutAttribute] T8[] pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y,

GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n + 2} \text{ (border)}$ for some integer *n*. All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m + 2} \text{ (border)}$ for some integer *m*. All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_-

UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8,
GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2,
and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

5.26.2.232 `static unsafe void OpenTK.Graphics.ES20.GL.TextureParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.TextureParameterName pname, Single
*@ params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.26.2.233 `static void OpenTK.Graphics.ES20.GL.TextureParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.TextureParameterName pname, Single
param) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.26.2.234 `static void OpenTK.Graphics.ES20.GL.TextureParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.TextureParameterName pname, Single
@[] params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.26.2.235 `static unsafe void OpenTK.Graphics.ES20.GL.TextureParameter
(OpenTK.Graphics.ES20.TextureTarget target,
OpenTK.Graphics.ES20.TextureParameterName pname, Int32 *@
params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

```
5.26.2.236 static void OpenTK.Graphics.ES20.GL.TextureParameter
( OpenTK.Graphics.ES20.TextureTarget target,
  OpenTK.Graphics.ES20.TextureParameterName pname, Int32
  @[] params ) [static]
```

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

```
5.26.2.237 static void OpenTK.Graphics.ES20.GL.TextureParameter
( OpenTK.Graphics.ES20.TextureTarget target,
  OpenTK.Graphics.ES20.TextureParameterName pname, Int32
  param ) [static]
```

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

```
5.26.2.238 static void OpenTK.Graphics.ES20.GL.TexSubImage2D
( OpenTK.Graphics.ES20.TextureTarget target, Int32
  level, Int32 xoffset, Int32 yoffset, Int32 width, Int32
  height, OpenTK.Graphics.ES20.PixelFormat format,
  OpenTK.Graphics.ES20.PixelType type, IntPtr pixels )
[static]
```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8,

GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.26.2.239 `static void OpenTK.Graphics.ES20.GL.TexSubImage2D< T8 > (OpenTK.Graphics.ES20.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.ES20.PixelFormat format, OpenTK.Graphics.ES20.PixelType type, [InAttribute, OutAttribute] T8[] pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.26.2.240 static void OpenTK.Graphics.ES20.GL.TexSubImage2D<
T8 > ( OpenTK.Graphics.ES20.TextureTarget target,
Int32 level, Int32 xoffset, Int32 yoffset, Int32 width,
Int32 height, OpenTK.Graphics.ES20.PixelFormat format,
OpenTK.Graphics.ES20.PixelType type, [InAttribute,
OutAttribute] T8 pixels[, ] ) [static]
```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```

5.26.2.241 static void OpenTK.Graphics.ES20.GL.TexSubImage2D<
    T8 > ( OpenTK.Graphics.ES20.TextureTarget target,
    Int32 level, Int32 xoffset, Int32 yoffset, Int32 width,
    Int32 height, OpenTK.Graphics.ES20.PixelFormat format,
    OpenTK.Graphics.ES20.PixelType type, [InAttribute,
    OutAttribute] ref T8 pixels ) [static]

```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

```

5.26.2.242 static void OpenTK.Graphics.ES20.GL.TexSubImage2D<
            T8 > ( OpenTK.Graphics.ES20.TextureTarget target,
                  Int32 level, Int32 xoffset, Int32 yoffset, Int32 width,
                  Int32 height, OpenTK.Graphics.ES20.PixelFormat format,
                  OpenTK.Graphics.ES20.PixelType type, [InAttribute,
                  OutAttribute] T8 pixels[,]) [static]

```

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.26.2.243 `static void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 x) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.244 `static void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 count, ref Single v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.245 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 count, Single * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.246 `static void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 count, ref Int32 v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.247 `static void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Single x) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.248 `static void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 count, Single[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.249 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 count, Int32 * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.250 `static void OpenTK.Graphics.ES20.GL.Uniform1 (Int32 location, Int32 count, Int32[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.251 `static void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Int32 count, Int32[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.252 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Int32 count, Int32 * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.253 `static void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Int32 x, Int32 y) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.254 `static void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Single x, Single y) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.255 `static void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Int32 count, Single[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.256 `static void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Int32 count, ref Single v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.257 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform2 (Int32 location, Int32 count, Single * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.258 `static void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 x, Int32 y, Int32 z) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.259 `static void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 count, Int32[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.260 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 count, Int32 * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.261 `static void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 count, ref Single v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.262 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 count, Single * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.263 `static void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 count, ref Int32 v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.264 `static void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Int32 count, Single[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.265 `static void OpenTK.Graphics.ES20.GL.Uniform3 (Int32 location, Single x, Single y, Single z) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.266 `static void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 count, Int32[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.267 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 count, Single * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.268 `static void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Single x, Single y, Single z, Single w) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.269 `static void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 count, Single[] v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.270 `static void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 count, ref Int32 v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.271 `static unsafe void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 count, Int32 * v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.272 `static void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 x, Int32 y, Int32 z, Int32 w) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.273 `static void OpenTK.Graphics.ES20.GL.Uniform4 (Int32 location, Int32 count, ref Single v) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.26.2.274 `static void OpenTK.Graphics.ES20.GL.UseProgram (Int32 program) [static]`

Installs a program object as part of current rendering state.

Parameters

program Specifies the handle of the program object whose executables are to be used as part of current rendering state.

5.26.2.275 `static void OpenTK.Graphics.ES20.GL.UseProgram (UInt32
program) [static]`

Installs a program object as part of current rendering state.

Parameters

program Specifies the handle of the program object whose executables are to be used as part of current rendering state.

5.26.2.276 `static void OpenTK.Graphics.ES20.GL.ValidateProgram (Int32
program) [static]`

Validates a program object.

Parameters

program Specifies the handle of the program object to be validated.

5.26.2.277 `static void OpenTK.Graphics.ES20.GL.ValidateProgram (UInt32
program) [static]`

Validates a program object.

Parameters

program Specifies the handle of the program object to be validated.

5.26.2.278 `static void OpenTK.Graphics.ES20.GL.VertexAttrib1 (Int32 indx,
Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.279 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib1 (Int32 indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.280 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib1 (UInt32 indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.281 `static void OpenTK.Graphics.ES20.GL.VertexAttrib1 (Int32 indx, Single x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.282 `static void OpenTK.Graphics.ES20.GL.VertexAttrib1 (UInt32 indx, Single x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.283 `static void OpenTK.Graphics.ES20.GL.VertexAttrib1 (UInt32
indx, Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.284 `static void OpenTK.Graphics.ES20.GL.VertexAttrib2 (UInt32
indx, Single x, Single y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.285 `static void OpenTK.Graphics.ES20.GL.VertexAttrib2 (Int32 indx,
Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.286 `static void OpenTK.Graphics.ES20.GL.VertexAttrib2 (UInt32
indx, Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.287 `static void OpenTK.Graphics.ES20.GL.VertexAttrib2 (UInt32
indx, ref Single values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.288 `static void OpenTK.Graphics.ES20.GL.VertexAttrib2 (Int32 indx,
ref Single values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.289 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib2 (UInt32 indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.290 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib2 (Int32
indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.291 `static void OpenTK.Graphics.ES20.GL.VertexAttrib2 (Int32 indx,
Single x, Single y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.292 `static void OpenTK.Graphics.ES20.GL.VertexAttrib3 (UInt32
indx, Single x, Single y, Single z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.293 `static void OpenTK.Graphics.ES20.GL.VertexAttrib3 (UInt32
indx, ref Single values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.294 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib3 (UInt32
indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.295 `static void OpenTK.Graphics.ES20.GL.VertexAttrib3 (Int32 indx,
ref Single values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.296 `static void OpenTK.Graphics.ES20.GL.VertexAttrib3 (UInt32
indx, Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.297 `static void OpenTK.Graphics.ES20.GL.VertexAttrib3 (Int32 indx,
Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.298 `static void OpenTK.Graphics.ES20.GL.VertexAttrib3 (Int32 indx,
Single x, Single y, Single z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.299 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib3 (Int32 indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.300 `static void OpenTK.Graphics.ES20.GL.VertexAttrib4 (Int32 indx, ref Single values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.301 `static void OpenTK.Graphics.ES20.GL.VertexAttrib4 (Int32 indx, Single x, Single y, Single z, Single w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.302 `static void OpenTK.Graphics.ES20.GL.VertexAttrib4 (UInt32 indx, Single x, Single y, Single z, Single w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.303 `static void OpenTK.Graphics.ES20.GL.VertexAttrib4 (Int32 indx, Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.304 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib4 (UInt32 indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.305 `static unsafe void OpenTK.Graphics.ES20.GL.VertexAttrib4 (Int32 indx, Single * values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.306 `static void OpenTK.Graphics.ES20.GL.VertexAttrib4 (UInt32 indx, Single[] values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.26.2.307 `static void OpenTK.Graphics.ES20.GL.VertexAttrib4 (UInt32
indx, ref Single values) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

values Specifies the new values to be used for the specified vertex attribute.

5.26.2.308 `static void OpenTK.Graphics.ES20.GL.VertexAttribPointer
(UInt32 indx, Int32 size,
OpenTK.Graphics.ES20.VertexAttribPointerType
type, bool normalized, Int32 stride, IntPtr ptr) [static]`

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

5.26.2.309 `static void OpenTK.Graphics.ES20.GL.VertexAttribPointer (Int32
indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType
type, bool normalized, Int32 stride, IntPtr ptr) [static]`

Define an array of generic vertex attribute data.

Parameters

- index* Specifies the index of the generic vertex attribute to be modified.
- size* Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized* Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.
- stride* Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

```

5.26.2.310 static void OpenTK.Graphics.ES20.GL.VertexAttribPointer<
            T5 > ( Int32 indx, Int32 size,
            OpenTK.Graphics.ES20.VertexAttribPointerType
            type, bool normalized, Int32 stride, [InAttribute, OutAttribute]
            T5[] ptr ) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index* Specifies the index of the generic vertex attribute to be modified.
- size* Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized* Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.
- stride* Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints*T5 : struct*

5.26.2.311 `static void OpenTK.Graphics.ES20.GL.VertexAttribPointer< T5 > (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride, [InAttribute, OutAttribute] T5 ptr[,]) [static]`

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints*T5 : struct*

5.26.2.312 `static void OpenTK.Graphics.ES20.GL.VertexAttribPointer< T5 > (UInt32 indx, Int32 size, OpenTK.Graphics.ES20.VertexAttribPointerType type, bool normalized, Int32 stride, [InAttribute, OutAttribute] T5 ptr[,]) [static]`

Define an array of generic vertex attribute data.

Parameters

- index*** Specifies the index of the generic vertex attribute to be modified.
- size*** Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type*** Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized*** Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.
- stride*** Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer*** Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```
5.26.2.313 static void OpenTK.Graphics.ES20.GL.VertexAttribPointer<
           T5 > ( Int32 indx, Int32 size,
           OpenTK.Graphics.ES20.VertexAttribPointerType
           type, bool normalized, Int32 stride, [InAttribute, OutAttribute] ref
           T5 ptr ) [static]
```

Define an array of generic vertex attribute data.

Parameters

- index*** Specifies the index of the generic vertex attribute to be modified.
- size*** Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type*** Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized*** Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : *struct*

```
5.26.2.314 static void OpenTK.Graphics.ES20.GL.VertexAttribPointer<
    T5 > ( UInt32 indx, Int32 size,
    OpenTK.Graphics.ES20.VertexAttribPointerType type, bool
    normalized, Int32 stride, [InAttribute, OutAttribute] T5[] ptr )
    [static]
```

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : *struct*

```

5.26.2.315 static void OpenTK.Graphics.ES20.GL.VertexAttribPointer<
           T5 > ( Int32 indx, Int32 size,
           OpenTK.Graphics.ES20.VertexAttribPointerType
           type, bool normalized, Int32 stride, [InAttribute, OutAttribute] T5
           ptr[,,] ) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index* Specifies the index of the generic vertex attribute to be modified.
- size* Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- normalized* Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.
- stride* Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```

5.26.2.316 static void OpenTK.Graphics.ES20.GL.VertexAttribPointer<
           T5 > ( Int32 indx, Int32 size,
           OpenTK.Graphics.ES20.VertexAttribPointerType
           type, bool normalized, Int32 stride, [InAttribute, OutAttribute] T5
           ptr[,,] ) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index* Specifies the index of the generic vertex attribute to be modified.
- size* Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

normalized Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```
5.26.2.317 static void OpenTK.Graphics.ES20.GL.VertexAttribPointer<
    T5 > ( UInt32 indx, Int32 size,
    OpenTK.Graphics.ES20.VertexAttribPointerType type, bool
    normalized, Int32 stride, [InAttribute, OutAttribute] ref T5 ptr )
    [static]
```

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

normalized Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

5.26.2.318 `static void OpenTK.Graphics.ES20.GL.Viewport (Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Set the viewport.

Parameters

- x** Specify the lower left corner of the viewport rectangle, in pixels. The initial value is (0,0).
- width** Specify the width and height of the viewport. When a [GL](#) context is first attached to a window, width and height are set to the dimensions of that window.

5.26.3 Property Documentation

5.26.3.1 `override object OpenTK.Graphics.ES20.GL.SyncRoot [get, protected]`

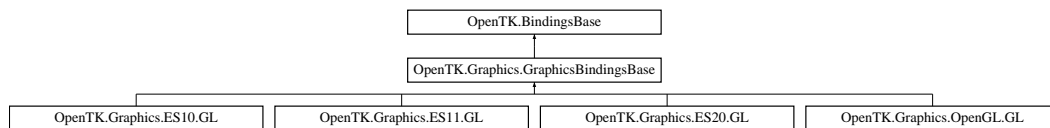
Returns a synchronization token unique for the [GL](#) class.

Reimplemented from [OpenTK.BindingsBase](#).

5.27 OpenTK.Graphics.GraphicsBindingsBase Class Reference

Implements [BindingsBase](#) for the [OpenTK.Graphics](#) namespace ([OpenGL](#) and [OpenGL|ES](#)).

Inheritance diagram for [OpenTK.Graphics.GraphicsBindingsBase](#):



Protected Member Functions

- override IntPtr [GetAddress](#) (string funcname)

Retrieves an unmanaged function pointer to the specified function.

5.27.1 Detailed Description

Implements [BindingsBase](#) for the [OpenTK.Graphics](#) namespace ([OpenGL](#) and [OpenGL|ES](#)).

5.27.2 Member Function Documentation

5.27.2.1 override IntPtr OpenTK.Graphics.GraphicsBindingsBase.GetAddress (string funcname) [protected, virtual]

Retrieves an unmanaged function pointer to the specified function.

Parameters

funcname A System.String that defines the name of the function.

Returns

A IntPtr that contains the address of funcname or IntPtr.Zero, if the function is not supported by the drivers.

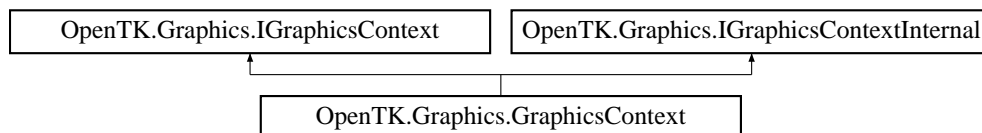
Note: some drivers are known to return non-zero values for unsupported functions. Typical values include 1 and 2 - inheritors are advised to check for and ignore these values.

Implements [OpenTK.BindingsBase](#).

5.28 OpenTK.Graphics.GraphicsContext Class Reference

Represents and provides methods to manipulate an [OpenGL](#) render context.

Inheritance diagram for OpenTK.Graphics.GraphicsContext:



Public Member Functions

- [GraphicsContext](#) ([GraphicsMode](#) mode, [IWindowInfo](#) window)
Constructs a new [GraphicsContext](#) with the specified [GraphicsMode](#) and attaches it to the specified window.
- [GraphicsContext](#) ([GraphicsMode](#) mode, [IWindowInfo](#) window, int major, int minor, [GraphicsContextFlags](#) flags)
Constructs a new [GraphicsContext](#) with the specified [GraphicsMode](#), version and flags, and attaches it to the specified window.
- [GraphicsContext](#) ([ContextHandle](#) handle, [IWindowInfo](#) window)
Constructs a new [GraphicsContext](#) from a pre-existing context created outside of OpenTK.
- [GraphicsContext](#) ([ContextHandle](#) handle, [IWindowInfo](#) window, [IGraphicsContext](#) shareContext, int major, int minor, [GraphicsContextFlags](#) flags)
Constructs a new [GraphicsContext](#) from a pre-existing context created outside of OpenTK.
- void [SwapBuffers](#) ()
Swaps buffers on a context. This presents the rendered scene to the user.
- void [MakeCurrent](#) ([IWindowInfo](#) window)
Makes the [GraphicsContext](#) the current rendering target.
- void [Update](#) ([IWindowInfo](#) window)
Updates the graphics context. This must be called when the render target is resized for proper behavior on Mac OS X.
- void [LoadAll](#) ()
Loads all [OpenGL](#) entry points.
- void [Dispose](#) ()
Disposes of the [GraphicsContext](#).

Static Public Member Functions

- static [GraphicsContext](#) [CreateDummyContext](#) ()
Creates a dummy [GraphicsContext](#) to allow OpenTK to work with contexts created by external libraries.

- static [GraphicsContext CreateDummyContext](#) ([ContextHandle](#) handle)
Creates a dummy [GraphicsContext](#) to allow OpenTK to work with contexts created by external libraries.
- static void [Assert](#) ()
Checks if a [GraphicsContext](#) exists in the calling thread and throws a [GraphicsContextMissingException](#) if it doesn't.

Properties

- static [IGraphicsContext CurrentContext](#) [get]
Gets the [GraphicsContext](#) that is current in the calling thread.
- static bool [ShareContexts](#) [get, set]
Gets or sets a [System.Boolean](#), indicating whether [GraphicsContext](#) resources are shared.
- static bool [DirectRendering](#) [get, set]
Gets or sets a [System.Boolean](#), indicating whether [GraphicsContexts](#) will perform direct rendering.
- bool [ErrorChecking](#) [get, set]
Gets or sets a [System.Boolean](#), indicating whether automatic error checking should be performed. Influences the debug version of OpenTK.dll, only.
- bool [IsCurrent](#) [get]
Gets a [System.Boolean](#) indicating whether this instance is current in the calling thread.
- bool [IsDisposed](#) [get, set]
Gets a [System.Boolean](#) indicating whether this instance has been disposed. It is an error to access any instance methods if this property returns true.
- bool [VSync](#) [get, set]
Gets or sets a value indicating whether VSync is enabled.
- [GraphicsMode GraphicsMode](#) [get]
Gets the [GraphicsMode](#) of the context.

5.28.1 Detailed Description

Represents and provides methods to manipulate an [OpenGL](#) render context.

5.28.2 Constructor & Destructor Documentation

5.28.2.1 `OpenTK.Graphics.GraphicsContext.GraphicsContext (GraphicsMode mode, IWindowInfo window)`

Constructs a new [GraphicsContext](#) with the specified [GraphicsMode](#) and attaches it to the specified window.

Parameters

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GraphicsContext](#).

window The [OpenTK.Platform.IWindowInfo](#) to attach the [GraphicsContext](#) to.

5.28.2.2 `OpenTK.Graphics.GraphicsContext.GraphicsContext (GraphicsMode mode, IWindowInfo window, int major, int minor, GraphicsContextFlags flags)`

Constructs a new [GraphicsContext](#) with the specified [GraphicsMode](#), version and flags, and attaches it to the specified window.

Parameters

mode The [OpenTK.Graphics.GraphicsMode](#) of the [GraphicsContext](#).

window The [OpenTK.Platform.IWindowInfo](#) to attach the [GraphicsContext](#) to.

major The major version of the new [GraphicsContext](#).

minor The minor version of the new [GraphicsContext](#).

flags The [GraphicsContextFlags](#) for the [GraphicsContext](#).

Different hardware supports different flags, major and minor versions. Invalid parameters will be silently ignored.

5.28.2.3 `OpenTK.Graphics.GraphicsContext.GraphicsContext (ContextHandle handle, IWindowInfo window)`

Constructs a new [GraphicsContext](#) from a pre-existing context created outside of OpenTK.

Parameters

handle The handle of the existing context. This must be a valid, unique handle that is not known to OpenTK.

window The window this context is bound to. This must be a valid window obtained through [Utilities.CreateWindowInfo](#).

Exceptions

GraphicsContextException Occurs if handle is identical to a context already registered with OpenTK.

5.28.2.4 OpenTK.Graphics.GraphicsContext.GraphicsContext (ContextHandle handle, IWindowInfo window, IGraphicsContext shareContext, int major, int minor, GraphicsContextFlags flags)

Constructs a new [GraphicsContext](#) from a pre-existing context created outside of OpenTK.

Parameters

handle The handle of the existing context. This must be a valid, unique handle that is not known to OpenTK.

window The window this context is bound to. This must be a valid window obtained through `Utilities.CreateWindowInfo`.

shareContext A different context that shares resources with this instance, if any. Pass null if the context is not shared or if this is the first [GraphicsContext](#) instruct you construct.

major The major version of the context (e.g. "2" for "2.1").

minor The minor version of the context (e.g. "1" for "2.1").

flags A bitwise combination of [GraphicsContextFlags](#) that describe this context.

Exceptions

GraphicsContextException Occurs if handle is identical to a context already registered with OpenTK.

5.28.3 Member Function Documentation

5.28.3.1 static void OpenTK.Graphics.GraphicsContext.Assert () [static]

Checks if a [GraphicsContext](#) exists in the calling thread and throws a [GraphicsContextMissingException](#) if it doesn't.

Exceptions

GraphicsContextMissingException Generated when no [GraphicsContext](#) is current in the calling thread.

5.28.3.2 static GraphicsContext OpenTK.Graphics.GraphicsContext.CreateDummyContext (ContextHandle *handle*) [static]

Creates a dummy [GraphicsContext](#) to allow OpenTK to work with contexts created by external libraries.

Parameters

handle The handle of a context.

Returns

A new, dummy [GraphicsContext](#) instance.

Instances created by this method will not be functional. Instance methods will have no effect.

5.28.3.3 static GraphicsContext OpenTK.Graphics.GraphicsContext.CreateDummyContext () [static]

Creates a dummy [GraphicsContext](#) to allow OpenTK to work with contexts created by external libraries.

Returns

A new, dummy [GraphicsContext](#) instance.

Instances created by this method will not be functional. Instance methods will have no effect.

This method requires that a context is current on the calling thread.

5.28.3.4 void OpenTK.Graphics.GraphicsContext.Dispose ()

Disposes of the [GraphicsContext](#).

5.28.3.5 void OpenTK.Graphics.GraphicsContext.LoadAll ()

Loads all [OpenGL](#) entry points.

Exceptions

[OpenTK.Graphics.GraphicsContextException](#) Occurs when this instance is not current on the calling thread.

Implements [OpenTK.Graphics.IGraphicsContextInternal](#).

5.28.3.6 void OpenTK.Graphics.GraphicsContext.MakeCurrent (IWindowInfo *window*)

Makes the [GraphicsContext](#) the current rendering target.

Parameters

window A valid [OpenTK.Platform.IWindowInfo](#) structure.

You can use this method to bind the [GraphicsContext](#) to a different window than the one it was created from.

5.28.3.7 void OpenTK.Graphics.GraphicsContext.SwapBuffers ()

Swaps buffers on a context. This presents the rendered scene to the user.

5.28.3.8 void OpenTK.Graphics.GraphicsContext.Update (IWindowInfo *window*)

Updates the graphics context. This must be called when the render target is resized for proper behavior on Mac OS X.

Parameters

window

5.28.4 Property Documentation

5.28.4.1 IGraphicsContext OpenTK.Graphics.GraphicsContext.CurrentContext [static, get]

Gets the [GraphicsContext](#) that is current in the calling thread.

Note: this property will not function correctly when both desktop and EGL contexts are available in the same process. This scenario is very unlikely to appear in practice.

5.28.4.2 bool OpenTK.Graphics.GraphicsContext.DirectRendering [static, get, set]

Gets or sets a System.Boolean, indicating whether GraphicsContexts will perform direct rendering.

If `DirectRendering` is true, new contexts will be constructed with direct rendering capabilities, if possible. If `DirectRendering` is false, new contexts will be constructed with indirect rendering capabilities.

This property does not affect existing `GraphicsContexts`, unless they are recreated.

This property is ignored on Operating Systems without support for indirect rendering, like Windows and OS X.

5.28.4.3 `bool OpenTK.Graphics.GraphicsContext.ErrorChecking [get, set]`

Gets or sets a `System.Boolean`, indicating whether automatic error checking should be performed. Influences the debug version of `OpenTK.dll`, only.

Automatic error checking will clear the [OpenGL](#) error state. Set `CheckErrors` to false if you use the [OpenGL](#) error state in your code flow (e.g. for checking supported texture formats).

5.28.4.4 `GraphicsMode OpenTK.Graphics.GraphicsContext.GraphicsMode [get]`

Gets the [GraphicsMode](#) of the context.

5.28.4.5 `bool OpenTK.Graphics.GraphicsContext.IsCurrent [get]`

Gets a `System.Boolean` indicating whether this instance is current in the calling thread.

5.28.4.6 `bool OpenTK.Graphics.GraphicsContext.IsDisposed [get, set]`

Gets a `System.Boolean` indicating whether this instance has been disposed. It is an error to access any instance methods if this property returns true.

5.28.4.7 `bool OpenTK.Graphics.GraphicsContext.ShareContexts [static, get, set]`

Gets or sets a `System.Boolean`, indicating whether [GraphicsContext](#) resources are shared.

If `ShareContexts` is true, new `GLContexts` will share resources. If this value is false, new `GLContexts` will not share resources.

Changing this value will not affect already created `GLContexts`.

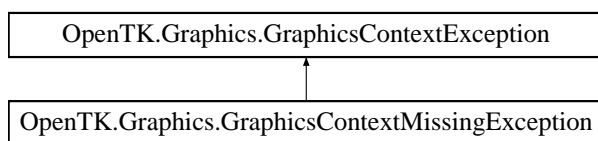
5.28.4.8 bool OpenTK.Graphics.GraphicsContext.VSync [get, set]

Gets or sets a value indicating whether VSync is enabled.

5.29 OpenTK.Graphics.GraphicsContextException Class Reference

Represents errors related to a [GraphicsContext](#).

Inheritance diagram for OpenTK.Graphics.GraphicsContextException:



Public Member Functions

- [GraphicsContextException](#) ()
Constructs a new [GraphicsContextException](#).
- [GraphicsContextException](#) (string message)
Constructs a new [GraphicsContextException](#) with the given error message.

5.29.1 Detailed Description

Represents errors related to a [GraphicsContext](#).

5.29.2 Constructor & Destructor Documentation

5.29.2.1 OpenTK.Graphics.GraphicsContextException.GraphicsContextException ()

Constructs a new [GraphicsContextException](#).

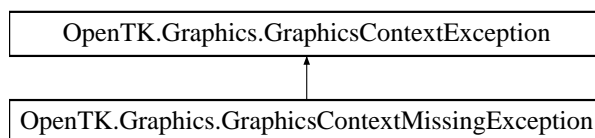
5.29.2.2 OpenTK.Graphics.GraphicsContextException.GraphicsContextException (string message)

Constructs a new [GraphicsContextException](#) with the given error message.

5.30 OpenTK.Graphics.GraphicsContextMissingException Class Reference

Thrown when an operation that required [GraphicsContext](#) is performed, when no [GraphicsContext](#) is current in the calling thread.

Inheritance diagram for OpenTK.Graphics.GraphicsContextMissingException:



Public Member Functions

- [GraphicsContextMissingException](#) ()

Constructs a new [GraphicsContextMissingException](#).

5.30.1 Detailed Description

Thrown when an operation that required [GraphicsContext](#) is performed, when no [GraphicsContext](#) is current in the calling thread.

5.30.2 Constructor & Destructor Documentation

5.30.2.1 OpenTK.Graphics.GraphicsContextMissingException.GraphicsContextMissingException ()

Constructs a new [GraphicsContextMissingException](#).

5.31 OpenTK.Graphics.GraphicsContextVersion Class Reference

Defines the version information of a [GraphicsContext](#).

Properties

- `int Minor [get, set]`
Gets a `System.Int32` indicating the minor version of a [GraphicsContext](#) instance.
- `int Major [get, set]`
Gets a `System.Int32` indicating the major version of a [GraphicsContext](#) instance.
- `string Vendor [get, set]`
Gets a `System.String` indicating the vendor of a [GraphicsContext](#) instance.
- `string Renderer [get, set]`
Gets a `System.String` indicating the renderer of a [GraphicsContext](#) instance.

5.31.1 Detailed Description

Defines the version information of a [GraphicsContext](#).

5.31.2 Property Documentation

5.31.2.1 `int OpenTK.Graphics.GraphicsContextVersion.Major [get, set]`

Gets a `System.Int32` indicating the major version of a [GraphicsContext](#) instance.

5.31.2.2 `int OpenTK.Graphics.GraphicsContextVersion.Minor [get, set]`

Gets a `System.Int32` indicating the minor version of a [GraphicsContext](#) instance.

5.31.2.3 `string OpenTK.Graphics.GraphicsContextVersion.Renderer [get, set]`

Gets a `System.String` indicating the renderer of a [GraphicsContext](#) instance.

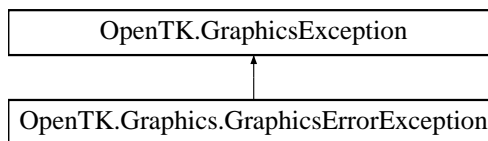
5.31.2.4 string OpenTK.Graphics.GraphicsContextVersion.Vendor [get, set]

Gets a System.String indicating the vendor of a [GraphicsContext](#) instance.

5.32 OpenTK.Graphics.GraphicsErrorException Class Reference

Identifies a specific [OpenGL](#) or OpenGL|ES error. Such exceptions are only thrown when [OpenGL](#) or OpenGL|ES automatic error checking is enabled - [GraphicsContext.ErrorChecking](#) property. Important: Do **not** catch this exception. Rather, fix the underlying issue that caused the error.

Inheritance diagram for OpenTK.Graphics.GraphicsErrorException:



Public Member Functions

- [GraphicsErrorException](#) (string message)

Constructs a new [GraphicsErrorException](#) instance with the specified error message.

5.32.1 Detailed Description

Identifies a specific [OpenGL](#) or OpenGL|ES error. Such exceptions are only thrown when [OpenGL](#) or OpenGL|ES automatic error checking is enabled - [GraphicsContext.ErrorChecking](#) property. Important: Do **not** catch this exception. Rather, fix the underlying issue that caused the error.

5.32.2 Constructor & Destructor Documentation

5.32.2.1 OpenTK.Graphics.GraphicsErrorException.GraphicsErrorException (string message)

Constructs a new [GraphicsErrorException](#) instance with the specified error message.

Parameters

message

5.33 OpenTK.Graphics.GraphicsMode Class Reference

Defines the format for graphics operations.

Public Member Functions

- [GraphicsMode](#) ()
Constructs a new [GraphicsMode](#) with sensible default parameters.
- [GraphicsMode](#) ([ColorFormat](#) color)
Constructs a new [GraphicsMode](#) with the specified parameters.
- [GraphicsMode](#) ([ColorFormat](#) color, int depth)
Constructs a new [GraphicsMode](#) with the specified parameters.
- [GraphicsMode](#) ([ColorFormat](#) color, int depth, int stencil)
Constructs a new [GraphicsMode](#) with the specified parameters.
- [GraphicsMode](#) ([ColorFormat](#) color, int depth, int stencil, int samples)
Constructs a new [GraphicsMode](#) with the specified parameters.
- [GraphicsMode](#) ([ColorFormat](#) color, int depth, int stencil, int samples, [ColorFormat](#) accum)
Constructs a new [GraphicsMode](#) with the specified parameters.
- [GraphicsMode](#) ([ColorFormat](#) color, int depth, int stencil, int samples, [ColorFormat](#) accum, int buffers)
Constructs a new [GraphicsMode](#) with the specified parameters.
- [GraphicsMode](#) ([ColorFormat](#) color, int depth, int stencil, int samples, [ColorFormat](#) accum, int buffers, bool stereo)
Constructs a new [GraphicsMode](#) with the specified parameters.
- override string [ToString](#) ()
Returns a [System.String](#) describing the current [GraphicsFormat](#).
- override int [GetHashCode](#) ()

Returns the hashcode for this instance.

- override bool [Equals](#) (object obj)
Indicates whether obj is equal to this instance.
- bool [Equals](#) ([GraphicsMode](#) other)
Indicates whether other represents the same mode as this instance.

Properties

- IntPtr [Index](#) [get, set]
Gets a nullable System.IntPtr value, indicating the platform-specific index for this GraphicsMode.
- ColorFormat [ColorFormat](#) [get, set]
Gets an [OpenTK.Graphics.ColorFormat](#) that describes the color format for this GraphicsFormat.
- ColorFormat [AccumulatorFormat](#) [get, set]
Gets an [OpenTK.Graphics.ColorFormat](#) that describes the accumulator format for this GraphicsFormat.
- int [Depth](#) [get, set]
Gets a System.Int32 that contains the bits per pixel for the depth buffer for this GraphicsFormat.
- int [Stencil](#) [get, set]
Gets a System.Int32 that contains the bits per pixel for the stencil buffer of this GraphicsFormat.
- int [Samples](#) [get, set]
Gets a System.Int32 that contains the number of FSAA samples per pixel for this GraphicsFormat.
- bool [Stereo](#) [get, set]
Gets a System.Boolean indicating whether this DisplayMode is stereoscopic.
- int [Buffers](#) [get, set]
Gets a System.Int32 containing the number of buffers associated with this DisplayMode.
- static [GraphicsMode Default](#) [get]
Returns an [OpenTK.GraphicsFormat](#) compatible with the underlying platform.

5.33.1 Detailed Description

Defines the format for graphics operations.

5.33.2 Constructor & Destructor Documentation

5.33.2.1 OpenTK.Graphics.GraphicsMode.GraphicsMode ()

Constructs a new [GraphicsMode](#) with sensible default parameters.

5.33.2.2 OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat *color*)

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

color The [ColorFormat](#) of the color buffer.

5.33.2.3 OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat *color*, int *depth*)

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

color The [ColorFormat](#) of the color buffer.

depth The number of bits in the depth buffer.

5.33.2.4 OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat *color*, int *depth*, int *stencil*)

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

color The [ColorFormat](#) of the color buffer.

depth The number of bits in the depth buffer.

stencil The number of bits in the stencil buffer.

5.33.2.5 `OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat color, int depth, int stencil, int samples)`

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

- color* The [ColorFormat](#) of the color buffer.
- depth* The number of bits in the depth buffer.
- stencil* The number of bits in the stencil buffer.
- samples* The number of samples for FSAA.

5.33.2.6 `OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat color, int depth, int stencil, int samples, ColorFormat accum)`

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

- color* The [ColorFormat](#) of the color buffer.
- depth* The number of bits in the depth buffer.
- stencil* The number of bits in the stencil buffer.
- samples* The number of samples for FSAA.
- accum* The [ColorFormat](#) of the accumilliary buffer.

5.33.2.7 `OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat color, int depth, int stencil, int samples, ColorFormat accum, int buffers)`

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

- color* The [ColorFormat](#) of the color buffer.
- depth* The number of bits in the depth buffer.
- stencil* The number of bits in the stencil buffer.
- samples* The number of samples for FSAA.
- accum* The [ColorFormat](#) of the accumilliary buffer.
- buffers* The number of render buffers. Typical values include one (single-), two (double-) or three (triple-buffering).

5.33.2.8 OpenTK.Graphics.GraphicsMode.GraphicsMode (ColorFormat *color*, int *depth*, int *stencil*, int *samples*, ColorFormat *accum*, int *buffers*, bool *stereo*)

Constructs a new [GraphicsMode](#) with the specified parameters.

Parameters

color The [ColorFormat](#) of the color buffer.

depth The number of bits in the depth buffer.

stencil The number of bits in the stencil buffer.

samples The number of samples for FSAA.

accum The [ColorFormat](#) of the accumilliary buffer.

stereo Set to true for a [GraphicsMode](#) with stereographic capabilities.

buffers The number of render buffers. Typical values include one (single-), two (double-) or three (triple-buffering).

5.33.3 Member Function Documentation

5.33.3.1 override bool OpenTK.Graphics.GraphicsMode.Equals (object *obj*)

Indicates whether *obj* is equal to this instance.

Parameters

obj An object instance to compare for equality.

Returns

True, if *obj* equals this instance; false otherwise.

5.33.3.2 bool OpenTK.Graphics.GraphicsMode.Equals (GraphicsMode *other*)

Indicates whether *other* represents the same mode as this instance.

Parameters

other The [GraphicsMode](#) to compare to.

Returns

True, if *other* is equal to this instance; false otherwise.

5.33.3.3 **override int OpenTK.Graphics.GraphicsMode.GetHashCode ()**

Returns the hashcode for this instance.

Returns

A System.Int32 hashcode for this instance.

5.33.3.4 **override string OpenTK.Graphics.GraphicsMode.ToString ()**

Returns a System.String describing the current GraphicsFormat.

Returns

! System.String describing the current GraphicsFormat.

5.33.4 **Property Documentation**

5.33.4.1 **ColorFormat OpenTK.Graphics.GraphicsMode.AccumulatorFormat [get, set]**

Gets an [OpenTK.Graphics.ColorFormat](#) that describes the accumulator format for this GraphicsFormat.

5.33.4.2 **int OpenTK.Graphics.GraphicsMode.Buffers [get, set]**

Gets a System.Int32 containing the number of buffers associated with this DisplayMode.

5.33.4.3 **ColorFormat OpenTK.Graphics.GraphicsMode.ColorFormat [get, set]**

Gets an [OpenTK.Graphics.ColorFormat](#) that describes the color format for this GraphicsFormat.

5.33.4.4 **GraphicsFormat OpenTK.Graphics.GraphicsMode.Default [static, get]**

Returns an OpenTK.GraphicsFormat compatible with the underlying platform.

5.33.4.5 int OpenTK.Graphics.GraphicsMode.Depth [get, set]

Gets a System.Int32 that contains the bits per pixel for the depth buffer for this GraphicsFormat.

5.33.4.6 IntPtr OpenTK.Graphics.GraphicsMode.Index [get, set]

Gets a nullable System.IntPtr value, indicating the platform-specific index for this [GraphicsMode](#).

5.33.4.7 int OpenTK.Graphics.GraphicsMode.Samples [get, set]

Gets a System.Int32 that contains the number of FSAA samples per pixel for this GraphicsFormat.

5.33.4.8 int OpenTK.Graphics.GraphicsMode.Stencil [get, set]

Gets a System.Int32 that contains the bits per pixel for the stencil buffer of this GraphicsFormat.

5.33.4.9 bool OpenTK.Graphics.GraphicsMode.Stereo [get, set]

Gets a System.Boolean indicating whether this DisplayMode is stereoscopic.

5.34 OpenTK.Graphics.GraphicsModeException Class Reference

Represents errors related to unavailable graphics parameters.

Public Member Functions

- [GraphicsModeException](#) ()
Constructs a new [GraphicsModeException](#).
- [GraphicsModeException](#) (string message)
Constructs a new [GraphicsModeException](#) with the given error message.

5.34.1 Detailed Description

Represents errors related to unavailable graphics parameters.

5.34.2 Constructor & Destructor Documentation

5.34.2.1 OpenTK.Graphics.GraphicsModeException.GraphicsModeException ()

Constructs a new [GraphicsModeException](#).

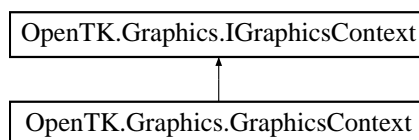
5.34.2.2 OpenTK.Graphics.GraphicsModeException.GraphicsModeException (string message)

Constructs a new [GraphicsModeException](#) with the given error message.

5.35 OpenTK.Graphics.IGraphicsContext Interface Reference

Provides methods for creating and interacting with an [OpenGL](#) context.

Inheritance diagram for OpenTK.Graphics.IGraphicsContext:



Public Member Functions

- void [SwapBuffers](#) ()
Swaps buffers, presenting the rendered scene to the user.
- void [MakeCurrent](#) ([IWindowInfo](#) window)
Makes the [GraphicsContext](#) current in the calling thread.
- void [Update](#) ([IWindowInfo](#) window)
Updates the graphics context. This must be called when the region the graphics context is drawn to is resized.

- void [LoadAll](#) ()
Loads all [OpenGL](#) entry points. Requires this instance to be current on the calling thread.

Properties

- bool [IsCurrent](#) [get]
Gets a System.Boolean indicating whether this instance is current in the calling thread.
- bool [IsDisposed](#) [get]
Gets a System.Boolean indicating whether this instance has been disposed. It is an error to access any instance methods if this property returns true.
- bool [VSync](#) [get, set]
Gets or sets a value indicating whether VSyncing is enabled.
- [GraphicsMode GraphicsMode](#) [get]
Gets the [GraphicsMode](#) of this instance.
- bool [ErrorChecking](#) [get, set]
Gets or sets a System.Boolean, indicating whether automatic error checking should be performed.

5.35.1 Detailed Description

Provides methods for creating and interacting with an [OpenGL](#) context.

5.35.2 Member Function Documentation

5.35.2.1 void OpenTK.Graphics.IGraphicsContext.LoadAll ()

Loads all [OpenGL](#) entry points. Requires this instance to be current on the calling thread.

5.35.2.2 void OpenTK.Graphics.IGraphicsContext.MakeCurrent (IWindowInfo window)

Makes the [GraphicsContext](#) current in the calling thread.

Parameters

window An [OpenTK.Platform.IWindowInfo](#) structure that points to a valid window.

[OpenGL](#) commands in one thread, affect the [GraphicsContext](#) which is current in that thread.

It is an error to issue an [OpenGL](#) command in a thread without a current [GraphicsContext](#).

5.35.2.3 void OpenTK.Graphics.IGraphicsContext.SwapBuffers ()

Swaps buffers, presenting the rendered scene to the user.

5.35.2.4 void OpenTK.Graphics.IGraphicsContext.Update (IWindowInfo *window*)

Updates the graphics context. This must be called when the region the graphics context is drawn to is resized.

Parameters

window

5.35.3 Property Documentation**5.35.3.1 bool OpenTK.Graphics.IGraphicsContext.ErrorChecking [get, set]**

Gets or sets a System.Boolean, indicating whether automatic error checking should be performed.

It is an error to enable error checking inside a Begin()-End() region.

This method only affects the debug version of OpenTK.dll.

5.35.3.2 GraphicsMode OpenTK.Graphics.IGraphicsContext.GraphicsMode [get]

Gets the [GraphicsMode](#) of this instance.

5.35.3.3 bool OpenTK.Graphics.IGraphicsContext.IsCurrent [get]

Gets a System.Boolean indicating whether this instance is current in the calling thread.

5.35.3.4 bool OpenTK.Graphics.IGraphicsContext.IsDisposed [get]

Gets a System.Boolean indicating whether this instance has been disposed. It is an error to access any instance methods if this property returns true.

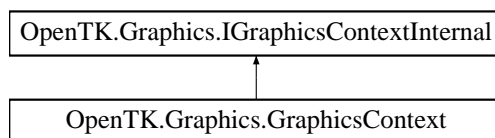
5.35.3.5 bool OpenTK.Graphics.IGraphicsContext.VSync [get, set]

Gets or sets a value indicating whether VSyncing is enabled.

5.36 OpenTK.Graphics.IGraphicsContextInternal Interface Reference

Provides methods to create new GraphicsContexts. Should only be used for extending OpenTK.

Inheritance diagram for OpenTK.Graphics.IGraphicsContextInternal:



Public Member Functions

- void [LoadAll](#) ()
Loads all [OpenGL](#) entry points. Requires this instance to be current on the calling thread.
- IntPtr [GetAddress](#) (string function)
Gets the address of an [OpenGL](#) extension function.

Properties

- [IGraphicsContext Implementation](#) [get]
Gets the internal implementation of the current instance.
- [ContextHandle Context](#) [get]
Gets a handle to the [OpenGL](#) rendering context.

5.36.1 Detailed Description

Provides methods to create new GraphicsContexts. Should only be used for extending OpenTK.

5.36.2 Member Function Documentation

5.36.2.1 IntPtr OpenTK.Graphics.IGraphicsContextInternal.GetAddress (string *function*)

Gets the address of an [OpenGL](#) extension function.

Parameters

function The name of the [OpenGL](#) function (e.g. "glGetString")

Returns

A pointer to the specified function or IntPtr.Zero if the function isn't available in the current opengl context.

5.36.2.2 void OpenTK.Graphics.IGraphicsContextInternal.LoadAll ()

Loads all [OpenGL](#) entry points. Requires this instance to be current on the calling thread.

Implemented in [OpenTK.Graphics.GraphicsContext](#).

5.36.3 Property Documentation

5.36.3.1 ContextHandle OpenTK.Graphics.IGraphicsContextInternal.Context [get]

Gets a handle to the [OpenGL](#) rendering context.

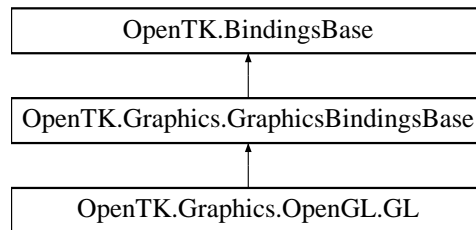
5.36.3.2 IGraphicsContext OpenTK.Graphics.IGraphicsContextInternal.Implementation [get]

Gets the internal implementation of the current instance.

5.37 OpenTK.Graphics.OpenGL.GL Class Reference

[OpenGL](#) bindings for .NET, implementing the full [OpenGL](#) API, including extensions.

Inheritance diagram for OpenTK.Graphics.OpenGL.GL:



Static Public Member Functions

- static void [Accum](#) (OpenTK.Graphics.OpenGL.AccumOp op, Single value)
Operate on the accumulation buffer.
- static void [ActiveTexture](#) (OpenTK.Graphics.OpenGL.TextureUnit texture)
Select active texture unit.
- static void [AlphaFunc](#) (OpenTK.Graphics.OpenGL.AlphaFunction func, Single @ref)
Specify the alpha test function.
- static bool [AreTexturesResident](#) (Int32 n, Int32[] textures,[OutAttribute] bool[] residences)
Determine if textures are loaded in texture memory.
- static bool [AreTexturesResident](#) (Int32 n, ref Int32 textures,[OutAttribute] out bool residences)
Determine if textures are loaded in texture memory.
- static unsafe bool [AreTexturesResident](#) (Int32 n, Int32 *textures,[OutAttribute] bool *residences)
Determine if textures are loaded in texture memory.
- static bool [AreTexturesResident](#) (Int32 n, UInt32[] textures,[OutAttribute] bool[] residences)
Determine if textures are loaded in texture memory.

- static bool [AreTexturesResident](#) (Int32 n, ref UInt32 textures,[OutAttribute] out bool residences)
Determine if textures are loaded in texture memory.
- static unsafe bool [AreTexturesResident](#) (Int32 n, UInt32 *textures,[OutAttribute] bool *residences)
Determine if textures are loaded in texture memory.
- static void [ArrayElement](#) (Int32 i)
Render a vertex using the specified vertex array element.
- static void [AttachShader](#) (Int32 program, Int32 shader)
Attaches a shader object to a program object.
- static void [AttachShader](#) (UInt32 program, UInt32 shader)
Attaches a shader object to a program object.
- static void [Begin](#) (OpenTK.Graphics.OpenGL.BeginMode mode)
Delimit the vertices of a primitive or a group of like primitives.
- static void **BeginConditionalRender** (Int32 id, OpenTK.Graphics.OpenGL.ConditionalRenderMode mode)
- static void **BeginConditionalRender** (UInt32 id, OpenTK.Graphics.OpenGL.ConditionalRenderMode mode)
- static void [BeginQuery](#) (OpenTK.Graphics.OpenGL.QueryTarget target, Int32 id)
Delimit the boundaries of a query object.
- static void [BeginQuery](#) (OpenTK.Graphics.OpenGL.QueryTarget target, UInt32 id)
Delimit the boundaries of a query object.
- static void **BeginTransformFeedback** (OpenTK.Graphics.OpenGL.BeginFeedbackMode primitiveMode)
- static void [BindAttribLocation](#) (Int32 program, Int32 index, String name)
Associates a generic vertex attribute index with a named attribute variable.
- static void [BindAttribLocation](#) (UInt32 program, UInt32 index, String name)
Associates a generic vertex attribute index with a named attribute variable.
- static void [BindBuffer](#) (OpenTK.Graphics.OpenGL.BufferTarget target, Int32 buffer)
Bind a named buffer object.

- static void [BindBuffer](#) (OpenTK.Graphics.OpenGL.BufferTarget target, UInt32 buffer)

Bind a named buffer object.

- static void **BindBufferBase** (OpenTK.Graphics.OpenGL.BufferTarget target, Int32 index, Int32 buffer)
- static void **BindBufferBase** (OpenTK.Graphics.OpenGL.BufferTarget target, UInt32 index, UInt32 buffer)
- static void **BindBufferRange** (OpenTK.Graphics.OpenGL.BufferTarget target, Int32 index, Int32 buffer, IntPtr offset, IntPtr size)
- static void **BindBufferRange** (OpenTK.Graphics.OpenGL.BufferTarget target, UInt32 index, UInt32 buffer, IntPtr offset, IntPtr size)
- static void **BindFragDataLocation** (Int32 program, Int32 color, String name)
- static void **BindFragDataLocation** (UInt32 program, UInt32 color, String name)
- static void **BindFramebuffer** (OpenTK.Graphics.OpenGL.FramebufferTarget target, Int32 framebuffer)
- static void **BindFramebuffer** (OpenTK.Graphics.OpenGL.FramebufferTarget target, UInt32 framebuffer)
- static void **BindRenderbuffer** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, Int32 renderbuffer)
- static void **BindRenderbuffer** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, UInt32 renderbuffer)
- static void [BindTexture](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 texture)

Bind a named texture to a texturing target.

- static void [BindTexture](#) (OpenTK.Graphics.OpenGL.TextureTarget target, UInt32 texture)

Bind a named texture to a texturing target.

- static void **BindVertexArray** (Int32 array)
- static void **BindVertexArray** (UInt32 array)
- static void [Bitmap](#) (Int32 width, Int32 height, Single xorig, Single yorig, Single xmove, Single ymove, Byte[] bitmap)

Draw a bitmap.

- static void [Bitmap](#) (Int32 width, Int32 height, Single xorig, Single yorig, Single xmove, Single ymove, ref Byte bitmap)

Draw a bitmap.

- static unsafe void [Bitmap](#) (Int32 width, Int32 height, Single xorig, Single yorig, Single xmove, Single ymove, Byte *bitmap)

Draw a bitmap.

- static void **BlendColor** (Single red, Single green, Single blue, Single alpha)
Set the blend color.
- static void **BlendEquation** (OpenTK.Graphics.OpenGL.BlendEquationMode mode)
Specify the equation used for both the RGB blend equation and the Alpha blend equation.
- static void **BlendEquation** (Int32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend mode)
Specify the equation used for both the RGB blend equation and the Alpha blend equation.
- static void **BlendEquation** (UInt32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend mode)
Specify the equation used for both the RGB blend equation and the Alpha blend equation.
- static void **BlendEquationSeparate** (OpenTK.Graphics.OpenGL.BlendEquationMode modeRGB, OpenTK.Graphics.OpenGL.BlendEquationMode modeAlpha)
Set the RGB blend equation and the alpha blend equation separately.
- static void **BlendEquationSeparate** (Int32 buf, OpenTK.Graphics.OpenGL.BlendEquationMode modeRGB, OpenTK.Graphics.OpenGL.BlendEquationMode modeAlpha)
Set the RGB blend equation and the alpha blend equation separately.
- static void **BlendEquationSeparate** (UInt32 buf, OpenTK.Graphics.OpenGL.BlendEquationMode modeRGB, OpenTK.Graphics.OpenGL.BlendEquationMode modeAlpha)
Set the RGB blend equation and the alpha blend equation separately.
- static void **BlendFunc** (OpenTK.Graphics.OpenGL.BlendingFactorSrc sfactor, OpenTK.Graphics.OpenGL.BlendingFactorDest dfactor)
Specify pixel arithmetic.
- static void **BlendFunc** (Int32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend src, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dst)
Specify pixel arithmetic.
- static void **BlendFunc** (UInt32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend src, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dst)

Specify pixel arithmetic.

- static void **BlendFuncSeparate** (OpenTK.Graphics.OpenGL.BlendingFactorSrc sfactorRGB, OpenTK.Graphics.OpenGL.BlendingFactorDest dfactorRGB, OpenTK.Graphics.OpenGL.BlendingFactorSrc sfactorAlpha, OpenTK.Graphics.OpenGL.BlendingFactorDest dfactorAlpha)

Specify pixel arithmetic for RGB and alpha components separately.

- static void **BlendFuncSeparate** (Int32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcAlpha, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstAlpha)

Specify pixel arithmetic for RGB and alpha components separately.

- static void **BlendFuncSeparate** (UInt32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcAlpha, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstAlpha)

Specify pixel arithmetic for RGB and alpha components separately.

- static void **BlitFramebuffer** (Int32 srcX0, Int32 srcY0, Int32 srcX1, Int32 srcY1, Int32 dstX0, Int32 dstY0, Int32 dstX1, Int32 dstY1, OpenTK.Graphics.OpenGL.ClearBufferMask mask, OpenTK.Graphics.OpenGL.BlitFramebufferFilter filter)
- static void **BufferData** (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size, IntPtr data, OpenTK.Graphics.OpenGL.BufferUsageHint usage)

Creates and initializes a buffer object's data store.

- static void **BufferData< T2 >** (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] T2[] data, OpenTK.Graphics.OpenGL.BufferUsageHint usage)

Creates and initializes a buffer object's data store.

- static void **BufferData< T2 >** (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] T2[,] data, OpenTK.Graphics.OpenGL.BufferUsageHint usage)

Creates and initializes a buffer object's data store.

- static void **BufferData< T2 >** (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] T2[, ,] data, OpenTK.Graphics.OpenGL.BufferUsageHint usage)

Creates and initializes a buffer object's data store.

- static void [BufferData](#)< T2 > (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size,[InAttribute, OutAttribute] ref T2 data, OpenTK.Graphics.OpenGL.BufferUsageHint usage)

Creates and initializes a buffer object's data store.

- static void [BufferSubData](#) (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size, IntPtr data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData](#)< T3 > (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[] data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData](#)< T3 > (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[,] data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData](#)< T3 > (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[,], data)

Updates a subset of a buffer object's data store.

- static void [BufferSubData](#)< T3 > (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] ref T3 data)

Updates a subset of a buffer object's data store.

- static void [CallList](#) (Int32 list)

Execute a display list.

- static void [CallList](#) (UInt32 list)

Execute a display list.

- static void [CallLists](#) (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type, IntPtr lists)

Execute a list of display lists.

- static void [CallLists](#)< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type,[InAttribute, OutAttribute] T2[] lists)

Execute a list of display lists.

- static void [CallLists](#)< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type,[InAttribute, OutAttribute] T2[,] lists)

Execute a list of display lists.

- static void **CallLists**< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type,[InAttribute, OutAttribute] T2[,..] lists)
Execute a list of display lists.
- static void **CallLists**< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type,[InAttribute, OutAttribute] ref T2 lists)
Execute a list of display lists.
- static OpenTK.Graphics.OpenGL.FramebufferErrorCode **CheckFramebufferStatus** (OpenTK.Graphics.OpenGL.FramebufferTarget target)
- static void **ClampColor** (OpenTK.Graphics.OpenGL.ClampColorTarget target, OpenTK.Graphics.OpenGL.ClampColorMode clamp)
- static void **Clear** (OpenTK.Graphics.OpenGL.ClearBufferMask mask)
Clear buffers to preset values.
- static void **ClearAccum** (Single red, Single green, Single blue, Single alpha)
Specify clear values for the accumulation buffer.
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, Single depth, Int32 stencil)
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, Single[] value)
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, ref Single value)
- static unsafe void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, Single *value)
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, Int32[] value)
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, ref Int32 value)
- static unsafe void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, Int32 *value)
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, UInt32[] value)
- static void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, ref UInt32 value)
- static unsafe void **ClearBuffer** (OpenTK.Graphics.OpenGL.ClearBuffer buffer, Int32 drawbuffer, UInt32 *value)
- static void **ClearColor** (Single red, Single green, Single blue, Single alpha)
Specify clear values for the color buffers.
- static void **ClearDepth** (Double depth)

Specify the clear value for the depth buffer.

- static void **ClearIndex** (Single c)
Specify the clear value for the color index buffers.
- static void **ClearStencil** (Int32 s)
Specify the clear value for the stencil buffer.
- static void **ClientActiveTexture** (OpenTK.Graphics.OpenGL.TextureUnit texture)
Select active texture unit.
- static OpenTK.Graphics.OpenGL.ArbSync **ClientWaitSync** (IntPtr sync, Int32 flags, Int64 timeout)
- static OpenTK.Graphics.OpenGL.ArbSync **ClientWaitSync** (IntPtr sync, UInt32 flags, UInt64 timeout)
- static void **ClipPlane** (OpenTK.Graphics.OpenGL.ClipPlaneName plane, Double[] equation)
Specify a plane against which all geometry is clipped.
- static void **ClipPlane** (OpenTK.Graphics.OpenGL.ClipPlaneName plane, ref Double equation)
Specify a plane against which all geometry is clipped.
- static unsafe void **ClipPlane** (OpenTK.Graphics.OpenGL.ClipPlaneName plane, Double *equation)
Specify a plane against which all geometry is clipped.
- static void **Color3** (SByte red, SByte green, SByte blue)
Set the current color.
- static void **Color3** (SByte[] v)
Set the current color.
- static void **Color3** (ref SByte v)
Set the current color.
- static unsafe void **Color3** (SByte *v)
Set the current color.
- static void **Color3** (Double red, Double green, Double blue)
Set the current color.

- static void [Color3](#) (Double[] v)
Set the current color.
- static void [Color3](#) (ref Double v)
Set the current color.
- static unsafe void [Color3](#) (Double *v)
Set the current color.
- static void [Color3](#) (Single red, Single green, Single blue)
Set the current color.
- static void [Color3](#) (Single[] v)
Set the current color.
- static void [Color3](#) (ref Single v)
Set the current color.
- static unsafe void [Color3](#) (Single *v)
Set the current color.
- static void [Color3](#) (Int32 red, Int32 green, Int32 blue)
Set the current color.
- static void [Color3](#) (Int32[] v)
Set the current color.
- static void [Color3](#) (ref Int32 v)
Set the current color.
- static unsafe void [Color3](#) (Int32 *v)
Set the current color.
- static void [Color3](#) (Int16 red, Int16 green, Int16 blue)
Set the current color.
- static void [Color3](#) (Int16[] v)
Set the current color.
- static void [Color3](#) (ref Int16 v)
Set the current color.

- static unsafe void [Color3](#) (Int16 *v)
Set the current color.
- static void [Color3](#) (Byte red, Byte green, Byte blue)
Set the current color.
- static void [Color3](#) (Byte[] v)
Set the current color.
- static void [Color3](#) (ref Byte v)
Set the current color.
- static unsafe void [Color3](#) (Byte *v)
Set the current color.
- static void [Color3](#) (UInt32 red, UInt32 green, UInt32 blue)
Set the current color.
- static void [Color3](#) (UInt32[] v)
Set the current color.
- static void [Color3](#) (ref UInt32 v)
Set the current color.
- static unsafe void [Color3](#) (UInt32 *v)
Set the current color.
- static void [Color3](#) (UInt16 red, UInt16 green, UInt16 blue)
Set the current color.
- static void [Color3](#) (UInt16[] v)
Set the current color.
- static void [Color3](#) (ref UInt16 v)
Set the current color.
- static unsafe void [Color3](#) (UInt16 *v)
Set the current color.
- static void [Color4](#) (SByte red, SByte green, SByte blue, SByte alpha)
Set the current color.

- static void [Color4](#) (SByte[] v)
Set the current color.
- static void [Color4](#) (ref SByte v)
Set the current color.
- static unsafe void [Color4](#) (SByte *v)
Set the current color.
- static void [Color4](#) (Double red, Double green, Double blue, Double alpha)
Set the current color.
- static void [Color4](#) (Double[] v)
Set the current color.
- static void [Color4](#) (ref Double v)
Set the current color.
- static unsafe void [Color4](#) (Double *v)
Set the current color.
- static void [Color4](#) (Single red, Single green, Single blue, Single alpha)
Set the current color.
- static void [Color4](#) (Single[] v)
Set the current color.
- static void [Color4](#) (ref Single v)
Set the current color.
- static unsafe void [Color4](#) (Single *v)
Set the current color.
- static void [Color4](#) (Int32 red, Int32 green, Int32 blue, Int32 alpha)
Set the current color.
- static void [Color4](#) (Int32[] v)
Set the current color.
- static void [Color4](#) (ref Int32 v)
Set the current color.

- static unsafe void [Color4](#) (Int32 *v)
Set the current color.
- static void [Color4](#) (Int16 red, Int16 green, Int16 blue, Int16 alpha)
Set the current color.
- static void [Color4](#) (Int16[] v)
Set the current color.
- static void [Color4](#) (ref Int16 v)
Set the current color.
- static unsafe void [Color4](#) (Int16 *v)
Set the current color.
- static void [Color4](#) (Byte red, Byte green, Byte blue, Byte alpha)
Set the current color.
- static void [Color4](#) (Byte[] v)
Set the current color.
- static void [Color4](#) (ref Byte v)
Set the current color.
- static unsafe void [Color4](#) (Byte *v)
Set the current color.
- static void [Color4](#) (UInt32 red, UInt32 green, UInt32 blue, UInt32 alpha)
Set the current color.
- static void [Color4](#) (UInt32[] v)
Set the current color.
- static void [Color4](#) (ref UInt32 v)
Set the current color.
- static unsafe void [Color4](#) (UInt32 *v)
Set the current color.
- static void [Color4](#) (UInt16 red, UInt16 green, UInt16 blue, UInt16 alpha)
Set the current color.

- static void [Color4](#) (UInt16[] v)
Set the current color.
- static void [Color4](#) (ref UInt16 v)
Set the current color.
- static unsafe void [Color4](#) (UInt16 *v)
Set the current color.
- static void [ColorMask](#) (bool red, bool green, bool blue, bool alpha)
Enable and disable writing of frame buffer color components.
- static void [ColorMask](#) (Int32 index, bool r, bool g, bool b, bool a)
Enable and disable writing of frame buffer color components.
- static void [ColorMask](#) (UInt32 index, bool r, bool g, bool b, bool a)
Enable and disable writing of frame buffer color components.
- static void [ColorMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.ColorMaterialParameter mode)
Cause a material color to track the current color.
- static void [ColorPointer](#) (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, IntPtr pointer)
Define an array of colors.
- static void [ColorPointer](#)< T3 > (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer)
Define an array of colors.
- static void [ColorPointer](#)< T3 > (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[,] pointer)
Define an array of colors.
- static void [ColorPointer](#)< T3 > (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[, ,] pointer)
Define an array of colors.
- static void [ColorPointer](#)< T3 > (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer)

Define an array of colors.

- static void [ColorSubTable](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr data)

Respecify a portion of a color table.

- static void [ColorSubTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T5[] data)

Respecify a portion of a color table.

- static void [ColorSubTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T5[,] data)

Respecify a portion of a color table.

- static void [ColorSubTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T5[, ,] data)

Respecify a portion of a color table.

- static void [ColorSubTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T5 data)

Respecify a portion of a color table.

- static void [ColorTable](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr table)

Define a color lookup table.

- static void [ColorTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T5[] table)

Define a color lookup table.

- static void [ColorTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T5[, table])

Define a color lookup table.

- static void [ColorTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T5[, table])

Define a color lookup table.

- static void [ColorTable< T5 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T5 table)

Define a color lookup table.

- static void [ColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, Single[] @params)

Set color lookup table parameters.

- static void [ColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, ref Single[] @params)

Set color lookup table parameters.

- static unsafe void [ColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, Single* @params)

Set color lookup table parameters.

- static void [ColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, Int32[] @params)

Set color lookup table parameters.

- static void [ColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, ref Int32[] @params)

Set color lookup table parameters.

- static unsafe void [ColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, Int32 *@params)

Set color lookup table parameters.

- static void [CompileShader](#) (Int32 shader)

Compiles a shader object.

- static void [CompileShader](#) (UInt32 shader)

Compiles a shader object.

- static void [CompressedTexImage1D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, Int32 imageSize, IntPtr data)

Specify a one-dimensional texture image in a compressed format.

- static void [CompressedTexImage1D](#)< [T6](#) > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T6[] data)

Specify a one-dimensional texture image in a compressed format.

- static void [CompressedTexImage1D](#)< [T6](#) > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T6[,] data)

Specify a one-dimensional texture image in a compressed format.

- static void [CompressedTexImage1D](#)< [T6](#) > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T6[, ,] data)

Specify a one-dimensional texture image in a compressed format.

- static void [CompressedTexImage1D](#)< [T6](#) > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] ref T6 data)

Specify a one-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[]
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[],
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] T7[,]
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage2D](#)< [T7](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width,
Int32 height, Int32 border, Int32 imageSize,[InAttribute, OutAttribute] ref T7
data)

Specify a two-dimensional texture image in a compressed format.

- static void [CompressedTexImage3D](#) (OpenTK.Graphics.OpenGL.TextureTarget
target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalfor-
mat, Int32 width, Int32 height, Int32 depth, Int32 border, Int32 imageSize, IntPtr
data)

Specify a three-dimensional texture image in a compressed format.

- static void [CompressedTexImage3D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32
width, Int32 height, Int32 depth, Int32 border, Int32 imageSize,[InAttribute,
OutAttribute] T8[] data)

Specify a three-dimensional texture image in a compressed format.

- static void [CompressedTexImage3D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32
width, Int32 height, Int32 depth, Int32 border, Int32 imageSize,[InAttribute,
OutAttribute] T8[,] data)

Specify a three-dimensional texture image in a compressed format.

- static void [CompressedTexImage3D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32
width, Int32 height, Int32 depth, Int32 border, Int32 imageSize,[InAttribute,
OutAttribute] T8[,] data)

Specify a three-dimensional texture image in a compressed format.

- static void [CompressedTexImage3D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32
width, Int32 height, Int32 depth, Int32 border, Int32 imageSize,[InAttribute,
OutAttribute] ref T8 data)

Specify a three-dimensional texture image in a compressed format.

- static void [CompressedTexSubImage1D](#) (OpenTK.Graphics.OpenGL.TextureTarget
target, Int32 level, Int32 xoffset, Int32 width,
OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize, IntPtr
data)

Specify a one-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage1D](#)< [T6](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoff-
set, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, Int32
imageSize,[InAttribute, OutAttribute] T6[] data)

Specify a one-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage1D](#)< [T6](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoff-
set, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, Int32
imageSize,[InAttribute, OutAttribute] T6[,] data)

Specify a one-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage1D](#)< [T6](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoff-
set, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, Int32
imageSize,[InAttribute, OutAttribute] T6[,] data)

Specify a one-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage1D](#)< [T6](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] ref T6 data)
Specify a one-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize, IntPtr data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T8[] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T8[,] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T8[, ,] data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage2D](#)< [T8](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] ref T8 data)
Specify a two-dimensional texture subimage in a compressed format.
- static void [CompressedTexSubImage3D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize, IntPtr data)

Specify a three-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage3D](#)< [T10](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T10[] data)

Specify a three-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage3D](#)< [T10](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T10[,] data)

Specify a three-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage3D](#)< [T10](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] T10[,], data)

Specify a three-dimensional texture subimage in a compressed format.

- static void [CompressedTexSubImage3D](#)< [T10](#) >
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,[InAttribute, OutAttribute] ref T10 data)

Specify a three-dimensional texture subimage in a compressed format.

- static void [ConvolutionFilter1D](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr image)

Define a one-dimensional convolution filter.

- static void [ConvolutionFilter1D](#)< [T5](#) > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T5[] image)

Define a one-dimensional convolution filter.

- static void [ConvolutionFilter1D< T5 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T5[, image])

Define a one-dimensional convolution filter.

- static void [ConvolutionFilter1D< T5 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T5[, image])

Define a one-dimensional convolution filter.

- static void [ConvolutionFilter1D< T5 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T5 image)

Define a one-dimensional convolution filter.

- static void [ConvolutionFilter2D](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr image)

Define a two-dimensional convolution filter.

- static void [ConvolutionFilter2D< T6 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[] image)

Define a two-dimensional convolution filter.

- static void [ConvolutionFilter2D< T6 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[, image])

Define a two-dimensional convolution filter.

- static void [ConvolutionFilter2D< T6 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format,

OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[,]
image)

Define a two-dimensional convolution filter.

- static void [ConvolutionFilter2D< T6 >](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T6 image)

Define a two-dimensional convolution filter.

- static void [ConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Single @params)

Set convolution parameters.

- static void [ConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Single[] @params)

Set convolution parameters.

- static unsafe void [ConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Single *@params)

Set convolution parameters.

- static void [ConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Int32 @params)

Set convolution parameters.

- static void [ConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Int32[] @params)

Set convolution parameters.

- static unsafe void [ConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Int32 *@params)

Set convolution parameters.

- static void **CopyBufferSubData** (OpenTK.Graphics.OpenGL.BufferTarget readTarget, OpenTK.Graphics.OpenGL.BufferTarget writeTarget, IntPtr readOffset, IntPtr writeOffset, IntPtr size)

- static void [CopyColorSubTable](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 x, Int32 y, Int32 width)
Respecify a portion of a color table.
- static void [CopyColorTable](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width)
Copy pixels into a color table.
- static void [CopyConvolutionFilter1D](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width)
Copy pixels into a one-dimensional convolution filter.
- static void [CopyConvolutionFilter2D](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width, Int32 height)
Copy pixels into a two-dimensional convolution filter.
- static void [CopyPixels](#) (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelCopyType type)
Copy pixels in the frame buffer.
- static void [CopyTexImage1D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width, Int32 border)
Copy pixels into a 1D texture image.
- static void [CopyTexImage2D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width, Int32 height, Int32 border)
Copy pixels into a 2D texture image.
- static void [CopyTexSubImage1D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 x, Int32 y, Int32 width)
Copy a one-dimensional texture subimage.
- static void [CopyTexSubImage2D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32 height)
Copy a two-dimensional texture subimage.

- static void [CopyTexSubImage3D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 x, Int32 y, Int32 width, Int32 height)
Copy a three-dimensional texture subimage.
- static Int32 [CreateProgram](#) ()
Creates a program object.
- static Int32 [CreateShader](#) (OpenTK.Graphics.OpenGL.ShaderType type)
Creates a shader object.
- static void [CullFace](#) (OpenTK.Graphics.OpenGL.CullFaceMode mode)
Specify whether front- or back-facing facets can be culled.
- static void [DeleteBuffers](#) (Int32 n, Int32[] buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, ref Int32 buffers)
Delete named buffer objects.
- static unsafe void [DeleteBuffers](#) (Int32 n, Int32 *buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, UInt32[] buffers)
Delete named buffer objects.
- static void [DeleteBuffers](#) (Int32 n, ref UInt32 buffers)
Delete named buffer objects.
- static unsafe void [DeleteBuffers](#) (Int32 n, UInt32 *buffers)
Delete named buffer objects.
- static void **DeleteFramebuffers** (Int32 n, Int32[] framebuffers)
- static void **DeleteFramebuffers** (Int32 n, ref Int32 framebuffers)
- static unsafe void **DeleteFramebuffers** (Int32 n, Int32 *framebuffers)
- static void **DeleteFramebuffers** (Int32 n, UInt32[] framebuffers)
- static void **DeleteFramebuffers** (Int32 n, ref UInt32 framebuffers)
- static unsafe void **DeleteFramebuffers** (Int32 n, UInt32 *framebuffers)
- static void [DeleteLists](#) (Int32 list, Int32 range)
Delete a contiguous group of display lists.
- static void [DeleteLists](#) (UInt32 list, Int32 range)

Delete a contiguous group of display lists.

- static void [DeleteProgram](#) (Int32 program)
Deletes a program object.
- static void [DeleteProgram](#) (UInt32 program)
Deletes a program object.
- static void [DeleteQueries](#) (Int32 n, Int32[] ids)
Delete named query objects.
- static void [DeleteQueries](#) (Int32 n, ref Int32 ids)
Delete named query objects.
- static unsafe void [DeleteQueries](#) (Int32 n, Int32 *ids)
Delete named query objects.
- static void [DeleteQueries](#) (Int32 n, UInt32[] ids)
Delete named query objects.
- static void [DeleteQueries](#) (Int32 n, ref UInt32 ids)
Delete named query objects.
- static unsafe void [DeleteQueries](#) (Int32 n, UInt32 *ids)
Delete named query objects.
- static void **DeleteRenderbuffers** (Int32 n, Int32[] renderbuffers)
- static void **DeleteRenderbuffers** (Int32 n, ref Int32 renderbuffers)
- static unsafe void **DeleteRenderbuffers** (Int32 n, Int32 *renderbuffers)
- static void **DeleteRenderbuffers** (Int32 n, UInt32[] renderbuffers)
- static void **DeleteRenderbuffers** (Int32 n, ref UInt32 renderbuffers)
- static unsafe void **DeleteRenderbuffers** (Int32 n, UInt32 *renderbuffers)
- static void [DeleteShader](#) (Int32 shader)
Deletes a shader object.
- static void [DeleteShader](#) (UInt32 shader)
Deletes a shader object.
- static void **DeleteSync** (IntPtr sync)
- static void [DeleteTextures](#) (Int32 n, Int32[] textures)
Delete named textures.
- static void [DeleteTextures](#) (Int32 n, ref Int32 textures)

Delete named textures.

- static unsafe void [DeleteTextures](#) (Int32 n, Int32 *textures)

Delete named textures.

- static void [DeleteTextures](#) (Int32 n, UInt32[] textures)

Delete named textures.

- static void [DeleteTextures](#) (Int32 n, ref UInt32 textures)

Delete named textures.

- static unsafe void [DeleteTextures](#) (Int32 n, UInt32 *textures)

Delete named textures.

- static void **DeleteVertexArrays** (Int32 n, Int32[] arrays)
- static void **DeleteVertexArrays** (Int32 n, ref Int32 arrays)
- static unsafe void **DeleteVertexArrays** (Int32 n, Int32 *arrays)
- static void **DeleteVertexArrays** (Int32 n, UInt32[] arrays)
- static void **DeleteVertexArrays** (Int32 n, ref UInt32 arrays)
- static unsafe void **DeleteVertexArrays** (Int32 n, UInt32 *arrays)
- static void [DepthFunc](#) (OpenTK.Graphics.OpenGL.DepthFunction func)

Specify the value used for depth buffer comparisons.

- static void [DepthMask](#) (bool flag)

Enable or disable writing into the depth buffer.

- static void [DepthRange](#) (Double near, Double far)

Specify mapping of depth values from normalized device coordinates to window coordinates.

- static void [DetachShader](#) (Int32 program, Int32 shader)

Detaches a shader object from a program object to which it is attached.

- static void [DetachShader](#) (UInt32 program, UInt32 shader)

Detaches a shader object from a program object to which it is attached.

- static void **Disable** (OpenTK.Graphics.OpenGL.EnableCap cap)
- static void **DisableClientState** (OpenTK.Graphics.OpenGL.ArrayCap array)
- static void **Disable** (OpenTK.Graphics.OpenGL.IndexedEnableCap target, Int32 index)
- static void **Disable** (OpenTK.Graphics.OpenGL.IndexedEnableCap target, UInt32 index)
- static void **DisableVertexAttribArray** (Int32 index)

- static void **DisableVertexAttribArray** (UInt32 index)
- static void **DrawArrays** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 first, Int32 count)
Render primitives from array data.
- static void **DrawArraysInstanced** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 first, Int32 count, Int32 primcount)
- static void **DrawBuffer** (OpenTK.Graphics.OpenGL.DrawBufferMode mode)
Specify which color buffers are to be drawn into.
- static void **DrawBuffers** (Int32 n, OpenTK.Graphics.OpenGL.DrawBuffersEnum[] bufs)
Specifies a list of color buffers to be drawn into.
- static void **DrawBuffers** (Int32 n, ref OpenTK.Graphics.OpenGL.DrawBuffersEnum bufs)
Specifies a list of color buffers to be drawn into.
- static unsafe void **DrawBuffers** (Int32 n, OpenTK.Graphics.OpenGL.DrawBuffersEnum *bufs)
Specifies a list of color buffers to be drawn into.
- static void **DrawElements** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices)
Render primitives from array data.
- static void **DrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[] indices)
Render primitives from array data.
- static void **DrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[,] indices)
Render primitives from array data.
- static void **DrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[,] indices)
Render primitives from array data.
- static void **DrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] ref T3 indices)

Render primitives from array data.

- static void **DrawElementsBaseVertex** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 basevertex)
- static void **DrawElementsBaseVertex**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[] indices, Int32 basevertex)
- static void **DrawElementsBaseVertex**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 basevertex)
- static void **DrawElementsBaseVertex**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 basevertex)
- static void **DrawElementsBaseVertex**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T3 indices, Int32 basevertex)
- static void **DrawElementsInstanced** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount)
- static void **DrawElementsInstanced**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[] indices, Int32 primcount)
- static void **DrawElementsInstanced**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount)
- static void **DrawElementsInstanced**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount)
- static void **DrawElementsInstanced**< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T3 indices, Int32 primcount)
- static void **DrawElementsInstancedBaseVertex** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount, Int32 basevertex)

- static void **DrawElementsInstancedBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[] indices, Int32 primcount, Int32 basevertex)
- static void **DrawElementsInstancedBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[] indices, Int32 primcount, Int32 basevertex)
- static void **DrawElementsInstancedBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[,] indices, Int32 primcount, Int32 basevertex)
- static void **DrawElementsInstancedBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] ref T3 indices, Int32 primcount, Int32 basevertex)
- static void **DrawPixels** (Int32 width, Int32
height, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Write a block of pixels to the frame buffer.

- static void **DrawPixels**< **T4** > (Int32 width, Int32
height, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[]
pixels)

Write a block of pixels to the frame buffer.

- static void **DrawPixels**< **T4** > (Int32 width, Int32
height, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[,]
pixels)

Write a block of pixels to the frame buffer.

- static void **DrawPixels**< **T4** > (Int32 width, Int32
height, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[,]
pixels)

Write a block of pixels to the frame buffer.

- static void **DrawPixels**< **T4** > (Int32 width, Int32
height, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref
T4 pixels)

Write a block of pixels to the frame buffer.

- static void [DrawRangeElements](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices)
Render primitives from array data.
- static void [DrawRangeElements< T5 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[] indices)
Render primitives from array data.
- static void [DrawRangeElements< T5 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[,] indices)
Render primitives from array data.
- static void [DrawRangeElements< T5 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[,] indices)
Render primitives from array data.
- static void [DrawRangeElements< T5 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] ref T5 indices)
Render primitives from array data.
- static void [DrawRangeElements](#) (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices)
Render primitives from array data.
- static void [DrawRangeElements< T5 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[] indices)
Render primitives from array data.
- static void [DrawRangeElements< T5 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[,] indices)

Render primitives from array data.

- static void **DrawRangeElements**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[, indices])

Render primitives from array data.

- static void **DrawRangeElements**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T5 indices)

Render primitives from array data.

- static void **DrawRangeElementsBaseVertex** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[] indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[, indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[, indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T5 indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex** (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[] indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end,

Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[,], indices, Int32 basevertex)

- static void **DrawRangeElementsBaseVertex**< **T5** > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T5[,], indices, Int32 basevertex)
- static void **DrawRangeElementsBaseVertex**< **T5** > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T5 indices, Int32 basevertex)
- static void [EdgeFlag](#) (bool flag)

Flag edges as either boundary or nonboundary.

- static void [EdgeFlagPointer](#) (Int32 stride, IntPtr pointer)

Define an array of edge flags.

- static void [EdgeFlagPointer](#)< **T1** > (Int32 stride,[InAttribute, OutAttribute] T1[] pointer)

Define an array of edge flags.

- static void [EdgeFlagPointer](#)< **T1** > (Int32 stride,[InAttribute, OutAttribute] T1[,] pointer)

Define an array of edge flags.

- static void [EdgeFlagPointer](#)< **T1** > (Int32 stride,[InAttribute, OutAttribute] T1[,], pointer)

Define an array of edge flags.

- static void [EdgeFlagPointer](#)< **T1** > (Int32 stride,[InAttribute, OutAttribute] ref T1 pointer)

Define an array of edge flags.

- static unsafe void [EdgeFlag](#) (bool *flag)

Flag edges as either boundary or nonboundary.

- static void [Enable](#) (OpenTK.Graphics.OpenGL.EnableCap cap)

Enable or disable server-side [GL](#) capabilities.

- static void [EnableClientState](#) (OpenTK.Graphics.OpenGL.ArrayCap array)

Enable or disable client-side capability.

- static void [Enable](#) (OpenTK.Graphics.OpenGL.IndexedEnableCap target, Int32 index)

Enable or disable server-side [GL](#) capabilities.

- static void [Enable](#) (OpenTK.Graphics.OpenGL.IndexedEnableCap target, UInt32 index)

Enable or disable server-side [GL](#) capabilities.

- static void [EnableVertexAttribArray](#) (Int32 index)

Enable or disable a generic vertex attribute array.

- static void [EnableVertexAttribArray](#) (UInt32 index)

Enable or disable a generic vertex attribute array.

- static void **End** ()
- static void **EndConditionalRender** ()
- static void **EndList** ()
- static void **EndQuery** (OpenTK.Graphics.OpenGL.QueryTarget target)
- static void **EndTransformFeedback** ()
- static void [EvalCoord1](#) (Double u)

Evaluate enabled one- and two-dimensional maps.

- static unsafe void [EvalCoord1](#) (Double *u)

Evaluate enabled one- and two-dimensional maps.

- static void [EvalCoord1](#) (Single u)

Evaluate enabled one- and two-dimensional maps.

- static unsafe void [EvalCoord1](#) (Single *u)

Evaluate enabled one- and two-dimensional maps.

- static void [EvalCoord2](#) (Double u, Double v)

Evaluate enabled one- and two-dimensional maps.

- static void [EvalCoord2](#) (Double[] u)

Evaluate enabled one- and two-dimensional maps.

- static void [EvalCoord2](#) (ref Double u)

Evaluate enabled one- and two-dimensional maps.

- static unsafe void [EvalCoord2](#) (Double *u)

Evaluate enabled one- and two-dimensional maps.

- static void [EvalCoord2](#) (Single u, Single v)

Evaluate enabled one- and two-dimensional maps.

- static void [EvalCoord2](#) (Single[] u)
Evaluate enabled one- and two-dimensional maps.
- static void [EvalCoord2](#) (ref Single u)
Evaluate enabled one- and two-dimensional maps.
- static unsafe void [EvalCoord2](#) (Single *u)
Evaluate enabled one- and two-dimensional maps.
- static void [EvalMesh1](#) (OpenTK.Graphics.OpenGL.MeshMode1 mode, Int32 i1, Int32 i2)
Compute a one- or two-dimensional grid of points or lines.
- static void [EvalMesh2](#) (OpenTK.Graphics.OpenGL.MeshMode2 mode, Int32 i1, Int32 i2, Int32 j1, Int32 j2)
Compute a one- or two-dimensional grid of points or lines.
- static void [EvalPoint1](#) (Int32 i)
Generate and evaluate a single point in a mesh.
- static void [EvalPoint2](#) (Int32 i, Int32 j)
Generate and evaluate a single point in a mesh.
- static void [FeedbackBuffer](#) (Int32 size, OpenTK.Graphics.OpenGL.FeedbackType type,[OutAttribute] Single[] buffer)
Controls feedback mode.
- static void [FeedbackBuffer](#) (Int32 size, OpenTK.Graphics.OpenGL.FeedbackType type,[OutAttribute] out Single buffer)
Controls feedback mode.
- static unsafe void [FeedbackBuffer](#) (Int32 size, OpenTK.Graphics.OpenGL.FeedbackType type,[OutAttribute] Single *buffer)
Controls feedback mode.
- static IntPtr **FenceSync** (OpenTK.Graphics.OpenGL.ArbSync condition, Int32 flags)
- static IntPtr **FenceSync** (OpenTK.Graphics.OpenGL.ArbSync condition, UInt32 flags)
- static void [Finish](#) ()
Block until all [GL](#) execution is complete.

- static void **Flush** ()
Force execution of GL commands in finite time.
- static void **FlushMappedBufferRange** (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr length)
- static void **FogCoord** (Double coord)
Set the current fog coordinates.
- static unsafe void **FogCoord** (Double *coord)
Set the current fog coordinates.
- static void **FogCoord** (Single coord)
Set the current fog coordinates.
- static unsafe void **FogCoord** (Single *coord)
Set the current fog coordinates.
- static void **FogCoordPointer** (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride, IntPtr pointer)
Define an array of fog coordinates.
- static void **FogCoordPointer< T2 >** (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,[InAttribute, OutAttribute] T2[] pointer)
Define an array of fog coordinates.
- static void **FogCoordPointer< T2 >** (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,[InAttribute, OutAttribute] T2[,] pointer)
Define an array of fog coordinates.
- static void **FogCoordPointer< T2 >** (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,[InAttribute, OutAttribute] T2[,] pointer)
Define an array of fog coordinates.
- static void **FogCoordPointer< T2 >** (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,[InAttribute, OutAttribute] ref T2 pointer)
Define an array of fog coordinates.
- static void **Fog** (OpenTK.Graphics.OpenGL.FogParameter pname, Single param)
Specify fog parameters.

- static void **Fog** (OpenTK.Graphics.OpenGL.FogParameter pname, Single[] @params)
Specify fog parameters.
- static unsafe void **Fog** (OpenTK.Graphics.OpenGL.FogParameter pname, Single* @params)
Specify fog parameters.
- static void **Fog** (OpenTK.Graphics.OpenGL.FogParameter pname, Int32 param)
Specify fog parameters.
- static void **Fog** (OpenTK.Graphics.OpenGL.FogParameter pname, Int32[] @params)
Specify fog parameters.
- static unsafe void **Fog** (OpenTK.Graphics.OpenGL.FogParameter pname, Int32* @params)
Specify fog parameters.
- static void **FramebufferRenderbuffer** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.RenderbufferTarget renderbuffertarget, Int32 renderbuffer)
- static void **FramebufferRenderbuffer** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.RenderbufferTarget renderbuffertarget, UInt32 renderbuffer)
- static void **FramebufferTexture** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, Int32 texture, Int32 level)
- static void **FramebufferTexture** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, UInt32 texture, Int32 level)
- static void **FramebufferTexture1D** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.TextureTarget textarget, Int32 texture, Int32 level)
- static void **FramebufferTexture1D** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.TextureTarget textarget, UInt32 texture, Int32 level)
- static void **FramebufferTexture2D** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment,

- OpenTK.Graphics.OpenGL.TextureTarget textarget, Int32 texture, Int32 level)
- static void **FramebufferTexture2D** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.TextureTarget textarget, UInt32 texture, Int32 level)
 - static void **FramebufferTexture3D** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.TextureTarget textarget, Int32 texture, Int32 level, Int32 zoffset)
 - static void **FramebufferTexture3D** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.TextureTarget textarget, UInt32 texture, Int32 level, Int32 zoffset)
 - static void **FramebufferTextureFace** (OpenTK.Graphics.OpenGL.Version32 target, OpenTK.Graphics.OpenGL.Version32 attachment, Int32 texture, Int32 level, OpenTK.Graphics.OpenGL.Version32 face)
 - static void **FramebufferTextureFace** (OpenTK.Graphics.OpenGL.Version32 target, OpenTK.Graphics.OpenGL.Version32 attachment, UInt32 texture, Int32 level, OpenTK.Graphics.OpenGL.Version32 face)
 - static void **FramebufferTextureLayer** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, Int32 texture, Int32 level, Int32 layer)
 - static void **FramebufferTextureLayer** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, UInt32 texture, Int32 level, Int32 layer)
 - static void **FrontFace** (OpenTK.Graphics.OpenGL.FrontFaceDirection mode)
Define front- and back-facing polygons.
 - static void **Frustum** (Double left, Double right, Double bottom, Double top, Double zNear, Double zFar)
Multiply the current matrix by a perspective matrix.
 - static void **GenBuffers** (Int32 n,[OutAttribute] Int32[] buffers)
Generate buffer object names.
 - static void **GenBuffers** (Int32 n,[OutAttribute] out Int32 buffers)
Generate buffer object names.
 - static unsafe void **GenBuffers** (Int32 n,[OutAttribute] Int32 *buffers)
Generate buffer object names.
 - static void **GenBuffers** (Int32 n,[OutAttribute] UInt32[] buffers)
Generate buffer object names.

- static void [GenBuffers](#) (Int32 n,[OutAttribute] out UInt32 buffers)
Generate buffer object names.
- static unsafe void [GenBuffers](#) (Int32 n,[OutAttribute] UInt32 *buffers)
Generate buffer object names.
- static void **GenerateMipmap** (OpenTK.Graphics.OpenGL.GenerateMipmapTarget target)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] Int32[] framebuffers)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] out Int32 framebuffers)
- static unsafe void **GenFramebuffers** (Int32 n,[OutAttribute] Int32 *framebuffers)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] UInt32[] framebuffers)
- static void **GenFramebuffers** (Int32 n,[OutAttribute] out UInt32 framebuffers)
- static unsafe void **GenFramebuffers** (Int32 n,[OutAttribute] UInt32 *framebuffers)
- static Int32 [GenLists](#) (Int32 range)
Generate a contiguous set of empty display lists.
- static void [GenQueries](#) (Int32 n,[OutAttribute] Int32[] ids)
Generate query object names.
- static void [GenQueries](#) (Int32 n,[OutAttribute] out Int32 ids)
Generate query object names.
- static unsafe void [GenQueries](#) (Int32 n,[OutAttribute] Int32 *ids)
Generate query object names.
- static void [GenQueries](#) (Int32 n,[OutAttribute] UInt32[] ids)
Generate query object names.
- static void [GenQueries](#) (Int32 n,[OutAttribute] out UInt32 ids)
Generate query object names.
- static unsafe void [GenQueries](#) (Int32 n,[OutAttribute] UInt32 *ids)
Generate query object names.
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] Int32[] renderbuffers)
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] out Int32 renderbuffers)
- static unsafe void **GenRenderbuffers** (Int32 n,[OutAttribute] Int32 *renderbuffers)
- static void **GenRenderbuffers** (Int32 n,[OutAttribute] UInt32[] renderbuffers)

- static void **GenRenderbuffers** (Int32 n,[OutAttribute] out UInt32 renderbuffers)
- static unsafe void **GenRenderbuffers** (Int32 n,[OutAttribute] UInt32 *renderbuffers)
- static void **GenTextures** (Int32 n,[OutAttribute] Int32[] textures)

Generate texture names.
- static void **GenTextures** (Int32 n,[OutAttribute] out Int32 textures)

Generate texture names.
- static unsafe void **GenTextures** (Int32 n,[OutAttribute] Int32 *textures)

Generate texture names.
- static void **GenTextures** (Int32 n,[OutAttribute] UInt32[] textures)

Generate texture names.
- static void **GenTextures** (Int32 n,[OutAttribute] out UInt32 textures)

Generate texture names.
- static unsafe void **GenTextures** (Int32 n,[OutAttribute] UInt32 *textures)

Generate texture names.
- static void **GenVertexArrays** (Int32 n,[OutAttribute] Int32[] arrays)
- static void **GenVertexArrays** (Int32 n,[OutAttribute] out Int32 arrays)
- static unsafe void **GenVertexArrays** (Int32 n,[OutAttribute] Int32 *arrays)
- static void **GenVertexArrays** (Int32 n,[OutAttribute] UInt32[] arrays)
- static void **GenVertexArrays** (Int32 n,[OutAttribute] out UInt32 arrays)
- static unsafe void **GenVertexArrays** (Int32 n,[OutAttribute] UInt32 *arrays)
- static void **GetActiveAttrib** (Int32 program, Int32 index, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.OpenGL.ActiveAttribType type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.
- static unsafe void **GetActiveAttrib** (Int32 program, Int32 index, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.OpenGL.ActiveAttribType *type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static void [GetActiveAttrib](#) (UInt32 program, UInt32 index, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.OpenGL.ActiveAttribType type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static unsafe void [GetActiveAttrib](#) (UInt32 program, UInt32 index, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.OpenGL.ActiveAttribType *type,[OutAttribute] StringBuilder name)

Returns information about an active attribute variable for the specified program object.

- static void [GetActiveUniform](#) (Int32 program, Int32 index, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.OpenGL.ActiveUniformType type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static unsafe void [GetActiveUniform](#) (Int32 program, Int32 index, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.OpenGL.ActiveUniformType *type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static void [GetActiveUniform](#) (UInt32 program, UInt32 index, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.OpenGL.ActiveUniformType type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static unsafe void [GetActiveUniform](#) (UInt32 program, UInt32 index, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.OpenGL.ActiveUniformType *type,[OutAttribute] StringBuilder name)

Returns information about an active uniform variable for the specified program object.

- static void [GetActiveUniformBlock](#) (Int32 program, Int32 uniformBlockIndex, OpenTK.Graphics.OpenGL.ActiveUniformBlockParameter pname,[OutAttribute] Int32[]@params)

- static void **GetActiveUniformBlock** (Int32 program, Int32 uniformBlockIndex, OpenTK.Graphics.OpenGL.ActiveUniformBlockParameter pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetActiveUniformBlock** (Int32 program, Int32 uniformBlockIndex, OpenTK.Graphics.OpenGL.ActiveUniformBlockParameter pname,[OutAttribute] Int32 *@params)
- static void **GetActiveUniformBlock** (UInt32 program, UInt32 uniformBlockIndex, OpenTK.Graphics.OpenGL.ActiveUniformBlockParameter pname,[OutAttribute] Int32[] @params)
- static void **GetActiveUniformBlock** (UInt32 program, UInt32 uniformBlockIndex, OpenTK.Graphics.OpenGL.ActiveUniformBlockParameter pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetActiveUniformBlock** (UInt32 program, UInt32 uniformBlockIndex, OpenTK.Graphics.OpenGL.ActiveUniformBlockParameter pname,[OutAttribute] Int32 *@params)
- static void **GetActiveUniformBlockName** (Int32 program, Int32 uniformBlockIndex, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder uniformBlockName)
- static unsafe void **GetActiveUniformBlockName** (Int32 program, Int32 uniformBlockIndex, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder uniformBlockName)
- static void **GetActiveUniformBlockName** (UInt32 program, UInt32 uniformBlockIndex, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder uniformBlockName)
- static unsafe void **GetActiveUniformBlockName** (UInt32 program, UInt32 uniformBlockIndex, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder uniformBlockName)
- static void **GetActiveUniformName** (Int32 program, Int32 uniformIndex, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder uniformName)
- static unsafe void **GetActiveUniformName** (Int32 program, Int32 uniformIndex, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder uniformName)
- static void **GetActiveUniformName** (UInt32 program, UInt32 uniformIndex, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder uniformName)
- static unsafe void **GetActiveUniformName** (UInt32 program, UInt32 uniformIndex, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder uniformName)
- static void **GetActiveUniforms** (Int32 program, Int32 uniformCount, Int32[] uniformIndices, OpenTK.Graphics.OpenGL.ActiveUniformParameter pname,[OutAttribute] Int32[] @params)
- static void **GetActiveUniforms** (Int32 program, Int32 uniformCount, ref Int32 uniformIndices, OpenTK.Graphics.OpenGL.ActiveUniformParameter pname,[OutAttribute] out Int32 @params)

- static unsafe void **GetActiveUniforms** (Int32 program, Int32 uniformCount, Int32 *uniformIndices, OpenTK.Graphics.OpenGL.ActiveUniformParameter pname,[OutAttribute] Int32 *@params)
- static void **GetActiveUniforms** (UInt32 program, Int32 uniformCount, UInt32[] uniformIndices, OpenTK.Graphics.OpenGL.ActiveUniformParameter pname,[OutAttribute] Int32[] @params)
- static void **GetActiveUniforms** (UInt32 program, Int32 uniformCount, ref UInt32 uniformIndices, OpenTK.Graphics.OpenGL.ActiveUniformParameter pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetActiveUniforms** (UInt32 program, Int32 uniformCount, UInt32 *uniformIndices, OpenTK.Graphics.OpenGL.ActiveUniformParameter pname,[OutAttribute] Int32 *@params)
- static void **GetAttachedShaders** (Int32 program, Int32 maxCount,[OutAttribute] out Int32 count,[OutAttribute] out Int32 obj)

Returns the handles of the shader objects attached to a program object.

- static unsafe void **GetAttachedShaders** (Int32 program, Int32 maxCount,[OutAttribute] Int32 *count,[OutAttribute] Int32[] obj)

Returns the handles of the shader objects attached to a program object.

- static unsafe void **GetAttachedShaders** (Int32 program, Int32 maxCount,[OutAttribute] Int32 *count,[OutAttribute] Int32 *obj)

Returns the handles of the shader objects attached to a program object.

- static void **GetAttachedShaders** (UInt32 program, Int32 maxCount,[OutAttribute] out Int32 count,[OutAttribute] out UInt32 obj)

Returns the handles of the shader objects attached to a program object.

- static unsafe void **GetAttachedShaders** (UInt32 program, Int32 maxCount,[OutAttribute] Int32 *count,[OutAttribute] UInt32[] obj)

Returns the handles of the shader objects attached to a program object.

- static unsafe void **GetAttachedShaders** (UInt32 program, Int32 maxCount,[OutAttribute] Int32 *count,[OutAttribute] UInt32 *obj)

Returns the handles of the shader objects attached to a program object.

- static Int32 **GetAttribLocation** (Int32 program, String name)

Returns the location of an attribute variable.

- static Int32 **GetAttribLocation** (UInt32 program, String name)

Returns the location of an attribute variable.

- static void **GetBoolean** (OpenTK.Graphics.OpenGL.GetIndexedPName target, Int32 index,[OutAttribute] bool[] data)

- static void **GetBoolean** (OpenTK.Graphics.OpenGL.GetIndexedPName target, Int32 index,[OutAttribute] out bool data)
- static unsafe void **GetBoolean** (OpenTK.Graphics.OpenGL.GetIndexedPName target, Int32 index,[OutAttribute] bool *data)
- static void **GetBoolean** (OpenTK.Graphics.OpenGL.GetIndexedPName target, UInt32 index,[OutAttribute] bool[] data)
- static void **GetBoolean** (OpenTK.Graphics.OpenGL.GetIndexedPName target, UInt32 index,[OutAttribute] out bool data)
- static unsafe void **GetBoolean** (OpenTK.Graphics.OpenGL.GetIndexedPName target, UInt32 index,[OutAttribute] bool *data)
- static void **GetBoolean** (OpenTK.Graphics.OpenGL.GetPName pname,[OutAttribute] bool[] @params)
- static void **GetBoolean** (OpenTK.Graphics.OpenGL.GetPName pname,[OutAttribute] out bool @params)
- static unsafe void **GetBoolean** (OpenTK.Graphics.OpenGL.GetPName pname,[OutAttribute] bool *@params)
- static void **GetBufferParameteri64** (OpenTK.Graphics.OpenGL.Version32 target, OpenTK.Graphics.OpenGL.Version32 pname,[OutAttribute] Int64[] @params)
- static void **GetBufferParameteri64** (OpenTK.Graphics.OpenGL.Version32 target, OpenTK.Graphics.OpenGL.Version32 pname,[OutAttribute] out Int64 @params)
- static unsafe void **GetBufferParameteri64** (OpenTK.Graphics.OpenGL.Version32 target, OpenTK.Graphics.OpenGL.Version32 pname,[OutAttribute] Int64 *@params)
- static void **GetBufferParameter** (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferParameterName pname,[OutAttribute] Int32[] @params)

Return parameters of a buffer object.
- static void **GetBufferParameter** (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferParameterName pname,[OutAttribute] out Int32 @params)

Return parameters of a buffer object.
- static unsafe void **GetBufferParameter** (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferParameterName pname,[OutAttribute] Int32 *@params)

Return parameters of a buffer object.
- static void **GetBufferPointer** (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferPointer pname,[OutAttribute] IntPtr @params)

Return the pointer to a mapped buffer object's data store.

- static void [GetBufferPointer< T2 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferPointer pname,[InAttribute, OutAttribute] T2[]@params)
Return the pointer to a mapped buffer object's data store.
- static void [GetBufferPointer< T2 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferPointer pname,[InAttribute, OutAttribute] T2[,]@params)
Return the pointer to a mapped buffer object's data store.
- static void [GetBufferPointer< T2 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferPointer pname,[InAttribute, OutAttribute] T2[, ,]@params)
Return the pointer to a mapped buffer object's data store.
- static void [GetBufferPointer< T2 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferPointer pname,[InAttribute, OutAttribute] ref T2 @params)
Return the pointer to a mapped buffer object's data store.
- static void [GetBufferSubData](#) (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[OutAttribute] IntPtr data)
Returns a subset of a buffer object's data store.
- static void [GetBufferSubData< T3 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[] data)
Returns a subset of a buffer object's data store.
- static void [GetBufferSubData< T3 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[,] data)
Returns a subset of a buffer object's data store.
- static void [GetBufferSubData< T3 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] T3[, ,] data)
Returns a subset of a buffer object's data store.
- static void [GetBufferSubData< T3 >](#) (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size,[InAttribute, OutAttribute] ref T3 data)
Returns a subset of a buffer object's data store.
- static void [GetClipPlane](#) (OpenTK.Graphics.OpenGL.ClipPlaneName plane,[OutAttribute] Double[] equation)

Return the coefficients of the specified clipping plane.

- static void [GetClipPlane](#) (OpenTK.Graphics.OpenGL.ClipPlaneName plane,[OutAttribute] out Double equation)

Return the coefficients of the specified clipping plane.

- static unsafe void [GetClipPlane](#) (OpenTK.Graphics.OpenGL.ClipPlaneName plane,[OutAttribute] Double *equation)

Return the coefficients of the specified clipping plane.

- static void [GetColorTable](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr table)

Retrieve contents of a color lookup table.

- static void [GetColorTable< T3 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T3[] table)

Retrieve contents of a color lookup table.

- static void [GetColorTable< T3 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T3[,] table)

Retrieve contents of a color lookup table.

- static void [GetColorTable< T3 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T3[,] table)

Retrieve contents of a color lookup table.

- static void [GetColorTable< T3 >](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T3 table)

Retrieve contents of a color lookup table.

- static void [GetColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.GetColorTableParameterPName pname,[OutAttribute] Single[] @params)

Get color lookup table parameters.

- static void [GetColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.GetColorTableParameterPName pname,[OutAttribute] out Single @params)
Get color lookup table parameters.
- static unsafe void [GetColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.GetColorTableParameterPName pname,[OutAttribute] Single *@params)
Get color lookup table parameters.
- static void [GetColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.GetColorTableParameterPName pname,[OutAttribute] Int32[] @params)
Get color lookup table parameters.
- static void [GetColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.GetColorTableParameterPName pname,[OutAttribute] out Int32 @params)
Get color lookup table parameters.
- static unsafe void [GetColorTableParameter](#) (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.GetColorTableParameterPName pname,[OutAttribute] Int32 *@params)
Get color lookup table parameters.
- static void [GetCompressedTexImage](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,[OutAttribute] IntPtr img)
Return a compressed texture image.
- static void [GetCompressedTexImage<T2>](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,[InAttribute, OutAttribute] T2[] img)
Return a compressed texture image.
- static void [GetCompressedTexImage<T2>](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,[InAttribute, OutAttribute] T2[,] img)
Return a compressed texture image.
- static void [GetCompressedTexImage<T2>](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,[InAttribute, OutAttribute] T2[,,,] img)

Return a compressed texture image.

- static void [GetCompressedTexImage](#)< [T2](#) > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,[InAttribute, OutAttribute] ref [T2](#) img)

Return a compressed texture image.

- static void [GetConvolutionFilter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr image)

Get current 1D or 2D convolution filter kernel.

- static void [GetConvolutionFilter](#)< [T3](#) > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] [T3](#)[] image)

Get current 1D or 2D convolution filter kernel.

- static void [GetConvolutionFilter](#)< [T3](#) > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] [T3](#)[,] image)

Get current 1D or 2D convolution filter kernel.

- static void [GetConvolutionFilter](#)< [T3](#) > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] [T3](#)[,] image)

Get current 1D or 2D convolution filter kernel.

- static void [GetConvolutionFilter](#)< [T3](#) > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref [T3](#) image)

Get current 1D or 2D convolution filter kernel.

- static void [GetConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.GetConvolutionParameterPName pname,[OutAttribute] Single[] @params)

Get convolution parameters.

- static void [GetConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.GetConvolutionParameterPName pname,[OutAttribute] out Single @params)

Get convolution parameters.

- static unsafe void [GetConvolutionParameter](#)
(OpenTK.Graphics.OpenGL.ConvolutionTarget target,
OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
pname,[OutAttribute] Single *@params)

Get convolution parameters.

- static void [GetConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget
target, OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
pname,[OutAttribute] Int32[] @params)

Get convolution parameters.

- static void [GetConvolutionParameter](#) (OpenTK.Graphics.OpenGL.ConvolutionTarget
target, OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
pname,[OutAttribute] out Int32 @params)

Get convolution parameters.

- static unsafe void [GetConvolutionParameter](#)
(OpenTK.Graphics.OpenGL.ConvolutionTarget target,
OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
pname,[OutAttribute] Int32 *@params)

Get convolution parameters.

- static void **GetDouble** (OpenTK.Graphics.OpenGL.GetPName
pname,[OutAttribute] Double[] @params)
- static void **GetDouble** (OpenTK.Graphics.OpenGL.GetPName
pname,[OutAttribute] out Double @params)
- static unsafe void **GetDouble** (OpenTK.Graphics.OpenGL.GetPName
pname,[OutAttribute] Double *@params)
- static OpenTK.Graphics.OpenGL.ErrorCode [GetError](#) ()

Return error information.

- static void **GetFloat** (OpenTK.Graphics.OpenGL.GetPName
pname,[OutAttribute] Single[] @params)
- static void **GetFloat** (OpenTK.Graphics.OpenGL.GetPName
pname,[OutAttribute] out Single @params)
- static unsafe void **GetFloat** (OpenTK.Graphics.OpenGL.GetPName
pname,[OutAttribute] Single *@params)
- static Int32 **GetFragDataLocation** (Int32 program, String name)
- static Int32 **GetFragDataLocation** (UInt32 program, String name)
- static void **GetFramebufferAttachmentParameter**
(OpenTK.Graphics.OpenGL.FramebufferTarget target,
OpenTK.Graphics.OpenGL.FramebufferAttachment attachment,

OpenTK.Graphics.OpenGL.FramebufferParameterName pname,[OutAttribute] Int32[]@params)

- static void **GetFramebufferAttachmentParameter** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.FramebufferParameterName pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetFramebufferAttachmentParameter** (OpenTK.Graphics.OpenGL.FramebufferTarget target, OpenTK.Graphics.OpenGL.FramebufferAttachment attachment, OpenTK.Graphics.OpenGL.FramebufferParameterName pname,[OutAttribute] Int32 *@params)
- static void **GetHistogram** (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr values)
Get histogram table.
- static void **GetHistogram< T4 >** (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[] values)
Get histogram table.
- static void **GetHistogram< T4 >** (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[,] values)
Get histogram table.
- static void **GetHistogram< T4 >** (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[,.] values)
Get histogram table.
- static void **GetHistogram< T4 >** (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T4 values)
Get histogram table.
- static void **GetHistogramParameter** (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname,[OutAttribute] Single[]@params)

Get histogram parameters.

- static void **GetHistogramParameter** (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname,[OutAttribute] out Single @params)

Get histogram parameters.

- static unsafe void **GetHistogramParameter** (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname,[OutAttribute] Single *@params)

Get histogram parameters.

- static void **GetHistogramParameter** (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname,[OutAttribute] Int32[] @params)

Get histogram parameters.

- static void **GetHistogramParameter** (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname,[OutAttribute] out Int32 @params)

Get histogram parameters.

- static unsafe void **GetHistogramParameter** (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname,[OutAttribute] Int32 *@params)

Get histogram parameters.

- static void **GetInteger64** (OpenTK.Graphics.OpenGL.Version32 target, Int32 index,[OutAttribute] Int64[] data)
- static void **GetInteger64** (OpenTK.Graphics.OpenGL.Version32 target, Int32 index,[OutAttribute] out Int64 data)
- static unsafe void **GetInteger64** (OpenTK.Graphics.OpenGL.Version32 target, Int32 index,[OutAttribute] Int64 *data)
- static void **GetInteger64** (OpenTK.Graphics.OpenGL.Version32 target, UInt32 index,[OutAttribute] Int64[] data)
- static void **GetInteger64** (OpenTK.Graphics.OpenGL.Version32 target, UInt32 index,[OutAttribute] out Int64 data)
- static unsafe void **GetInteger64** (OpenTK.Graphics.OpenGL.Version32 target, UInt32 index,[OutAttribute] Int64 *data)
- static void **GetInteger64** (OpenTK.Graphics.OpenGL.ArbSync pname,[OutAttribute] Int64[] @params)
- static void **GetInteger64** (OpenTK.Graphics.OpenGL.ArbSync pname,[OutAttribute] out Int64 @params)

- static unsafe void **GetInteger64** (OpenTK.Graphics.OpenGL.ArbSync pname,[OutAttribute] Int64 *@params)
- static void **GetInteger** (OpenTK.Graphics.OpenGL.GetIndexedPName target, Int32 index,[OutAttribute] Int32[] data)
- static void **GetInteger** (OpenTK.Graphics.OpenGL.GetIndexedPName target, Int32 index,[OutAttribute] out Int32 data)
- static unsafe void **GetInteger** (OpenTK.Graphics.OpenGL.GetIndexedPName target, Int32 index,[OutAttribute] Int32 *data)
- static void **GetInteger** (OpenTK.Graphics.OpenGL.GetIndexedPName target, UInt32 index,[OutAttribute] Int32[] data)
- static void **GetInteger** (OpenTK.Graphics.OpenGL.GetIndexedPName target, UInt32 index,[OutAttribute] out Int32 data)
- static unsafe void **GetInteger** (OpenTK.Graphics.OpenGL.GetIndexedPName target, UInt32 index,[OutAttribute] Int32 *data)
- static void **GetInteger** (OpenTK.Graphics.OpenGL.GetPName pname,[OutAttribute] Int32[] @params)
- static void **GetInteger** (OpenTK.Graphics.OpenGL.GetPName pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetInteger** (OpenTK.Graphics.OpenGL.GetPName pname,[OutAttribute] Int32 *@params)
- static void **GetLight** (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname,[OutAttribute] Single[] @params)

Return light source parameter values.

- static void **GetLight** (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname,[OutAttribute] out Single @params)

Return light source parameter values.

- static unsafe void **GetLight** (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname,[OutAttribute] Single *@params)

Return light source parameter values.

- static void **GetLight** (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname,[OutAttribute] Int32[] @params)

Return light source parameter values.

- static void **GetLight** (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname,[OutAttribute] out Int32 @params)

Return light source parameter values.

- static unsafe void [GetLight](#) (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname,[OutAttribute] Int32 *@params)

Return light source parameter values.

- static void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] Double[] v)

Return evaluator parameters.

- static void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] out Double v)

Return evaluator parameters.

- static unsafe void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] Double *v)

Return evaluator parameters.

- static void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] Single[] v)

Return evaluator parameters.

- static void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] out Single v)

Return evaluator parameters.

- static unsafe void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] Single *v)

Return evaluator parameters.

- static void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] Int32[] v)

Return evaluator parameters.

- static void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] out Int32 v)

Return evaluator parameters.

- static unsafe void [GetMap](#) (OpenTK.Graphics.OpenGL.MapTarget target, OpenTK.Graphics.OpenGL.GetMapQuery query,[OutAttribute] Int32 *v)
Return evaluator parameters.
- static void [GetMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname,[OutAttribute] Single[] @params)
Return material parameters.
- static void [GetMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname,[OutAttribute] out Single[] @params)
Return material parameters.
- static unsafe void [GetMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname,[OutAttribute] Single[] *@params)
Return material parameters.
- static void [GetMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname,[OutAttribute] Int32[] @params)
Return material parameters.
- static void [GetMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname,[OutAttribute] out Int32[] @params)
Return material parameters.
- static unsafe void [GetMaterial](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname,[OutAttribute] Int32[] *@params)
Return material parameters.
- static void [GetMinmax](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr values)
Get minimum and maximum pixel values.
- static void [GetMinmax< T4 >](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[] values)
Get minimum and maximum pixel values.

- static void [GetMinmax< T4 >](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[, values])

Get minimum and maximum pixel values.

- static void [GetMinmax< T4 >](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[, values])

Get minimum and maximum pixel values.

- static void [GetMinmax< T4 >](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T4 values)

Get minimum and maximum pixel values.

- static void [GetMinmaxParameter](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname,[OutAttribute] Single[]@params)

Get minmax parameters.

- static void [GetMinmaxParameter](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname,[OutAttribute] out Single @params)

Get minmax parameters.

- static unsafe void [GetMinmaxParameter](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname,[OutAttribute] Single *@params)

Get minmax parameters.

- static void [GetMinmaxParameter](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname,[OutAttribute] Int32[]@params)

Get minmax parameters.

- static void [GetMinmaxParameter](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname,[OutAttribute] out Int32 @params)

Get minmax parameters.

- static unsafe void [GetMinmaxParameter](#) (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname, [OutAttribute] Int32 *@params)
Get minmax parameters.
- static void **GetMultisample** (OpenTK.Graphics.OpenGL.GetMultisamplePName pname, Int32 index, [OutAttribute] Single[] val)
- static void **GetMultisample** (OpenTK.Graphics.OpenGL.GetMultisamplePName pname, Int32 index, [OutAttribute] out Single val)
- static unsafe void **GetMultisample** (OpenTK.Graphics.OpenGL.GetMultisamplePName pname, Int32 index, [OutAttribute] Single *val)
- static void **GetMultisample** (OpenTK.Graphics.OpenGL.GetMultisamplePName pname, UInt32 index, [OutAttribute] Single[] val)
- static void **GetMultisample** (OpenTK.Graphics.OpenGL.GetMultisamplePName pname, UInt32 index, [OutAttribute] out Single val)
- static unsafe void **GetMultisample** (OpenTK.Graphics.OpenGL.GetMultisamplePName pname, UInt32 index, [OutAttribute] Single *val)
- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Single[] values)
Return the specified pixel map.
- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] out Single values)
Return the specified pixel map.
- static unsafe void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Single *values)
Return the specified pixel map.
- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Int32[] values)
Return the specified pixel map.
- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] out Int32 values)
Return the specified pixel map.
- static unsafe void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Int32 *values)
Return the specified pixel map.
- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] UInt32[] values)

Return the specified pixel map.

- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] out UInt32 values)

Return the specified pixel map.

- static unsafe void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] UInt32 *values)

Return the specified pixel map.

- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] Int16[] values)

Return the specified pixel map.

- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] out Int16 values)

Return the specified pixel map.

- static unsafe void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] Int16 *values)

Return the specified pixel map.

- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] UInt16[] values)

Return the specified pixel map.

- static void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] out UInt16 values)

Return the specified pixel map.

- static unsafe void [GetPixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map,[OutAttribute] UInt16 *values)

Return the specified pixel map.

- static void [GetPointer](#) (OpenTK.Graphics.OpenGL.GetPointervPName pname,[OutAttribute] IntPtr @params)

Return the address of the specified pointer.

- static void [GetPointer< T1 >](#) (OpenTK.Graphics.OpenGL.GetPointervPName pname,[InAttribute, OutAttribute] T1[] @params)

Return the address of the specified pointer.

- static void [GetPointer< T1 >](#) (OpenTK.Graphics.OpenGL.GetPointervPName pname,[InAttribute, OutAttribute] T1[,] @params)

Return the address of the specified pointer.

- static void [GetPointer< T1 >](#) (OpenTK.Graphics.OpenGL.GetPointervPName pname,[InAttribute, OutAttribute] T1[,,@params)

Return the address of the specified pointer.

- static void [GetPointer< T1 >](#) (OpenTK.Graphics.OpenGL.GetPointervPName pname,[InAttribute, OutAttribute] ref T1 @params)

Return the address of the specified pointer.

- static void [GetPolygonStipple](#) ([OutAttribute] Byte[] mask)

Return the polygon stipple pattern.

- static void [GetPolygonStipple](#) ([OutAttribute] out Byte mask)

Return the polygon stipple pattern.

- static unsafe void [GetPolygonStipple](#) ([OutAttribute] Byte *mask)

Return the polygon stipple pattern.

- static void [GetProgramInfoLog](#) (Int32 program, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infoLog)

Returns the information log for a program object.

- static unsafe void [GetProgramInfoLog](#) (Int32 program, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infoLog)

Returns the information log for a program object.

- static void [GetProgramInfoLog](#) (UInt32 program, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infoLog)

Returns the information log for a program object.

- static unsafe void [GetProgramInfoLog](#) (UInt32 program, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infoLog)

Returns the information log for a program object.

- static void [GetProgram](#) (Int32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname,[OutAttribute] Int32[] @params)

Returns a parameter from a program object.

- static void [GetProgram](#) (Int32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname,[OutAttribute] out Int32 @params)

Returns a parameter from a program object.

- static unsafe void [GetProgram](#) (Int32 program,
 OpenTK.Graphics.OpenGL.ProgramParameter pname,[OutAttribute] Int32
 *@params)
Returns a parameter from a program object.
- static void [GetProgram](#) (UInt32 program,
 OpenTK.Graphics.OpenGL.ProgramParameter pname,[OutAttribute]
 Int32[]@params)
Returns a parameter from a program object.
- static void [GetProgram](#) (UInt32 program,
 OpenTK.Graphics.OpenGL.ProgramParameter pname,[OutAttribute] out
 Int32 @params)
Returns a parameter from a program object.
- static unsafe void [GetProgram](#) (UInt32 program,
 OpenTK.Graphics.OpenGL.ProgramParameter pname,[OutAttribute] Int32
 *@params)
Returns a parameter from a program object.
- static void [GetQuery](#) (OpenTK.Graphics.OpenGL.QueryTarget tar-
 get, OpenTK.Graphics.OpenGL.GetQueryParam pname,[OutAttribute]
 Int32[]@params)
Return parameters of a query object target.
- static void [GetQuery](#) (OpenTK.Graphics.OpenGL.QueryTarget target,
 OpenTK.Graphics.OpenGL.GetQueryParam pname,[OutAttribute] out Int32
 @params)
Return parameters of a query object target.
- static unsafe void [GetQuery](#) (OpenTK.Graphics.OpenGL.QueryTarget tar-
 get, OpenTK.Graphics.OpenGL.GetQueryParam pname,[OutAttribute] Int32
 *@params)
Return parameters of a query object target.
- static void [GetQueryObject](#) (Int32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam
 pname,[OutAttribute] Int32[]@params)
Return parameters of a query object.
- static void [GetQueryObject](#) (Int32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam
 pname,[OutAttribute] out Int32 @params)
Return parameters of a query object.

- static unsafe void [GetQueryObject](#) (Int32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] Int32 *@params)
Return parameters of a query object.
- static void [GetQueryObject](#) (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] Int32[] @params)
Return parameters of a query object.
- static void [GetQueryObject](#) (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] out Int32 @params)
Return parameters of a query object.
- static unsafe void [GetQueryObject](#) (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] Int32 *@params)
Return parameters of a query object.
- static void [GetQueryObject](#) (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] UInt32[] @params)
Return parameters of a query object.
- static void [GetQueryObject](#) (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] out UInt32 @params)
Return parameters of a query object.
- static unsafe void [GetQueryObject](#) (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,[OutAttribute] UInt32 *@params)
Return parameters of a query object.
- static void **GetRenderbufferParameter** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, OpenTK.Graphics.OpenGL.RenderbufferParameterName pname,[OutAttribute] Int32[] @params)
- static void **GetRenderbufferParameter** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, OpenTK.Graphics.OpenGL.RenderbufferParameterName pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetRenderbufferParameter** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, OpenTK.Graphics.OpenGL.RenderbufferParameterName pname,[OutAttribute] Int32 *@params)
- static void [GetSeparableFilter](#) (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[OutAttribute] IntPtr column,[OutAttribute] IntPtr span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[OutAttribute] IntPtr column,[InAttribute, OutAttribute] T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[OutAttribute] IntPtr column,[InAttribute, OutAttribute] T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[OutAttribute] IntPtr column,[InAttribute, OutAttribute] T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[OutAttribute] IntPtr column,[InAttribute, OutAttribute] ref T5 span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute] T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute] T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format,

OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr
 row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute]
 T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T4, T5 >
 (OpenTK.Graphics.OpenGL.SeparableTarget target,
 OpenTK.Graphics.OpenGL.PixelFormat format,
 OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr
 row,[InAttribute, OutAttribute] ref T4 column,[InAttribute, OutAttribute]
 T5[,] span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T3, T4, T5 >
 (OpenTK.Graphics.OpenGL.SeparableTarget target,
 OpenTK.Graphics.OpenGL.PixelFormat format,
 OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T3[]
 row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute] T5[,]
 span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T3, T4, T5 >
 (OpenTK.Graphics.OpenGL.SeparableTarget target,
 OpenTK.Graphics.OpenGL.PixelFormat format,
 OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T3[,]
 row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute] T5[,]
 span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T3, T4, T5 >
 (OpenTK.Graphics.OpenGL.SeparableTarget target,
 OpenTK.Graphics.OpenGL.PixelFormat format,
 OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T3[,]
 row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute] T5[,]
 span)

Get separable convolution filter kernel images.

- static void [GetSeparableFilter](#)< T3, T4, T5 >
 (OpenTK.Graphics.OpenGL.SeparableTarget target,
 OpenTK.Graphics.OpenGL.PixelFormat format,
 OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref
 T3 row,[InAttribute, OutAttribute] T4[,] column,[InAttribute, OutAttribute]
 T5[,] span)

Get separable convolution filter kernel images.

- static void [GetShaderInfoLog](#) (Int32 shader, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infoLog)
Returns the information log for a shader object.
- static unsafe void [GetShaderInfoLog](#) (Int32 shader, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infoLog)
Returns the information log for a shader object.
- static void [GetShaderInfoLog](#) (UInt32 shader, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder infoLog)
Returns the information log for a shader object.
- static unsafe void [GetShaderInfoLog](#) (UInt32 shader, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder infoLog)
Returns the information log for a shader object.
- static void [GetShader](#) (Int32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname,[OutAttribute] Int32[] @params)
Returns a parameter from a shader object.
- static void [GetShader](#) (Int32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname,[OutAttribute] out Int32 @params)
Returns a parameter from a shader object.
- static unsafe void [GetShader](#) (Int32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname,[OutAttribute] Int32 *@params)
Returns a parameter from a shader object.
- static void [GetShader](#) (UInt32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname,[OutAttribute] Int32[] @params)
Returns a parameter from a shader object.
- static void [GetShader](#) (UInt32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname,[OutAttribute] out Int32 @params)
Returns a parameter from a shader object.
- static unsafe void [GetShader](#) (UInt32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname,[OutAttribute] Int32 *@params)
Returns a parameter from a shader object.

- static void [GetShaderSource](#) (Int32 shader, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static unsafe void [GetShaderSource](#) (Int32 shader, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static void [GetShaderSource](#) (UInt32 shader, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static unsafe void [GetShaderSource](#) (UInt32 shader, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] StringBuilder source)
Returns the source code string from a shader object.
- static System.String [GetString](#) (OpenTK.Graphics.OpenGL.StringName name)
Return a string describing the current [GL](#) connection.
- static System.String [GetString](#) (OpenTK.Graphics.OpenGL.StringName name, Int32 index)
Return a string describing the current [GL](#) connection.
- static System.String [GetString](#) (OpenTK.Graphics.OpenGL.StringName name, UInt32 index)
Return a string describing the current [GL](#) connection.
- static void **GetSync** (IntPtr sync, OpenTK.Graphics.OpenGL.ArbSync pname, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 values)
- static unsafe void **GetSync** (IntPtr sync, OpenTK.Graphics.OpenGL.ArbSync pname, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32[] values)
- static unsafe void **GetSync** (IntPtr sync, OpenTK.Graphics.OpenGL.ArbSync pname, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *values)
- static void [GetTexEnv](#) (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname,[OutAttribute] Single[] @params)
Return texture environment parameters.
- static void [GetTexEnv](#) (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname,[OutAttribute] out Single @params)

Return texture environment parameters.

- static unsafe void [GetTexEnv](#) (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname,[OutAttribute] Single *@params)

Return texture environment parameters.

- static void [GetTexEnv](#) (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname,[OutAttribute] Int32[] @params)

Return texture environment parameters.

- static void [GetTexEnv](#) (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname,[OutAttribute] out Int32 @params)

Return texture environment parameters.

- static unsafe void [GetTexEnv](#) (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname,[OutAttribute] Int32 *@params)

Return texture environment parameters.

- static void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] Double[] @params)

Return texture coordinate generation parameters.

- static void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] out Double @params)

Return texture coordinate generation parameters.

- static unsafe void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] Double *@params)

Return texture coordinate generation parameters.

- static void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] Single[] @params)

Return texture coordinate generation parameters.

- static void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] out Single @params)

Return texture coordinate generation parameters.

- static unsafe void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] Single *@params)

Return texture coordinate generation parameters.

- static void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] Int32[] @params)

Return texture coordinate generation parameters.

- static void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] out Int32 @params)

Return texture coordinate generation parameters.

- static unsafe void [GetTexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname,[OutAttribute] Int32 *@params)

Return texture coordinate generation parameters.

- static void [GetTexImage](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[OutAttribute] IntPtr pixels)

Return a texture image.

- static void [GetTexImage< T4 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[] pixels)

Return a texture image.

- static void [GetTexImage< T4 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[,] pixels)

Return a texture image.

- static void [GetTexImage< T4 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T4[,]) pixels)

Return a texture image.

- static void [GetTexImage< T4 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T4 pixels)
Return a texture image.
- static void [GetTexLevelParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Single[]@params)
Return texture parameter values for a specific level of detail.
- static void [GetTexLevelParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out Single @params)
Return texture parameter values for a specific level of detail.
- static unsafe void [GetTexLevelParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Single *@params)
Return texture parameter values for a specific level of detail.
- static void [GetTexLevelParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Int32[]@params)
Return texture parameter values for a specific level of detail.
- static void [GetTexLevelParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out Int32 @params)
Return texture parameter values for a specific level of detail.
- static unsafe void [GetTexLevelParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Int32 *@params)
Return texture parameter values for a specific level of detail.
- static void [GetTexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Single[]@params)
Return texture parameter values.

- static void [GetTexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out Single @params)

Return texture parameter values.

- static unsafe void [GetTexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Single *@params)

Return texture parameter values.

- static void **GetTexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Int32[] @params)
- static void **GetTexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetTexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Int32 *@params)
- static void **GetTexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] UInt32[] @params)
- static void **GetTexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out UInt32 @params)
- static unsafe void **GetTexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] UInt32 *@params)
- static void [GetTexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out Int32[] @params)

Return texture parameter values.

- static void [GetTexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] out Int32 @params)

Return texture parameter values.

- static unsafe void [GetTexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.GetTextureParameter pname,[OutAttribute] Int32 *@params)

Return texture parameter values.

- static void **GetTransformFeedbackVarying** (Int32 program, Int32 index, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.OpenGL.ActiveAttribType type,[OutAttribute] StringBuilder name)
- static unsafe void **GetTransformFeedbackVarying** (Int32 program, Int32 index, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.OpenGL.ActiveAttribType *type,[OutAttribute] StringBuilder name)
- static void **GetTransformFeedbackVarying** (UInt32 program, UInt32 index, Int32 bufSize,[OutAttribute] out Int32 length,[OutAttribute] out Int32 size,[OutAttribute] out OpenTK.Graphics.OpenGL.ActiveAttribType type,[OutAttribute] StringBuilder name)
- static unsafe void **GetTransformFeedbackVarying** (UInt32 program, UInt32 index, Int32 bufSize,[OutAttribute] Int32 *length,[OutAttribute] Int32 *size,[OutAttribute] OpenTK.Graphics.OpenGL.ActiveAttribType *type,[OutAttribute] StringBuilder name)
- static Int32 **GetUniformBlockIndex** (Int32 program, String uniformBlock-Name)
- static Int32 **GetUniformBlockIndex** (UInt32 program, String uniformBlock-Name)
- static void **GetUniform** (Int32 program, Int32 location,[OutAttribute] Single[] @params)
Returns the value of a uniform variable.
- static void **GetUniform** (Int32 program, Int32 location,[OutAttribute] out Single @params)
Returns the value of a uniform variable.
- static unsafe void **GetUniform** (Int32 program, Int32 location,[OutAttribute] Single *@params)
Returns the value of a uniform variable.
- static void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] Single[] @params)
Returns the value of a uniform variable.
- static void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] out Single @params)
Returns the value of a uniform variable.
- static unsafe void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] Single *@params)
Returns the value of a uniform variable.

- static void **GetUniformIndices** (Int32 program, Int32 uniformCount, String[] uniformNames,[OutAttribute] Int32[] uniformIndices)
- static void **GetUniformIndices** (Int32 program, Int32 uniformCount, String[] uniformNames,[OutAttribute] out Int32 uniformIndices)
- static unsafe void **GetUniformIndices** (Int32 program, Int32 uniformCount, String[] uniformNames,[OutAttribute] Int32 *uniformIndices)
- static void **GetUniformIndices** (UInt32 program, Int32 uniformCount, String[] uniformNames,[OutAttribute] UInt32[] uniformIndices)
- static void **GetUniformIndices** (UInt32 program, Int32 uniformCount, String[] uniformNames,[OutAttribute] out UInt32 uniformIndices)
- static unsafe void **GetUniformIndices** (UInt32 program, Int32 uniformCount, String[] uniformNames,[OutAttribute] UInt32 *uniformIndices)
- static void **GetUniform** (Int32 program, Int32 location,[OutAttribute] Int32[] @params)
Returns the value of a uniform variable.
- static void **GetUniform** (Int32 program, Int32 location,[OutAttribute] out Int32 @params)
Returns the value of a uniform variable.
- static unsafe void **GetUniform** (Int32 program, Int32 location,[OutAttribute] Int32 * @params)
Returns the value of a uniform variable.
- static void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] Int32[] @params)
Returns the value of a uniform variable.
- static void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] out Int32 @params)
Returns the value of a uniform variable.
- static unsafe void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] Int32 * @params)
Returns the value of a uniform variable.
- static Int32 **GetUniformLocation** (Int32 program, String name)
Returns the location of a uniform variable.
- static Int32 **GetUniformLocation** (UInt32 program, String name)
Returns the location of a uniform variable.
- static void **GetUniform** (UInt32 program, Int32 location,[OutAttribute] UInt32[] @params)

Returns the value of a uniform variable.

- static void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] out UInt32 @params)

Returns the value of a uniform variable.

- static unsafe void [GetUniform](#) (UInt32 program, Int32 location,[OutAttribute] UInt32 *@params)

Returns the value of a uniform variable.

- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Double[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] out Double @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Double *@params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Double[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] out Double @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Double *@params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Single[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] out Single @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Single *@params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Single[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter out Single @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter Single *@params)

Return a generic vertex attribute parameter.

- static void **GetVertexAttribI** (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter out Int32 @params)

- static unsafe void **GetVertexAttribI** (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter Int32 *@params)

- static void **GetVertexAttribI** (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter out Int32 @params)

- static unsafe void **GetVertexAttribI** (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter Int32 *@params)

- static void **GetVertexAttribI** (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter out UInt32 @params)

- static unsafe void **GetVertexAttribI** (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter UInt32 *@params)

- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Int32[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] out Int32 @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Int32 *@params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Int32[] @params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] out Int32 @params)

Return a generic vertex attribute parameter.

- static unsafe void [GetVertexAttrib](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname,[OutAttribute] Int32 *@params)

Return a generic vertex attribute parameter.

- static void [GetVertexAttribPointer](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[OutAttribute] IntPtr pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer< T2 >](#) (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[,,,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] ref T2 pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#) (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[OutAttribute] IntPtr pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] T2[, ,] pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [GetVertexAttribPointer](#)< T2 > (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,[InAttribute, OutAttribute] ref T2 pointer)

Return the address of the specified generic vertex attribute pointer.

- static void [Hint](#) (OpenTK.Graphics.OpenGL.HintTarget target, OpenTK.Graphics.OpenGL.HintMode mode)

Specify implementation-specific hints.

- static void [Histogram](#) (OpenTK.Graphics.OpenGL.HistogramTarget target, Int32 width, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, bool sink)

Define histogram table.

- static void [Index](#) (Double c)

Set the current color index.

- static unsafe void [Index](#) (Double *c)

Set the current color index.

- static void [Index](#) (Single c)
Set the current color index.
- static unsafe void [Index](#) (Single *c)
Set the current color index.
- static void [Index](#) (Int32 c)
Set the current color index.
- static unsafe void [Index](#) (Int32 *c)
Set the current color index.
- static void [IndexMask](#) (Int32 mask)
Control the writing of individual bits in the color index buffers.
- static void [IndexMask](#) (UInt32 mask)
Control the writing of individual bits in the color index buffers.
- static void [IndexPointer](#) (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, IntPtr pointer)
Define an array of color indexes.
- static void [IndexPointer< T2 >](#) (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer)
Define an array of color indexes.
- static void [IndexPointer< T2 >](#) (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[,] pointer)
Define an array of color indexes.
- static void [IndexPointer< T2 >](#) (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[, ,] pointer)
Define an array of color indexes.
- static void [IndexPointer< T2 >](#) (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer)
Define an array of color indexes.
- static void [Index](#) (Int16 c)
Set the current color index.

- static unsafe void [Index](#) (Int16 *c)
Set the current color index.
- static void [Index](#) (Byte c)
Set the current color index.
- static unsafe void [Index](#) (Byte *c)
Set the current color index.
- static void [InitNames](#) ()
Initialize the name stack.
- static void [InterleavedArrays](#) (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, IntPtr pointer)
Simultaneously specify and enable several interleaved arrays.
- static void [InterleavedArrays< T2 >](#) (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, [InAttribute, OutAttribute] T2[] pointer)
Simultaneously specify and enable several interleaved arrays.
- static void [InterleavedArrays< T2 >](#) (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, [InAttribute, OutAttribute] T2[,] pointer)
Simultaneously specify and enable several interleaved arrays.
- static void [InterleavedArrays< T2 >](#) (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, [InAttribute, OutAttribute] T2[, ,] pointer)
Simultaneously specify and enable several interleaved arrays.
- static void [InterleavedArrays< T2 >](#) (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer)
Simultaneously specify and enable several interleaved arrays.
- static bool [IsBuffer](#) (Int32 buffer)
Determine if a name corresponds to a buffer object.
- static bool [IsBuffer](#) (UInt32 buffer)
Determine if a name corresponds to a buffer object.
- static bool [IsEnabled](#) (OpenTK.Graphics.OpenGL.EnableCap cap)
Test whether a capability is enabled.
- static bool [IsEnabled](#) (OpenTK.Graphics.OpenGL.IndexedEnableCap target, Int32 index)

Test whether a capability is enabled.

- static bool **IsEnabled** (OpenTK.Graphics.OpenGL.IndexedEnableCap target, UInt32 index)

Test whether a capability is enabled.

- static bool **IsFramebuffer** (Int32 framebuffer)
- static bool **IsFramebuffer** (UInt32 framebuffer)
- static bool **IsList** (Int32 list)

Determine if a name corresponds to a display list.

- static bool **IsList** (UInt32 list)

Determine if a name corresponds to a display list.

- static bool **IsProgram** (Int32 program)

Determines if a name corresponds to a program object.

- static bool **IsProgram** (UInt32 program)

Determines if a name corresponds to a program object.

- static bool **IsQuery** (Int32 id)

Determine if a name corresponds to a query object.

- static bool **IsQuery** (UInt32 id)

Determine if a name corresponds to a query object.

- static bool **IsRenderbuffer** (Int32 renderbuffer)
- static bool **IsRenderbuffer** (UInt32 renderbuffer)
- static bool **IsShader** (Int32 shader)

Determines if a name corresponds to a shader object.

- static bool **IsShader** (UInt32 shader)

Determines if a name corresponds to a shader object.

- static bool **IsSync** (IntPtr sync)
- static bool **IsTexture** (Int32 texture)

Determine if a name corresponds to a texture.

- static bool **IsTexture** (UInt32 texture)

Determine if a name corresponds to a texture.

- static bool **IsVertexArray** (Int32 array)
- static bool **IsVertexArray** (UInt32 array)

- static void [Light](#) (OpenTK.Graphics.OpenGL.LightName pname, Single param) light,
OpenTK.Graphics.OpenGL.LightParameter pname, Single param)
Set light source parameters.
- static void [Light](#) (OpenTK.Graphics.OpenGL.LightName pname, Single[] @params) light,
OpenTK.Graphics.OpenGL.LightParameter pname, Single[] @params)
Set light source parameters.
- static unsafe void [Light](#) (OpenTK.Graphics.OpenGL.LightName pname, Single * @params) light,
OpenTK.Graphics.OpenGL.LightParameter pname, Single * @params)
Set light source parameters.
- static void [Light](#) (OpenTK.Graphics.OpenGL.LightName pname, Int32 param) light,
OpenTK.Graphics.OpenGL.LightParameter pname, Int32 param)
Set light source parameters.
- static void [Light](#) (OpenTK.Graphics.OpenGL.LightName pname, Int32[] @params) light,
OpenTK.Graphics.OpenGL.LightParameter pname, Int32[] @params)
Set light source parameters.
- static unsafe void [Light](#) (OpenTK.Graphics.OpenGL.LightName pname, Int32 * @params) light,
OpenTK.Graphics.OpenGL.LightParameter pname, Int32 * @params)
Set light source parameters.
- static void [LightModel](#) (OpenTK.Graphics.OpenGL.LightModelParameter pname, Single param)
Set the lighting model parameters.
- static void [LightModel](#) (OpenTK.Graphics.OpenGL.LightModelParameter pname, Single[] @params)
Set the lighting model parameters.
- static unsafe void [LightModel](#) (OpenTK.Graphics.OpenGL.LightModelParameter pname, Single * @params)
Set the lighting model parameters.
- static void [LightModel](#) (OpenTK.Graphics.OpenGL.LightModelParameter pname, Int32 param)
Set the lighting model parameters.
- static void [LightModel](#) (OpenTK.Graphics.OpenGL.LightModelParameter pname, Int32[] @params)
Set the lighting model parameters.

- static unsafe void [LightModel](#) (OpenTK.Graphics.OpenGL.LightModelParameter pname, Int32 *@params)
Set the lighting model parameters.
- static void [LineStipple](#) (Int32 factor, Int16 pattern)
Specify the line stipple pattern.
- static void [LineStipple](#) (Int32 factor, UInt16 pattern)
Specify the line stipple pattern.
- static void [LineWidth](#) (Single width)
Specify the width of rasterized lines.
- static void [LinkProgram](#) (Int32 program)
Links a program object.
- static void [LinkProgram](#) (UInt32 program)
Links a program object.
- static void [ListBase](#) (Int32 @base)
Set the display-list base for glCallLists.
- static void [ListBase](#) (UInt32 @base)
Set the display-list base for glCallLists.
- static void [LoadIdentity](#) ()
Replace the current matrix with the identity matrix.
- static void [LoadMatrix](#) (Double[] m)
Replace the current matrix with the specified matrix.
- static void [LoadMatrix](#) (ref Double m)
Replace the current matrix with the specified matrix.
- static unsafe void [LoadMatrix](#) (Double *m)
Replace the current matrix with the specified matrix.
- static void [LoadMatrix](#) (Single[] m)
Replace the current matrix with the specified matrix.
- static void [LoadMatrix](#) (ref Single m)

Replace the current matrix with the specified matrix.

- static unsafe void [LoadMatrix](#) (Single *m)
Replace the current matrix with the specified matrix.
- static void [LoadName](#) (Int32 name)
Load a name onto the name stack.
- static void [LoadName](#) (UInt32 name)
Load a name onto the name stack.
- static void [LoadTransposeMatrix](#) (Double[] m)
Replace the current matrix with the specified row-major ordered matrix.
- static void [LoadTransposeMatrix](#) (ref Double m)
Replace the current matrix with the specified row-major ordered matrix.
- static unsafe void [LoadTransposeMatrix](#) (Double *m)
Replace the current matrix with the specified row-major ordered matrix.
- static void [LoadTransposeMatrix](#) (Single[] m)
Replace the current matrix with the specified row-major ordered matrix.
- static void [LoadTransposeMatrix](#) (ref Single m)
Replace the current matrix with the specified row-major ordered matrix.
- static unsafe void [LoadTransposeMatrix](#) (Single *m)
Replace the current matrix with the specified row-major ordered matrix.
- static void [LogicOp](#) (OpenTK.Graphics.OpenGL.LogicOp opcode)
Specify a logical pixel operation for color index rendering.
- static void [Map1](#) (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 stride, Int32 order, Double[] points)
Define a one-dimensional evaluator.
- static void [Map1](#) (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 stride, Int32 order, ref Double points)
Define a one-dimensional evaluator.
- static unsafe void [Map1](#) (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 stride, Int32 order, Double *points)
Define a one-dimensional evaluator.

- static void [Map1](#) (OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single u2, Int32 stride, Int32 order, Single[] points)

Define a one-dimensional evaluator.

- static void [Map1](#) (OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single u2, Int32 stride, Int32 order, ref Single points)

Define a one-dimensional evaluator.

- static unsafe void [Map1](#) (OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single u2, Int32 stride, Int32 order, Single *points)

Define a one-dimensional evaluator.

- static void [Map2](#) (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 ustride, Int32 uorder, Double v1, Double v2, Int32 vstride, Int32 vorder, Double[] points)

Define a two-dimensional evaluator.

- static void [Map2](#) (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 ustride, Int32 uorder, Double v1, Double v2, Int32 vstride, Int32 vorder, ref Double points)

Define a two-dimensional evaluator.

- static unsafe void [Map2](#) (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 ustride, Int32 uorder, Double v1, Double v2, Int32 vstride, Int32 vorder, Double *points)

Define a two-dimensional evaluator.

- static void [Map2](#) (OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single u2, Int32 ustride, Int32 uorder, Single v1, Single v2, Int32 vstride, Int32 vorder, Single[] points)

Define a two-dimensional evaluator.

- static void [Map2](#) (OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single u2, Int32 ustride, Int32 uorder, Single v1, Single v2, Int32 vstride, Int32 vorder, ref Single points)

Define a two-dimensional evaluator.

- static unsafe void [Map2](#) (OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single u2, Int32 ustride, Int32 uorder, Single v1, Single v2, Int32 vstride, Int32 vorder, Single *points)

Define a two-dimensional evaluator.

- static unsafe System.IntPtr **MapBuffer** (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferAccess access)
Map a buffer object's data store.
- static unsafe System.IntPtr **MapBufferRange** (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr length, OpenTK.Graphics.OpenGL.BufferAccessMask access)
- static void **MapGrid1** (Int32 un, Double u1, Double u2)
Define a one- or two-dimensional mesh.
- static void **MapGrid1** (Int32 un, Single u1, Single u2)
Define a one- or two-dimensional mesh.
- static void **MapGrid2** (Int32 un, Double u1, Double u2, Int32 vn, Double v1, Double v2)
Define a one- or two-dimensional mesh.
- static void **MapGrid2** (Int32 un, Single u1, Single u2, Int32 vn, Single v1, Single v2)
Define a one- or two-dimensional mesh.
- static void **Material** (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname, Single param)
Specify material parameters for the lighting model.
- static void **Material** (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname, Single[] @params)
Specify material parameters for the lighting model.
- static unsafe void **Material** (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname, Single * @params)
Specify material parameters for the lighting model.
- static void **Material** (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname, Int32 param)
Specify material parameters for the lighting model.
- static void **Material** (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname, Int32[] @params)
Specify material parameters for the lighting model.
- static unsafe void **Material** (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.MaterialParameter pname, Int32 * @params)

Specify material parameters for the lighting model.

- static void **MatrixMode** (OpenTK.Graphics.OpenGL.MatrixMode mode)
Specify which matrix is the current matrix.
- static void **Minmax** (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, bool sink)
Define minmax table.
- static void **MinSampleShading** (Single value)
- static void **MultiDrawArrays** (OpenTK.Graphics.OpenGL.BeginMode mode, [OutAttribute] Int32[] first, [OutAttribute] Int32[] count, Int32 primcount)
Render multiple sets of primitives from array data.
- static void **MultiDrawArrays** (OpenTK.Graphics.OpenGL.BeginMode mode, [OutAttribute] out Int32 first, [OutAttribute] out Int32 count, Int32 primcount)
Render multiple sets of primitives from array data.
- static unsafe void **MultiDrawArrays** (OpenTK.Graphics.OpenGL.BeginMode mode, [OutAttribute] Int32 *first, [OutAttribute] Int32 *count, Int32 primcount)
Render multiple sets of primitives from array data.
- static void **MultiDrawElements** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void **MultiDrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void **MultiDrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[,] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void **MultiDrawElements< T3 >** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[,.] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.

- static void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T3 indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void [MultiDrawElements](#) (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,.] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T3 indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static unsafe void [MultiDrawElements](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static unsafe void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[] indices, Int32 primcount)
Render multiple sets of primitives by specifying indices of array data elements.
- static unsafe void [MultiDrawElements< T3 >](#) (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount)

Render multiple sets of primitives by specifying indices of array data elements.

- static unsafe void **MultiDrawElements**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[,] indices, Int32 primcount)

Render multiple sets of primitives by specifying indices of array data elements.

- static unsafe void **MultiDrawElements**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] ref T3 indices, Int32 primcount)

Render multiple sets of primitives by specifying indices of array data elements.

- static void **MultiDrawElementsBaseVertex**
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count,
OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32
primcount, Int32[] basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[] indices, Int32 primcount, Int32[] basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[,] indices, Int32 primcount, Int32[] basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[,] indices, Int32 primcount, Int32[] basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] ref T3 indices, Int32 primcount, Int32[] basevertex)
- static void **MultiDrawElementsBaseVertex**
(OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count,
OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32
primcount, ref Int32 basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count,
OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute,
OutAttribute] T3[] indices, Int32 primcount, ref Int32 basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** >
(OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count,

- OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount, ref Int32 basevertex)
- static void **MultiDrawElementsBaseVertex**< **T3** > (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount, ref Int32 basevertex)
 - static void **MultiDrawElementsBaseVertex**< **T3** > (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T3 indices, Int32 primcount, ref Int32 basevertex)
 - static unsafe void **MultiDrawElementsBaseVertex** (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount, Int32 *basevertex)
 - static unsafe void **MultiDrawElementsBaseVertex**< **T3** > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[] indices, Int32 primcount, Int32 *basevertex)
 - static unsafe void **MultiDrawElementsBaseVertex**< **T3** > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount, Int32 *basevertex)
 - static unsafe void **MultiDrawElementsBaseVertex**< **T3** > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] T3[,] indices, Int32 primcount, Int32 *basevertex)
 - static unsafe void **MultiDrawElementsBaseVertex**< **T3** > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 *count, OpenTK.Graphics.OpenGL.DrawElementsType type,[InAttribute, OutAttribute] ref T3 indices, Int32 primcount, Int32 *basevertex)
 - static void **MultiTexCoord1** (OpenTK.Graphics.OpenGL.TextureUnit target, Double s)
Set the current texture coordinates.
 - static unsafe void **MultiTexCoord1** (OpenTK.Graphics.OpenGL.TextureUnit target, Double *v)
Set the current texture coordinates.
 - static void **MultiTexCoord1** (OpenTK.Graphics.OpenGL.TextureUnit target, Single s)
Set the current texture coordinates.
 - static unsafe void **MultiTexCoord1** (OpenTK.Graphics.OpenGL.TextureUnit target, Single *v)

Set the current texture coordinates.

- static void [MultiTexCoord1](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord1](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 *v)

Set the current texture coordinates.

- static void [MultiTexCoord1](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord1](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 *v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double s, Double t)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double[] v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Double v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double *v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single s, Single t)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single[] v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Single v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single *v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s, Int32 t)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32[] v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int32 v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 *v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s, Int16 t)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16[] v)

Set the current texture coordinates.

- static void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int16 v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord2](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 *v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double s, Double t, Double r)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double[] v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Double v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double *v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single s, Single t, Single r)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single[] v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Single v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single *v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s, Int32 t, Int32 r)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32[] v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int32 v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 *v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s, Int16 t, Int16 r)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16[] v)

Set the current texture coordinates.

- static void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int16 v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord3](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 *v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double s, Double t, Double r, Double q)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double[] v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Double v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Double *v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single s, Single t, Single r, Single q)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single[] v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Single v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Single *v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s, Int32 t, Int32 r, Int32 q)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32[] v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int32 v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 *v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s, Int16 t, Int16 r, Int16 q)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16[] v)

Set the current texture coordinates.

- static void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int16 v)

Set the current texture coordinates.

- static unsafe void [MultiTexCoord4](#) (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 *v)

Set the current texture coordinates.

- static void [MultMatrix](#) (Double[] m)

Multiply the current matrix with the specified matrix.

- static void [MultMatrix](#) (ref Double m)

Multiply the current matrix with the specified matrix.

- static unsafe void [MultMatrix](#) (Double *m)

Multiply the current matrix with the specified matrix.

- static void [MultMatrix](#) (Single[] m)
Multiply the current matrix with the specified matrix.
- static void [MultMatrix](#) (ref Single m)
Multiply the current matrix with the specified matrix.
- static unsafe void [MultMatrix](#) (Single *m)
Multiply the current matrix with the specified matrix.
- static void [MultTransposeMatrix](#) (Double[] m)
Multiply the current matrix with the specified row-major ordered matrix.
- static void [MultTransposeMatrix](#) (ref Double m)
Multiply the current matrix with the specified row-major ordered matrix.
- static unsafe void [MultTransposeMatrix](#) (Double *m)
Multiply the current matrix with the specified row-major ordered matrix.
- static void [MultTransposeMatrix](#) (Single[] m)
Multiply the current matrix with the specified row-major ordered matrix.
- static void [MultTransposeMatrix](#) (ref Single m)
Multiply the current matrix with the specified row-major ordered matrix.
- static unsafe void [MultTransposeMatrix](#) (Single *m)
Multiply the current matrix with the specified row-major ordered matrix.
- static void [NewList](#) (Int32 list, OpenTK.Graphics.OpenGL.ListMode mode)
Create or replace a display list.
- static void [NewList](#) (UInt32 list, OpenTK.Graphics.OpenGL.ListMode mode)
Create or replace a display list.
- static void [Normal3](#) (Byte nx, Byte ny, Byte nz)
Set the current normal vector.
- static void [Normal3](#) (SByte nx, SByte ny, SByte nz)
Set the current normal vector.
- static void [Normal3](#) (Byte[] v)
Set the current normal vector.

- static void [Normal3](#) (ref Byte v)
Set the current normal vector.
- static unsafe void [Normal3](#) (Byte *v)
Set the current normal vector.
- static void [Normal3](#) (SByte[] v)
Set the current normal vector.
- static void [Normal3](#) (ref SByte v)
Set the current normal vector.
- static unsafe void [Normal3](#) (SByte *v)
Set the current normal vector.
- static void [Normal3](#) (Double nx, Double ny, Double nz)
Set the current normal vector.
- static void [Normal3](#) (Double[] v)
Set the current normal vector.
- static void [Normal3](#) (ref Double v)
Set the current normal vector.
- static unsafe void [Normal3](#) (Double *v)
Set the current normal vector.
- static void [Normal3](#) (Single nx, Single ny, Single nz)
Set the current normal vector.
- static void [Normal3](#) (Single[] v)
Set the current normal vector.
- static void [Normal3](#) (ref Single v)
Set the current normal vector.
- static unsafe void [Normal3](#) (Single *v)
Set the current normal vector.
- static void [Normal3](#) (Int32 nx, Int32 ny, Int32 nz)
Set the current normal vector.

- static void [Normal3](#) (Int32[] v)
Set the current normal vector.
- static void [Normal3](#) (ref Int32 v)
Set the current normal vector.
- static unsafe void [Normal3](#) (Int32 *v)
Set the current normal vector.
- static void [Normal3](#) (Int16 nx, Int16 ny, Int16 nz)
Set the current normal vector.
- static void [Normal3](#) (Int16[] v)
Set the current normal vector.
- static void [Normal3](#) (ref Int16 v)
Set the current normal vector.
- static unsafe void [Normal3](#) (Int16 *v)
Set the current normal vector.
- static void [NormalPointer](#) (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, IntPtr pointer)
Define an array of normals.
- static void [NormalPointer< T2 >](#) (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer)
Define an array of normals.
- static void [NormalPointer< T2 >](#) (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[,] pointer)
Define an array of normals.
- static void [NormalPointer< T2 >](#) (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[, ,] pointer)
Define an array of normals.
- static void [NormalPointer< T2 >](#) (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer)
Define an array of normals.
- static void [Ortho](#) (Double left, Double right, Double bottom, Double top, Double zNear, Double zFar)

Multiply the current matrix with an orthographic matrix.

- static void [PassThrough](#) (Single token)

Place a marker in the feedback buffer.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, Single[] values)

Set up pixel transfer maps.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, ref Single values)

Set up pixel transfer maps.

- static unsafe void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Single *values)

Set up pixel transfer maps.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, Int32[] values)

Set up pixel transfer maps.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, ref Int32 values)

Set up pixel transfer maps.

- static unsafe void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Int32 *values)

Set up pixel transfer maps.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, UInt32[] values)

Set up pixel transfer maps.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, ref UInt32 values)

Set up pixel transfer maps.

- static unsafe void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, UInt32 *values)

Set up pixel transfer maps.

- static void [PixelMap](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 map-size, Int16[] values)

Set up pixel transfer maps.

- static void [PixelFormat](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref Int16 values)

Set up pixel transfer maps.

- static unsafe void [PixelFormat](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Int16 *values)

Set up pixel transfer maps.

- static void [PixelFormat](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, UInt16[] values)

Set up pixel transfer maps.

- static void [PixelFormat](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref UInt16 values)

Set up pixel transfer maps.

- static unsafe void [PixelFormat](#) (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, UInt16 *values)

Set up pixel transfer maps.

- static void [PixelStore](#) (OpenTK.Graphics.OpenGL.PixelStoreParameter pname, Single param)

Set pixel storage modes.

- static void [PixelStore](#) (OpenTK.Graphics.OpenGL.PixelStoreParameter pname, Int32 param)

Set pixel storage modes.

- static void [PixelTransfer](#) (OpenTK.Graphics.OpenGL.PixelTransferParameter pname, Single param)

Set pixel transfer modes.

- static void [PixelTransfer](#) (OpenTK.Graphics.OpenGL.PixelTransferParameter pname, Int32 param)

Set pixel transfer modes.

- static void [PixelZoom](#) (Single xfactor, Single yfactor)

Specify the pixel zoom factors.

- static void [PointParameter](#) (OpenTK.Graphics.OpenGL.PointParameterName pname, Single param)

Specify point parameters.

- static void [PointSize](#) (OpenTK.Graphics.OpenGL.PointParameterName pname, Single[] @params)
Specify point parameters.
- static unsafe void [PointSize](#) (OpenTK.Graphics.OpenGL.PointParameterName pname, Single * @params)
Specify point parameters.
- static void [PointParameterName](#) (OpenTK.Graphics.OpenGL.PointParameterName pname, Int32 param)
Specify point parameters.
- static void [PointParameterName](#) (OpenTK.Graphics.OpenGL.PointParameterName pname, Int32[] @params)
Specify point parameters.
- static unsafe void [PointParameterName](#) (OpenTK.Graphics.OpenGL.PointParameterName pname, Int32 * @params)
Specify point parameters.
- static void [PointSize](#) (Single size)
Specify the diameter of rasterized points.
- static void [PolygonMode](#) (OpenTK.Graphics.OpenGL.MaterialFace face, OpenTK.Graphics.OpenGL.PolygonMode mode)
Select a polygon rasterization mode.
- static void [PolygonOffset](#) (Single factor, Single units)
Set the scale and units used to calculate depth values.
- static void [PolygonStipple](#) (Byte[] mask)
Set the polygon stippling pattern.
- static void [PolygonStipple](#) (ref Byte mask)
Set the polygon stippling pattern.
- static unsafe void [PolygonStipple](#) (Byte *mask)
Set the polygon stippling pattern.
- static void **PopAttrib** ()
- static void **PopClientAttrib** ()

- static void **PopMatrix** ()
- static void **PopName** ()
- static void **PrimitiveRestartIndex** (Int32 index)
- static void **PrimitiveRestartIndex** (UInt32 index)
- static void **PrioritizeTextures** (Int32 n, Int32[] textures, Single[] priorities)
Set texture residence priority.
- static void **PrioritizeTextures** (Int32 n, ref Int32 textures, ref Single priorities)
Set texture residence priority.
- static unsafe void **PrioritizeTextures** (Int32 n, Int32 *textures, Single *priorities)
Set texture residence priority.
- static void **PrioritizeTextures** (Int32 n, UInt32[] textures, Single[] priorities)
Set texture residence priority.
- static void **PrioritizeTextures** (Int32 n, ref UInt32 textures, ref Single priorities)
Set texture residence priority.
- static unsafe void **PrioritizeTextures** (Int32 n, UInt32 *textures, Single *priorities)
Set texture residence priority.
- static void **ProgramParameter** (Int32 program, OpenTK.Graphics.OpenGL.Version32 pname, Int32 value)
- static void **ProgramParameter** (UInt32 program, OpenTK.Graphics.OpenGL.Version32 pname, Int32 value)
- static void **ProvokingVertex** (OpenTK.Graphics.OpenGL.ProvokingVertexMode mode)
- static void **PushAttrib** (OpenTK.Graphics.OpenGL.AttribMask mask)
Push and pop the server attribute stack.
- static void **PushClientAttrib** (OpenTK.Graphics.OpenGL.ClientAttribMask mask)
Push and pop the client attribute stack.
- static void **PushMatrix** ()
Push and pop the current matrix stack.
- static void **PushName** (Int32 name)
Push and pop the name stack.

- static void [PushName](#) (UInt32 name)
Push and pop the name stack.
- static void [RasterPos2](#) (Double x, Double y)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (Double[] v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (ref Double v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos2](#) (Double *v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (Single x, Single y)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (Single[] v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (ref Single v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos2](#) (Single *v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (Int32 x, Int32 y)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (Int32[] v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (ref Int32 v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos2](#) (Int32 *v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (Int16 x, Int16 y)
Specify the raster position for pixel operations.

- static void [RasterPos2](#) (Int16[] v)
Specify the raster position for pixel operations.
- static void [RasterPos2](#) (ref Int16 v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos2](#) (Int16 *v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Double x, Double y, Double z)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Double[] v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (ref Double v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos3](#) (Double *v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Single x, Single y, Single z)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Single[] v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (ref Single v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos3](#) (Single *v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Int32 x, Int32 y, Int32 z)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Int32[] v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (ref Int32 v)
Specify the raster position for pixel operations.

- static unsafe void [RasterPos3](#) (Int32 *v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Int16 x, Int16 y, Int16 z)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (Int16[] v)
Specify the raster position for pixel operations.
- static void [RasterPos3](#) (ref Int16 v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos3](#) (Int16 *v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Double x, Double y, Double z, Double w)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Double[] v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (ref Double v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos4](#) (Double *v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Single x, Single y, Single z, Single w)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Single[] v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (ref Single v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos4](#) (Single *v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Int32 x, Int32 y, Int32 z, Int32 w)
Specify the raster position for pixel operations.

- static void [RasterPos4](#) (Int32[] v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (ref Int32 v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos4](#) (Int32 *v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Int16 x, Int16 y, Int16 z, Int16 w)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (Int16[] v)
Specify the raster position for pixel operations.
- static void [RasterPos4](#) (ref Int16 v)
Specify the raster position for pixel operations.
- static unsafe void [RasterPos4](#) (Int16 *v)
Specify the raster position for pixel operations.
- static void [ReadBuffer](#) (OpenTK.Graphics.OpenGL.ReadBufferMode mode)
Select a color buffer source for pixels.
- static void [ReadPixels](#) (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr pixels)
Read a block of pixels from the frame buffer.
- static void [ReadPixels< T6 >](#) (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[] pixels)
Read a block of pixels from the frame buffer.
- static void [ReadPixels< T6 >](#) (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[] pixels)
Read a block of pixels from the frame buffer.

- static void [ReadPixels](#)< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[,] pixels)

Read a block of pixels from the frame buffer.

- static void [ReadPixels](#)< T6 > (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T6 pixels)

Read a block of pixels from the frame buffer.

- static void [Rect](#) (Double x1, Double y1, Double x2, Double y2)

Draw a rectangle.

- static void [Rect](#) (Double[] v1, Double[] v2)

Draw a rectangle.

- static void [Rect](#) (ref Double v1, ref Double v2)

Draw a rectangle.

- static unsafe void [Rect](#) (Double *v1, Double *v2)

Draw a rectangle.

- static void [Rect](#) (Single x1, Single y1, Single x2, Single y2)

Draw a rectangle.

- static void [Rect](#) (Single[] v1, Single[] v2)

Draw a rectangle.

- static void [Rect](#) (ref Single v1, ref Single v2)

Draw a rectangle.

- static unsafe void [Rect](#) (Single *v1, Single *v2)

Draw a rectangle.

- static void [Rect](#) (Int32 x1, Int32 y1, Int32 x2, Int32 y2)

Draw a rectangle.

- static void [Rect](#) (Int32[] v1, Int32[] v2)

Draw a rectangle.

- static void [Rect](#) (ref Int32 v1, ref Int32 v2)

Draw a rectangle.

- static unsafe void **Rect** (Int32 *v1, Int32 *v2)

Draw a rectangle.

- static void **Rects** (Int16 x1, Int16 y1, Int16 x2, Int16 y2)
- static void **Rect** (Int16[] v1, Int16[] v2)

Draw a rectangle.

- static void **Rect** (ref Int16 v1, ref Int16 v2)

Draw a rectangle.

- static unsafe void **Rect** (Int16 *v1, Int16 *v2)

Draw a rectangle.

- static void **RenderbufferStorage** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, OpenTK.Graphics.OpenGL.RenderbufferStorage internalformat, Int32 width, Int32 height)

- static void **RenderbufferStorageMultisample** (OpenTK.Graphics.OpenGL.RenderbufferTarget target, Int32 samples, OpenTK.Graphics.OpenGL.RenderbufferStorage internalformat, Int32 width, Int32 height)

- static Int32 **RenderMode** (OpenTK.Graphics.OpenGL.RenderingMode mode)

Set rasterization mode.

- static void **ResetHistogram** (OpenTK.Graphics.OpenGL.HistogramTarget target)

Reset histogram table entries to zero.

- static void **ResetMinmax** (OpenTK.Graphics.OpenGL.MinmaxTarget target)

Reset minmax table entries to initial values.

- static void **Rotate** (Double angle, Double x, Double y, Double z)

Multiply the current matrix by a rotation matrix.

- static void **Rotate** (Single angle, Single x, Single y, Single z)

Multiply the current matrix by a rotation matrix.

- static void **SampleCoverage** (Single value, bool invert)

Specify multisample coverage parameters.

- static void **SampleMask** (Int32 index, Int32 mask)
- static void **SampleMask** (UInt32 index, UInt32 mask)

- static void [Scale](#) (Double x, Double y, Double z)
Multiply the current matrix by a general scaling matrix.
- static void [Scale](#) (Single x, Single y, Single z)
Multiply the current matrix by a general scaling matrix.
- static void [Scissor](#) (Int32 x, Int32 y, Int32 width, Int32 height)
Define the scissor box.
- static void [SecondaryColor3](#) (SByte red, SByte green, SByte blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (SByte[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref SByte v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (SByte *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (Double red, Double green, Double blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (Double[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref Double v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (Double *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (Single red, Single green, Single blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (Single[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref Single v)
Set the current secondary color.

- static unsafe void [SecondaryColor3](#) (Single *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (Int32 red, Int32 green, Int32 blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (Int32[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref Int32 v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (Int32 *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (Int16 red, Int16 green, Int16 blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (Int16[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref Int16 v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (Int16 *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (Byte red, Byte green, Byte blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (Byte[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref Byte v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (Byte *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (UInt32 red, UInt32 green, UInt32 blue)
Set the current secondary color.

- static void [SecondaryColor3](#) (UInt32[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref UInt32 v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (UInt32 *v)
Set the current secondary color.
- static void [SecondaryColor3](#) (UInt16 red, UInt16 green, UInt16 blue)
Set the current secondary color.
- static void [SecondaryColor3](#) (UInt16[] v)
Set the current secondary color.
- static void [SecondaryColor3](#) (ref UInt16 v)
Set the current secondary color.
- static unsafe void [SecondaryColor3](#) (UInt16 *v)
Set the current secondary color.
- static void [SecondaryColorPointer](#) (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, IntPtr pointer)
Define an array of secondary colors.
- static void [SecondaryColorPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer)
Define an array of secondary colors.
- static void [SecondaryColorPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[,] pointer)
Define an array of secondary colors.
- static void [SecondaryColorPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[,] pointer)
Define an array of secondary colors.
- static void [SecondaryColorPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer)

Define an array of secondary colors.

- static void [SelectBuffer](#) (Int32 size,[OutAttribute] Int32[] buffer)
Establish a buffer for selection mode values.
- static void [SelectBuffer](#) (Int32 size,[OutAttribute] out Int32 buffer)
Establish a buffer for selection mode values.
- static unsafe void [SelectBuffer](#) (Int32 size,[OutAttribute] Int32 *buffer)
Establish a buffer for selection mode values.
- static void [SelectBuffer](#) (Int32 size,[OutAttribute] UInt32[] buffer)
Establish a buffer for selection mode values.
- static void [SelectBuffer](#) (Int32 size,[OutAttribute] out UInt32 buffer)
Establish a buffer for selection mode values.
- static unsafe void [SelectBuffer](#) (Int32 size,[OutAttribute] UInt32 *buffer)
Establish a buffer for selection mode values.
- static void [SeparableFilter2D](#) (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row, IntPtr column)
Define a separable two-dimensional convolution filter.
- static void [SeparableFilter2D<T7>](#) (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,[InAttribute, OutAttribute] T7[] column)
Define a separable two-dimensional convolution filter.
- static void [SeparableFilter2D<T7>](#) (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,[InAttribute, OutAttribute] T7[,] column)
Define a separable two-dimensional convolution filter.
- static void [SeparableFilter2D<T7>](#) (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,[InAttribute, OutAttribute] T7[,„] column)

Define a separable two-dimensional convolution filter.

- static void [SeparableFilter2D](#)< T7 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,[InAttribute, OutAttribute] ref T7 column)

Define a separable two-dimensional convolution filter.

- static void [SeparableFilter2D](#)< T6, T7 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[,] row,[InAttribute, OutAttribute] T7[,] column)

Define a separable two-dimensional convolution filter.

- static void [SeparableFilter2D](#)< T6, T7 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[,] row,[InAttribute, OutAttribute] T7[,] column)

Define a separable two-dimensional convolution filter.

- static void [SeparableFilter2D](#)< T6, T7 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T6[,] row,[InAttribute, OutAttribute] T7[,] column)

Define a separable two-dimensional convolution filter.

- static void [SeparableFilter2D](#)< T6, T7 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T6 row,[InAttribute, OutAttribute] T7[,] column)

Define a separable two-dimensional convolution filter.

- static void [ShadeModel](#) (OpenTK.Graphics.OpenGL.ShadingModel mode)

Select flat or smooth shading.

- static void [ShaderSource](#) (Int32 shader, Int32 count, String[] @string, ref Int32 length)
Replaces the source code in a shader object.
- static unsafe void [ShaderSource](#) (Int32 shader, Int32 count, String[] @string, Int32 *length)
Replaces the source code in a shader object.
- static void [ShaderSource](#) (UInt32 shader, Int32 count, String[] @string, ref Int32 length)
Replaces the source code in a shader object.
- static unsafe void [ShaderSource](#) (UInt32 shader, Int32 count, String[] @string, Int32 *length)
Replaces the source code in a shader object.
- static void [StencilFunc](#) (OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ref, Int32 mask)
Set front and back function and reference value for stencil testing.
- static void [StencilFunc](#) (OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ref, UInt32 mask)
Set front and back function and reference value for stencil testing.
- static void [StencilFuncSeparate](#) (OpenTK.Graphics.OpenGL.StencilFace face, OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ref, Int32 mask)
Set front and/or back function and reference value for stencil testing.
- static void [StencilFuncSeparate](#) (OpenTK.Graphics.OpenGL.StencilFace face, OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ref, UInt32 mask)
Set front and/or back function and reference value for stencil testing.
- static void [StencilMask](#) (Int32 mask)
Control the front and back writing of individual bits in the stencil planes.
- static void [StencilMask](#) (UInt32 mask)
Control the front and back writing of individual bits in the stencil planes.
- static void [StencilMaskSeparate](#) (OpenTK.Graphics.OpenGL.StencilFace face, Int32 mask)
Control the front and/or back writing of individual bits in the stencil planes.

- static void [StencilMaskSeparate](#) (OpenTK.Graphics.OpenGL.StencilFace face, UInt32 mask)
Control the front and/or back writing of individual bits in the stencil planes.
- static void [StencilOp](#) (OpenTK.Graphics.OpenGL.StencilOp fail, OpenTK.Graphics.OpenGL.StencilOp zfail, OpenTK.Graphics.OpenGL.StencilOp zpass)
Set front and back stencil test actions.
- static void [StencilOpSeparate](#) (OpenTK.Graphics.OpenGL.StencilFace face, OpenTK.Graphics.OpenGL.StencilOp sfail, OpenTK.Graphics.OpenGL.StencilOp dpfail, OpenTK.Graphics.OpenGL.StencilOp dppass)
Set front and/or back stencil test actions.
- static void **TexBuffer** (OpenTK.Graphics.OpenGL.TextureBufferTarget target, OpenTK.Graphics.OpenGL.SizedInternalFormat internalformat, Int32 buffer)
- static void **TexBuffer** (OpenTK.Graphics.OpenGL.TextureBufferTarget target, OpenTK.Graphics.OpenGL.SizedInternalFormat internalformat, UInt32 buffer)
- static void [TexCoord1](#) (Double s)
Set the current texture coordinates.
- static unsafe void [TexCoord1](#) (Double *v)
Set the current texture coordinates.
- static void [TexCoord1](#) (Single s)
Set the current texture coordinates.
- static unsafe void [TexCoord1](#) (Single *v)
Set the current texture coordinates.
- static void [TexCoord1](#) (Int32 s)
Set the current texture coordinates.
- static unsafe void [TexCoord1](#) (Int32 *v)
Set the current texture coordinates.
- static void [TexCoord1](#) (Int16 s)
Set the current texture coordinates.
- static unsafe void [TexCoord1](#) (Int16 *v)
Set the current texture coordinates.

- static void [TexCoord2](#) (Double s, Double t)
Set the current texture coordinates.
- static void [TexCoord2](#) (Double[] v)
Set the current texture coordinates.
- static void [TexCoord2](#) (ref Double v)
Set the current texture coordinates.
- static unsafe void [TexCoord2](#) (Double *v)
Set the current texture coordinates.
- static void [TexCoord2](#) (Single s, Single t)
Set the current texture coordinates.
- static void [TexCoord2](#) (Single[] v)
Set the current texture coordinates.
- static void [TexCoord2](#) (ref Single v)
Set the current texture coordinates.
- static unsafe void [TexCoord2](#) (Single *v)
Set the current texture coordinates.
- static void [TexCoord2](#) (Int32 s, Int32 t)
Set the current texture coordinates.
- static void [TexCoord2](#) (Int32[] v)
Set the current texture coordinates.
- static void [TexCoord2](#) (ref Int32 v)
Set the current texture coordinates.
- static unsafe void [TexCoord2](#) (Int32 *v)
Set the current texture coordinates.
- static void [TexCoord2](#) (Int16 s, Int16 t)
Set the current texture coordinates.
- static void [TexCoord2](#) (Int16[] v)
Set the current texture coordinates.

- static void [TexCoord2](#) (ref Int16 v)
Set the current texture coordinates.
- static unsafe void [TexCoord2](#) (Int16 *v)
Set the current texture coordinates.
- static void [TexCoord3](#) (Double s, Double t, Double r)
Set the current texture coordinates.
- static void [TexCoord3](#) (Double[] v)
Set the current texture coordinates.
- static void [TexCoord3](#) (ref Double v)
Set the current texture coordinates.
- static unsafe void [TexCoord3](#) (Double *v)
Set the current texture coordinates.
- static void [TexCoord3](#) (Single s, Single t, Single r)
Set the current texture coordinates.
- static void [TexCoord3](#) (Single[] v)
Set the current texture coordinates.
- static void [TexCoord3](#) (ref Single v)
Set the current texture coordinates.
- static unsafe void [TexCoord3](#) (Single *v)
Set the current texture coordinates.
- static void [TexCoord3](#) (Int32 s, Int32 t, Int32 r)
Set the current texture coordinates.
- static void [TexCoord3](#) (Int32[] v)
Set the current texture coordinates.
- static void [TexCoord3](#) (ref Int32 v)
Set the current texture coordinates.
- static unsafe void [TexCoord3](#) (Int32 *v)
Set the current texture coordinates.

- static void [TexCoord3](#) (Int16 s, Int16 t, Int16 r)
Set the current texture coordinates.
- static void [TexCoord3](#) (Int16[] v)
Set the current texture coordinates.
- static void [TexCoord3](#) (ref Int16 v)
Set the current texture coordinates.
- static unsafe void [TexCoord3](#) (Int16 *v)
Set the current texture coordinates.
- static void [TexCoord4](#) (Double s, Double t, Double r, Double q)
Set the current texture coordinates.
- static void [TexCoord4](#) (Double[] v)
Set the current texture coordinates.
- static void [TexCoord4](#) (ref Double v)
Set the current texture coordinates.
- static unsafe void [TexCoord4](#) (Double *v)
Set the current texture coordinates.
- static void [TexCoord4](#) (Single s, Single t, Single r, Single q)
Set the current texture coordinates.
- static void [TexCoord4](#) (Single[] v)
Set the current texture coordinates.
- static void [TexCoord4](#) (ref Single v)
Set the current texture coordinates.
- static unsafe void [TexCoord4](#) (Single *v)
Set the current texture coordinates.
- static void [TexCoord4](#) (Int32 s, Int32 t, Int32 r, Int32 q)
Set the current texture coordinates.
- static void [TexCoord4](#) (Int32[] v)
Set the current texture coordinates.

- static void [TexCoord4](#) (ref Int32 v)
Set the current texture coordinates.
- static unsafe void [TexCoord4](#) (Int32 *v)
Set the current texture coordinates.
- static void [TexCoord4](#) (Int16 s, Int16 t, Int16 r, Int16 q)
Set the current texture coordinates.
- static void [TexCoord4](#) (Int16[] v)
Set the current texture coordinates.
- static void [TexCoord4](#) (ref Int16 v)
Set the current texture coordinates.
- static unsafe void [TexCoord4](#) (Int16 *v)
Set the current texture coordinates.
- static void [TexCoordPointer](#) (Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType type, Int32 stride, IntPtr pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[,] pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType type, Int32 stride, [InAttribute, OutAttribute] T3[,], pointer)
Define an array of texture coordinates.
- static void [TexCoordPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer)
Define an array of texture coordinates.

- static void **TexEnv** (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Single param)
Set texture environment parameters.
- static void **TexEnv** (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Single[] @params)
Set texture environment parameters.
- static unsafe void **TexEnv** (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Single * @params)
Set texture environment parameters.
- static void **TexEnv** (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Int32 param)
Set texture environment parameters.
- static void **TexEnv** (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Int32[] @params)
Set texture environment parameters.
- static unsafe void **TexEnv** (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Int32 * @params)
Set texture environment parameters.
- static void **TexGend** (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Double param)
- static void **TexGen** (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Double[] @params)
Control the generation of texture coordinates.
- static void **TexGen** (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, ref Double @params)
Control the generation of texture coordinates.
- static unsafe void **TexGen** (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Double * @params)
Control the generation of texture coordinates.
- static void **TexGen** (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Single param)

Control the generation of texture coordinates.

- static void [TexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Single[] @params)

Control the generation of texture coordinates.

- static unsafe void [TexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Single* @params)

Control the generation of texture coordinates.

- static void [TexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Int32 param)

Control the generation of texture coordinates.

- static void [TexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Int32[] @params)

Control the generation of texture coordinates.

- static unsafe void [TexGen](#) (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Int32* @params)

Control the generation of texture coordinates.

- static void [TexImage1D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Specify a one-dimensional texture image.

- static void [TexImage1D< T7 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T7[] pixels)

Specify a one-dimensional texture image.

- static void [TexImage1D< T7 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T7[,] pixels)

Specify a one-dimensional texture image.

- static void [TexImage1D< T7 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T7[,], pixels)

Specify a one-dimensional texture image.

- static void [TexImage1D< T7 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T7 pixels)

Specify a one-dimensional texture image.

- static void [TexImage2D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T8[,], pixels)

Specify a two-dimensional texture image.

- static void [TexImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat

format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute]
ref T8 pixels)

Specify a two-dimensional texture image.

- static void **TexImage2DMultisample** (OpenTK.Graphics.OpenGL.TextureTargetMultisample target, Int32 samples, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, bool fixedsamplelocations)
- static void **TexImage3D** (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Specify a three-dimensional texture image.

- static void **TexImage3D< T9 >** (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T9[] pixels)

Specify a three-dimensional texture image.

- static void **TexImage3D< T9 >** (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T9[,] pixels)

Specify a three-dimensional texture image.

- static void **TexImage3D< T9 >** (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T9[,]) pixels)

Specify a three-dimensional texture image.

- static void **TexImage3D< T9 >** (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T9 pixels)

Specify a three-dimensional texture image.

- static void **TexImage3DMultisample** (OpenTK.Graphics.OpenGL.TextureTargetMultisample target, Int32 samples, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, bool fixedsamplelocations)
- static void **TexParameter** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Single param)

Set texture parameters.

- static void **TexParameter** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Single[] @params)

Set texture parameters.

- static unsafe void **TexParameter** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Single* @params)

Set texture parameters.

- static void **TexParameter** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Int32 param)

Set texture parameters.

- static void **TexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Int32[] @params)

- static void **TexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, ref Int32 @params)

- static unsafe void **TexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Int32* @params)

- static void **TexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, UInt32[] @params)

- static void **TexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, ref UInt32 @params)

- static unsafe void **TexParameterI** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, UInt32* @params)

- static void **TexParameter** (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Int32[] @params)

Set texture parameters.

- static unsafe void [TexParameter](#) (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Int32 *@params)

Set texture parameters.

- static void [TexSubImage1D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Specify a one-dimensional texture subimage.

- static void [TexSubImage1D< T6 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[] pixels)

Specify a one-dimensional texture subimage.

- static void [TexSubImage1D< T6 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[,] pixels)

Specify a one-dimensional texture subimage.

- static void [TexSubImage1D< T6 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[,]) pixels)

Specify a one-dimensional texture subimage.

- static void [TexSubImage1D< T6 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T6 pixels)

Specify a one-dimensional texture subimage.

- static void [TexSubImage2D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T8[] pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T8[,] pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T8[,], pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage2D< T8 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T8 pixels)

Specify a two-dimensional texture subimage.

- static void [TexSubImage3D](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels)

Specify a three-dimensional texture subimage.

- static void [TexSubImage3D< T10 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T10[] pixels)

Specify a three-dimensional texture subimage.

- static void [TexSubImage3D< T10 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width,

Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T10[, pixels])

Specify a three-dimensional texture subimage.

- static void [TexSubImage3D< T10 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] T10[, pixels])

Specify a three-dimensional texture subimage.

- static void [TexSubImage3D< T10 >](#) (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32 height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type,[InAttribute, OutAttribute] ref T10 pixels)

Specify a three-dimensional texture subimage.

- static void **TransformFeedbackVaryings** (Int32 program, Int32 count, String[] varyings, OpenTK.Graphics.OpenGL.TransformFeedbackMode bufferMode)
- static void **TransformFeedbackVaryings** (UInt32 program, Int32 count, String[] varyings, OpenTK.Graphics.OpenGL.TransformFeedbackMode bufferMode)
- static void [Translate](#) (Double x, Double y, Double z)

Multiply the current matrix by a translation matrix.

- static void [Translate](#) (Single x, Single y, Single z)

Multiply the current matrix by a translation matrix.

- static void [Uniform1](#) (Int32 location, Single v0)

Specify the value of a uniform variable for the current program object.

- static void [Uniform1](#) (Int32 location, Int32 count, Single[] value)

Specify the value of a uniform variable for the current program object.

- static void [Uniform1](#) (Int32 location, Int32 count, ref Single value)

Specify the value of a uniform variable for the current program object.

- static unsafe void [Uniform1](#) (Int32 location, Int32 count, Single *value)

Specify the value of a uniform variable for the current program object.

- static void [Uniform1](#) (Int32 location, Int32 v0)

Specify the value of a uniform variable for the current program object.

- static void **Uniform1** (Int32 location, Int32 count, Int32[] value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, ref Int32 value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform1** (Int32 location, Int32 count, Int32 *value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, UInt32 v0)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, UInt32[] value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform1** (Int32 location, Int32 count, ref UInt32 value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform1** (Int32 location, Int32 count, UInt32 *value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Single v0, Single v1)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Int32 count, Single[] value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Int32 count, ref Single value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform2** (Int32 location, Int32 count, Single *value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Int32 v0, Int32 v1)
Specify the value of a uniform variable for the current program object.
- static void **Uniform2** (Int32 location, Int32 count, Int32[] value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform2** (Int32 location, Int32 count, Int32 *value)

Specify the value of a uniform variable for the current program object.

- static void [Uniform2](#) (Int32 location, UInt32 v0, UInt32 v1)
Specify the value of a uniform variable for the current program object.
- static void [Uniform2](#) (Int32 location, Int32 count, UInt32[] value)
Specify the value of a uniform variable for the current program object.
- static void [Uniform2](#) (Int32 location, Int32 count, ref UInt32 value)
Specify the value of a uniform variable for the current program object.
- static unsafe void [Uniform2](#) (Int32 location, Int32 count, UInt32 *value)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Single v0, Single v1, Single v2)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, Single[] value)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, ref Single value)
Specify the value of a uniform variable for the current program object.
- static unsafe void [Uniform3](#) (Int32 location, Int32 count, Single *value)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 v0, Int32 v1, Int32 v2)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, Int32[] value)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, ref Int32 value)
Specify the value of a uniform variable for the current program object.
- static unsafe void [Uniform3](#) (Int32 location, Int32 count, Int32 *value)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, UInt32 v0, UInt32 v1, UInt32 v2)
Specify the value of a uniform variable for the current program object.
- static void [Uniform3](#) (Int32 location, Int32 count, UInt32[] value)

Specify the value of a uniform variable for the current program object.

- static void **Uniform3** (Int32 location, Int32 count, ref UInt32 value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform3** (Int32 location, Int32 count, UInt32 *value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Single v0, Single v1, Single v2, Single v3)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, Single[] value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, ref Single value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform4** (Int32 location, Int32 count, Single *value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 v0, Int32 v1, Int32 v2, Int32 v3)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, Int32[] value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, ref Int32 value)
Specify the value of a uniform variable for the current program object.
- static unsafe void **Uniform4** (Int32 location, Int32 count, Int32 *value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, UInt32 v0, UInt32 v1, UInt32 v2, UInt32 v3)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, UInt32[] value)
Specify the value of a uniform variable for the current program object.
- static void **Uniform4** (Int32 location, Int32 count, ref UInt32 value)
Specify the value of a uniform variable for the current program object.

- static unsafe void [Uniform4](#) (Int32 location, Int32 count, UInt32 *value)

Specify the value of a uniform variable for the current program object.

- static void **UniformBlockBinding** (Int32 program, Int32 uniformBlockIndex, Int32 uniformBlockBinding)
- static void **UniformBlockBinding** (UInt32 program, UInt32 uniformBlockIndex, UInt32 uniformBlockBinding)
- static void **UniformMatrix2** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix2** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix2** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix2x3** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix2x3** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix2x3** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix2x4** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix2x4** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix2x4** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix3** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix3** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix3** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix3x2** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix3x2** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix3x2** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix3x4** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix3x4** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix3x4** (Int32 location, Int32 count, bool transpose, Single *value)

- static void **UniformMatrix4** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix4** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix4** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix4x2** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix4x2** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix4x2** (Int32 location, Int32 count, bool transpose, Single *value)
- static void **UniformMatrix4x3** (Int32 location, Int32 count, bool transpose, Single[] value)
- static void **UniformMatrix4x3** (Int32 location, Int32 count, bool transpose, ref Single value)
- static unsafe void **UniformMatrix4x3** (Int32 location, Int32 count, bool transpose, Single *value)
- static bool **UnmapBuffer** (OpenTK.Graphics.OpenGL.BufferTarget target)
- static void **UseProgram** (Int32 program)

Installs a program object as part of current rendering state.

- static void **UseProgram** (UInt32 program)
Installs a program object as part of current rendering state.
- static void **ValidateProgram** (Int32 program)
Validates a program object.
- static void **ValidateProgram** (UInt32 program)
Validates a program object.
- static void **Vertex2** (Double x, Double y)
Specify a vertex.
- static void **Vertex2** (Double[] v)
Specify a vertex.
- static void **Vertex2** (ref Double v)
Specify a vertex.
- static unsafe void **Vertex2** (Double *v)
Specify a vertex.

- static void [Vertex2](#) (Single x, Single y)
Specify a vertex.
- static void [Vertex2](#) (Single[] v)
Specify a vertex.
- static void [Vertex2](#) (ref Single v)
Specify a vertex.
- static unsafe void [Vertex2](#) (Single *v)
Specify a vertex.
- static void [Vertex2](#) (Int32 x, Int32 y)
Specify a vertex.
- static void [Vertex2](#) (Int32[] v)
Specify a vertex.
- static void [Vertex2](#) (ref Int32 v)
Specify a vertex.
- static unsafe void [Vertex2](#) (Int32 *v)
Specify a vertex.
- static void [Vertex2](#) (Int16 x, Int16 y)
Specify a vertex.
- static void [Vertex2](#) (Int16[] v)
Specify a vertex.
- static void [Vertex2](#) (ref Int16 v)
Specify a vertex.
- static unsafe void [Vertex2](#) (Int16 *v)
Specify a vertex.
- static void [Vertex3](#) (Double x, Double y, Double z)
Specify a vertex.
- static void [Vertex3](#) (Double[] v)
Specify a vertex.

- static void [Vertex3](#) (ref Double v)
Specify a vertex.
- static unsafe void [Vertex3](#) (Double *v)
Specify a vertex.
- static void [Vertex3](#) (Single x, Single y, Single z)
Specify a vertex.
- static void [Vertex3](#) (Single[] v)
Specify a vertex.
- static void [Vertex3](#) (ref Single v)
Specify a vertex.
- static unsafe void [Vertex3](#) (Single *v)
Specify a vertex.
- static void [Vertex3](#) (Int32 x, Int32 y, Int32 z)
Specify a vertex.
- static void [Vertex3](#) (Int32[] v)
Specify a vertex.
- static void [Vertex3](#) (ref Int32 v)
Specify a vertex.
- static unsafe void [Vertex3](#) (Int32 *v)
Specify a vertex.
- static void [Vertex3](#) (Int16 x, Int16 y, Int16 z)
Specify a vertex.
- static void [Vertex3](#) (Int16[] v)
Specify a vertex.
- static void [Vertex3](#) (ref Int16 v)
Specify a vertex.
- static unsafe void [Vertex3](#) (Int16 *v)
Specify a vertex.

- static void [Vertex4](#) (Double x, Double y, Double z, Double w)
Specify a vertex.
- static void [Vertex4](#) (Double[] v)
Specify a vertex.
- static void [Vertex4](#) (ref Double v)
Specify a vertex.
- static unsafe void [Vertex4](#) (Double *v)
Specify a vertex.
- static void [Vertex4](#) (Single x, Single y, Single z, Single w)
Specify a vertex.
- static void [Vertex4](#) (Single[] v)
Specify a vertex.
- static void [Vertex4](#) (ref Single v)
Specify a vertex.
- static unsafe void [Vertex4](#) (Single *v)
Specify a vertex.
- static void [Vertex4](#) (Int32 x, Int32 y, Int32 z, Int32 w)
Specify a vertex.
- static void [Vertex4](#) (Int32[] v)
Specify a vertex.
- static void [Vertex4](#) (ref Int32 v)
Specify a vertex.
- static unsafe void [Vertex4](#) (Int32 *v)
Specify a vertex.
- static void [Vertex4](#) (Int16 x, Int16 y, Int16 z, Int16 w)
Specify a vertex.
- static void [Vertex4](#) (Int16[] v)
Specify a vertex.

- static void [Vertex4](#) (ref Int16 v)
Specify a vertex.
- static unsafe void [Vertex4](#) (Int16 *v)
Specify a vertex.
- static void [VertexAttrib1](#) (Int32 index, Double x)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (UInt32 index, Double x)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (Int32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (UInt32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (Int32 index, Single x)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (UInt32 index, Single x)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (Int32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (UInt32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (Int32 index, Int16 x)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib1](#) (UInt32 index, Int16 x)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (Int32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib1](#) (UInt32 index, Int16 *v)
Specifies the value of a generic vertex attribute.

- static void [VertexAttrib2](#) (Int32 index, Double x, Double y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, Double x, Double y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, Double[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, ref Double v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (Int32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, Double[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, ref Double v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (UInt32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, Single x, Single y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, Single x, Single y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, Single[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, ref Single v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (Int32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, Single[] v)
Specifies the value of a generic vertex attribute.

- static void [VertexAttrib2](#) (UInt32 index, ref Single v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (UInt32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, Int16 x, Int16 y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, Int16 x, Int16 y)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, Int16[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (Int32 index, ref Int16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (Int32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, Int16[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib2](#) (UInt32 index, ref Int16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib2](#) (UInt32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, Double x, Double y, Double z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, Double x, Double y, Double z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, Double[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, ref Double v)
Specifies the value of a generic vertex attribute.

- static unsafe void [VertexAttrib3](#) (Int32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, Double[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, ref Double v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (UInt32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, Single x, Single y, Single z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, Single x, Single y, Single z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, Single[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, ref Single v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (Int32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, Single[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, ref Single v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (UInt32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, Int16 x, Int16 y, Int16 z)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, Int16 x, Int16 y, Int16 z)
Specifies the value of a generic vertex attribute.

- static void [VertexAttrib3](#) (Int32 index, Int16[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (Int32 index, ref Int16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (Int32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, Int16[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib3](#) (UInt32 index, ref Int16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib3](#) (UInt32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, SByte[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref SByte v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 index, SByte *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, Double x, Double y, Double z, Double w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, Double x, Double y, Double z, Double w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, Double[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, ref Double v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (Int32 index, Double *v)
Specifies the value of a generic vertex attribute.

- static void [VertexAttrib4](#) (UInt32 index, Double[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref Double v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 index, Double *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, Single x, Single y, Single z, Single w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, Single x, Single y, Single z, Single w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, Single[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, ref Single v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (Int32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, Single[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref Single v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 index, Single *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, Int32[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, ref Int32 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (Int32 index, Int32 *v)
Specifies the value of a generic vertex attribute.

- static void [VertexAttrib4](#) (UInt32 index, Int32[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref Int32 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 index, Int32 *v)
Specifies the value of a generic vertex attribute.
- static void **VertexAttrib4N** (UInt32 index, SByte[] v)
- static void **VertexAttrib4N** (UInt32 index, ref SByte v)
- static unsafe void **VertexAttrib4N** (UInt32 index, SByte *v)
- static void **VertexAttrib4N** (Int32 index, Int32[] v)
- static void **VertexAttrib4N** (Int32 index, ref Int32 v)
- static unsafe void **VertexAttrib4N** (Int32 index, Int32 *v)
- static void **VertexAttrib4N** (UInt32 index, Int32[] v)
- static void **VertexAttrib4N** (UInt32 index, ref Int32 v)
- static unsafe void **VertexAttrib4N** (UInt32 index, Int32 *v)
- static void **VertexAttrib4N** (Int32 index, Int16[] v)
- static void **VertexAttrib4N** (Int32 index, ref Int16 v)
- static unsafe void **VertexAttrib4N** (Int32 index, Int16 *v)
- static void **VertexAttrib4N** (UInt32 index, Int16[] v)
- static void **VertexAttrib4N** (UInt32 index, ref Int16 v)
- static unsafe void **VertexAttrib4N** (UInt32 index, Int16 *v)
- static void **VertexAttrib4N** (Int32 index, Byte x, Byte y, Byte z, Byte w)
- static void **VertexAttrib4N** (UInt32 index, Byte x, Byte y, Byte z, Byte w)
- static void **VertexAttrib4N** (Int32 index, Byte[] v)
- static void **VertexAttrib4N** (Int32 index, ref Byte v)
- static unsafe void **VertexAttrib4N** (Int32 index, Byte *v)
- static void **VertexAttrib4N** (UInt32 index, Byte[] v)
- static void **VertexAttrib4N** (UInt32 index, ref Byte v)
- static unsafe void **VertexAttrib4N** (UInt32 index, Byte *v)
- static void **VertexAttrib4N** (UInt32 index, UInt32[] v)
- static void **VertexAttrib4N** (UInt32 index, ref UInt32 v)
- static unsafe void **VertexAttrib4N** (UInt32 index, UInt32 *v)
- static void **VertexAttrib4N** (UInt32 index, UInt16[] v)
- static void **VertexAttrib4N** (UInt32 index, ref UInt16 v)
- static unsafe void **VertexAttrib4N** (UInt32 index, UInt16 *v)
- static void [VertexAttrib4](#) (Int32 index, Int16 x, Int16 y, Int16 z, Int16 w)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, Int16 x, Int16 y, Int16 z, Int16 w)

Specifies the value of a generic vertex attribute.

- static void [VertexAttrib4](#) (Int32 index, Int16[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, ref Int16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (Int32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, Int16[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref Int16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 index, Int16 *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, Byte[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (Int32 index, ref Byte v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (Int32 index, Byte *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, Byte[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref Byte v)
Specifies the value of a generic vertex attribute.
- static unsafe void [VertexAttrib4](#) (UInt32 index, Byte *v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, UInt32[] v)
Specifies the value of a generic vertex attribute.
- static void [VertexAttrib4](#) (UInt32 index, ref UInt32 v)

Specifies the value of a generic vertex attribute.

- static unsafe void **VertexAttrib4** (UInt32 index, UInt32 *v)
Specifies the value of a generic vertex attribute.
- static void **VertexAttrib4** (UInt32 index, UInt16[] v)
Specifies the value of a generic vertex attribute.
- static void **VertexAttrib4** (UInt32 index, ref UInt16 v)
Specifies the value of a generic vertex attribute.
- static unsafe void **VertexAttrib4** (UInt32 index, UInt16 *v)
Specifies the value of a generic vertex attribute.
- static void **VertexAttribI1** (Int32 index, Int32 x)
- static void **VertexAttribI1** (UInt32 index, Int32 x)
- static unsafe void **VertexAttribI1** (Int32 index, Int32 *v)
- static unsafe void **VertexAttribI1** (UInt32 index, Int32 *v)
- static void **VertexAttribI1** (UInt32 index, UInt32 x)
- static unsafe void **VertexAttribI1** (UInt32 index, UInt32 *v)
- static void **VertexAttribI2** (Int32 index, Int32 x, Int32 y)
- static void **VertexAttribI2** (UInt32 index, Int32 x, Int32 y)
- static void **VertexAttribI2** (Int32 index, Int32[] v)
- static void **VertexAttribI2** (Int32 index, ref Int32 v)
- static unsafe void **VertexAttribI2** (Int32 index, Int32 *v)
- static void **VertexAttribI2** (UInt32 index, Int32[] v)
- static void **VertexAttribI2** (UInt32 index, ref Int32 v)
- static unsafe void **VertexAttribI2** (UInt32 index, Int32 *v)
- static void **VertexAttribI2** (UInt32 index, UInt32 x, UInt32 y)
- static void **VertexAttribI2** (UInt32 index, UInt32[] v)
- static void **VertexAttribI2** (UInt32 index, ref UInt32 v)
- static unsafe void **VertexAttribI2** (UInt32 index, UInt32 *v)
- static void **VertexAttribI3** (Int32 index, Int32 x, Int32 y, Int32 z)
- static void **VertexAttribI3** (UInt32 index, Int32 x, Int32 y, Int32 z)
- static void **VertexAttribI3** (Int32 index, Int32[] v)
- static void **VertexAttribI3** (Int32 index, ref Int32 v)
- static unsafe void **VertexAttribI3** (Int32 index, Int32 *v)
- static void **VertexAttribI3** (UInt32 index, Int32[] v)
- static void **VertexAttribI3** (UInt32 index, ref Int32 v)
- static unsafe void **VertexAttribI3** (UInt32 index, Int32 *v)
- static void **VertexAttribI3** (UInt32 index, UInt32 x, UInt32 y, UInt32 z)
- static void **VertexAttribI3** (UInt32 index, UInt32[] v)
- static void **VertexAttribI3** (UInt32 index, ref UInt32 v)

- static unsafe void **VertexAttribI3** (UInt32 index, UInt32 *v)
- static void **VertexAttribI4** (UInt32 index, SByte[] v)
- static void **VertexAttribI4** (UInt32 index, ref SByte v)
- static unsafe void **VertexAttribI4** (UInt32 index, SByte *v)
- static void **VertexAttribI4** (Int32 index, Int32 x, Int32 y, Int32 z, Int32 w)
- static void **VertexAttribI4** (UInt32 index, Int32 x, Int32 y, Int32 z, Int32 w)
- static void **VertexAttribI4** (Int32 index, Int32[] v)
- static void **VertexAttribI4** (Int32 index, ref Int32 v)
- static unsafe void **VertexAttribI4** (Int32 index, Int32 *v)
- static void **VertexAttribI4** (UInt32 index, Int32[] v)
- static void **VertexAttribI4** (UInt32 index, ref Int32 v)
- static unsafe void **VertexAttribI4** (UInt32 index, Int32 *v)
- static void **VertexAttribI4** (Int32 index, Int16[] v)
- static void **VertexAttribI4** (Int32 index, ref Int16 v)
- static unsafe void **VertexAttribI4** (Int32 index, Int16 *v)
- static void **VertexAttribI4** (UInt32 index, Int16[] v)
- static void **VertexAttribI4** (UInt32 index, ref Int16 v)
- static unsafe void **VertexAttribI4** (UInt32 index, Int16 *v)
- static void **VertexAttribI4** (Int32 index, Byte[] v)
- static void **VertexAttribI4** (Int32 index, ref Byte v)
- static unsafe void **VertexAttribI4** (Int32 index, Byte *v)
- static void **VertexAttribI4** (UInt32 index, Byte[] v)
- static void **VertexAttribI4** (UInt32 index, ref Byte v)
- static unsafe void **VertexAttribI4** (UInt32 index, Byte *v)
- static void **VertexAttribI4** (UInt32 index, UInt32 x, UInt32 y, UInt32 z, UInt32 w)
- static void **VertexAttribI4** (UInt32 index, UInt32[] v)
- static void **VertexAttribI4** (UInt32 index, ref UInt32 v)
- static unsafe void **VertexAttribI4** (UInt32 index, UInt32 *v)
- static void **VertexAttribI4** (UInt32 index, UInt16[] v)
- static void **VertexAttribI4** (UInt32 index, ref UInt16 v)
- static unsafe void **VertexAttribI4** (UInt32 index, UInt16 *v)
- static void **VertexAttribIPointer** (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride, IntPtr pointer)
- static void **VertexAttribIPointer**< T4 > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] T4[] pointer)
- static void **VertexAttribIPointer**< T4 > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] T4[,] pointer)
- static void **VertexAttribIPointer**< T4 > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] T4[,.] pointer)

- static void **VertexAttribIPointer**< **T4** > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] ref T4 pointer)
- static void **VertexAttribIPointer** (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride, IntPtr pointer)
- static void **VertexAttribIPointer**< **T4** > (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] T4[] pointer)
- static void **VertexAttribIPointer**< **T4** > (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] T4[,] pointer)
- static void **VertexAttribIPointer**< **T4** > (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] T4[,], pointer)
- static void **VertexAttribIPointer**< **T4** > (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, Int32 stride,[InAttribute, OutAttribute] ref T4 pointer)
- static void **VertexAttribPointer** (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride, IntPtr pointer)

Define an array of generic vertex attribute data.

- static void **VertexAttribPointer**< **T5** > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[] pointer)

Define an array of generic vertex attribute data.

- static void **VertexAttribPointer**< **T5** > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,] pointer)

Define an array of generic vertex attribute data.

- static void **VertexAttribPointer**< **T5** > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,], pointer)

Define an array of generic vertex attribute data.

- static void **VertexAttribPointer**< **T5** > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] ref T5 pointer)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer](#) (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride, IntPtr pointer)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer< T5 >](#) (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[] pointer)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer< T5 >](#) (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,] pointer)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer< T5 >](#) (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] T5[,.] pointer)

Define an array of generic vertex attribute data.

- static void [VertexAttribPointer< T5 >](#) (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointer type, bool normalized, Int32 stride,[InAttribute, OutAttribute] ref T5 pointer)

Define an array of generic vertex attribute data.

- static void [VertexPointer](#) (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, IntPtr pointer)

Define an array of vertex data.

- static void [VertexPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, Int32 stride,[InAttribute, OutAttribute] T3[] pointer)

Define an array of vertex data.

- static void [VertexPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, Int32 stride,[InAttribute, OutAttribute] T3[,] pointer)

Define an array of vertex data.

- static void [VertexPointer< T3 >](#) (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, Int32 stride,[InAttribute, OutAttribute] T3[,.] pointer)

Define an array of vertex data.

- static void [VertexPointer](#)< [T3](#) > (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, Int32 stride,[InAttribute, OutAttribute] ref T3 pointer)
Define an array of vertex data.
- static void [Viewport](#) (Int32 x, Int32 y, Int32 width, Int32 height)
Set the viewport.
- static void **WaitSync** (IntPtr sync, Int32 flags, Int64 timeout)
- static void **WaitSync** (IntPtr sync, UInt32 flags, UInt64 timeout)
- static void [WindowPos2](#) (Double x, Double y)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Double[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (ref Double v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos2](#) (Double *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Single x, Single y)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Single[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (ref Single v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos2](#) (Single *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Int32 x, Int32 y)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Int32[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (ref Int32 v)
Specify the raster position in window coordinates for pixel operations.

- static unsafe void [WindowPos2](#) (Int32 *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Int16 x, Int16 y)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (Int16[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos2](#) (ref Int16 v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos2](#) (Int16 *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Double x, Double y, Double z)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Double[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (ref Double v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos3](#) (Double *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Single x, Single y, Single z)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Single[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (ref Single v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos3](#) (Single *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Int32 x, Int32 y, Int32 z)
Specify the raster position in window coordinates for pixel operations.

- static void [WindowPos3](#) (Int32[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (ref Int32 v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos3](#) (Int32 *v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Int16 x, Int16 y, Int16 z)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (Int16[] v)
Specify the raster position in window coordinates for pixel operations.
- static void [WindowPos3](#) (ref Int16 v)
Specify the raster position in window coordinates for pixel operations.
- static unsafe void [WindowPos3](#) (Int16 *v)
Specify the raster position in window coordinates for pixel operations.
- static void [LoadAll](#) ()
Loads all [OpenGL](#) entry points (core and extension). This method is provided for compatibility purposes with older OpenTK versions.
- static void **Color3** (System.Drawing.Color color)
- static void **Color4** (System.Drawing.Color color)
- static void **Color3** ([Vector3](#) color)
- static void **Color4** ([Vector4](#) color)
- static void **Color4** ([Color4](#) color)
- static void **ClearColor** (System.Drawing.Color color)
- static void **ClearColor** ([Color4](#) color)
- static void **BlendColor** (System.Drawing.Color color)
- static void **BlendColor** ([Color4](#) color)
- static void **Material** (MaterialFace face, MaterialParameter pname, [Vector4](#) @params)
- static void **Material** (MaterialFace face, MaterialParameter pname, [Color4](#) @params)
- static void **Light** (LightName name, LightParameter pname, [Vector4](#) @params)
- static void **Light** (LightName name, LightParameter pname, [Color4](#) @params)
- static void **Normal3** ([Vector3](#) normal)
- static void **RasterPos2** ([Vector2](#) pos)
- static void **RasterPos3** ([Vector3](#) pos)

- static void **RasterPos4** ([Vector4](#) pos)
- static void **Vertex2** ([Vector2](#) v)
- static void **Vertex3** ([Vector3](#) v)
- static void **Vertex4** ([Vector4](#) v)
- static void **TexCoord2** ([Vector2](#) v)
- static void **TexCoord3** ([Vector3](#) v)
- static void **TexCoord4** ([Vector4](#) v)
- static void **Rotate** (Single angle, [Vector3](#) axis)
- static void **Scale** ([Vector3](#) scale)
- static void **Translate** ([Vector3](#) trans)
- static void **MultMatrix** (ref [Matrix4](#) mat)
- static void **LoadMatrix** (ref [Matrix4](#) mat)
- static void **LoadTransposeMatrix** (ref [Matrix4](#) mat)
- static void **MultTransposeMatrix** (ref [Matrix4](#) mat)
- static void **UniformMatrix4** (int location, bool transpose, ref [Matrix4](#) matrix)
- static void **Normal3** ([Vector3d](#) normal)
- static void **RasterPos2** ([Vector2d](#) pos)
- static void **RasterPos3** ([Vector3d](#) pos)
- static void **RasterPos4** ([Vector4d](#) pos)
- static void **Vertex2** ([Vector2d](#) v)
- static void **Vertex3** ([Vector3d](#) v)
- static void **Vertex4** ([Vector4d](#) v)
- static void **TexCoord2** ([Vector2d](#) v)
- static void **TexCoord3** ([Vector3d](#) v)
- static void **TexCoord4** ([Vector4d](#) v)
- static void **Rotate** (double angle, [Vector3d](#) axis)
- static void **Scale** ([Vector3d](#) scale)
- static void **Translate** ([Vector3d](#) trans)
- static void **MultMatrix** (ref [Matrix4d](#) mat)
- static void **LoadMatrix** (ref [Matrix4d](#) mat)
- static void **LoadTransposeMatrix** (ref [Matrix4d](#) mat)
- static void **MultTransposeMatrix** (ref [Matrix4d](#) mat)
- static void **Uniform2** (int location, ref [Vector2](#) vector)
- static void **Uniform3** (int location, ref [Vector3](#) vector)
- static void **Uniform4** (int location, ref [Vector4](#) vector)
- static void **Uniform2** (int location, [Vector2](#) vector)
- static void **Uniform3** (int location, [Vector3](#) vector)
- static void **Uniform4** (int location, [Vector4](#) vector)
- static void **Uniform4** (int location, [Color4](#) color)
- static void **Uniform4** (int location, [Quaternion](#) quaternion)
- static string **GetActiveAttrib** (int program, int index, out int size, out [ActiveAttribType](#) type)

- static string **GetActiveUniform** (int program, int uniformIndex, out int size, out ActiveUniformType type)
- static string **GetActiveUniformName** (int program, int uniformIndex)
- static string **GetActiveUniformBlockName** (int program, int uniformIndex)
- static void **ShaderSource** (Int32 shader, System.String @string)
- static string **GetShaderInfoLog** (Int32 shader)
- static void **GetShaderInfoLog** (Int32 shader, out string info)
- static string **GetProgramInfoLog** (Int32 program)
- static void **GetProgramInfoLog** (Int32 program, out string info)
- static void **PointParameter** (PointSpriteCoordOriginParameter param)

Helper function that defines the coordinate origin of the Point Sprite.

- static void **VertexAttrib2** (Int32 index, ref [Vector2](#) v)
- static void **VertexAttrib3** (Int32 index, ref [Vector3](#) v)
- static void **VertexAttrib4** (Int32 index, ref [Vector4](#) v)
- static void **VertexAttrib2** (Int32 index, [Vector2](#) v)
- static void **VertexAttrib3** (Int32 index, [Vector3](#) v)
- static void **VertexAttrib4** (Int32 index, [Vector4](#) v)
- static void **MultiTexCoord2** (TextureUnit target, ref [Vector2](#) v)
- static void **MultiTexCoord3** (TextureUnit target, ref [Vector3](#) v)
- static void **MultiTexCoord4** (TextureUnit target, ref [Vector4](#) v)
- static void **VertexAttrib2** (Int32 index, ref [Vector2d](#) v)
- static void **VertexAttrib3** (Int32 index, ref [Vector3d](#) v)
- static void **VertexAttrib4** (Int32 index, ref [Vector4d](#) v)
- static void **VertexAttrib2** (Int32 index, [Vector2d](#) v)
- static void **VertexAttrib3** (Int32 index, [Vector3d](#) v)
- static void **VertexAttrib4** (Int32 index, [Vector4d](#) v)
- static void **MultiTexCoord2** (TextureUnit target, ref [Vector2d](#) v)
- static void **MultiTexCoord3** (TextureUnit target, ref [Vector3d](#) v)
- static void **MultiTexCoord4** (TextureUnit target, ref [Vector4d](#) v)
- static void **Rect** (System.Drawing.RectangleF rect)
- static void **Rect** (System.Drawing.Rectangle rect)
- static void **Rect** (ref System.Drawing.RectangleF rect)
- static void **Rect** (ref System.Drawing.Rectangle rect)
- static int **GenTexture** ()
- static void **DeleteTexture** (int id)
- static void **DeleteTexture** (uint id)
- static void **VertexPointer** (int size, VertexPointerType type, int stride, int offset)
- static void **NormalPointer** (NormalPointerType type, int stride, int offset)
- static void **IndexPointer** (IndexPointerType type, int stride, int offset)
- static void **ColorPointer** (int size, ColorPointerType type, int stride, int offset)
- static void **FogCoordPointer** (FogPointerType type, int stride, int offset)

- static void **EdgeFlagPointer** (int stride, int offset)
- static void **TexCoordPointer** (int size, TexCoordPointerType type, int stride, int offset)
- static void **VertexAttribPointer** (int index, int size, VertexAttribPointerType type, bool normalized, int stride, int offset)
- static void **DrawElements** (BeginMode mode, int count, DrawElementsType type, int offset)
- static void **GetFloat** (GetPName pname, out [Vector2](#) vector)
- static void **GetFloat** (GetPName pname, out [Vector3](#) vector)
- static void **GetFloat** (GetPName pname, out [Vector4](#) vector)
- static void **GetFloat** (GetPName pname, out [Matrix4](#) matrix)
- static void **GetDouble** (GetPName pname, out [Vector2d](#) vector)
- static void **GetDouble** (GetPName pname, out [Vector3d](#) vector)
- static void **GetDouble** (GetPName pname, out [Vector4d](#) vector)
- static void **GetDouble** (GetPName pname, out [Matrix4d](#) matrix)
- static void **Viewport** (System.Drawing.Size size)
- static void **Viewport** (System.Drawing.Point location, System.Drawing.Size size)
- static void **Viewport** (System.Drawing.Rectangle rectangle)
- static void **TexEnv** (TextureEnvTarget target, TextureEnvParameter pname, System.Drawing.Color color)
- static void **TexEnv** (TextureEnvTarget target, TextureEnvParameter pname, [Color4](#) color)
- static void **DisableClientState** (OpenTK.Graphics.OpenGL.EnableCap array)
- static void **EnableClientState** (OpenTK.Graphics.OpenGL.EnableCap array)
- static void **GetActiveUniforms** (Int32 program, Int32 uniformCount, Int32[] uniformIndices, ArbUniformBufferObject pname,[OutAttribute] Int32[] @params)
- static void **GetActiveUniforms** (Int32 program, Int32 uniformCount, ref Int32 uniformIndices, ArbUniformBufferObject pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetActiveUniforms** (Int32 program, Int32 uniformCount, Int32 *uniformIndices, ArbUniformBufferObject pname,[OutAttribute] Int32 * @params)
- static void **GetActiveUniforms** (UInt32 program, Int32 uniformCount, UInt32[] uniformIndices, ArbUniformBufferObject pname,[OutAttribute] Int32[] @params)
- static void **GetActiveUniforms** (UInt32 program, Int32 uniformCount, ref UInt32 uniformIndices, ArbUniformBufferObject pname,[OutAttribute] out Int32 @params)
- static unsafe void **GetActiveUniforms** (UInt32 program, Int32 uniformCount, UInt32 *uniformIndices, ArbUniformBufferObject pname,[OutAttribute] Int32 * @params)

Properties

- override object [SyncRoot](#) [get]

Returns a synchronization token unique for the [GL](#) class.

5.37.1 Detailed Description

[OpenGL](#) bindings for .NET, implementing the full [OpenGL](#) API, including extensions. This class contains all [OpenGL](#) enums and functions defined in the latest [OpenGL](#) specification. The official .spec files can be found at: <http://opengl.org/registry/>.

A valid [OpenGL](#) context must be created before calling any [OpenGL](#) function.

Use the [GL.Load](#) and [GL.LoadAll](#) methods to prepare function entry points prior to use. To maintain cross-platform compatibility, this must be done for both core and extension functions. The [GameWindow](#) and the [GLControl](#) class will take care of this automatically.

You can use the [GL.SupportsExtension](#) method to check whether any given category of extension functions exists in the current [OpenGL](#) context. Keep in mind that different [OpenGL](#) contexts may support different extensions, and under different entry points. Always check if all required extensions are still supported when changing visuals or pixel formats.

You may retrieve the entry point for an [OpenGL](#) function using the [GL.GetDelegate](#) method.

5.37.2 Member Function Documentation

5.37.2.1 static void [OpenTK.Graphics.OpenGL.GL.Accum](#) (
 [OpenTK.Graphics.OpenGL.AccumOp](#) *op*, Single *value*)
 [static]

Operate on the accumulation buffer.

Parameters

op Specifies the accumulation buffer operation. Symbolic constants [GL_ACCUM](#), [GL_LOAD](#), [GL_ADD](#), [GL_MULT](#), and [GL_RETURN](#) are accepted.

value Specifies a floating-point value used in the accumulation buffer operation. *op* determines how value is used.

5.37.2.2 `static void OpenTK.Graphics.OpenGL.GL.ActiveTexture (OpenTK.Graphics.OpenGL.TextureUnit texture) [static]`

Select active texture unit.

Parameters

texture Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. texture must be one of GL_TEXTURE*i*, where *i* ranges from 0 to the larger of (GL_MAX_TEXTURE_COORDS - 1) and (GL_MAX_COMBINED_TEXTURE_IMAGE_UNITS - 1). The initial value is GL_TEXTURE0.

5.37.2.3 `static void OpenTK.Graphics.OpenGL.GL.AlphaFunc (OpenTK.Graphics.OpenGL.AlphaFunction func, Single @ ref) [static]`

Specify the alpha test function.

Parameters

func Specifies the alpha comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_ALWAYS.

ref Specifies the reference value that incoming alpha values are compared to. This value is clamped to the range [0,1], where 0 represents the lowest possible alpha value and 1 the highest possible value. The initial reference value is 0.

5.37.2.4 `static bool OpenTK.Graphics.OpenGL.GL.AreTexturesResident (Int32 n, Int32[] textures, [OutAttribute] bool[] residences) [static]`

Determine if textures are loaded in texture memory.

Parameters

n Specifies the number of textures to be queried.

textures Specifies an array containing the names of the textures to be queried.

residences Specifies an array in which the texture residence status is returned. The residence status of a texture named by an element of textures is returned in the corresponding element of residences.

5.37.2.5 `static bool OpenTK.Graphics.OpenGL.GL.AreTexturesResident (Int32 n, ref UInt32 textures, [OutAttribute] out bool residences) [static]`

Determine if textures are loaded in texture memory.

Parameters

n Specifies the number of textures to be queried.

textures Specifies an array containing the names of the textures to be queried.

residences Specifies an array in which the texture residence status is returned. The residence status of a texture named by an element of textures is returned in the corresponding element of residences.

5.37.2.6 `static unsafe bool OpenTK.Graphics.OpenGL.GL.AreTexturesResident (Int32 n, UInt32 * textures, [OutAttribute] bool * residences) [static]`

Determine if textures are loaded in texture memory.

Parameters

n Specifies the number of textures to be queried.

textures Specifies an array containing the names of the textures to be queried.

residences Specifies an array in which the texture residence status is returned. The residence status of a texture named by an element of textures is returned in the corresponding element of residences.

5.37.2.7 `static bool OpenTK.Graphics.OpenGL.GL.AreTexturesResident (Int32 n, ref Int32 textures, [OutAttribute] out bool residences) [static]`

Determine if textures are loaded in texture memory.

Parameters

n Specifies the number of textures to be queried.

textures Specifies an array containing the names of the textures to be queried.

residences Specifies an array in which the texture residence status is returned. The residence status of a texture named by an element of textures is returned in the corresponding element of residences.

5.37.2.8 static unsafe bool
OpenTK.Graphics.OpenGL.GL.AreTexturesResident
(**Int32** *n*, **Int32** * *textures*, [**OutAttribute**] **bool** * *residences*)
[**static**]

Determine if textures are loaded in texture memory.

Parameters

- n* Specifies the number of textures to be queried.
- textures* Specifies an array containing the names of the textures to be queried.
- residences* Specifies an array in which the texture residence status is returned. The residence status of a texture named by an element of textures is returned in the corresponding element of residences.

5.37.2.9 static bool OpenTK.Graphics.OpenGL.GL.AreTexturesResident (
Int32 *n*, **UInt32**[] *textures*, [**OutAttribute**] **bool**[] *residences*)
[**static**]

Determine if textures are loaded in texture memory.

Parameters

- n* Specifies the number of textures to be queried.
- textures* Specifies an array containing the names of the textures to be queried.
- residences* Specifies an array in which the texture residence status is returned. The residence status of a texture named by an element of textures is returned in the corresponding element of residences.

5.37.2.10 static void OpenTK.Graphics.OpenGL.GL.ArrayElement (**Int32** *i*)
[**static**]

Render a vertex using the specified vertex array element.

Parameters

- i* Specifies an index into the enabled vertex data arrays.

5.37.2.11 static void OpenTK.Graphics.OpenGL.GL.AttachShader (**Int32**
program, **Int32** *shader*) [**static**]

Attaches a shader object to a program object.

Parameters

program Specifies the program object to which a shader object will be attached.

shader Specifies the shader object that is to be attached.

5.37.2.12 `static void OpenTK.Graphics.OpenGL.GL.AttachShader (UInt32
program, UInt32 shader) [static]`

Attaches a shader object to a program object.

Parameters

program Specifies the program object to which a shader object will be attached.

shader Specifies the shader object that is to be attached.

5.37.2.13 `static void OpenTK.Graphics.OpenGL.GL.Begin (
OpenTK.Graphics.OpenGL.BeginMode mode) [static]`

Delimit the vertices of a primitive or a group of like primitives.

Parameters

mode Specifies the primitive or primitives that will be created from vertices presented between glBegin and the subsequent glEnd. Ten symbolic constants are accepted: GL_POINTS, GL_LINES, GL_LINE_STRIP, GL_LINE_LOOP, GL_TRIANGLES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_QUADS, GL_QUAD_STRIP, and GL_POLYGON.

5.37.2.14 `static void OpenTK.Graphics.OpenGL.GL.BeginQuery (
OpenTK.Graphics.OpenGL.QueryTarget target, Int32 id)
[static]`

Delimit the boundaries of a query object.

Parameters

target Specifies the target type of query object established between glBeginQuery and the subsequent glEndQuery. The symbolic constant must be GL_SAMPLES_PASSED.

id Specifies the name of a query object.

5.37.2.15 `static void OpenTK.Graphics.OpenGL.GL.BeginQuery (`
`OpenTK.Graphics.OpenGL.QueryTarget target, UInt32 id)`
`[static]`

Delimit the boundaries of a query object.

Parameters

target Specifies the target type of query object established between `glBeginQuery` and the subsequent `glEndQuery`. The symbolic constant must be `GL_SAMPLES_PASSED`.

id Specifies the name of a query object.

5.37.2.16 `static void OpenTK.Graphics.OpenGL.GL.BindAttribLocation (`
`Int32 program, Int32 index, String name) [static]`

Associates a generic vertex attribute index with a named attribute variable.

Parameters

program Specifies the handle of the program object in which the association is to be made.

index Specifies the index of the generic vertex attribute to be bound.

name Specifies a null terminated string containing the name of the vertex shader attribute variable to which index is to be bound.

5.37.2.17 `static void OpenTK.Graphics.OpenGL.GL.BindAttribLocation (`
`UInt32 program, UInt32 index, String name) [static]`

Associates a generic vertex attribute index with a named attribute variable.

Parameters

program Specifies the handle of the program object in which the association is to be made.

index Specifies the index of the generic vertex attribute to be bound.

name Specifies a null terminated string containing the name of the vertex shader attribute variable to which index is to be bound.

5.37.2.18 `static void OpenTK.Graphics.OpenGL.GL.BindBuffer (`
`OpenTK.Graphics.OpenGL.BufferTarget target, Int32 buffer)`
`[static]`

Bind a named buffer object.

Parameters

target Specifies the target to which the buffer object is bound. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

buffer Specifies the name of a buffer object.

5.37.2.19 `static void OpenTK.Graphics.OpenGL.GL.BindBuffer (`
`OpenTK.Graphics.OpenGL.BufferTarget target, UInt32 buffer)`
`[static]`

Bind a named buffer object.

Parameters

target Specifies the target to which the buffer object is bound. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

buffer Specifies the name of a buffer object.

5.37.2.20 `static void OpenTK.Graphics.OpenGL.GL.BindTexture (`
`OpenTK.Graphics.OpenGL.TextureTarget target, Int32 texture)`
`[static]`

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.37.2.21 `static void OpenTK.Graphics.OpenGL.GL.BindTexture (OpenTK.Graphics.OpenGL.TextureTarget target, UInt32 texture) [static]`

Bind a named texture to a texturing target.

Parameters

target Specifies the target to which the texture is bound. Must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

texture Specifies the name of a texture.

5.37.2.22 `static void OpenTK.Graphics.OpenGL.GL.Bitmap (Int32 width, Int32 height, Single xorig, Single yorig, Single xmove, Single ymove, Byte[] bitmap) [static]`

Draw a bitmap.

Parameters

width Specify the pixel width and height of the bitmap image.

xorig Specify the location of the origin in the bitmap image. The origin is measured from the lower left corner of the bitmap, with right and up being the positive axes.

xmove Specify the x and y offsets to be added to the current raster position after the bitmap is drawn.

bitmap Specifies the address of the bitmap image.

5.37.2.23 `static void OpenTK.Graphics.OpenGL.GL.Bitmap (Int32 width, Int32 height, Single xorig, Single yorig, Single xmove, Single ymove, ref Byte bitmap) [static]`

Draw a bitmap.

Parameters

width Specify the pixel width and height of the bitmap image.

xorig Specify the location of the origin in the bitmap image. The origin is measured from the lower left corner of the bitmap, with right and up being the positive axes.

xmove Specify the x and y offsets to be added to the current raster position after the bitmap is drawn.

bitmap Specifies the address of the bitmap image.

5.37.2.24 `static unsafe void OpenTK.Graphics.OpenGL.GL.Bitmap (Int32 width, Int32 height, Single xorig, Single yorig, Single xmove, Single ymove, Byte * bitmap) [static]`

Draw a bitmap.

Parameters

width Specify the pixel width and height of the bitmap image.

xorig Specify the location of the origin in the bitmap image. The origin is measured from the lower left corner of the bitmap, with right and up being the positive axes.

xmove Specify the x and y offsets to be added to the current raster position after the bitmap is drawn.

bitmap Specifies the address of the bitmap image.

5.37.2.25 `static void OpenTK.Graphics.OpenGL.GL.BlendColor (Single red, Single green, Single blue, Single alpha) [static]`

Set the blend color.

Parameters

red specify the components of GL_BLEND_COLOR

5.37.2.26 `static void OpenTK.Graphics.OpenGL.GL.BlendEquation (OpenTK.Graphics.OpenGL.BlendEquationMode mode) [static]`

Specify the equation used for both the RGB blend equation and the Alpha blend equation.

Parameters

mode specifies how source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.37.2.27 `static void OpenTK.Graphics.OpenGL.GL.BlendEquation (Int32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend mode) [static]`

Specify the equation used for both the RGB blend equation and the Alpha blend equation.

Parameters

mode specifies how source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.37.2.28 static void OpenTK.Graphics.OpenGL.GL.BlendEquation (UInt32 *buf*, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend *mode*) [static]

Specify the equation used for both the RGB blend equation and the Alpha blend equation.

Parameters

mode specifies how source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.37.2.29 static void OpenTK.Graphics.OpenGL.GL.BlendEquationSeparate (OpenTK.Graphics.OpenGL.BlendEquationMode *modeRGB*, OpenTK.Graphics.OpenGL.BlendEquationMode *modeAlpha*) [static]

Set the RGB blend equation and the alpha blend equation separately.

Parameters

modeRGB specifies the RGB blend equation, how the red, green, and blue components of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

modeAlpha specifies the alpha blend equation, how the alpha component of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.37.2.30 static void OpenTK.Graphics.OpenGL.GL.BlendEquationSeparate (Int32 *buf*, OpenTK.Graphics.OpenGL.BlendEquationMode *modeRGB*, OpenTK.Graphics.OpenGL.BlendEquationMode *modeAlpha*) [static]

Set the RGB blend equation and the alpha blend equation separately.

Parameters

modeRGB specifies the RGB blend equation, how the red, green, and blue components of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

modeAlpha specifies the alpha blend equation, how the alpha component of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.37.2.31 `static void OpenTK.Graphics.OpenGL.GL.BlendEquationSeparate (UInt32 buf, OpenTK.Graphics.OpenGL.BlendEquationMode modeRGB, OpenTK.Graphics.OpenGL.BlendEquationMode modeAlpha) [static]`

Set the RGB blend equation and the alpha blend equation separately.

Parameters

modeRGB specifies the RGB blend equation, how the red, green, and blue components of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

modeAlpha specifies the alpha blend equation, how the alpha component of the source and destination colors are combined. It must be GL_FUNC_ADD, GL_FUNC_SUBTRACT, GL_FUNC_REVERSE_SUBTRACT, GL_MIN, GL_MAX.

5.37.2.32 `static void OpenTK.Graphics.OpenGL.GL.BlendFunc (OpenTK.Graphics.OpenGL.BlendingFactorSrc sfactor, OpenTK.Graphics.OpenGL.BlendingFactorDest dfactor) [static]`

Specify pixel arithmetic.

Parameters

sfactor Specifies how the red, green, blue, and alpha source blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA,

GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dfactor Specifies how the red, green, blue, and alpha destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

5.37.2.33 static void OpenTK.Graphics.OpenGL.GL.BlendFunc (Int32 *buf*, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend *src*, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend *dst*) [static]

Specify pixel arithmetic.

Parameters

sfactor Specifies how the red, green, blue, and alpha source blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dfactor Specifies how the red, green, blue, and alpha destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

5.37.2.34 `static void OpenTK.Graphics.OpenGL.GL.BlendFunc (UInt32
buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend src,
OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dst)
[static]`

Specify pixel arithmetic.

Parameters

sfactor Specifies how the red, green, blue, and alpha source blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dfactor Specifies how the red, green, blue, and alpha destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

5.37.2.35 `static void OpenTK.Graphics.OpenGL.GL.BlendFuncSeparate
(OpenTK.Graphics.OpenGL.BlendingFactorSrc sfactorRGB,
OpenTK.Graphics.OpenGL.BlendingFactorDest dfactorRGB,
OpenTK.Graphics.OpenGL.BlendingFactorSrc sfactorAlpha,
OpenTK.Graphics.OpenGL.BlendingFactorDest dfactorAlpha)
[static]`

Specify pixel arithmetic for RGB and alpha components separately.

Parameters

srcRGB Specifies how the red, green, and blue blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE.

CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dstRGB Specifies how the red, green, and blue destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

srcAlpha Specified how the alpha source blending factor is computed. The same symbolic constants are accepted as for srcRGB. The initial value is GL_ONE.

dstAlpha Specified how the alpha destination blending factor is computed. The same symbolic constants are accepted as for dstRGB. The initial value is GL_ZERO.

5.37.2.36 static void OpenTK.Graphics.OpenGL.GL.BlendFuncSeparate (Int32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcAlpha, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstAlpha) [static]

Specify pixel arithmetic for RGB and alpha components separately.

Parameters

srcRGB Specifies how the red, green, and blue blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA, GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, GL_ONE_MINUS_CONSTANT_ALPHA, and GL_SRC_ALPHA_SATURATE. The initial value is GL_ONE.

dstRGB Specifies how the red, green, and blue destination blending factors are computed. The following symbolic constants are accepted: GL_ZERO, GL_ONE, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_DST_COLOR, GL_ONE_MINUS_DST_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, GL_DST_ALPHA, GL_ONE_MINUS_DST_ALPHA. GL_CONSTANT_COLOR, GL_ONE_MINUS_CONSTANT_COLOR, GL_CONSTANT_ALPHA, and GL_ONE_MINUS_CONSTANT_ALPHA. The initial value is GL_ZERO.

srcAlpha Specified how the alpha source blending factor is computed. The same symbolic constants are accepted as for *srcRGB*. The initial value is *GL_ONE*.

dstAlpha Specified how the alpha destination blending factor is computed. The same symbolic constants are accepted as for *dstRGB*. The initial value is *GL_ZERO*.

5.37.2.37 `static void OpenTK.Graphics.OpenGL.GL.BlendFuncSeparate (UInt32 buf, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstRGB, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend srcAlpha, OpenTK.Graphics.OpenGL.ArbDrawBuffersBlend dstAlpha) [static]`

Specify pixel arithmetic for RGB and alpha components separately.

Parameters

srcRGB Specifies how the red, green, and blue blending factors are computed. The following symbolic constants are accepted: *GL_ZERO*, *GL_ONE*, *GL_SRC_COLOR*, *GL_ONE_MINUS_SRC_COLOR*, *GL_DST_COLOR*, *GL_ONE_MINUS_DST_COLOR*, *GL_SRC_ALPHA*, *GL_ONE_MINUS_SRC_ALPHA*, *GL_DST_ALPHA*, *GL_ONE_MINUS_DST_ALPHA*, *GL_CONSTANT_COLOR*, *GL_ONE_MINUS_CONSTANT_COLOR*, *GL_CONSTANT_ALPHA*, *GL_ONE_MINUS_CONSTANT_ALPHA*, and *GL_SRC_ALPHA_SATURATE*. The initial value is *GL_ONE*.

dstRGB Specifies how the red, green, and blue destination blending factors are computed. The following symbolic constants are accepted: *GL_ZERO*, *GL_ONE*, *GL_SRC_COLOR*, *GL_ONE_MINUS_SRC_COLOR*, *GL_DST_COLOR*, *GL_ONE_MINUS_DST_COLOR*, *GL_SRC_ALPHA*, *GL_ONE_MINUS_SRC_ALPHA*, *GL_DST_ALPHA*, *GL_ONE_MINUS_DST_ALPHA*, *GL_CONSTANT_COLOR*, *GL_ONE_MINUS_CONSTANT_COLOR*, *GL_CONSTANT_ALPHA*, and *GL_ONE_MINUS_CONSTANT_ALPHA*. The initial value is *GL_ZERO*.

srcAlpha Specified how the alpha source blending factor is computed. The same symbolic constants are accepted as for *srcRGB*. The initial value is *GL_ONE*.

dstAlpha Specified how the alpha destination blending factor is computed. The same symbolic constants are accepted as for *dstRGB*. The initial value is *GL_ZERO*.

5.37.2.38 `static void OpenTK.Graphics.OpenGL.GL.BufferData (`
`OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size,`
`IntPtr data, OpenTK.Graphics.OpenGL.BufferUsageHint usage)`
`[static]`

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

5.37.2.39 `static void OpenTK.Graphics.OpenGL.GL.BufferData<`
`T2 > (OpenTK.Graphics.OpenGL.BufferTarget target,`
`IntPtr size, [InAttribute, OutAttribute] T2[] data,`
`OpenTK.Graphics.OpenGL.BufferUsageHint usage) [static]`

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : *struct*

```

5.37.2.40 static void OpenTK.Graphics.OpenGL.GL.BufferData<
           T2 > ( OpenTK.Graphics.OpenGL.BufferTarget target,
                 IntPtr size, [InAttribute, OutAttribute] T2 data[,],
                 OpenTK.Graphics.OpenGL.BufferUsageHint usage ) [static]

```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints

T2 : struct

```

5.37.2.41 static void OpenTK.Graphics.OpenGL.GL.BufferData<
           T2 > ( OpenTK.Graphics.OpenGL.BufferTarget target,
                 IntPtr size, [InAttribute, OutAttribute] T2 data[,],
                 OpenTK.Graphics.OpenGL.BufferUsageHint usage ) [static]

```

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints*T2 : struct*

5.37.2.42 `static void OpenTK.Graphics.OpenGL.GL.BufferData< T2 > (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr size, [InAttribute, OutAttribute] ref T2 data, OpenTK.Graphics.OpenGL.BufferUsageHint usage) [static]`

Creates and initializes a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

size Specifies the size in bytes of the buffer object's new data store.

data Specifies a pointer to data that will be copied into the data store for initialization, or NULL if no data is to be copied.

usage Specifies the expected usage pattern of the data store. The symbolic constant must be GL_STREAM_DRAW, GL_STREAM_READ, GL_STREAM_COPY, GL_STATIC_DRAW, GL_STATIC_READ, GL_STATIC_COPY, GL_DYNAMIC_DRAW, GL_DYNAMIC_READ, or GL_DYNAMIC_COPY.

Type Constraints*T2 : struct*

5.37.2.43 `static void OpenTK.Graphics.OpenGL.GL.BufferSubData (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset, IntPtr size, IntPtr data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

5.37.2.44 `static void OpenTK.Graphics.OpenGL.GL.BufferSubData< T3 >
(OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
IntPtr size, [InAttribute, OutAttribute] T3[] data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.37.2.45 `static void OpenTK.Graphics.OpenGL.GL.BufferSubData< T3 >
(OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
IntPtr size, [InAttribute, OutAttribute] T3 data,) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.37.2.46 `static void OpenTK.Graphics.OpenGL.GL.BufferSubData< T3 >
 (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
 IntPtr size, [InAttribute, OutAttribute] T3 data[,,]) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.37.2.47 `static void OpenTK.Graphics.OpenGL.GL.BufferSubData< T3 >
 (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
 IntPtr size, [InAttribute, OutAttribute] ref T3 data) [static]`

Updates a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store where data replacement will begin, measured in bytes.

size Specifies the size in bytes of the data store region being replaced.

data Specifies a pointer to the new data that will be copied into the data store.

Type Constraints

T3 : *struct*

5.37.2.48 `static void OpenTK.Graphics.OpenGL.GL.CallList (Int32 list)`
`[static]`

Execute a display list.

Parameters

list Specifies the integer name of the display list to be executed.

5.37.2.49 `static void OpenTK.Graphics.OpenGL.GL.CallList (UInt32 list)`
`[static]`

Execute a display list.

Parameters

list Specifies the integer name of the display list to be executed.

5.37.2.50 `static void OpenTK.Graphics.OpenGL.GL.CallLists (Int32 n,`
`OpenTK.Graphics.OpenGL.ListNameType type, IntPtr lists)`
`[static]`

Execute a list of display lists.

Parameters

n Specifies the number of display lists to be executed.

type Specifies the type of values in lists. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, GL_2_BYTES, GL_3_BYTES, and GL_4_BYTES are accepted.

lists Specifies the address of an array of name offsets in the display list. The pointer type is void because the offsets can be bytes, shorts, ints, or floats, depending on the value of type.

5.37.2.51 `static void OpenTK.Graphics.OpenGL.GL.CallLists< T2 > (Int32`
`n, OpenTK.Graphics.OpenGL.ListNameType type, [InAttribute,`
`OutAttribute] T2[] lists) [static]`

Execute a list of display lists.

Parameters

- n* Specifies the number of display lists to be executed.
- type* Specifies the type of values in lists. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, GL_2_BYTES, GL_3_BYTES, and GL_4_BYTES are accepted.
- lists* Specifies the address of an array of name offsets in the display list. The pointer type is void because the offsets can be bytes, shorts, ints, or floats, depending on the value of type.

Type Constraints

T2 : *struct*

5.37.2.52 `static void OpenTK.Graphics.OpenGL.GL.CallLists< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type, [InAttribute, OutAttribute] T2 lists[,]) [static]`

Execute a list of display lists.

Parameters

- n* Specifies the number of display lists to be executed.
- type* Specifies the type of values in lists. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, GL_2_BYTES, GL_3_BYTES, and GL_4_BYTES are accepted.
- lists* Specifies the address of an array of name offsets in the display list. The pointer type is void because the offsets can be bytes, shorts, ints, or floats, depending on the value of type.

Type Constraints

T2 : *struct*

5.37.2.53 `static void OpenTK.Graphics.OpenGL.GL.CallLists< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type, [InAttribute, OutAttribute] T2 lists[,]) [static]`

Execute a list of display lists.

Parameters

- n* Specifies the number of display lists to be executed.

type Specifies the type of values in lists. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, GL_2_BYTES, GL_3_BYTES, and GL_4_BYTES are accepted.

lists Specifies the address of an array of name offsets in the display list. The pointer type is void because the offsets can be bytes, shorts, ints, or floats, depending on the value of type.

Type Constraints

T2 : struct

5.37.2.54 `static void OpenTK.Graphics.OpenGL.GL.CallLists< T2 > (Int32 n, OpenTK.Graphics.OpenGL.ListNameType type, [InAttribute, OutAttribute] ref T2 lists) [static]`

Execute a list of display lists.

Parameters

n Specifies the number of display lists to be executed.

type Specifies the type of values in lists. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, GL_2_BYTES, GL_3_BYTES, and GL_4_BYTES are accepted.

lists Specifies the address of an array of name offsets in the display list. The pointer type is void because the offsets can be bytes, shorts, ints, or floats, depending on the value of type.

Type Constraints

T2 : struct

5.37.2.55 `static void OpenTK.Graphics.OpenGL.GL.Clear (OpenTK.Graphics.OpenGL.ClearBufferMask mask) [static]`

Clear buffers to preset values.

Parameters

mask Bitwise OR of masks that indicate the buffers to be cleared. The four masks are GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT, GL_ACCUM_BUFFER_BIT, and GL_STENCIL_BUFFER_BIT.

5.37.2.56 `static void OpenTK.Graphics.OpenGL.GL.ClearAccum (Single red,
Single green, Single blue, Single alpha) [static]`

Specify clear values for the accumulation buffer.

Parameters

red Specify the red, green, blue, and alpha values used when the accumulation buffer is cleared. The initial values are all 0.

5.37.2.57 `static void OpenTK.Graphics.OpenGL.GL.ClearColor (Single red,
Single green, Single blue, Single alpha) [static]`

Specify clear values for the color buffers.

Parameters

red Specify the red, green, blue, and alpha values used when the color buffers are cleared. The initial values are all 0.

5.37.2.58 `static void OpenTK.Graphics.OpenGL.GL.ClearDepth (Double
depth) [static]`

Specify the clear value for the depth buffer.

Parameters

depth Specifies the depth value used when the depth buffer is cleared. The initial value is 1.

5.37.2.59 `static void OpenTK.Graphics.OpenGL.GL.ClearIndex (Single c)
[static]`

Specify the clear value for the color index buffers.

Parameters

c Specifies the index used when the color index buffers are cleared. The initial value is 0.

**5.37.2.60 static void OpenTK.Graphics.OpenGL.GL.ClearStencil (Int32 *s*)
[static]**

Specify the clear value for the stencil buffer.

Parameters

- s* Specifies the index used when the stencil buffer is cleared. The initial value is 0.

5.37.2.61 static void OpenTK.Graphics.OpenGL.GL.ClientActiveTexture (OpenTK.Graphics.OpenGL.TextureUnit *texture*) [static]

Select active texture unit.

Parameters

- texture* Specifies which texture unit to make active. The number of texture units is implementation dependent, but must be at least two. *texture* must be one of GL_TEXTURE*i*, where *i* ranges from 0 to the value of GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value. The initial value is GL_TEXTURE0.

5.37.2.62 static void OpenTK.Graphics.OpenGL.GL.ClipPlane (OpenTK.Graphics.OpenGL.ClipPlaneName *plane*, Double[] *equation*) [static]

Specify a plane against which all geometry is clipped.

Parameters

- plane* Specifies which clipping plane is being positioned. Symbolic names of the form GL_CLIP_PLANE*i*, where *i* is an integer between 0 and GL_MAX_CLIP_PLANES - 1, are accepted.
- equation* Specifies the address of an array of four double-precision floating-point values. These values are interpreted as a plane equation.

5.37.2.63 static void OpenTK.Graphics.OpenGL.GL.ClipPlane (OpenTK.Graphics.OpenGL.ClipPlaneName *plane*, ref Double *equation*) [static]

Specify a plane against which all geometry is clipped.

Parameters

plane Specifies which clipping plane is being positioned. Symbolic names of the form GL_CLIP_PLANEi, where i is an integer between 0 and GL_MAX_CLIP_PLANES - 1, are accepted.

equation Specifies the address of an array of four double-precision floating-point values. These values are interpreted as a plane equation.

5.37.2.64 static unsafe void OpenTK.Graphics.OpenGL.GL.ClipPlane (OpenTK.Graphics.OpenGL.ClipPlaneName *plane*, Double * *equation*) [static]

Specify a plane against which all geometry is clipped.

Parameters

plane Specifies which clipping plane is being positioned. Symbolic names of the form GL_CLIP_PLANEi, where i is an integer between 0 and GL_MAX_CLIP_PLANES - 1, are accepted.

equation Specifies the address of an array of four double-precision floating-point values. These values are interpreted as a plane equation.

5.37.2.65 static void OpenTK.Graphics.OpenGL.GL.Color3 (SByte *red*, SByte *green*, SByte *blue*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.66 static void OpenTK.Graphics.OpenGL.GL.Color3 (SByte[] *v*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.67 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref SByte *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.68 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (SByte * *v*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.69 static void OpenTK.Graphics.OpenGL.GL.Color3 (Double *red*,
Double *green*, Double *blue*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.70 static void OpenTK.Graphics.OpenGL.GL.Color3 (Double[] *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.71 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref Double *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.72 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (Double *
v) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.73 static void OpenTK.Graphics.OpenGL.GL.Color3 (Single *red*,
Single *green*, Single *blue*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.74 static void OpenTK.Graphics.OpenGL.GL.Color3 (Single[] *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.75 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref Single *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.76 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (Single *
v) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.77 static void OpenTK.Graphics.OpenGL.GL.Color3 (Int32 *red*, Int32
green, Int32 *blue*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.78 static void OpenTK.Graphics.OpenGL.GL.Color3 (Int32[] *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.79 **static void OpenTK.Graphics.OpenGL.GL.Color3 (ref Int32 *v*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.80 **static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (Int32 * *v*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.81 **static void OpenTK.Graphics.OpenGL.GL.Color3 (Int16 *red*, Int16 *green*, Int16 *blue*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.82 **static void OpenTK.Graphics.OpenGL.GL.Color3 (Int16[] *v*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.83 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref Int16 *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.84 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (Int16 * *v*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.85 static void OpenTK.Graphics.OpenGL.GL.Color3 (Byte *red*, Byte *green*, Byte *blue*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.86 static void OpenTK.Graphics.OpenGL.GL.Color3 (Byte[] *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.87 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref Byte *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.88 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (Byte * *v*) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.89 static void OpenTK.Graphics.OpenGL.GL.Color3 (UInt32 *red*,
UInt32 *green*, UInt32 *blue*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.90 static void OpenTK.Graphics.OpenGL.GL.Color3 (UInt32[] *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.91 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref UInt32 *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.92 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (UInt32 *
v) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.93 static void OpenTK.Graphics.OpenGL.GL.Color3 (UInt16 *red*,
UInt16 *green*, UInt16 *blue*) [static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.94 static void OpenTK.Graphics.OpenGL.GL.Color3 (UInt16[] *v*)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.95 static void OpenTK.Graphics.OpenGL.GL.Color3 (ref UInt16 v)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.96 static unsafe void OpenTK.Graphics.OpenGL.GL.Color3 (UInt16 * v) [static]

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.97 static void OpenTK.Graphics.OpenGL.GL.Color4 (SByte[] v)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

**5.37.2.98 static void OpenTK.Graphics.OpenGL.GL.Color4 (ref SByte v)
[static]**

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.99 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (SByte * v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.100 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Double red, Double green, Double blue, Double alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.101 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Double[] v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.102 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref Double v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.103 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (Double * v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.104 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Single red, Single green, Single blue, Single alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.105 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Single[] v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.106 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref Single v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.107 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (Single *
v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.108 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Int32 red,
Int32 green, Int32 blue, Int32 alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.109 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Int32[] v)
[static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.110 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref Int32 v)
[static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.111 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (Int32 *
v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.112 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Int16 red,
Int16 green, Int16 blue, Int16 alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.113 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Int16[] v)
[static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.114 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref Int16 v)
[static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.115 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (Int16 *
v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.116 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Byte red, Byte
green, Byte blue, Byte alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.117 `static void OpenTK.Graphics.OpenGL.GL.Color4 (Byte[] v)
[static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.118 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref Byte v)
[static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.119 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (Byte * v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.120 `static void OpenTK.Graphics.OpenGL.GL.Color4 (UInt32 red, UInt32 green, UInt32 blue, UInt32 alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.121 `static void OpenTK.Graphics.OpenGL.GL.Color4 (UInt32[] v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.122 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref UInt32 v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.123 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (UInt32 * v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.124 `static void OpenTK.Graphics.OpenGL.GL.Color4 (UInt16 red, UInt16 green, UInt16 blue, UInt16 alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.125 `static void OpenTK.Graphics.OpenGL.GL.Color4 (UInt16[] v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.126 `static void OpenTK.Graphics.OpenGL.GL.Color4 (ref UInt16 v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.127 `static unsafe void OpenTK.Graphics.OpenGL.GL.Color4 (UInt16 * v) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.128 `static void OpenTK.Graphics.OpenGL.GL.Color4 (SByte red, SByte green, SByte blue, SByte alpha) [static]`

Set the current color.

Parameters

red Specify new red, green, and blue values for the current color.

alpha Specifies a new alpha value for the current color. Included only in the four-argument glColor4 commands.

5.37.2.129 `static void OpenTK.Graphics.OpenGL.GL.ColorMask (bool red, bool green, bool blue, bool alpha) [static]`

Enable and disable writing of frame buffer color components.

Parameters

red Specify whether red, green, blue, and alpha can or cannot be written into the frame buffer. The initial values are all GL_TRUE, indicating that the color components can be written.

5.37.2.130 `static void OpenTK.Graphics.OpenGL.GL.ColorMask (Int32 index, bool r, bool g, bool b, bool a) [static]`

Enable and disable writing of frame buffer color components.

Parameters

red Specify whether red, green, blue, and alpha can or cannot be written into the frame buffer. The initial values are all GL_TRUE, indicating that the color components can be written.

5.37.2.131 `static void OpenTK.Graphics.OpenGL.GL.ColorMask (UInt32
index, bool r, bool g, bool b, bool a) [static]`

Enable and disable writing of frame buffer color components.

Parameters

red Specify whether red, green, blue, and alpha can or cannot be written into the frame buffer. The initial values are all GL_TRUE, indicating that the color components can be written.

5.37.2.132 `static void OpenTK.Graphics.OpenGL.GL.ColorMaterial
(OpenTK.Graphics.OpenGL.MaterialFace face,
OpenTK.Graphics.OpenGL.ColorMaterialParameter mode)
[static]`

Cause a material color to track the current color.

Parameters

face Specifies whether front, back, or both front and back material parameters should track the current color. Accepted values are GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK. The initial value is GL_FRONT_AND_BACK.

mode Specifies which of several material parameters track the current color. Accepted values are GL_EMISSION, GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, and GL_AMBIENT_AND_DIFFUSE. The initial value is GL_AMBIENT_AND_DIFFUSE.

5.37.2.133 `static void OpenTK.Graphics.OpenGL.GL.ColorPointer (Int32
size, OpenTK.Graphics.OpenGL.ColorPointerType type, Int32
stride, IntPtr pointer) [static]`

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

```
5.37.2.134 static void OpenTK.Graphics.OpenGL.GL.ColorPointer< T3 >
( Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer )
[static]
```

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.135 static void OpenTK.Graphics.OpenGL.GL.ColorPointer< T3 >
( Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,] )
[static]
```

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : *struct*

```
5.37.2.136 static void OpenTK.Graphics.OpenGL.GL.ColorPointer< T3 >
( Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,])
[static]
```

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : *struct*

```
5.37.2.137 static void OpenTK.Graphics.OpenGL.GL.ColorPointer< T3 >
( Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer )
[static]
```

Define an array of colors.

Parameters

size Specifies the number of components per color. Must be 3 or 4. The initial value is 4.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.37.2.138 `static void OpenTK.Graphics.OpenGL.GL.ColorSubTable (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr data) [static]`

Respecify a portion of a color table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

start The starting index of the portion of the color table to be replaced.

count The number of table entries to replace.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to replace the specified region of the color table.

5.37.2.139 `static void OpenTK.Graphics.OpenGL.GL.ColorSubTable< T5
> (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32
start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T5[] data) [static]`

Respecify a portion of a color table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

start The starting index of the portion of the color table to be replaced.

count The number of table entries to replace.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to replace the specified region of the color table.

Type Constraints

T5 : *struct*

5.37.2.140 `static void OpenTK.Graphics.OpenGL.GL.ColorSubTable< T5
> (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32
start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T5 data[,]) [static]`

Respecify a portion of a color table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

start The starting index of the portion of the color table to be replaced.

count The number of table entries to replace.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to replace the specified region of the color table.

Type Constraints

T5 : struct

```
5.37.2.141 static void OpenTK.Graphics.OpenGL.GL.ColorSubTable< T5
> ( OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32
start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T5 data[,], ) [static]
```

Respecify a portion of a color table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

start The starting index of the portion of the color table to be replaced.

count The number of table entries to replace.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT,

GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to replace the specified region of the color table.

Type Constraints

T5 : *struct*

5.37.2.142 `static void OpenTK.Graphics.OpenGL.GL.ColorSubTable< T5
> (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32
start, Int32 count, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] ref T5 data) [static]`

Respecify a portion of a color table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

start The starting index of the portion of the color table to be replaced.

count The number of table entries to replace.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to replace the specified region of the color table.

Type Constraints

T5 : struct

```
5.37.2.143 static void OpenTK.Graphics.OpenGL.GL.ColorTable
( OpenTK.Graphics.OpenGL.ColorTableTarget target,
  OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
  Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
  OpenTK.Graphics.OpenGL.PixelType type, IntPtr table )
[static]
```

Define a color lookup table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

internalformat The internal format of the color table. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, and GL_RGBA16.

width The number of entries in the color lookup table specified by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV,

GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to build the color table.

```
5.37.2.144 static void OpenTK.Graphics.OpenGL.GL.ColorTable< T5
    > ( OpenTK.Graphics.OpenGL.ColorTableTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T5[] table ) [static]
```

Define a color lookup table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

internalformat The internal format of the color table. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, and GL_RGBA16.

width The number of entries in the color lookup table specified by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_

SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to build the color table.

Type Constraints

T5 : *struct*

```
5.37.2.145 static void OpenTK.Graphics.OpenGL.GL.ColorTable< T5
> ( OpenTK.Graphics.OpenGL.ColorTableTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T5 table[, ] ) [static]
```

Define a color lookup table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

internalformat The internal format of the color table. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, and GL_RGBA16.

width The number of entries in the color lookup table specified by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to build the color table.

Type Constraints

T5 : struct

```
5.37.2.146 static void OpenTK.Graphics.OpenGL.GL.ColorTable< T5
> ( OpenTK.Graphics.OpenGL.ColorTableTarget target,
  OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
  Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
  OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
  OutAttribute] T5 table[,]) [static]
```

Define a color lookup table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

internalformat The internal format of the color table. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, and GL_RGBA16.

width The number of entries in the color lookup table specified by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to build the color table.

Type Constraints

T5 : struct

```
5.37.2.147 static void OpenTK.Graphics.OpenGL.GL.ColorTable< T5
> ( OpenTK.Graphics.OpenGL.ColorTableTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] ref T5 table ) [static]
```

Define a color lookup table.

Parameters

target Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

internalformat The internal format of the color table. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12,

GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, and GL_RGBA16.

width The number of entries in the color lookup table specified by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in data. The allowable values are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

data Pointer to a one-dimensional array of pixel data that is processed to build the color table.

Type Constraints

T5 : struct

```
5.37.2.148 static void OpenTK.Graphics.OpenGL.GL.ColorTableParameter
( OpenTK.Graphics.OpenGL.ColorTableTarget target,
  OpenTK.Graphics.OpenGL.ColorTableParameterPName pname,
  Single @[] params ) [static]
```

Set color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a texture color lookup table parameter. Must be one of GL_COLOR_TABLE_SCALE or GL_COLOR_TABLE_BIAS.

params A pointer to an array where the values of the parameters are stored.

5.37.2.149 `static void OpenTK.Graphics.OpenGL.GL.ColorTableParameter
(OpenTK.Graphics.OpenGL.ColorTableTarget target,
OpenTK.Graphics.OpenGL.ColorTableParameterPName pname,
ref Single @ params) [static]`

Set color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a texture color lookup table parameter. Must be one of GL_COLOR_TABLE_SCALE or GL_COLOR_TABLE_BIAS.

params A pointer to an array where the values of the parameters are stored.

5.37.2.150 `static unsafe void
OpenTK.Graphics.OpenGL.GL.ColorTableParameter
(OpenTK.Graphics.OpenGL.ColorTableTarget target,
OpenTK.Graphics.OpenGL.ColorTableParameterPName pname,
Single *@ params) [static]`

Set color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a texture color lookup table parameter. Must be one of GL_COLOR_TABLE_SCALE or GL_COLOR_TABLE_BIAS.

params A pointer to an array where the values of the parameters are stored.

5.37.2.151 `static void OpenTK.Graphics.OpenGL.GL.ColorTableParameter
(OpenTK.Graphics.OpenGL.ColorTableTarget target,
OpenTK.Graphics.OpenGL.ColorTableParameterPName pname,
Int32 @[] params) [static]`

Set color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a texture color lookup table parameter. Must be one of GL_COLOR_TABLE_SCALE or GL_COLOR_TABLE_BIAS.

params A pointer to an array where the values of the parameters are stored.

5.37.2.152 `static void OpenTK.Graphics.OpenGL.GL.ColorTableParameter (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, ref Int32 @ params) [static]`

Set color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a texture color lookup table parameter. Must be one of GL_COLOR_TABLE_SCALE or GL_COLOR_TABLE_BIAS.

params A pointer to an array where the values of the parameters are stored.

5.37.2.153 `static unsafe void OpenTK.Graphics.OpenGL.GL.ColorTableParameter (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.ColorTableParameterPName pname, Int32 *@ params) [static]`

Set color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a texture color lookup table parameter. Must be one of GL_COLOR_TABLE_SCALE or GL_COLOR_TABLE_BIAS.

params A pointer to an array where the values of the parameters are stored.

5.37.2.154 `static void OpenTK.Graphics.OpenGL.GL.CompileShader (Int32 shader) [static]`

Compiles a shader object.

Parameters

shader Specifies the shader object to be compiled.

5.37.2.155 `static void OpenTK.Graphics.OpenGL.GL.CompileShader (UInt32
shader) [static]`

Compiles a shader object.

Parameters

shader Specifies the shader object to be compiled.

5.37.2.156 `static void OpenTK.Graphics.OpenGL.GL.CompressedTexImage1D
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, Int32 border, Int32 imageSize, IntPtr data)
[static]`

Specify a one-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be 2^{n+2} (*border*) for some integer n. All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.37.2.157 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage1D< T6 > (
 OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
 OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
 Int32 *width*, Int32 *border*, Int32 *imageSize*, [InAttribute,
 OutAttribute] T6[] *data*) [**static**]

Specify a one-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer *n*. All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T6 : struct

5.37.2.158 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage1D< T6 > (
 OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
 OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
 Int32 *width*, Int32 *border*, Int32 *imageSize*, [InAttribute,
 OutAttribute] T6 *data*[,]) [**static**]

Specify a one-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer *n*. All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T6 : struct

```
5.37.2.159 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage1D< T6 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 border, Int32 imageSize, [InAttribute,
    OutAttribute] T6 data[,]) [static]
```

Specify a one-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer *n*. All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T6 : *struct*

5.37.2.160 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage1D< **T6** > (
OpenTK.Graphics.OpenGL.TextureTarget *target*, **Int32** *level*,
OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
Int32 *width*, **Int32** *border*, **Int32** *imageSize*, [InAttribute,
 OutAttribute] ref **T6** *data*) [**static**]

Specify a one-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T6 : *struct*

5.37.2.161 `static void OpenTK.Graphics.OpenGL.GL.CompressedTexImage2D (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, IntPtr data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address *data*.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n} + 2 \cdot (\text{border})$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n} + 2 \cdot (\text{border})$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by *data*.

data Specifies a pointer to the compressed image data in memory.

5.37.2.162 `static void OpenTK.Graphics.OpenGL.GL.CompressedTexImage2D< T7 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 border, Int32 imageSize, [InAttribute, OutAttribute] T7[] data) [static]`

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

```
5.37.2.163 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage2D< T7 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, Int32 border, Int32 imageSize,
    [InAttribute, OutAttribute] T7 data[,]) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address *data*.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer *n*. All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by *data*.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

```
5.37.2.164 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage2D< T7 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, Int32 border, Int32 imageSize,
    [InAttribute, OutAttribute] T7 data[,]) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2}$ (border) for some integer *n*. All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2}$ (border) for some integer *n*. All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : struct

```
5.37.2.165 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage2D< T7 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, Int32 border, Int32 imageSize,
    [InAttribute, OutAttribute] ref T7 data ) [static]
```

Specify a two-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer n . All implementations support 2D texture images that are at least 64 texels wide and cube-mapped texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer n . All implementations support 2D texture images that are at least 64 texels high and cube-mapped texture images that are at least 16 texels high.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T7 : *struct*

5.37.2.166 `static void OpenTK.Graphics.OpenGL.GL.CompressedTexImage3D (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, Int32 imageSize, IntPtr data) [static]`

Specify a three-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be `GL_TEXTURE_3D` or `GL_PROXY_TEXTURE_3D`.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot \text{border}$ for some integer n . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2 \sup n + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2 \sup n + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.37.2.167 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage3D< T8 > (
OpenTK.Graphics.OpenGL.TextureTarget *target*, **Int32** *level*,
OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
Int32 *width*, **Int32** *height*, **Int32** *depth*, **Int32** *border*, **Int32**
imageSize, [**InAttribute**, **OutAttribute**] T8[] *data*) [**static**]

Specify a three-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2 \sup n + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2 \sup n + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2 \sup n + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

```
5.37.2.168 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage3D< T8 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, Int32 depth, Int32 border, Int32
    imageSize, [InAttribute, OutAttribute] T8 data[,]) [static]
```

Specify a three-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{sup n + 2} (border)$ for some integer . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{sup n + 2} (border)$ for some integer . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the [GL](#) version does not support non-power-of-two sizes, this value must be $2^{sup n + 2} (border)$ for some integer . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.37.2.169 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexImage3D< **T8** > (
OpenTK.Graphics.OpenGL.TextureTarget *target*, **Int32** *level*,
OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
Int32 *width*, **Int32** *height*, **Int32** *depth*, **Int32** *border*, **Int32**
imageSize, [**InAttribute**, **OutAttribute**] **T8** *data*[,,]) [**static**]

Specify a three-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer n . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer n . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2} \cdot (\text{border})$ for some integer n . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

```

5.37.2.170 static void
    OpenTK.Graphics.OpenGL.GL.CompressedTexImage3D< T8 > (
        OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
        OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
        Int32 width, Int32 height, Int32 depth, Int32 border, Int32
        imageSize, [InAttribute, OutAttribute] ref T8 data ) [static]

```

Specify a three-dimensional texture image in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalformat Specifies the format of the compressed image data stored at address data.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2} \cdot (\text{border})$ for some integer . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2} \cdot (\text{border})$ for some integer . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2} \cdot (\text{border})$ for some integer . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : struct

5.37.2.171 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage1D
 (**OpenTK.Graphics.OpenGL.TextureTarget**
target, **Int32** *level*, **Int32** *xoffset*, **Int32** *width*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
IntPtr *data*) [**static**]

Specify a one-dimensional texture subimage in a compressed format.

Parameters

- target* Specifies the target texture. Must be GL_TEXTURE_1D.
- level* Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
- xoffset* Specifies a texel offset in the x direction within the texture array.
- width* Specifies the width of the texture subimage.
- format* Specifies the format of the compressed image data stored at address data.
- imageSize* Specifies the number of unsigned bytes of image data starting at the address specified by data.
- data* Specifies a pointer to the compressed image data in memory.

5.37.2.172 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage1D<
T6 > (**OpenTK.Graphics.OpenGL.TextureTarget**
target, **Int32** *level*, **Int32** *xoffset*, **Int32** *width*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] T6[] *data*) [**static**]

Specify a one-dimensional texture subimage in a compressed format.

Parameters

- target* Specifies the target texture. Must be GL_TEXTURE_1D.
- level* Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
- xoffset* Specifies a texel offset in the x direction within the texture array.
- width* Specifies the width of the texture subimage.
- format* Specifies the format of the compressed image data stored at address data.
- imageSize* Specifies the number of unsigned bytes of image data starting at the address specified by data.
- data* Specifies a pointer to the compressed image data in memory.

Type Constraints*T6 : struct*

5.37.2.173 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage1D<
T6 > (**OpenTK.Graphics.OpenGL.TextureTarget**
target, **Int32** *level*, **Int32** *xoffset*, **Int32** *width*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] **T6** *data*[,,]) **[static]**

Specify a one-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.
level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
xoffset Specifies a texel offset in the x direction within the texture array.
width Specifies the width of the texture subimage.
format Specifies the format of the compressed image data stored at address data.
imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.
data Specifies a pointer to the compressed image data in memory.

Type Constraints*T6 : struct*

5.37.2.174 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage1D<
T6 > (**OpenTK.Graphics.OpenGL.TextureTarget**
target, **Int32** *level*, **Int32** *xoffset*, **Int32** *width*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] **T6** *data*[,,]) **[static]**

Specify a one-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.
level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

width Specifies the width of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T6 : *struct*

5.37.2.175 static void

```
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage1D<
T6 > ( OpenTK.Graphics.OpenGL.TextureTarget
target, Int32 level, Int32 xoffset, Int32 width,
OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,
[InAttribute, OutAttribute] ref T6 data ) [static]
```

Specify a one-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

width Specifies the width of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T6 : *struct*

5.37.2.176 static void

```
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage2D
( OpenTK.Graphics.OpenGL.TextureTarget target, Int32
level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height,
OpenTK.Graphics.OpenGL.PixelFormat format, Int32 imageSize,
IntPtr data ) [static]
```

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.37.2.177 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage2D<
T8 > (**OpenTK.Graphics.OpenGL.TextureTarget** *target*, **Int32**
level, **Int32** *xoffset*, **Int32** *yoffset*, **Int32** *width*, **Int32** *height*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] T8[] *data*) [**static**]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.37.2.178 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage2D<
T8 > (**OpenTK.Graphics.OpenGL.TextureTarget** *target*, **Int32**
level, **Int32** *xoffset*, **Int32** *yoffset*, **Int32** *width*, **Int32** *height*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] T8 *data*[,]) [**static**]

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.37.2.179 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage2D<
T8 > (**OpenTK.Graphics.OpenGL.TextureTarget** *target*, **Int32**
level, **Int32** *xoffset*, **Int32** *yoffset*, **Int32** *width*, **Int32** *height*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] **T8** *data*[,]) **[static]**

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.37.2.180 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage2D<
T8 > (**OpenTK.Graphics.OpenGL.TextureTarget** *target*, **Int32**
level, **Int32** *xoffset*, **Int32** *yoffset*, **Int32** *width*, **Int32** *height*,
OpenTK.Graphics.OpenGL.PixelFormat *format*, **Int32** *imageSize*,
[InAttribute, OutAttribute] **ref T8** *data*) **[static]**

Specify a two-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y,

GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T8 : *struct*

5.37.2.181 static void

OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage3D (
OpenTK.Graphics.OpenGL.TextureTarget *target*, **Int32** *level*,
Int32 *xoffset*, **Int32** *yoffset*, **Int32** *zoffset*, **Int32** *width*, **Int32**
height, **Int32** *depth*, **OpenTK.Graphics.OpenGL.PixelFormat**
format, **Int32** *imageSize*, **IntPtr** *data*) [**static**]

Specify a three-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

5.37.2.182 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage3D< T10
 > (**OpenTK.Graphics.OpenGL.TextureTarget** *target*, Int32 *level*,
 Int32 *xoffset*, Int32 *yoffset*, Int32 *zoffset*, Int32 *width*, Int32
height, Int32 *depth*, **OpenTK.Graphics.OpenGL.PixelFormat**
format, Int32 *imageSize*, [InAttribute, OutAttribute] T10[] *data*)
 [static]

Specify a three-dimensional texture subimage in a compressed format.

Parameters

- target* Specifies the target texture. Must be GL_TEXTURE_3D.
- level* Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
- xoffset* Specifies a texel offset in the x direction within the texture array.
- yoffset* Specifies a texel offset in the y direction within the texture array.
- width* Specifies the width of the texture subimage.
- height* Specifies the height of the texture subimage.
- depth* Specifies the depth of the texture subimage.
- format* Specifies the format of the compressed image data stored at address data.
- imageSize* Specifies the number of unsigned bytes of image data starting at the address specified by data.
- data* Specifies a pointer to the compressed image data in memory.

Type Constraints

T10 : struct

5.37.2.183 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage3D< T10
 > (**OpenTK.Graphics.OpenGL.TextureTarget** *target*, Int32 *level*,
 Int32 *xoffset*, Int32 *yoffset*, Int32 *zoffset*, Int32 *width*, Int32
height, Int32 *depth*, **OpenTK.Graphics.OpenGL.PixelFormat**
format, Int32 *imageSize*, [InAttribute, OutAttribute] T10 *data*[,])
 [static]

Specify a three-dimensional texture subimage in a compressed format.

Parameters

- target* Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints

T10 : *struct*

```
5.37.2.184 static void
OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage3D< T10
> ( OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32
height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat
format, Int32 imageSize, [InAttribute, OutAttribute] T10 data[,,] )
[static]
```

Specify a three-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the compressed image data stored at address data.

imageSize Specifies the number of unsigned bytes of image data starting at the address specified by data.

data Specifies a pointer to the compressed image data in memory.

Type Constraints*T10 : struct*

5.37.2.185 static void
 OpenTK.Graphics.OpenGL.GL.CompressedTexSubImage3D< T10
 > (OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
 Int32 *xoffset*, Int32 *yoffset*, Int32 *zoffset*, Int32 *width*, Int32
height, Int32 *depth*, OpenTK.Graphics.OpenGL.PixelFormat
format, Int32 *imageSize*, [InAttribute, OutAttribute] ref T10 *data*
) [static]

Specify a three-dimensional texture subimage in a compressed format.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.
level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.
xoffset Specifies a texel offset in the x direction within the texture array.
yoffset Specifies a texel offset in the y direction within the texture array.
width Specifies the width of the texture subimage.
height Specifies the height of the texture subimage.
depth Specifies the depth of the texture subimage.
format Specifies the format of the compressed image data stored at address *data*.
imageSize Specifies the number of unsigned bytes of image data starting at the address specified by *data*.
data Specifies a pointer to the compressed image data in memory.

Type Constraints*T10 : struct*

5.37.2.186 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter1D
 (OpenTK.Graphics.OpenGL.ConvolutionTarget *target*,
 OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
 Int32 *width*, OpenTK.Graphics.OpenGL.PixelFormat *format*,
 OpenTK.Graphics.OpenGL.PixelType *type*, IntPtr *image*)
 [static]

Define a one-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_1D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_INTENSITY, GL_RGB, and GL_RGBA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a one-dimensional array of pixel data that is processed to build the convolution filter kernel.

```
5.37.2.187 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter1D<
T5 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T5[] image ) [static]
```

Define a one-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_1D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_INTENSITY, GL_RGB, and GL_RGBA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a one-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T5 : struct

```
5.37.2.188 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter1D<
T5 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T5 image[, ] ) [static]
```

Define a one-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_1D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_INTENSITY, GL_RGB, and GL_RGBA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a one-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T5 : struct

```
5.37.2.189 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter1D<
    T5 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T5 image[, ] ) [static]
```

Define a one-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_1D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_INTENSITY, GL_RGB, and GL_RGBA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a one-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T5 : struct

```
5.37.2.190 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter1D<
T5 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] ref T5 image ) [static]
```

Define a one-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_1D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_INTENSITY, GL_RGB, and GL_RGBA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a one-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T5 : struct

```
5.37.2.191 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter2D
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
  Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
  format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr image
) [static]
```

Define a two-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

height The height of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a two-dimensional array of pixel data that is processed to build the convolution filter kernel.

```
5.37.2.192 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter2D<
            T6 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
                  OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
                  Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
                  format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
                  OutAttribute] T6[] image ) [static]
```

Define a two-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

height The height of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a two-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T6 : *struct*

```
5.37.2.193 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter2D<
    T6 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T6 image[,]) [static]
```

Define a two-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

height The height of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a two-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T6 : struct

```
5.37.2.194 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter2D<
T6 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T6 image[,]) [static]
```

Define a two-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

height The height of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a two-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T6 : struct

```
5.37.2.195 static void OpenTK.Graphics.OpenGL.GL.ConvolutionFilter2D<
    T6 > ( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] ref T6 image ) [static]
```

Define a two-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The width of the pixel array referenced by data.

height The height of the pixel array referenced by data.

format The format of the pixel data in data. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Pointer to a two-dimensional array of pixel data that is processed to build the convolution filter kernel.

Type Constraints

T6 : struct

```
5.37.2.196 static void OpenTK.Graphics.OpenGL.GL.ConvolutionParameter
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Single
  @ params ) [static]
```

Set convolution parameters.

Parameters

- target* The target for the convolution parameter. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.
- pname* The parameter to be set. Must be GL_CONVOLUTION_BORDER_MODE.
- params* The parameter value. Must be one of GL_REDUCE, GL_CONSTANT_BORDER, GL_REPLICATE_BORDER.

5.37.2.197 static void OpenTK.Graphics.OpenGL.GL.ConvolutionParameter (OpenTK.Graphics.OpenGL.ConvolutionTarget *target*, OpenTK.Graphics.OpenGL.ConvolutionParameter *pname*, Single @[] *params*) [static]

Set convolution parameters.

Parameters

- target* The target for the convolution parameter. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.
- pname* The parameter to be set. Must be GL_CONVOLUTION_BORDER_MODE.
- params* The parameter value. Must be one of GL_REDUCE, GL_CONSTANT_BORDER, GL_REPLICATE_BORDER.

5.37.2.198 static unsafe void OpenTK.Graphics.OpenGL.GL.ConvolutionParameter (OpenTK.Graphics.OpenGL.ConvolutionTarget *target*, OpenTK.Graphics.OpenGL.ConvolutionParameter *pname*, Single *@ *params*) [static]

Set convolution parameters.

Parameters

- target* The target for the convolution parameter. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.
- pname* The parameter to be set. Must be GL_CONVOLUTION_BORDER_MODE.
- params* The parameter value. Must be one of GL_REDUCE, GL_CONSTANT_BORDER, GL_REPLICATE_BORDER.

5.37.2.199 `static void OpenTK.Graphics.OpenGL.GL.ConvolutionParameter (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Int32 @ params) [static]`

Set convolution parameters.

Parameters

target The target for the convolution parameter. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be set. Must be GL_CONVOLUTION_BORDER_MODE.

params The parameter value. Must be one of GL_REDUCE, GL_CONSTANT_BORDER, GL_REPLICATE_BORDER.

5.37.2.200 `static void OpenTK.Graphics.OpenGL.GL.ConvolutionParameter (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Int32 @[] params) [static]`

Set convolution parameters.

Parameters

target The target for the convolution parameter. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be set. Must be GL_CONVOLUTION_BORDER_MODE.

params The parameter value. Must be one of GL_REDUCE, GL_CONSTANT_BORDER, GL_REPLICATE_BORDER.

5.37.2.201 `static unsafe void OpenTK.Graphics.OpenGL.GL.ConvolutionParameter (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.ConvolutionParameter pname, Int32 *@ params) [static]`

Set convolution parameters.

Parameters

- target* The target for the convolution parameter. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.
- pname* The parameter to be set. Must be GL_CONVOLUTION_BORDER_MODE.
- params* The parameter value. Must be one of GL_REDUCE, GL_CONSTANT_BORDER, GL_REPLICATE_BORDER.

5.37.2.202 `static void OpenTK.Graphics.OpenGL.GL.CopyColorSubTable (OpenTK.Graphics.OpenGL.ColorTableTarget target, Int32 start, Int32 x, Int32 y, Int32 width) [static]`

Respecify a portion of a color table.

Parameters

- target* Must be one of GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.
- start* The starting index of the portion of the color table to be replaced.
- x* The window coordinates of the left corner of the row of pixels to be copied.
- width* The number of table entries to replace.

5.37.2.203 `static void OpenTK.Graphics.OpenGL.GL.CopyColorTable (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 x, Int32 y, Int32 width) [static]`

Copy pixels into a color table.

Parameters

- target* The color table target. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.
- internalformat* The internal storage format of the texture image. Must be one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8,

GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

x The x coordinate of the lower-left corner of the pixel rectangle to be transferred to the color table.

y The y coordinate of the lower-left corner of the pixel rectangle to be transferred to the color table.

width The width of the pixel rectangle.

5.37.2.204 static void
OpenTK.Graphics.OpenGL.GL.CopyConvolutionFilter1D
 (**OpenTK.Graphics.OpenGL.ConvolutionTarget** *target*,
OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
Int32 *x*, **Int32** *y*, **Int32** *width*) [**static**]

Copy pixels into a one-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_1D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

x The window space coordinates of the lower-left coordinate of the pixel array to copy.

width The width of the pixel array to copy.

5.37.2.205 static void

OpenTK.Graphics.OpenGL.GL.CopyConvolutionFilter2D
 (**OpenTK.Graphics.OpenGL.ConvolutionTarget** *target*,
OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
Int32 *x*, **Int32** *y*, **Int32** *width*, **Int32** *height*) [**static**]

Copy pixels into a two-dimensional convolution filter.

Parameters

target Must be GL_CONVOLUTION_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

x The window space coordinates of the lower-left coordinate of the pixel array to copy.

width The width of the pixel array to copy.

height The height of the pixel array to copy.

5.37.2.206 static void OpenTK.Graphics.OpenGL.GL.CopyPixels

(**Int32** *x*, **Int32** *y*, **Int32** *width*, **Int32** *height*,
OpenTK.Graphics.OpenGL.PixelCopyType *type*) [**static**]

Copy pixels in the frame buffer.

Parameters

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specify the dimensions of the rectangular region of pixels to be copied. Both must be nonnegative.

type Specifies whether color values, depth values, or stencil values are to be copied. Symbolic constants GL_COLOR, GL_DEPTH, and GL_STENCIL are accepted.

5.37.2.207 `static void OpenTK.Graphics.OpenGL.GL.CopyTexImage1D (`
`OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,`
`OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,`
`Int32 x, Int32 y, Int32 width, Int32 border) [static]`

Copy pixels into a 1D texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the internal format of the texture. Must be one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_RGB, GL_R3_G3_B2, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

x Specify the window coordinates of the left corner of the row of pixels to be copied.

width Specifies the width of the texture image. Must be 0 or 2^{n+2} (*border*) for some integer *n*. The height of the texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

5.37.2.208 `static void OpenTK.Graphics.OpenGL.GL.CopyTexImage2D (`
`OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,`
`OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,`
`Int32 x, Int32 y, Int32 width, Int32 height, Int32 border)`
`[static]`

Copy pixels into a 2D texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalformat Specifies the internal format of the texture. Must be one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_RGB, GL_R3_G3_B2, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture image. Must be 0 or $2^{\sup n} + 2$ (*border*) for some integer .

height Specifies the height of the texture image. Must be 0 or $2^{\sup m} + 2$ (*border*) for some integer .

border Specifies the width of the border. Must be either 0 or 1.

5.37.2.209 `static void OpenTK.Graphics.OpenGL.GL.CopyTexSubImage1D (`
`OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 x, Int32 y, Int32 width) [static]`

Copy a one-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies the texel offset within the texture array.

x Specify the window coordinates of the left corner of the row of pixels to be copied.

width Specifies the width of the texture subimage.

5.37.2.210 `static void OpenTK.Graphics.OpenGL.GL.CopyTexSubImage2D (`
`OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 x, Int32 y, Int32 width, Int32
height) [static]`

Copy a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

x Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

5.37.2.211 `static void OpenTK.Graphics.OpenGL.GL.CopyTexSubImage3D (`
`OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,`
`Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 x, Int32 y, Int32`
`width, Int32 height) [static]`

Copy a three-dimensional texture subimage.

Parameters

- target* Specifies the target texture. Must be GL_TEXTURE_3D
- level* Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.
- xoffset* Specifies a texel offset in the x direction within the texture array.
- yoffset* Specifies a texel offset in the y direction within the texture array.
- zoffset* Specifies a texel offset in the z direction within the texture array.
- x* Specify the window coordinates of the lower left corner of the rectangular region of pixels to be copied.
- width* Specifies the width of the texture subimage.
- height* Specifies the height of the texture subimage.

5.37.2.212 `static Int32 OpenTK.Graphics.OpenGL.GL.CreateProgram ()`
`[static]`

Creates a program object.

5.37.2.213 `static Int32 OpenTK.Graphics.OpenGL.GL.CreateShader (`
`OpenTK.Graphics.OpenGL.ShaderType type) [static]`

Creates a shader object.

Parameters

- shaderType* Specifies the type of shader to be created. Must be either GL_VERTEX_SHADER or GL_FRAGMENT_SHADER.

5.37.2.214 `static void OpenTK.Graphics.OpenGL.GL.CullFace (`
`OpenTK.Graphics.OpenGL.CullFaceMode mode) [static]`

Specify whether front- or back-facing facets can be culled.

Parameters

mode Specifies whether front- or back-facing facets are candidates for culling. Symbolic constants GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK are accepted. The initial value is GL_BACK.

5.37.2.215 `static void OpenTK.Graphics.OpenGL.GL.DeleteBuffers (Int32 n, Int32[] buffers) [static]`

Delete named buffer objects.

Parameters

n Specifies the number of buffer objects to be deleted.

buffers Specifies an array of buffer objects to be deleted.

5.37.2.216 `static void OpenTK.Graphics.OpenGL.GL.DeleteBuffers (Int32 n, ref Int32 buffers) [static]`

Delete named buffer objects.

Parameters

n Specifies the number of buffer objects to be deleted.

buffers Specifies an array of buffer objects to be deleted.

5.37.2.217 `static unsafe void OpenTK.Graphics.OpenGL.GL.DeleteBuffers (Int32 n, Int32 * buffers) [static]`

Delete named buffer objects.

Parameters

n Specifies the number of buffer objects to be deleted.

buffers Specifies an array of buffer objects to be deleted.

5.37.2.218 `static void OpenTK.Graphics.OpenGL.GL.DeleteBuffers (Int32 n, UInt32[] buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.37.2.219 `static void OpenTK.Graphics.OpenGL.GL.DeleteBuffers (Int32 n,
ref UInt32 buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.37.2.220 `static unsafe void OpenTK.Graphics.OpenGL.GL.DeleteBuffers (Int32 n, UInt32 * buffers) [static]`

Delete named buffer objects.

Parameters

- n* Specifies the number of buffer objects to be deleted.
buffers Specifies an array of buffer objects to be deleted.

5.37.2.221 `static void OpenTK.Graphics.OpenGL.GL.DeleteLists (Int32 list,
Int32 range) [static]`

Delete a contiguous group of display lists.

Parameters

- list* Specifies the integer name of the first display list to delete.
range Specifies the number of display lists to delete.

5.37.2.222 `static void OpenTK.Graphics.OpenGL.GL.DeleteLists (UInt32
list, Int32 range) [static]`

Delete a contiguous group of display lists.

Parameters

- list* Specifies the integer name of the first display list to delete.
range Specifies the number of display lists to delete.

5.37.2.223 `static void OpenTK.Graphics.OpenGL.GL.DeleteProgram (Int32
program) [static]`

Deletes a program object.

Parameters

program Specifies the program object to be deleted.

5.37.2.224 `static void OpenTK.Graphics.OpenGL.GL.DeleteProgram (UInt32
program) [static]`

Deletes a program object.

Parameters

program Specifies the program object to be deleted.

5.37.2.225 `static void OpenTK.Graphics.OpenGL.GL.DeleteQueries (Int32 n,
Int32[] ids) [static]`

Delete named query objects.

Parameters

n Specifies the number of query objects to be deleted.

ids Specifies an array of query objects to be deleted.

5.37.2.226 `static void OpenTK.Graphics.OpenGL.GL.DeleteQueries (Int32 n,
ref Int32 ids) [static]`

Delete named query objects.

Parameters

n Specifies the number of query objects to be deleted.

ids Specifies an array of query objects to be deleted.

5.37.2.227 `static unsafe void OpenTK.Graphics.OpenGL.GL.DeleteQueries (Int32 n, Int32 * ids) [static]`

Delete named query objects.

Parameters

n Specifies the number of query objects to be deleted.

ids Specifies an array of query objects to be deleted.

5.37.2.228 `static void OpenTK.Graphics.OpenGL.GL.DeleteQueries (Int32 n, UInt32[] ids) [static]`

Delete named query objects.

Parameters

n Specifies the number of query objects to be deleted.

ids Specifies an array of query objects to be deleted.

5.37.2.229 `static void OpenTK.Graphics.OpenGL.GL.DeleteQueries (Int32 n, ref UInt32 ids) [static]`

Delete named query objects.

Parameters

n Specifies the number of query objects to be deleted.

ids Specifies an array of query objects to be deleted.

5.37.2.230 `static unsafe void OpenTK.Graphics.OpenGL.GL.DeleteQueries (Int32 n, UInt32 * ids) [static]`

Delete named query objects.

Parameters

n Specifies the number of query objects to be deleted.

ids Specifies an array of query objects to be deleted.

5.37.2.231 `static void OpenTK.Graphics.OpenGL.GL.DeleteShader (Int32
shader) [static]`

Deletes a shader object.

Parameters

shader Specifies the shader object to be deleted.

5.37.2.232 `static void OpenTK.Graphics.OpenGL.GL.DeleteShader (UInt32
shader) [static]`

Deletes a shader object.

Parameters

shader Specifies the shader object to be deleted.

5.37.2.233 `static void OpenTK.Graphics.OpenGL.GL.DeleteTextures (Int32
n, Int32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.37.2.234 `static void OpenTK.Graphics.OpenGL.GL.DeleteTextures (Int32
n, ref Int32 textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.37.2.235 `static unsafe void OpenTK.Graphics.OpenGL.GL.DeleteTextures (Int32 n, Int32 * textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.37.2.236 `static void OpenTK.Graphics.OpenGL.GL.DeleteTextures (Int32 n, UInt32[] textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.37.2.237 `static void OpenTK.Graphics.OpenGL.GL.DeleteTextures (Int32 n, ref UInt32 textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.37.2.238 `static unsafe void OpenTK.Graphics.OpenGL.GL.DeleteTextures (Int32 n, UInt32 * textures) [static]`

Delete named textures.

Parameters

n Specifies the number of textures to be deleted.

textures Specifies an array of textures to be deleted.

5.37.2.239 static void OpenTK.Graphics.OpenGL.GL.DepthFunc (
OpenTK.Graphics.OpenGL.DepthFunction *func*) [static]

Specify the value used for depth buffer comparisons.

Parameters

func Specifies the depth comparison function. Symbolic constants GL_NEVER, GL_LESS, GL_EQUAL, GL_LEQUAL, GL_GREATER, GL_NOTEQUAL, GL_GEQUAL, and GL_ALWAYS are accepted. The initial value is GL_LESS.

5.37.2.240 static void OpenTK.Graphics.OpenGL.GL.DepthMask (bool *flag*
) [static]

Enable or disable writing into the depth buffer.

Parameters

flag Specifies whether the depth buffer is enabled for writing. If flag is GL_FALSE, depth buffer writing is disabled. Otherwise, it is enabled. Initially, depth buffer writing is enabled.

5.37.2.241 static void OpenTK.Graphics.OpenGL.GL.DepthRange (Double
***near*, Double *far*) [static]**

Specify mapping of depth values from normalized device coordinates to window coordinates.

Parameters

nearVal Specifies the mapping of the near clipping plane to window coordinates. The initial value is 0.

farVal Specifies the mapping of the far clipping plane to window coordinates. The initial value is 1.

5.37.2.242 static void OpenTK.Graphics.OpenGL.GL.DetachShader (Int32
***program*, Int32 *shader*) [static]**

Detaches a shader object from a program object to which it is attached.

Parameters

program Specifies the program object from which to detach the shader object.

shader Specifies the shader object to be detached.

5.37.2.243 `static void OpenTK.Graphics.OpenGL.GL.DetachShader (UInt32 program, UInt32 shader) [static]`

Detaches a shader object from a program object to which it is attached.

Parameters

program Specifies the program object from which to detach the shader object.

shader Specifies the shader object to be detached.

5.37.2.244 `static void OpenTK.Graphics.OpenGL.GL.DrawArrays (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 first, Int32 count) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Specifies the starting index in the enabled arrays.

count Specifies the number of indices to be rendered.

5.37.2.245 `static void OpenTK.Graphics.OpenGL.GL.DrawBuffer (OpenTK.Graphics.OpenGL.DrawBufferMode mode) [static]`

Specify which color buffers are to be drawn into.

Parameters

mode Specifies up to four color buffers to be drawn into. Symbolic constants GL_NONE, GL_FRONT_LEFT, GL_FRONT_RIGHT, GL_BACK_LEFT, GL_BACK_RIGHT, GL_FRONT, GL_BACK, GL_LEFT, GL_RIGHT, GL_FRONT_AND_BACK, and GL_AUXi, where i is between 0 and the value of GL_AUX_BUFFERS minus 1, are accepted. (GL_AUX_BUFFERS is not the upper limit; use glGet to query the number of available aux buffers.) The initial value is GL_FRONT for single-buffered contexts, and GL_BACK for double-buffered contexts.

5.37.2.246 `static void OpenTK.Graphics.OpenGL.GL.DrawBuffers (Int32
n, OpenTK.Graphics.OpenGL.DrawBuffersEnum[] bufs)
[static]`

Specifies a list of color buffers to be drawn into.

Parameters

n Specifies the number of buffers in bufs.

bufs Points to an array of symbolic constants specifying the buffers into which fragment colors or data values will be written.

5.37.2.247 `static void OpenTK.Graphics.OpenGL.GL.DrawBuffers (Int32
n, ref OpenTK.Graphics.OpenGL.DrawBuffersEnum bufs)
[static]`

Specifies a list of color buffers to be drawn into.

Parameters

n Specifies the number of buffers in bufs.

bufs Points to an array of symbolic constants specifying the buffers into which fragment colors or data values will be written.

5.37.2.248 `static unsafe void OpenTK.Graphics.OpenGL.GL.DrawBuffers (Int32 n, OpenTK.Graphics.OpenGL.DrawBuffersEnum * bufs)
[static]`

Specifies a list of color buffers to be drawn into.

Parameters

n Specifies the number of buffers in bufs.

bufs Points to an array of symbolic constants specifying the buffers into which fragment colors or data values will be written.

5.37.2.249 `static void OpenTK.Graphics.OpenGL.GL.DrawElements (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

```
5.37.2.250 static void OpenTK.Graphics.OpenGL.GL.DrawElements< T3 >
( OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
  OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute,
  OutAttribute] T3[] indices ) [static]
```

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : *struct*

```
5.37.2.251 static void OpenTK.Graphics.OpenGL.GL.DrawElements< T3 >
( OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
  OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute,
  OutAttribute] T3 indices[, ] ) [static]
```

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : struct

```
5.37.2.252 static void OpenTK.Graphics.OpenGL.GL.DrawElements< T3 >
( OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
  OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute,
  OutAttribute] T3 indices[, ] ) [static]
```

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : struct

```
5.37.2.253 static void OpenTK.Graphics.OpenGL.GL.DrawElements< T3 >
( OpenTK.Graphics.OpenGL.BeginMode mode, Int32 count,
  OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute,
  OutAttribute] ref T3 indices ) [static]
```

Render primitives from array data.

Parameters

- mode* Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.
- count* Specifies the number of elements to be rendered.
- type* Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.
- indices* Specifies a pointer to the location where the indices are stored.

Type Constraints

T3 : struct

5.37.2.254 static void OpenTK.Graphics.OpenGL.GL.DrawPixels (Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels) [static]

Write a block of pixels to the frame buffer.

Parameters

- width* Specify the dimensions of the pixel rectangle to be written into the frame buffer.
- format* Specifies the format of the pixel data. Symbolic constants GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA are accepted.
- type* Specifies the data type for data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV are accepted.
- data* Specifies a pointer to the pixel data.

5.37.2.255 `static void OpenTK.Graphics.OpenGL.GL.DrawPixels< T4 > (Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4[] pixels) [static]`

Write a block of pixels to the frame buffer.

Parameters

width Specify the dimensions of the pixel rectangle to be written into the frame buffer.

format Specifies the format of the pixel data. Symbolic constants GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA are accepted.

type Specifies the data type for data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Specifies a pointer to the pixel data.

Type Constraints

T4 : *struct*

5.37.2.256 `static void OpenTK.Graphics.OpenGL.GL.DrawPixels< T4 > (Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4 pixels[,]) [static]`

Write a block of pixels to the frame buffer.

Parameters

width Specify the dimensions of the pixel rectangle to be written into the frame buffer.

format Specifies the format of the pixel data. Symbolic constants GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA are accepted.

type Specifies the data type for data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV are accepted.

data Specifies a pointer to the pixel data.

Type Constraints

T4 : struct

5.37.2.257 `static void OpenTK.Graphics.OpenGL.GL.DrawPixels< T4 > (Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4 pixels[,]) [static]`

Write a block of pixels to the frame buffer.

Parameters

width Specify the dimensions of the pixel rectangle to be written into the frame buffer.

format Specifies the format of the pixel data. Symbolic constants GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA are accepted.

type Specifies the data type for data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV,

GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Specifies a pointer to the pixel data.

Type Constraints

T4 : struct

5.37.2.258 `static void OpenTK.Graphics.OpenGL.GL.DrawPixels< T4 > (Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T4 pixels) [static]`

Write a block of pixels to the frame buffer.

Parameters

width Specify the dimensions of the pixel rectangle to be written into the frame buffer.

format Specifies the format of the pixel data. Symbolic constants GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA are accepted.

type Specifies the data type for data. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

data Specifies a pointer to the pixel data.

Type Constraints

T4 : struct

5.37.2.259 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

5.37.2.260 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

5.37.2.261 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[] indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : struct

5.37.2.262 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : *struct*

5.37.2.263 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : *struct*

5.37.2.264 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 start, Int32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] ref T5 indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : *struct*

5.37.2.265 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements<T5> (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5[] indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : *struct*

5.37.2.266 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : *struct*

5.37.2.267 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements< T5 > (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T5 indices[,]) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : struct

5.37.2.268 `static void OpenTK.Graphics.OpenGL.GL.DrawRangeElements<T5> (OpenTK.Graphics.OpenGL.BeginMode mode, UInt32 start, UInt32 end, Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] ref T5 indices) [static]`

Render primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

start Specifies the minimum array index contained in indices.

end Specifies the maximum array index contained in indices.

count Specifies the number of elements to be rendered.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

Type Constraints

T5 : struct

5.37.2.269 `static void OpenTK.Graphics.OpenGL.GL.EdgeFlag (bool flag) [static]`

Flag edges as either boundary or nonboundary.

Parameters

flag Specifies the current edge flag value, either GL_TRUE or GL_FALSE. The initial value is GL_TRUE.

5.37.2.270 `static unsafe void OpenTK.Graphics.OpenGL.GL.EdgeFlag (bool * flag) [static]`

Flag edges as either boundary or nonboundary.

Parameters

flag Specifies the current edge flag value, either GL_TRUE or GL_FALSE. The initial value is GL_TRUE.

5.37.2.271 `static void OpenTK.Graphics.OpenGL.GL.EdgeFlagPointer (Int32 stride, IntPtr pointer) [static]`

Define an array of edge flags.

Parameters

stride Specifies the byte offset between consecutive edge flags. If stride is 0, the edge flags are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first edge flag in the array. The initial value is 0.

5.37.2.272 `static void OpenTK.Graphics.OpenGL.GL.EdgeFlagPointer< T1 > (Int32 stride, [InAttribute, OutAttribute] T1[] pointer) [static]`

Define an array of edge flags.

Parameters

stride Specifies the byte offset between consecutive edge flags. If stride is 0, the edge flags are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first edge flag in the array. The initial value is 0.

Type Constraints

T1 : struct

5.37.2.273 `static void OpenTK.Graphics.OpenGL.GL.EdgeFlagPointer< T1 > (Int32 stride, [InAttribute, OutAttribute] T1 pointer [,]) [static]`

Define an array of edge flags.

Parameters

stride Specifies the byte offset between consecutive edge flags. If stride is 0, the edge flags are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first edge flag in the array. The initial value is 0.

Type Constraints

T1 : *struct*

```
5.37.2.274 static void OpenTK.Graphics.OpenGL.GL.EdgeFlagPointer< T1 >
( Int32 stride, [InAttribute, OutAttribute] T1 pointer[, ] )
[static]
```

Define an array of edge flags.

Parameters

stride Specifies the byte offset between consecutive edge flags. If stride is 0, the edge flags are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first edge flag in the array. The initial value is 0.

Type Constraints

T1 : *struct*

```
5.37.2.275 static void OpenTK.Graphics.OpenGL.GL.EdgeFlagPointer< T1
> ( Int32 stride, [InAttribute, OutAttribute] ref T1 pointer )
[static]
```

Define an array of edge flags.

Parameters

stride Specifies the byte offset between consecutive edge flags. If stride is 0, the edge flags are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first edge flag in the array. The initial value is 0.

Type Constraints

T1 : *struct*

5.37.2.276 `static void OpenTK.Graphics.OpenGL.GL.Enable (`
`OpenTK.Graphics.OpenGL.EnableCap cap) [static]`

Enable or disable server-side [GL](#) capabilities.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.37.2.277 `static void OpenTK.Graphics.OpenGL.GL.Enable (`
`OpenTK.Graphics.OpenGL.IndexedEnableCap target, Int32 index`
`) [static]`

Enable or disable server-side [GL](#) capabilities.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.37.2.278 `static void OpenTK.Graphics.OpenGL.GL.Enable (`
`OpenTK.Graphics.OpenGL.IndexedEnableCap target, UInt32`
`index) [static]`

Enable or disable server-side [GL](#) capabilities.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.37.2.279 `static void OpenTK.Graphics.OpenGL.GL.EnableClientState (`
`OpenTK.Graphics.OpenGL.ArrayCap array) [static]`

Enable or disable client-side capability.

Parameters

cap Specifies the capability to enable. Symbolic constants `GL_COLOR_ARRAY`, `GL_EDGE_FLAG_ARRAY`, `GL_FOG_COORD_ARRAY`, `GL_INDEX_ARRAY`, `GL_NORMAL_ARRAY`, `GL_SECONDARY_COLOR_ARRAY`, `GL_TEXTURE_COORD_ARRAY`, and `GL_VERTEX_ARRAY` are accepted.

5.37.2.280 **static void**
OpenTK.Graphics.OpenGL.GL.EnableVertexAttribArray (Int32
index) [static]

Enable or disable a generic vertex attribute array.

Parameters

index Specifies the index of the generic vertex attribute to be enabled or disabled.

5.37.2.281 **static void**
OpenTK.Graphics.OpenGL.GL.EnableVertexAttribArray (UInt32
index) [static]

Enable or disable a generic vertex attribute array.

Parameters

index Specifies the index of the generic vertex attribute to be enabled or disabled.

5.37.2.282 **static void OpenTK.Graphics.OpenGL.GL.EvalCoord1 (Double u**
) [static]

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.283 **static unsafe void OpenTK.Graphics.OpenGL.GL.EvalCoord1 (**
Double * u) [static]

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.284 static void OpenTK.Graphics.OpenGL.GL.EvalCoord1 (Single u) [static]

Evaluate enabled one- and two-dimensional maps.

Parameters

- u Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.285 static unsafe void OpenTK.Graphics.OpenGL.GL.EvalCoord1 (Single * u) [static]

Evaluate enabled one- and two-dimensional maps.

Parameters

- u Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.286 static void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (Double u , Double v) [static]

Evaluate enabled one- and two-dimensional maps.

Parameters

- u Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.287 `static void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (Double[]
u) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.288 `static void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (ref Double
u) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.289 `static unsafe void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (
Double * u) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.290 `static void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (Single u,
Single v) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.291 `static void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (Single[] u
) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.292 `static void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (ref Single
u) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.293 `static unsafe void OpenTK.Graphics.OpenGL.GL.EvalCoord2 (Single * u) [static]`

Evaluate enabled one- and two-dimensional maps.

Parameters

- u* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap1 or glMap2 command.
- v* Specifies a value that is the domain coordinate to the basis function defined in a previous glMap2 command. This argument is not present in a glEvalCoord1 command.

5.37.2.294 `static void OpenTK.Graphics.OpenGL.GL.EvalMesh1 (OpenTK.Graphics.OpenGL.MeshMode1 mode, Int32 i1, Int32 i2) [static]`

[Compute](#) a one- or two-dimensional grid of points or lines.

Parameters

- mode* In glEvalMesh1, specifies whether to compute a one-dimensional mesh of points or lines. Symbolic constants GL_POINT and GL_LINE are accepted.
- i1* Specify the first and last integer values for grid domain variable .

5.37.2.295 `static void OpenTK.Graphics.OpenGL.GL.EvalMesh2 (OpenTK.Graphics.OpenGL.MeshMode2 mode, Int32 i1, Int32 i2, Int32 j1, Int32 j2) [static]`

[Compute](#) a one- or two-dimensional grid of points or lines.

Parameters

- mode* In glEvalMesh1, specifies whether to compute a one-dimensional mesh of points or lines. Symbolic constants GL_POINT and GL_LINE are accepted.
- i1* Specify the first and last integer values for grid domain variable .

5.37.2.296 `static void OpenTK.Graphics.OpenGL.GL.EvalPoint1 (Int32 i) [static]`

Generate and evaluate a single point in a mesh.

Parameters

- i* Specifies the integer value for grid domain variable .
- j* Specifies the integer value for grid domain variable (glEvalPoint2 only).

5.37.2.297 `static void OpenTK.Graphics.OpenGL.GL.EvalPoint2 (Int32 i, Int32 j) [static]`

Generate and evaluate a single point in a mesh.

Parameters

- i* Specifies the integer value for grid domain variable .
- j* Specifies the integer value for grid domain variable (glEvalPoint2 only).

5.37.2.298 `static void OpenTK.Graphics.OpenGL.GL.FeedbackBuffer (Int32 size, OpenTK.Graphics.OpenGL.FeedbackType type, [OutAttribute] Single[] buffer) [static]`

Controls feedback mode.

Parameters

- size* Specifies the maximum number of values that can be written into buffer.
- type* Specifies a symbolic constant that describes the information that will be returned for each vertex. GL_2D, GL_3D, GL_3D_COLOR, GL_3D_COLOR_TEXTURE, and GL_4D_COLOR_TEXTURE are accepted.
- buffer* Returns the feedback data.

5.37.2.299 `static void OpenTK.Graphics.OpenGL.GL.FeedbackBuffer (Int32 size, OpenTK.Graphics.OpenGL.FeedbackType type, [OutAttribute] out Single buffer) [static]`

Controls feedback mode.

Parameters

- size* Specifies the maximum number of values that can be written into buffer.
- type* Specifies a symbolic constant that describes the information that will be returned for each vertex. GL_2D, GL_3D, GL_3D_COLOR, GL_3D_COLOR_TEXTURE, and GL_4D_COLOR_TEXTURE are accepted.
- buffer* Returns the feedback data.

5.37.2.300 `static unsafe void OpenTK.Graphics.OpenGL.GL.FeedbackBuffer
(Int32 size, OpenTK.Graphics.OpenGL.FeedbackType type,
[OutAttribute] Single * buffer) [static]`

Controls feedback mode.

Parameters

size Specifies the maximum number of values that can be written into buffer.

type Specifies a symbolic constant that describes the information that will be returned for each vertex. GL_2D, GL_3D, GL_3D_COLOR, GL_3D_COLOR_TEXTURE, and GL_4D_COLOR_TEXTURE are accepted.

buffer Returns the feedback data.

5.37.2.301 `static void OpenTK.Graphics.OpenGL.GL.Finish () [static]`

Block until all GL execution is complete.

5.37.2.302 `static void OpenTK.Graphics.OpenGL.GL.Flush () [static]`

Force execution of GL commands in finite time.

5.37.2.303 `static void OpenTK.Graphics.OpenGL.GL.Fog (
OpenTK.Graphics.OpenGL.FogParameter pname, Single param)
[static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that *pname* will be set to.

5.37.2.304 `static void OpenTK.Graphics.OpenGL.GL.Fog (
OpenTK.Graphics.OpenGL.FogParameter pname, Single @[
params]) [static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.37.2.305 `static unsafe void OpenTK.Graphics.OpenGL.GL.Fog (OpenTK.Graphics.OpenGL.FogParameter pname, Single *@ params) [static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.37.2.306 `static void OpenTK.Graphics.OpenGL.GL.Fog (OpenTK.Graphics.OpenGL.FogParameter pname, Int32 param) [static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.37.2.307 `static void OpenTK.Graphics.OpenGL.GL.Fog (OpenTK.Graphics.OpenGL.FogParameter pname, Int32 @[] params) [static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.37.2.308 `static unsafe void OpenTK.Graphics.OpenGL.GL.Fog (`
`OpenTK.Graphics.OpenGL.FogParameter pname, Int32 *@`
`params) [static]`

Specify fog parameters.

Parameters

pname Specifies a single-valued fog parameter. GL_FOG_MODE, GL_FOG_DENSITY, GL_FOG_START, GL_FOG_END, GL_FOG_INDEX, and GL_FOG_COORD_SRC are accepted.

param Specifies the value that pname will be set to.

5.37.2.309 `static void OpenTK.Graphics.OpenGL.GL.FogCoord (Double`
`coord) [static]`

Set the current fog coordinates.

Parameters

coord Specify the fog distance.

5.37.2.310 `static unsafe void OpenTK.Graphics.OpenGL.GL.FogCoord (`
`Double * coord) [static]`

Set the current fog coordinates.

Parameters

coord Specify the fog distance.

5.37.2.311 `static void OpenTK.Graphics.OpenGL.GL.FogCoord (Single`
`coord) [static]`

Set the current fog coordinates.

Parameters

coord Specify the fog distance.

5.37.2.312 `static unsafe void OpenTK.Graphics.OpenGL.GL.FogCoord (Single * coord) [static]`

Set the current fog coordinates.

Parameters

coord Specify the fog distance.

5.37.2.313 `static void OpenTK.Graphics.OpenGL.GL.FogCoordPointer (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride, IntPtr pointer) [static]`

Define an array of fog coordinates.

Parameters

type Specifies the data type of each fog coordinate. Symbolic constants GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive fog coordinates. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first fog coordinate in the array. The initial value is 0.

5.37.2.314 `static void OpenTK.Graphics.OpenGL.GL.FogCoordPointer< T2 > (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer) [static]`

Define an array of fog coordinates.

Parameters

type Specifies the data type of each fog coordinate. Symbolic constants GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive fog coordinates. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first fog coordinate in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.315 `static void OpenTK.Graphics.OpenGL.GL.FogCoordPointer< T2 >
 (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,
 [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of fog coordinates.

Parameters

type Specifies the data type of each fog coordinate. Symbolic constants GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive fog coordinates. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first fog coordinate in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.316 `static void OpenTK.Graphics.OpenGL.GL.FogCoordPointer< T2 >
 (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,
 [InAttribute, OutAttribute] T2 pointer[,,]) [static]`

Define an array of fog coordinates.

Parameters

type Specifies the data type of each fog coordinate. Symbolic constants GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive fog coordinates. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first fog coordinate in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.317 `static void OpenTK.Graphics.OpenGL.GL.FogCoordPointer< T2 >
 (OpenTK.Graphics.OpenGL.FogPointerType type, Int32 stride,
 [InAttribute, OutAttribute] ref T2 pointer) [static]`

Define an array of fog coordinates.

Parameters

- type* Specifies the data type of each fog coordinate. Symbolic constants GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive fog coordinates. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first fog coordinate in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.318 static void OpenTK.Graphics.OpenGL.GL.FrontFace (OpenTK.Graphics.OpenGL.FrontFaceDirection *mode*)
[static]

Define front- and back-facing polygons.

Parameters

- mode* Specifies the orientation of front-facing polygons. GL_CW and GL_CCW are accepted. The initial value is GL_CCW.

5.37.2.319 static void OpenTK.Graphics.OpenGL.GL.Frustum (Double *left*, Double *right*, Double *bottom*, Double *top*, Double *zNear*, Double *zFar*) [static]

Multiply the current matrix by a perspective matrix.

Parameters

- left* Specify the coordinates for the left and right vertical clipping planes.
- bottom* Specify the coordinates for the bottom and top horizontal clipping planes.
- nearVal* Specify the distances to the near and far depth clipping planes. Both distances must be positive.

5.37.2.320 static void OpenTK.Graphics.OpenGL.GL.GenBuffers (Int32 *n*, [OutAttribute] Int32[] *buffers*) [static]

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.37.2.321 `static void OpenTK.Graphics.OpenGL.GL.GenBuffers (Int32 n,
[OutAttribute] out Int32 buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.37.2.322 `static unsafe void OpenTK.Graphics.OpenGL.GL.GenBuffers (Int32 n, [OutAttribute] Int32 * buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.37.2.323 `static void OpenTK.Graphics.OpenGL.GL.GenBuffers (Int32 n,
[OutAttribute] UInt32[] buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.37.2.324 `static void OpenTK.Graphics.OpenGL.GL.GenBuffers (Int32 n,
[OutAttribute] out UInt32 buffers) [static]`

Generate buffer object names.

Parameters

- n* Specifies the number of buffer object names to be generated.
buffers Specifies an array in which the generated buffer object names are stored.

5.37.2.325 `static unsafe void OpenTK.Graphics.OpenGL.GL.GenBuffers (Int32 n, [OutAttribute] UInt32 * buffers) [static]`

Generate buffer object names.

Parameters

n Specifies the number of buffer object names to be generated.

buffers Specifies an array in which the generated buffer object names are stored.

5.37.2.326 `static Int32 OpenTK.Graphics.OpenGL.GL.GenLists (Int32 range) [static]`

Generate a contiguous set of empty display lists.

Parameters

range Specifies the number of contiguous empty display lists to be generated.

5.37.2.327 `static void OpenTK.Graphics.OpenGL.GL.GenQueries (Int32 n, [OutAttribute] Int32[] ids) [static]`

Generate query object names.

Parameters

n Specifies the number of query object names to be generated.

ids Specifies an array in which the generated query object names are stored.

5.37.2.328 `static void OpenTK.Graphics.OpenGL.GL.GenQueries (Int32 n, [OutAttribute] out Int32 ids) [static]`

Generate query object names.

Parameters

n Specifies the number of query object names to be generated.

ids Specifies an array in which the generated query object names are stored.

5.37.2.329 `static unsafe void OpenTK.Graphics.OpenGL.GL.GenQueries (Int32 n, [OutAttribute] Int32 * ids) [static]`

Generate query object names.

Parameters

n Specifies the number of query object names to be generated.

ids Specifies an array in which the generated query object names are stored.

5.37.2.330 `static void OpenTK.Graphics.OpenGL.GL.GenQueries (Int32 n, [OutAttribute] UInt32[] ids) [static]`

Generate query object names.

Parameters

n Specifies the number of query object names to be generated.

ids Specifies an array in which the generated query object names are stored.

5.37.2.331 `static void OpenTK.Graphics.OpenGL.GL.GenQueries (Int32 n, [OutAttribute] out UInt32 ids) [static]`

Generate query object names.

Parameters

n Specifies the number of query object names to be generated.

ids Specifies an array in which the generated query object names are stored.

5.37.2.332 `static unsafe void OpenTK.Graphics.OpenGL.GL.GenQueries (Int32 n, [OutAttribute] UInt32 * ids) [static]`

Generate query object names.

Parameters

n Specifies the number of query object names to be generated.

ids Specifies an array in which the generated query object names are stored.

5.37.2.333 `static void OpenTK.Graphics.OpenGL.GL.GenTextures (Int32 n,
[OutAttribute] Int32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.37.2.334 `static void OpenTK.Graphics.OpenGL.GL.GenTextures (Int32 n,
[OutAttribute] out Int32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.37.2.335 `static unsafe void OpenTK.Graphics.OpenGL.GL.GenTextures (Int32 n,
[OutAttribute] Int32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.37.2.336 `static void OpenTK.Graphics.OpenGL.GL.GenTextures (Int32 n,
[OutAttribute] UInt32[] textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.37.2.337 `static void OpenTK.Graphics.OpenGL.GL.GenTextures (Int32 n, [OutAttribute] out UInt32 textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.37.2.338 `static unsafe void OpenTK.Graphics.OpenGL.GL.GenTextures (Int32 n, [OutAttribute] UInt32 * textures) [static]`

Generate texture names.

Parameters

n Specifies the number of texture names to be generated.

textures Specifies an array in which the generated texture names are stored.

5.37.2.339 `static void OpenTK.Graphics.OpenGL.GL.GetActiveAttrib (UInt32 program, UInt32 index, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.OpenGL.ActiveAttribType type, [OutAttribute] out StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

program Specifies the program object to be queried.

index Specifies the index of the attribute variable to be queried.

bufSize Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.

length Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.

size Returns the size of the attribute variable.

type Returns the data type of the attribute variable.

name Returns a null terminated string containing the name of the attribute variable.

5.37.2.340 `static void OpenTK.Graphics.OpenGL.GL.GetActiveAttrib (Int32 program, Int32 index, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.OpenGL.ActiveAttribType type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.37.2.341 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetActiveAttrib (Int32 program, Int32 index, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] Int32 * size, [OutAttribute] OpenTK.Graphics.OpenGL.ActiveAttribType * type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the attribute variable to be queried.
- bufSize* Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the attribute variable.
- type* Returns the data type of the attribute variable.
- name* Returns a null terminated string containing the name of the attribute variable.

5.37.2.342 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetActiveAttrib (UInt32 program, UInt32 index, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] Int32 * size, [OutAttribute] OpenTK.Graphics.OpenGL.ActiveAttribType * type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active attribute variable for the specified program object.

Parameters

- program*** Specifies the program object to be queried.
- index*** Specifies the index of the attribute variable to be queried.
- bufSize*** Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length*** Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size*** Returns the size of the attribute variable.
- type*** Returns the data type of the attribute variable.
- name*** Returns a null terminated string containing the name of the attribute variable.

5.37.2.343 `static void OpenTK.Graphics.OpenGL.GL.GetActiveUniform (Int32 program, Int32 index, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] out Int32 size, [OutAttribute] out OpenTK.Graphics.OpenGL.ActiveUniformType type, [OutAttribute] StringBuilder name) [static]`

Returns information about an active uniform variable for the specified program object.

Parameters

- program*** Specifies the program object to be queried.
- index*** Specifies the index of the uniform variable to be queried.
- bufSize*** Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
- length*** Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size*** Returns the size of the uniform variable.
- type*** Returns the data type of the uniform variable.
- name*** Returns a null terminated string containing the name of the uniform variable.

5.37.2.344 static unsafe void
 OpenTK.Graphics.OpenGL.GL.GetActiveUniform
 (Int32 *program*, Int32 *index*, Int32 *bufSize*, [OutAttribute]
 Int32 * *length*, [OutAttribute] Int32 * *size*, [OutAttribute]
 OpenTK.Graphics.OpenGL.ActiveUniformType * *type*,
 [OutAttribute] StringBuilder *name*) [static]

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters OpenGL is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by OpenGL in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.
- name* Returns a null terminated string containing the name of the uniform variable.

5.37.2.345 static void OpenTK.Graphics.OpenGL.GL.GetActiveUniform (UInt32 *program*, UInt32 *index*, Int32 *bufSize*, [OutAttribute] out Int32 *length*, [OutAttribute] out Int32 *size*, [OutAttribute] out OpenTK.Graphics.OpenGL.ActiveUniformType *type*, [OutAttribute] StringBuilder *name*) [static]

Returns information about an active uniform variable for the specified program object.

Parameters

- program* Specifies the program object to be queried.
- index* Specifies the index of the uniform variable to be queried.
- bufSize* Specifies the maximum number of characters OpenGL is allowed to write in the character buffer indicated by name.
- length* Returns the number of characters actually written by OpenGL in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
- size* Returns the size of the uniform variable.
- type* Returns the data type of the uniform variable.

name Returns a null terminated string containing the name of the uniform variable.

5.37.2.346 static unsafe void
OpenTK.Graphics.OpenGL.GL.GetActiveUniform
 (UInt32 *program*, UInt32 *index*, Int32 *bufSize*, [OutAttribute]
 Int32 * *length*, [OutAttribute] Int32 * *size*, [OutAttribute]
 OpenTK.Graphics.OpenGL.ActiveUniformType * *type*,
 [OutAttribute] StringBuilder *name*) [static]

Returns information about an active uniform variable for the specified program object.

Parameters

program Specifies the program object to be queried.
index Specifies the index of the uniform variable to be queried.
bufSize Specifies the maximum number of characters [OpenGL](#) is allowed to write in the character buffer indicated by name.
length Returns the number of characters actually written by [OpenGL](#) in the string indicated by name (excluding the null terminator) if a value other than NULL is passed.
size Returns the size of the uniform variable.
type Returns the data type of the uniform variable.
name Returns a null terminated string containing the name of the uniform variable.

5.37.2.347 static void OpenTK.Graphics.OpenGL.GL.GetAttachedShaders (
Int32 program, Int32 maxCount, [OutAttribute] out Int32 count,
[OutAttribute] out Int32 obj) [static]

Returns the handles of the shader objects attached to a program object.

Parameters

program Specifies the program object to be queried.
maxCount Specifies the size of the array for storing the returned object names.
count Returns the number of names actually returned in objects.
shaders Specifies an array that is used to return the names of attached shader objects.

5.37.2.348 `static unsafe void
OpenTK.Graphics.OpenGL.GL.GetAttachedShaders (Int32
program, Int32 maxCount, [OutAttribute] Int32 * count,
[OutAttribute] Int32[] obj) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

program Specifies the program object to be queried.
maxCount Specifies the size of the array for storing the returned object names.
count Returns the number of names actually returned in objects.
shaders Specifies an array that is used to return the names of attached shader objects.

5.37.2.349 `static unsafe void
OpenTK.Graphics.OpenGL.GL.GetAttachedShaders (Int32
program, Int32 maxCount, [OutAttribute] Int32 * count,
[OutAttribute] Int32 * obj) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

program Specifies the program object to be queried.
maxCount Specifies the size of the array for storing the returned object names.
count Returns the number of names actually returned in objects.
shaders Specifies an array that is used to return the names of attached shader objects.

5.37.2.350 `static void OpenTK.Graphics.OpenGL.GL.GetAttachedShaders (
UInt32 program, Int32 maxCount, [OutAttribute] out Int32 count,
[OutAttribute] out UInt32 obj) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

program Specifies the program object to be queried.
maxCount Specifies the size of the array for storing the returned object names.
count Returns the number of names actually returned in objects.
shaders Specifies an array that is used to return the names of attached shader objects.

5.37.2.351 `static unsafe void
OpenTK.Graphics.OpenGL.GL.GetAttachedShaders (UInt32
program, Int32 maxCount, [OutAttribute] Int32 * count,
[OutAttribute] UInt32[] obj) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

program Specifies the program object to be queried.

maxCount Specifies the size of the array for storing the returned object names.

count Returns the number of names actually returned in objects.

shaders Specifies an array that is used to return the names of attached shader objects.

5.37.2.352 `static unsafe void
OpenTK.Graphics.OpenGL.GL.GetAttachedShaders (UInt32
program, Int32 maxCount, [OutAttribute] Int32 * count,
[OutAttribute] UInt32 * obj) [static]`

Returns the handles of the shader objects attached to a program object.

Parameters

program Specifies the program object to be queried.

maxCount Specifies the size of the array for storing the returned object names.

count Returns the number of names actually returned in objects.

shaders Specifies an array that is used to return the names of attached shader objects.

5.37.2.353 `static Int32 OpenTK.Graphics.OpenGL.GL.GetAttribLocation (`
`Int32 program, String name) [static]`

Returns the location of an attribute variable.

Parameters

program Specifies the program object to be queried.

name Points to a null terminated string containing the name of the attribute variable whose location is to be queried.

5.37.2.354 `static Int32 OpenTK.Graphics.OpenGL.GL.GetAttribLocation (UInt32 program, String name) [static]`

Returns the location of an attribute variable.

Parameters

program Specifies the program object to be queried.

name Points to a null terminated string containing the name of the attribute variable whose location is to be queried.

5.37.2.355 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetBufferParameter (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferParameterName pname, [OutAttribute] Int32 *@ params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.37.2.356 `static void OpenTK.Graphics.OpenGL.GL.GetBufferParameter (OpenTK.Graphics.OpenGL.BufferTarget target, OpenTK.Graphics.OpenGL.BufferParameterName pname, [OutAttribute] Int32 @[] params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.37.2.357 `static void OpenTK.Graphics.OpenGL.GL.GetBufferParameter
(OpenTK.Graphics.OpenGL.BufferTarget target,
OpenTK.Graphics.OpenGL.BufferParameterName pname,
[OutAttribute] out Int32 @ params) [static]`

Return parameters of a buffer object.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

value Specifies the symbolic name of a buffer object parameter. Accepted values are GL_BUFFER_ACCESS, GL_BUFFER_MAPPED, GL_BUFFER_SIZE, or GL_BUFFER_USAGE.

data Returns the requested parameter.

5.37.2.358 `static void OpenTK.Graphics.OpenGL.GL.GetBufferPointer
(OpenTK.Graphics.OpenGL.BufferTarget target,
OpenTK.Graphics.OpenGL.BufferPointer pname, [OutAttribute]
IntPtr @ params) [static]`

Return the pointer to a mapped buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

pname Specifies the pointer to be returned. The symbolic constant must be GL_BUFFER_MAP_POINTER.

params Returns the pointer value specified by *pname*.

5.37.2.359 `static void OpenTK.Graphics.OpenGL.GL.GetBufferPointer<
T2 > (OpenTK.Graphics.OpenGL.BufferTarget target,
OpenTK.Graphics.OpenGL.BufferPointer pname, [InAttribute,
OutAttribute] T2 @[] params) [static]`

Return the pointer to a mapped buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

pname Specifies the pointer to be returned. The symbolic constant must be GL_BUFFER_MAP_POINTER.

params Returns the pointer value specified by pname.

Type Constraints

T2 : *struct*

```
5.37.2.360 static void OpenTK.Graphics.OpenGL.GL.GetBufferPointer<
T2 > ( OpenTK.Graphics.OpenGL.BufferTarget target,
OpenTK.Graphics.OpenGL.BufferPointer pname, [InAttribute,
OutAttribute] T2 @ params[, ] ) [static]
```

Return the pointer to a mapped buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

pname Specifies the pointer to be returned. The symbolic constant must be GL_BUFFER_MAP_POINTER.

params Returns the pointer value specified by pname.

Type Constraints

T2 : *struct*

```
5.37.2.361 static void OpenTK.Graphics.OpenGL.GL.GetBufferPointer<
T2 > ( OpenTK.Graphics.OpenGL.BufferTarget target,
OpenTK.Graphics.OpenGL.BufferPointer pname, [InAttribute,
OutAttribute] T2 @ params[, ] ) [static]
```

Return the pointer to a mapped buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

pname Specifies the pointer to be returned. The symbolic constant must be GL_BUFFER_MAP_POINTER.

params Returns the pointer value specified by *pname*.

Type Constraints

T2 : *struct*

```
5.37.2.362 static void OpenTK.Graphics.OpenGL.GL.GetBufferPointer<
           T2 > ( OpenTK.Graphics.OpenGL.BufferTarget target,
           OpenTK.Graphics.OpenGL.BufferPointer pname, [InAttribute,
           OutAttribute] ref T2 @ params ) [static]
```

Return the pointer to a mapped buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

pname Specifies the pointer to be returned. The symbolic constant must be GL_BUFFER_MAP_POINTER.

params Returns the pointer value specified by *pname*.

Type Constraints

T2 : *struct*

```
5.37.2.363 static void OpenTK.Graphics.OpenGL.GL.GetBufferSubData (
           OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
           IntPtr size, [OutAttribute] IntPtr data ) [static]
```

Returns a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store from which data will be returned, measured in bytes.

size Specifies the size in bytes of the data store region being returned.

data Specifies a pointer to the location where buffer object data is returned.

5.37.2.364 `static void OpenTK.Graphics.OpenGL.GL.GetBufferSubData< T3
> (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
IntPtr size, [InAttribute, OutAttribute] T3[] data) [static]`

Returns a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store from which data will be returned, measured in bytes.

size Specifies the size in bytes of the data store region being returned.

data Specifies a pointer to the location where buffer object data is returned.

Type Constraints

T3 : *struct*

5.37.2.365 `static void OpenTK.Graphics.OpenGL.GL.GetBufferSubData< T3
> (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
IntPtr size, [InAttribute, OutAttribute] T3 data[,]) [static]`

Returns a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store from which data will be returned, measured in bytes.

size Specifies the size in bytes of the data store region being returned.

data Specifies a pointer to the location where buffer object data is returned.

Type Constraints

T3 : *struct*

5.37.2.366 `static void OpenTK.Graphics.OpenGL.GL.GetBufferSubData< T3
> (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
IntPtr size, [InAttribute, OutAttribute] T3 data[,,]) [static]`

Returns a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store from which data will be returned, measured in bytes.

size Specifies the size in bytes of the data store region being returned.

data Specifies a pointer to the location where buffer object data is returned.

Type Constraints

T3 : *struct*

5.37.2.367 `static void OpenTK.Graphics.OpenGL.GL.GetBufferSubData< T3
> (OpenTK.Graphics.OpenGL.BufferTarget target, IntPtr offset,
IntPtr size, [InAttribute, OutAttribute] ref T3 data) [static]`

Returns a subset of a buffer object's data store.

Parameters

target Specifies the target buffer object. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

offset Specifies the offset into the buffer object's data store from which data will be returned, measured in bytes.

size Specifies the size in bytes of the data store region being returned.

data Specifies a pointer to the location where buffer object data is returned.

Type Constraints

T3 : *struct*

5.37.2.368 `static void OpenTK.Graphics.OpenGL.GL.GetClipPlane (`
`OpenTK.Graphics.OpenGL.ClipPlaneName plane, [OutAttribute]`
`Double[] equation) [static]`

Return the coefficients of the specified clipping plane.

Parameters

plane Specifies a clipping plane. The number of clipping planes depends on the implementation, but at least six clipping planes are supported. They are identified by symbolic names of the form GL_CLIP_PLANE where i ranges from 0 to the value of GL_MAX_CLIP_PLANES - 1.

equation Returns four double-precision values that are the coefficients of the plane equation of plane in eye coordinates. The initial value is (0, 0, 0, 0).

5.37.2.369 `static void OpenTK.Graphics.OpenGL.GL.GetClipPlane (`
`OpenTK.Graphics.OpenGL.ClipPlaneName plane, [OutAttribute]`
`out Double equation) [static]`

Return the coefficients of the specified clipping plane.

Parameters

plane Specifies a clipping plane. The number of clipping planes depends on the implementation, but at least six clipping planes are supported. They are identified by symbolic names of the form GL_CLIP_PLANE where i ranges from 0 to the value of GL_MAX_CLIP_PLANES - 1.

equation Returns four double-precision values that are the coefficients of the plane equation of plane in eye coordinates. The initial value is (0, 0, 0, 0).

5.37.2.370 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetClipPlane (`
`OpenTK.Graphics.OpenGL.ClipPlaneName plane, [OutAttribute]`
`Double * equation) [static]`

Return the coefficients of the specified clipping plane.

Parameters

plane Specifies a clipping plane. The number of clipping planes depends on the implementation, but at least six clipping planes are supported. They are identified by symbolic names of the form GL_CLIP_PLANE where i ranges from 0 to the value of GL_MAX_CLIP_PLANES - 1.

equation Returns four double-precision values that are the coefficients of the plane equation of plane in eye coordinates. The initial value is (0, 0, 0, 0).

5.37.2.371 `static void OpenTK.Graphics.OpenGL.GL.GetColorTable
(OpenTK.Graphics.OpenGL.ColorTableTarget target,
OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
table) [static]`

Retrieve contents of a color lookup table.

Parameters

target Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

format The format of the pixel data in table. The possible values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in table. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV are accepted.

table Pointer to a one-dimensional array of pixel data containing the contents of the color table.

5.37.2.372 `static void OpenTK.Graphics.OpenGL.GL.GetColorTable<
T3 > (OpenTK.Graphics.OpenGL.ColorTableTarget
target, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T3[] table) [static]`

Retrieve contents of a color lookup table.

Parameters

target Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

format The format of the pixel data in table. The possible values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in table. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV are accepted.

table Pointer to a one-dimensional array of pixel data containing the contents of the color table.

Type Constraints

T3 : struct

```
5.37.2.373 static void OpenTK.Graphics.OpenGL.GL.GetColorTable<
    T3 > ( OpenTK.Graphics.OpenGL.ColorTableTarget
    target, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T3 table[, ] ) [static]
```

Retrieve contents of a color lookup table.

Parameters

target Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

format The format of the pixel data in table. The possible values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in table. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV are accepted.

table Pointer to a one-dimensional array of pixel data containing the contents of the color table.

Type Constraints*T3 : struct*

5.37.2.374 `static void OpenTK.Graphics.OpenGL.GL.GetColorTable< T3 > (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T3 table[,]) [static]`

Retrieve contents of a color lookup table.

Parameters

target Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

format The format of the pixel data in table. The possible values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in table. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV are accepted.

table Pointer to a one-dimensional array of pixel data containing the contents of the color table.

Type Constraints*T3 : struct*

5.37.2.375 `static void OpenTK.Graphics.OpenGL.GL.GetColorTable< T3 > (OpenTK.Graphics.OpenGL.ColorTableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T3 table) [static]`

Retrieve contents of a color lookup table.

Parameters

target Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, or GL_POST_COLOR_MATRIX_COLOR_TABLE.

format The format of the pixel data in table. The possible values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_LUMINANCE, GL_LUMINANCE_ALPHA, GL_RGB, GL_BGR, GL_RGBA, and GL_BGRA.

type The type of the pixel data in table. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

table Pointer to a one-dimensional array of pixel data containing the contents of the color table.

Type Constraints

T3 : struct

5.37.2.376 static void

OpenTK.Graphics.OpenGL.GL.GetColorTableParameter
 (**OpenTK.Graphics.OpenGL.ColorTableTarget** *target*,
OpenTK.Graphics.OpenGL.GetColorTableParameterPName
pname, [OutAttribute] Single @[] *params*) [**static**]

Get color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a color lookup table parameter. Must be one of GL_COLOR_TABLE_BIAS, GL_COLOR_TABLE_SCALE, GL_COLOR_TABLE_FORMAT, GL_COLOR_TABLE_WIDTH, GL_COLOR_TABLE_RED_SIZE, GL_COLOR_TABLE_GREEN_SIZE, GL_COLOR_TABLE_BLUE_SIZE, GL_COLOR_TABLE_ALPHA_SIZE, GL_COLOR_TABLE_LUMINANCE_SIZE, or GL_COLOR_TABLE_INTENSITY_SIZE.

params A pointer to an array where the values of the parameter will be stored.

5.37.2.377 static void
OpenTK.Graphics.OpenGL.GL.GetColorTableParameter
 (**OpenTK.Graphics.OpenGL.ColorTableTarget** *target*,
OpenTK.Graphics.OpenGL.GetColorTableParameterPName
pname, [OutAttribute] out Single @ *params*) [**static**]

Get color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a color lookup table parameter. Must be one of GL_COLOR_TABLE_BIAS, GL_COLOR_TABLE_SCALE, GL_COLOR_TABLE_FORMAT, GL_COLOR_TABLE_WIDTH, GL_COLOR_TABLE_RED_SIZE, GL_COLOR_TABLE_GREEN_SIZE, GL_COLOR_TABLE_BLUE_SIZE, GL_COLOR_TABLE_ALPHA_SIZE, GL_COLOR_TABLE_LUMINANCE_SIZE, or GL_COLOR_TABLE_INTENSITY_SIZE.

params A pointer to an array where the values of the parameter will be stored.

5.37.2.378 static unsafe void
OpenTK.Graphics.OpenGL.GL.GetColorTableParameter
 (**OpenTK.Graphics.OpenGL.ColorTableTarget** *target*,
OpenTK.Graphics.OpenGL.GetColorTableParameterPName
pname, [OutAttribute] Single * @ *params*) [**static**]

Get color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a color lookup table parameter. Must be one of GL_COLOR_TABLE_BIAS, GL_COLOR_TABLE_SCALE,

GL_COLOR_TABLE_FORMAT, GL_COLOR_TABLE_WIDTH, GL_COLOR_TABLE_RED_SIZE, GL_COLOR_TABLE_GREEN_SIZE, GL_COLOR_TABLE_BLUE_SIZE, GL_COLOR_TABLE_ALPHA_SIZE, GL_COLOR_TABLE_LUMINANCE_SIZE, or GL_COLOR_TABLE_INTENSITY_SIZE.

params A pointer to an array where the values of the parameter will be stored.

5.37.2.379 static void

```
OpenTK.Graphics.OpenGL.GL.GetColorTableParameter
( OpenTK.Graphics.OpenGL.ColorTableTarget target,
  OpenTK.Graphics.OpenGL.GetColorTableParameterPName
  pname, [OutAttribute] Int32 @[ ] params ) [static]
```

Get color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a color lookup table parameter. Must be one of GL_COLOR_TABLE_BIAS, GL_COLOR_TABLE_SCALE, GL_COLOR_TABLE_FORMAT, GL_COLOR_TABLE_WIDTH, GL_COLOR_TABLE_RED_SIZE, GL_COLOR_TABLE_GREEN_SIZE, GL_COLOR_TABLE_BLUE_SIZE, GL_COLOR_TABLE_ALPHA_SIZE, GL_COLOR_TABLE_LUMINANCE_SIZE, or GL_COLOR_TABLE_INTENSITY_SIZE.

params A pointer to an array where the values of the parameter will be stored.

5.37.2.380 static void

```
OpenTK.Graphics.OpenGL.GL.GetColorTableParameter
( OpenTK.Graphics.OpenGL.ColorTableTarget target,
  OpenTK.Graphics.OpenGL.GetColorTableParameterPName
  pname, [OutAttribute] out Int32 @ params ) [static]
```

Get color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX-

COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a color lookup table parameter. Must be one of GL_COLOR_TABLE_BIAS, GL_COLOR_TABLE_SCALE, GL_COLOR_TABLE_FORMAT, GL_COLOR_TABLE_WIDTH, GL_COLOR_TABLE_RED_SIZE, GL_COLOR_TABLE_GREEN_SIZE, GL_COLOR_TABLE_BLUE_SIZE, GL_COLOR_TABLE_ALPHA_SIZE, GL_COLOR_TABLE_LUMINANCE_SIZE, or GL_COLOR_TABLE_INTENSITY_SIZE.

params A pointer to an array where the values of the parameter will be stored.

5.37.2.381 static unsafe void

```
OpenTK.Graphics.OpenGL.GL.GetColorTableParameter
( OpenTK.Graphics.OpenGL.ColorTableTarget target,
  OpenTK.Graphics.OpenGL.GetColorTableParameterPName
  pname, [OutAttribute] Int32 *@ params ) [static]
```

Get color lookup table parameters.

Parameters

target The target color table. Must be GL_COLOR_TABLE, GL_POST_CONVOLUTION_COLOR_TABLE, GL_POST_COLOR_MATRIX_COLOR_TABLE, GL_PROXY_COLOR_TABLE, GL_PROXY_POST_CONVOLUTION_COLOR_TABLE, or GL_PROXY_POST_COLOR_MATRIX_COLOR_TABLE.

pname The symbolic name of a color lookup table parameter. Must be one of GL_COLOR_TABLE_BIAS, GL_COLOR_TABLE_SCALE, GL_COLOR_TABLE_FORMAT, GL_COLOR_TABLE_WIDTH, GL_COLOR_TABLE_RED_SIZE, GL_COLOR_TABLE_GREEN_SIZE, GL_COLOR_TABLE_BLUE_SIZE, GL_COLOR_TABLE_ALPHA_SIZE, GL_COLOR_TABLE_LUMINANCE_SIZE, or GL_COLOR_TABLE_INTENSITY_SIZE.

params A pointer to an array where the values of the parameter will be stored.

5.37.2.382 static void

```
OpenTK.Graphics.OpenGL.GL.GetCompressedTexImage (
  OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
  [OutAttribute] IntPtr img ) [static]
```

Return a compressed texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, and GL_TEXTURE_3D GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

lod Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

img Returns the compressed texture image.

5.37.2.383 static void
OpenTK.Graphics.OpenGL.GL.GetCompressedTexImage< T2 > (
OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
[InAttribute, OutAttribute] T2[] img) [static]

Return a compressed texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, and GL_TEXTURE_3D GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

lod Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

img Returns the compressed texture image.

Type Constraints

T2 : struct

5.37.2.384 static void
OpenTK.Graphics.OpenGL.GL.GetCompressedTexImage< T2 > (
OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
[InAttribute, OutAttribute] T2 img[,]) [static]

Return a compressed texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, and GL_TEXTURE_3D GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

lod Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

img Returns the compressed texture image.

Type Constraints

T2 : struct

```
5.37.2.385 static void
OpenTK.Graphics.OpenGL.GL.GetCompressedTexImage< T2 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    [InAttribute, OutAttribute] T2 img[,,,] ) [static]
```

Return a compressed texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, and GL_TEXTURE_3D GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

lod Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

img Returns the compressed texture image.

Type Constraints

T2 : struct

```
5.37.2.386 static void
OpenTK.Graphics.OpenGL.GL.GetCompressedTexImage< T2 > (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    [InAttribute, OutAttribute] ref T2 img ) [static]
```

Return a compressed texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, and GL_TEXTURE_3D GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

lod Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

img Returns the compressed texture image.

Type Constraints

T2 : *struct*

5.37.2.387 static void OpenTK.Graphics.OpenGL.GL.GetConvolutionFilter
(OpenTK.Graphics.OpenGL.ConvolutionTarget
target, OpenTK.Graphics.OpenGL.PixelFormat *format*,
OpenTK.Graphics.OpenGL.PixelType *type*, [OutAttribute] IntPtr
image) [static]

Get current 1D or 2D convolution filter kernel.

Parameters

target The filter to be retrieved. Must be one of GL_CONVOLUTION_1D or GL_CONVOLUTION_2D.

format Format of the output image. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output image. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

image Pointer to storage for the output image.

5.37.2.388 `static void OpenTK.Graphics.OpenGL.GL.GetConvolutionFilter< T3 > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T3[] image) [static]`

Get current 1D or 2D convolution filter kernel.

Parameters

target The filter to be retrieved. Must be one of GL_CONVOLUTION_1D or GL_CONVOLUTION_2D.

format Format of the output image. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output image. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

image Pointer to storage for the output image.

Type Constraints

T3 : *struct*

5.37.2.389 `static void OpenTK.Graphics.OpenGL.GL.GetConvolutionFilter< T3 > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T3 image[,]) [static]`

Get current 1D or 2D convolution filter kernel.

Parameters

target The filter to be retrieved. Must be one of GL_CONVOLUTION_1D or GL_CONVOLUTION_2D.

format Format of the output image. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output image. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

image Pointer to storage for the output image.

Type Constraints

T3 : *struct*

5.37.2.390 static void OpenTK.Graphics.OpenGL.GL.GetConvolutionFilter< T3 > (OpenTK.Graphics.OpenGL.ConvolutionTarget *target*, OpenTK.Graphics.OpenGL.PixelFormat *format*, OpenTK.Graphics.OpenGL.PixelType *type*, [InAttribute, OutAttribute] T3 *image*[,]) [static]

Get current 1D or 2D convolution filter kernel.

Parameters

target The filter to be retrieved. Must be one of GL_CONVOLUTION_1D or GL_CONVOLUTION_2D.

format Format of the output image. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output image. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

image Pointer to storage for the output image.

Type Constraints

T3 : *struct*

5.37.2.391 `static void OpenTK.Graphics.OpenGL.GL.GetConvolutionFilter< T3 > (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T3 image) [static]`

Get current 1D or 2D convolution filter kernel.

Parameters

target The filter to be retrieved. Must be one of GL_CONVOLUTION_1D or GL_CONVOLUTION_2D.

format Format of the output image. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output image. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

image Pointer to storage for the output image.

Type Constraints

T3 : *struct*

5.37.2.392 `static void OpenTK.Graphics.OpenGL.GL.GetConvolutionParameter (OpenTK.Graphics.OpenGL.ConvolutionTarget target, OpenTK.Graphics.OpenGL.GetConvolutionParameterPName pname, [OutAttribute] Single @[] params) [static]`

Get convolution parameters.

Parameters

target The filter whose parameters are to be retrieved. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be retrieved. Must be one of GL_CONVOLUTION_BORDER_MODE, GL_CONVOLUTION_BORDER_COLOR, GL_CONVOLUTION_FILTER_SCALE, GL_CONVOLUTION_FILTER_BIAS, GL_CONVOLUTION_FORMAT, GL_CONVOLUTION_WIDTH, GL_CONVOLUTION_HEIGHT, GL_MAX_CONVOLUTION_WIDTH, or GL_MAX_CONVOLUTION_HEIGHT.

params Pointer to storage for the parameters to be retrieved.

5.37.2.393 static void

```
OpenTK.Graphics.OpenGL.GL.GetConvolutionParameter
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
  pname, [OutAttribute] out Single @ params ) [static]
```

Get convolution parameters.

Parameters

target The filter whose parameters are to be retrieved. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be retrieved. Must be one of GL_CONVOLUTION_BORDER_MODE, GL_CONVOLUTION_BORDER_COLOR, GL_CONVOLUTION_FILTER_SCALE, GL_CONVOLUTION_FILTER_BIAS, GL_CONVOLUTION_FORMAT, GL_CONVOLUTION_WIDTH, GL_CONVOLUTION_HEIGHT, GL_MAX_CONVOLUTION_WIDTH, or GL_MAX_CONVOLUTION_HEIGHT.

params Pointer to storage for the parameters to be retrieved.

5.37.2.394 static unsafe void

```
OpenTK.Graphics.OpenGL.GL.GetConvolutionParameter
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
  pname, [OutAttribute] Single *@ params ) [static]
```

Get convolution parameters.

Parameters

target The filter whose parameters are to be retrieved. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be retrieved. Must be one of GL_CONVOLUTION_BORDER_MODE, GL_CONVOLUTION_BORDER_COLOR, GL_CONVOLUTION_FILTER_SCALE, GL_CONVOLUTION_FILTER_BIAS, GL_CONVOLUTION_FORMAT, GL_CONVOLUTION_WIDTH, GL_CONVOLUTION_HEIGHT, GL_MAX_CONVOLUTION_WIDTH, or GL_MAX_CONVOLUTION_HEIGHT.

params Pointer to storage for the parameters to be retrieved.

5.37.2.395 static void

```
OpenTK.Graphics.OpenGL.GL.GetConvolutionParameter
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
  pname, [OutAttribute] Int32 @[] params ) [static]
```

Get convolution parameters.

Parameters

target The filter whose parameters are to be retrieved. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be retrieved. Must be one of GL_CONVOLUTION_BORDER_MODE, GL_CONVOLUTION_BORDER_COLOR, GL_CONVOLUTION_FILTER_SCALE, GL_CONVOLUTION_FILTER_BIAS, GL_CONVOLUTION_FORMAT, GL_CONVOLUTION_WIDTH, GL_CONVOLUTION_HEIGHT, GL_MAX_CONVOLUTION_WIDTH, or GL_MAX_CONVOLUTION_HEIGHT.

params Pointer to storage for the parameters to be retrieved.

5.37.2.396 static void

```
OpenTK.Graphics.OpenGL.GL.GetConvolutionParameter
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
  pname, [OutAttribute] out Int32 @ params ) [static]
```

Get convolution parameters.

Parameters

target The filter whose parameters are to be retrieved. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be retrieved. Must be one of GL_CONVOLUTION_BORDER_MODE, GL_CONVOLUTION_BORDER_COLOR, GL_CONVOLUTION_FILTER_SCALE, GL_CONVOLUTION_FILTER_BIAS, GL_CONVOLUTION_FORMAT, GL_CONVOLUTION_WIDTH, GL_CONVOLUTION_HEIGHT, GL_MAX_CONVOLUTION_WIDTH, or GL_MAX_CONVOLUTION_HEIGHT.

params Pointer to storage for the parameters to be retrieved.

5.37.2.397 static unsafe void

```
OpenTK.Graphics.OpenGL.GL.GetConvolutionParameter
( OpenTK.Graphics.OpenGL.ConvolutionTarget target,
  OpenTK.Graphics.OpenGL.GetConvolutionParameterPName
  pname, [OutAttribute] Int32 *@ params ) [static]
```

Get convolution parameters.

Parameters

target The filter whose parameters are to be retrieved. Must be one of GL_CONVOLUTION_1D, GL_CONVOLUTION_2D, or GL_SEPARABLE_2D.

pname The parameter to be retrieved. Must be one of GL_CONVOLUTION_BORDER_MODE, GL_CONVOLUTION_BORDER_COLOR, GL_CONVOLUTION_FILTER_SCALE, GL_CONVOLUTION_FILTER_BIAS, GL_CONVOLUTION_FORMAT, GL_CONVOLUTION_WIDTH, GL_CONVOLUTION_HEIGHT, GL_MAX_CONVOLUTION_WIDTH, or GL_MAX_CONVOLUTION_HEIGHT.

params Pointer to storage for the parameters to be retrieved.

5.37.2.398 static OpenTK.Graphics.OpenGL.ErrorCode

```
OpenTK.Graphics.OpenGL.GL.GetError ( ) [static]
```

Return error information.

```
5.37.2.399 static void OpenTK.Graphics.OpenGL.GL.GetHistogram (
  OpenTK.Graphics.OpenGL.HistogramTarget target, bool
  reset, OpenTK.Graphics.OpenGL.PixelFormat format,
  OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
  values ) [static]
```

Get histogram table.

Parameters

target Must be GL_HISTOGRAM.

reset If GL_TRUE, each component counter that is actually returned is reset to zero. (Other counters are unaffected.) If GL_FALSE, none of the counters in the histogram table is modified.

format The format of values to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type The type of values to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned histogram table.

```
5.37.2.400 static void OpenTK.Graphics.OpenGL.GL.GetHistogram<
            T4 > ( OpenTK.Graphics.OpenGL.HistogramTarget target,
            bool reset, OpenTK.Graphics.OpenGL.PixelFormat format,
            OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
            OutAttribute] T4[] values ) [static]
```

Get histogram table.

Parameters

target Must be GL_HISTOGRAM.

reset If GL_TRUE, each component counter that is actually returned is reset to zero. (Other counters are unaffected.) If GL_FALSE, none of the counters in the histogram table is modified.

format The format of values to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type The type of values to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV,

GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_5_5_5_1_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned histogram table.

Type Constraints

T4 : *struct*

5.37.2.401 static void OpenTK.Graphics.OpenGL.GL.GetHistogram< T4 > (OpenTK.Graphics.OpenGL.HistogramTarget *target*, bool *reset*, OpenTK.Graphics.OpenGL.PixelFormat *format*, OpenTK.Graphics.OpenGL.PixelType *type*, [InAttribute, OutAttribute] T4 *values*[,]) [static]

Get histogram table.

Parameters

target Must be GL_HISTOGRAM.

reset If GL_TRUE, each component counter that is actually returned is reset to zero. (Other counters are unaffected.) If GL_FALSE, none of the counters in the histogram table is modified.

format The format of values to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type The type of values to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_5_5_5_1_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned histogram table.

Type Constraints

T4 : *struct*

5.37.2.402 `static void OpenTK.Graphics.OpenGL.GL.GetHistogram< T4 > (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4 values[,]) [static]`

Get histogram table.

Parameters

target Must be GL_HISTOGRAM.

reset If GL_TRUE, each component counter that is actually returned is reset to zero. (Other counters are unaffected.) If GL_FALSE, none of the counters in the histogram table is modified.

format The format of values to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type The type of values to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned histogram table.

Type Constraints

T4 : *struct*

5.37.2.403 `static void OpenTK.Graphics.OpenGL.GL.GetHistogram< T4 > (OpenTK.Graphics.OpenGL.HistogramTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T4 values) [static]`

Get histogram table.

Parameters

target Must be GL_HISTOGRAM.

reset If GL_TRUE, each component counter that is actually returned is reset to zero. (Other counters are unaffected.) If GL_FALSE, none of the counters in the histogram table is modified.

format The format of values to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type The type of values to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned histogram table.

Type Constraints

T4 : struct

5.37.2.404 `static void OpenTK.Graphics.OpenGL.GL.GetHistogramParameter (OpenTK.Graphics.OpenGL.HistogramTarget target, OpenTK.Graphics.OpenGL.GetHistogramParameterPName pname, [OutAttribute] Single @[] params) [static]`

Get histogram parameters.

Parameters

target Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

pname The name of the parameter to be retrieved. Must be one of GL_HISTOGRAM_WIDTH, GL_HISTOGRAM_FORMAT, GL_HISTOGRAM_RED_SIZE, GL_HISTOGRAM_GREEN_SIZE, GL_HISTOGRAM_BLUE_SIZE, GL_HISTOGRAM_ALPHA_SIZE, GL_HISTOGRAM_LUMINANCE_SIZE, or GL_HISTOGRAM_SINK.

params Pointer to storage for the returned values.

5.37.2.405 `static void OpenTK.Graphics.OpenGL.GL.GetHistogramParameter
(OpenTK.Graphics.OpenGL.HistogramTarget target,
OpenTK.Graphics.OpenGL.GetHistogramParameterPName
pname, [OutAttribute] out Single @ params) [static]`

Get histogram parameters.

Parameters

target Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

pname The name of the parameter to be retrieved. Must be one of GL_HISTOGRAM_WIDTH, GL_HISTOGRAM_FORMAT, GL_HISTOGRAM_RED_SIZE, GL_HISTOGRAM_GREEN_SIZE, GL_HISTOGRAM_BLUE_SIZE, GL_HISTOGRAM_ALPHA_SIZE, GL_HISTOGRAM_LUMINANCE_SIZE, or GL_HISTOGRAM_SINK.

params Pointer to storage for the returned values.

5.37.2.406 `static unsafe void
OpenTK.Graphics.OpenGL.GL.GetHistogramParameter
(OpenTK.Graphics.OpenGL.HistogramTarget target,
OpenTK.Graphics.OpenGL.GetHistogramParameterPName
pname, [OutAttribute] Single *@ params) [static]`

Get histogram parameters.

Parameters

target Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

pname The name of the parameter to be retrieved. Must be one of GL_HISTOGRAM_WIDTH, GL_HISTOGRAM_FORMAT, GL_HISTOGRAM_RED_SIZE, GL_HISTOGRAM_GREEN_SIZE, GL_HISTOGRAM_BLUE_SIZE, GL_HISTOGRAM_ALPHA_SIZE, GL_HISTOGRAM_LUMINANCE_SIZE, or GL_HISTOGRAM_SINK.

params Pointer to storage for the returned values.

5.37.2.407 `static void OpenTK.Graphics.OpenGL.GL.GetHistogramParameter
(OpenTK.Graphics.OpenGL.HistogramTarget target,
OpenTK.Graphics.OpenGL.GetHistogramParameterPName
pname, [OutAttribute] Int32 @[] params) [static]`

Get histogram parameters.

Parameters

target Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

pname The name of the parameter to be retrieved. Must be one of GL_HISTOGRAM_WIDTH, GL_HISTOGRAM_FORMAT, GL_HISTOGRAM_RED_SIZE, GL_HISTOGRAM_GREEN_SIZE, GL_HISTOGRAM_BLUE_SIZE, GL_HISTOGRAM_ALPHA_SIZE, GL_HISTOGRAM_LUMINANCE_SIZE, or GL_HISTOGRAM_SINK.

params Pointer to storage for the returned values.

5.37.2.408 static void OpenTK.Graphics.OpenGL.GL.GetHistogramParameter (OpenTK.Graphics.OpenGL.HistogramTarget *target*, OpenTK.Graphics.OpenGL.GetHistogramParameterPName *pname*, [OutAttribute] out Int32 @ *params*) [static]

Get histogram parameters.

Parameters

target Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

pname The name of the parameter to be retrieved. Must be one of GL_HISTOGRAM_WIDTH, GL_HISTOGRAM_FORMAT, GL_HISTOGRAM_RED_SIZE, GL_HISTOGRAM_GREEN_SIZE, GL_HISTOGRAM_BLUE_SIZE, GL_HISTOGRAM_ALPHA_SIZE, GL_HISTOGRAM_LUMINANCE_SIZE, or GL_HISTOGRAM_SINK.

params Pointer to storage for the returned values.

5.37.2.409 static unsafe void OpenTK.Graphics.OpenGL.GL.GetHistogramParameter (OpenTK.Graphics.OpenGL.HistogramTarget *target*, OpenTK.Graphics.OpenGL.GetHistogramParameterPName *pname*, [OutAttribute] Int32 *@ *params*) [static]

Get histogram parameters.

Parameters

target Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

pname The name of the parameter to be retrieved. Must be one of GL_HISTOGRAM_WIDTH, GL_HISTOGRAM_FORMAT, GL_HISTOGRAM_RED_SIZE, GL_HISTOGRAM_GREEN_SIZE, GL_HISTOGRAM_BLUE_SIZE, GL_HISTOGRAM_ALPHA_SIZE, GL_HISTOGRAM_LUMINANCE_SIZE, or GL_HISTOGRAM_SINK.

params Pointer to storage for the returned values.

5.37.2.410 `static void OpenTK.Graphics.OpenGL.GL.GetLight
(OpenTK.Graphics.OpenGL.LightName light,
OpenTK.Graphics.OpenGL.LightParameter pname,
[OutAttribute] Single @[] params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT where ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.37.2.411 `static void OpenTK.Graphics.OpenGL.GL.GetLight
(OpenTK.Graphics.OpenGL.LightName light,
OpenTK.Graphics.OpenGL.LightParameter pname,
[OutAttribute] out Single @ params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT where ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.37.2.412 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetLight
(OpenTK.Graphics.OpenGL.LightName light,
OpenTK.Graphics.OpenGL.LightParameter pname,
[OutAttribute] Single *@ params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT where ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.37.2.413 `static void OpenTK.Graphics.OpenGL.GL.GetLight
(OpenTK.Graphics.OpenGL.LightName light,
OpenTK.Graphics.OpenGL.LightParameter pname,
[OutAttribute] Int32 @[] params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT where ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.37.2.414 `static void OpenTK.Graphics.OpenGL.GL.GetLight
(OpenTK.Graphics.OpenGL.LightName light,
OpenTK.Graphics.OpenGL.LightParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT where ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.37.2.415 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetLight
(OpenTK.Graphics.OpenGL.LightName light,
OpenTK.Graphics.OpenGL.LightParameter pname,
[OutAttribute] Int32 *@ params) [static]`

Return light source parameter values.

Parameters

light Specifies a light source. The number of possible lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT where ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a light source parameter for light. Accepted symbolic names are GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_POSITION, GL_SPOT_DIRECTION, GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION.

params Returns the requested data.

5.37.2.416 `static void OpenTK.Graphics.OpenGL.GL.GetMap`
 (`OpenTK.Graphics.OpenGL.MapTarget target`,
`OpenTK.Graphics.OpenGL.GetMapQuery query`, `[OutAttribute]`
`Double[] v`) `[static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.417 `static void OpenTK.Graphics.OpenGL.GL.GetMap`
 (`OpenTK.Graphics.OpenGL.MapTarget target`,
`OpenTK.Graphics.OpenGL.GetMapQuery query`, `[OutAttribute]`
`out Double v`) `[static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.418 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetMap
(OpenTK.Graphics.OpenGL.MapTarget target,
OpenTK.Graphics.OpenGL.GetMapQuery query, [OutAttribute]
Double * v) [static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.419 `static void OpenTK.Graphics.OpenGL.GL.GetMap
(OpenTK.Graphics.OpenGL.MapTarget target,
OpenTK.Graphics.OpenGL.GetMapQuery query, [OutAttribute]
Single[] v) [static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.420 `static void OpenTK.Graphics.OpenGL.GL.GetMap`
 (`OpenTK.Graphics.OpenGL.MapTarget target`,
`OpenTK.Graphics.OpenGL.GetMapQuery query`, `[OutAttribute]`
`out Single v`) `[static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.421 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetMap`
 (`OpenTK.Graphics.OpenGL.MapTarget target`,
`OpenTK.Graphics.OpenGL.GetMapQuery query`, `[OutAttribute]`
`Single * v`) `[static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.422 `static void OpenTK.Graphics.OpenGL.GL.GetMap
(OpenTK.Graphics.OpenGL.MapTarget target,
OpenTK.Graphics.OpenGL.GetMapQuery query, [OutAttribute]
Int32[] v) [static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.423 `static void OpenTK.Graphics.OpenGL.GL.GetMap
(OpenTK.Graphics.OpenGL.MapTarget target,
OpenTK.Graphics.OpenGL.GetMapQuery query, [OutAttribute]
out Int32 v) [static]`

Return evaluator parameters.

Parameters

target Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.

query Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.

v Returns the requested data.

5.37.2.424 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetMap`
 (`OpenTK.Graphics.OpenGL.MapTarget` *target*,
`OpenTK.Graphics.OpenGL.GetMapQuery` *query*, `[OutAttribute]`
`Int32 * v`) `[static]`

Return evaluator parameters.

Parameters

- target* Specifies the symbolic name of a map. Accepted values are GL_MAP1_COLOR_4, GL_MAP1_INDEX, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, GL_MAP1_TEXTURE_COORD_4, GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP2_COLOR_4, GL_MAP2_INDEX, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, GL_MAP2_TEXTURE_COORD_4, GL_MAP2_VERTEX_3, and GL_MAP2_VERTEX_4.
- query* Specifies which parameter to return. Symbolic names GL_COEFF, GL_ORDER, and GL_DOMAIN are accepted.
- v* Returns the requested data.

5.37.2.425 `static void OpenTK.Graphics.OpenGL.GL.GetMaterial`
 (`OpenTK.Graphics.OpenGL.MaterialFace` *face*,
`OpenTK.Graphics.OpenGL.MaterialParameter` *pname*,
`[OutAttribute] Single @[] params`) `[static]`

Return material parameters.

Parameters

- face* Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.
- pname* Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.
- params* Returns the requested data.

5.37.2.426 `static void OpenTK.Graphics.OpenGL.GL.GetMaterial`
 (`OpenTK.Graphics.OpenGL.MaterialFace` *face*,
`OpenTK.Graphics.OpenGL.MaterialParameter` *pname*,
`[OutAttribute] out Single @ params`) `[static]`

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

```
5.37.2.427 static unsafe void OpenTK.Graphics.OpenGL.GL.GetMaterial
( OpenTK.Graphics.OpenGL.MaterialFace face,
  OpenTK.Graphics.OpenGL.MaterialParameter pname,
  [OutAttribute] Single *@ params ) [static]
```

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

```
5.37.2.428 static void OpenTK.Graphics.OpenGL.GL.GetMaterial
( OpenTK.Graphics.OpenGL.MaterialFace face,
  OpenTK.Graphics.OpenGL.MaterialParameter pname,
  [OutAttribute] Int32 @[] params ) [static]
```

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

5.37.2.429 static void OpenTK.Graphics.OpenGL.GL.GetMaterial
(OpenTK.Graphics.OpenGL.MaterialFace *face*,
OpenTK.Graphics.OpenGL.MaterialParameter *pname*,
[OutAttribute] out Int32 @ *params*) [static]

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

5.37.2.430 static unsafe void OpenTK.Graphics.OpenGL.GL.GetMaterial
(OpenTK.Graphics.OpenGL.MaterialFace *face*,
OpenTK.Graphics.OpenGL.MaterialParameter *pname*,
[OutAttribute] Int32 *@ *params*) [static]

Return material parameters.

Parameters

face Specifies which of the two materials is being queried. GL_FRONT or GL_BACK are accepted, representing the front and back materials, respectively.

pname Specifies the material parameter to return. GL_AMBIENT, GL_DIFFUSE, GL_SPECULAR, GL_EMISSION, GL_SHININESS, and GL_COLOR_INDEXES are accepted.

params Returns the requested data.

5.37.2.431 static void OpenTK.Graphics.OpenGL.GL.GetMinmax (
OpenTK.Graphics.OpenGL.MinmaxTarget *target*, bool
reset, OpenTK.Graphics.OpenGL.PixelFormat *format*,
OpenTK.Graphics.OpenGL.PixelType *type*, [OutAttribute] IntPtr
values) [static]

Get minimum and maximum pixel values.

Parameters

target Must be GL_MINMAX.

reset If GL_TRUE, all entries in the minmax table that are actually returned are reset to their initial values. (Other entries are unaltered.) If GL_FALSE, the minmax table is unaltered.

format The format of the data to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

types The type of the data to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned values.

```
5.37.2.432 static void OpenTK.Graphics.OpenGL.GL.GetMinmax<
    T4 > ( OpenTK.Graphics.OpenGL.MinmaxTarget target,
    bool reset, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T4[] values ) [static]
```

Get minimum and maximum pixel values.

Parameters

target Must be GL_MINMAX.

reset If GL_TRUE, all entries in the minmax table that are actually returned are reset to their initial values. (Other entries are unaltered.) If GL_FALSE, the minmax table is unaltered.

format The format of the data to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

types The type of the data to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

INT_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned values.

Type Constraints

T4 : *struct*

5.37.2.433 `static void OpenTK.Graphics.OpenGL.GL.GetMinmax<T4> (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4 values[,]) [static]`

Get minimum and maximum pixel values.

Parameters

target Must be GL_MINMAX.

reset If GL_TRUE, all entries in the minmax table that are actually returned are reset to their initial values. (Other entries are unaltered.) If GL_FALSE, the minmax table is unaltered.

format The format of the data to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

types The type of the data to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned values.

Type Constraints

T4 : *struct*

5.37.2.434 `static void OpenTK.Graphics.OpenGL.GL.GetMinmax< T4 > (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4 values[,]) [static]`

Get minimum and maximum pixel values.

Parameters

target Must be GL_MINMAX.

reset If GL_TRUE, all entries in the minmax table that are actually returned are reset to their initial values. (Other entries are unaltered.) If GL_FALSE, the minmax table is unaltered.

format The format of the data to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

types The type of the data to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned values.

Type Constraints

T4 : *struct*

5.37.2.435 `static void OpenTK.Graphics.OpenGL.GL.GetMinmax< T4 > (OpenTK.Graphics.OpenGL.MinmaxTarget target, bool reset, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T4 values) [static]`

Get minimum and maximum pixel values.

Parameters

target Must be GL_MINMAX.

reset If GL_TRUE, all entries in the minmax table that are actually returned are reset to their initial values. (Other entries are unaltered.) If GL_FALSE, the minmax table is unaltered.

format The format of the data to be returned in values. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

types The type of the data to be returned in values. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

values A pointer to storage for the returned values.

Type Constraints

T4 : struct

5.37.2.436 `static void OpenTK.Graphics.OpenGL.GL.GetMinmaxParameter (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname, [OutAttribute] Single @[] params) [static]`

Get minmax parameters.

Parameters

target Must be GL_MINMAX.

pname The parameter to be retrieved. Must be one of GL_MINMAX_FORMAT or GL_MINMAX_SINK.

params A pointer to storage for the retrieved parameters.

5.37.2.437 `static void OpenTK.Graphics.OpenGL.GL.GetMinmaxParameter (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.GetMinmaxParameterPName pname, [OutAttribute] out Single @ params) [static]`

Get minmax parameters.

Parameters

target Must be GL_MINMAX.
pname The parameter to be retrieved. Must be one of GL_MINMAX_FORMAT or GL_MINMAX_SINK.
params A pointer to storage for the retrieved parameters.

5.37.2.438 static unsafe void
OpenTK.Graphics.OpenGL.GL.GetMinmaxParameter
 (OpenTK.Graphics.OpenGL.MinmaxTarget *target*,
 OpenTK.Graphics.OpenGL.GetMinmaxParameterPName *pname*,
 [OutAttribute] Single *@ *params*) [static]

Get minmax parameters.

Parameters

target Must be GL_MINMAX.
pname The parameter to be retrieved. Must be one of GL_MINMAX_FORMAT or GL_MINMAX_SINK.
params A pointer to storage for the retrieved parameters.

5.37.2.439 static void OpenTK.Graphics.OpenGL.GL.GetMinmaxParameter
 (OpenTK.Graphics.OpenGL.MinmaxTarget *target*,
 OpenTK.Graphics.OpenGL.GetMinmaxParameterPName *pname*,
 [OutAttribute] Int32 @[] *params*) [static]

Get minmax parameters.

Parameters

target Must be GL_MINMAX.
pname The parameter to be retrieved. Must be one of GL_MINMAX_FORMAT or GL_MINMAX_SINK.
params A pointer to storage for the retrieved parameters.

5.37.2.440 static void OpenTK.Graphics.OpenGL.GL.GetMinmaxParameter
 (OpenTK.Graphics.OpenGL.MinmaxTarget *target*,
 OpenTK.Graphics.OpenGL.GetMinmaxParameterPName *pname*,
 [OutAttribute] out Int32 @ *params*) [static]

Get minmax parameters.

Parameters

- target* Must be GL_MINMAX.
- pname* The parameter to be retrieved. Must be one of GL_MINMAX_FORMAT or GL_MINMAX_SINK.
- params* A pointer to storage for the retrieved parameters.

5.37.2.441 static unsafe void
OpenTK.Graphics.OpenGL.GL.GetMinmaxParameter
 (**OpenTK.Graphics.OpenGL.MinmaxTarget** *target*,
OpenTK.Graphics.OpenGL.GetMinmaxParameterPName *pname*,
 [OutAttribute] Int32 *@ *params*) [static]

Get minmax parameters.

Parameters

- target* Must be GL_MINMAX.
- pname* The parameter to be retrieved. Must be one of GL_MINMAX_FORMAT or GL_MINMAX_SINK.
- params* A pointer to storage for the retrieved parameters.

5.37.2.442 static void **OpenTK.Graphics.OpenGL.GL.GetPixelMap** (
OpenTK.Graphics.OpenGL.PixelMap *map*, [OutAttribute]
 Single[] *values*) [static]

Return the specified pixel map.

Parameters

- map* Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.
- data* Returns the pixel map contents.

5.37.2.443 static void **OpenTK.Graphics.OpenGL.GL.GetPixelMap** (
OpenTK.Graphics.OpenGL.PixelMap *map*, [OutAttribute] out
 Single *values*) [static]

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.444 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelFormat map, [OutAttribute] Single * values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.445 `static void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelFormat map, [OutAttribute] Int32[] values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.446 `static void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelFormat map, [OutAttribute] out Int32[] values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.447 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetPixelMap (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Int32 * values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.448 `static void OpenTK.Graphics.OpenGL.GL.GetPixelMap (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] UInt32[] values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.449 `static void OpenTK.Graphics.OpenGL.GL.GetPixelMap (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] out UInt32 values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.450 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] UInt32 * values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.451 `static void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Int16[] values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.452 `static void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] out Int16 values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.453 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetPixelMap (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] Int16 * values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.454 `static void OpenTK.Graphics.OpenGL.GL.GetPixelMap (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] UInt16[] values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.455 `static void OpenTK.Graphics.OpenGL.GL.GetPixelMap (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] out UInt16 values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.456 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetPixelFormat (OpenTK.Graphics.OpenGL.PixelMap map, [OutAttribute] UInt16 * values) [static]`

Return the specified pixel map.

Parameters

map Specifies the name of the pixel map to return. Accepted values are GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, and GL_PIXEL_MAP_A_TO_A.

data Returns the pixel map contents.

5.37.2.457 `static void OpenTK.Graphics.OpenGL.GL.GetPointer (OpenTK.Graphics.OpenGL.GetPointervPName pname, [OutAttribute] IntPtr @ params) [static]`

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

5.37.2.458 `static void OpenTK.Graphics.OpenGL.GL.GetPointer< T1 >
 (OpenTK.Graphics.OpenGL.GetPointervPName pname,
 [InAttribute, OutAttribute] T1 @[] params) [static]`

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

Type Constraints

T1 : *struct*

5.37.2.459 `static void OpenTK.Graphics.OpenGL.GL.GetPointer< T1 >
 (OpenTK.Graphics.OpenGL.GetPointervPName pname,
 [InAttribute, OutAttribute] T1 @ params[,]) [static]`

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

Type Constraints

T1 : *struct*

```

5.37.2.460 static void OpenTK.Graphics.OpenGL.GL.GetPointer< T1 >
( OpenTK.Graphics.OpenGL.GetPointervPName pname,
  [InAttribute, OutAttribute] T1 @ params[, ] ) [static]

```

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

Type Constraints

T1 : struct

```

5.37.2.461 static void OpenTK.Graphics.OpenGL.GL.GetPointer< T1 >
( OpenTK.Graphics.OpenGL.GetPointervPName pname,
  [InAttribute, OutAttribute] ref T1 @ params ) [static]

```

Return the address of the specified pointer.

Parameters

pname Specifies the array or buffer pointer to be returned. Symbolic constants GL_COLOR_ARRAY_POINTER, GL_EDGE_FLAG_ARRAY_POINTER, GL_FOG_COORD_ARRAY_POINTER, GL_FEEDBACK_BUFFER_POINTER, GL_INDEX_ARRAY_POINTER, GL_NORMAL_ARRAY_POINTER, GL_SECONDARY_COLOR_ARRAY_POINTER, GL_SELECTION_BUFFER_POINTER, GL_TEXTURE_COORD_ARRAY_POINTER, or GL_VERTEX_ARRAY_POINTER are accepted.

params Returns the pointer value specified by *pname*.

Type Constraints

T1 : struct

5.37.2.462 `static void OpenTK.Graphics.OpenGL.GL.GetPolygonStipple ([OutAttribute] Byte[] mask) [static]`

Return the polygon stipple pattern.

Parameters

pattern Returns the stipple pattern. The initial value is all 1's.

5.37.2.463 `static void OpenTK.Graphics.OpenGL.GL.GetPolygonStipple ([OutAttribute] out Byte mask) [static]`

Return the polygon stipple pattern.

Parameters

pattern Returns the stipple pattern. The initial value is all 1's.

5.37.2.464 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetPolygonStipple ([OutAttribute] Byte * mask) [static]`

Return the polygon stipple pattern.

Parameters

pattern Returns the stipple pattern. The initial value is all 1's.

5.37.2.465 `static void OpenTK.Graphics.OpenGL.GL.GetProgram (Int32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname, [OutAttribute] Int32 @[] params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.37.2.466 `static void OpenTK.Graphics.OpenGL.GL.GetProgram (Int32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname, [OutAttribute] out Int32 @ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.37.2.467 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetProgram (Int32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname, [OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.37.2.468 `static void OpenTK.Graphics.OpenGL.GL.GetProgram (UInt32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname, [OutAttribute] Int32 @[] params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.37.2.469 `static void OpenTK.Graphics.OpenGL.GL.GetProgram (UInt32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname, [OutAttribute] out Int32 @ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.37.2.470 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetProgram (UInt32 program, OpenTK.Graphics.OpenGL.ProgramParameter pname, [OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a program object.

Parameters

program Specifies the program object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_DELETE_STATUS, GL_LINK_STATUS, GL_VALIDATE_STATUS, GL_INFO_LOG_LENGTH, GL_ATTACHED_SHADERS, GL_ACTIVE_ATTRIBUTES, GL_ACTIVE_ATTRIBUTE_MAX_LENGTH, GL_ACTIVE_UNIFORMS, GL_ACTIVE_UNIFORM_MAX_LENGTH.

params Returns the requested object parameter.

5.37.2.471 `static void OpenTK.Graphics.OpenGL.GL.GetProgramInfoLog (Int32 program, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.37.2.472 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetProgramInfoLog (Int32 program, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.37.2.473 `static void OpenTK.Graphics.OpenGL.GL.GetProgramInfoLog (UInt32 program, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.37.2.474 static unsafe void
OpenTK.Graphics.OpenGL.GL.GetProgramInfoLog (UInt32
program, Int32 bufSize, [OutAttribute] Int32 * length,
[OutAttribute] StringBuilder infoLog) [static]

Returns the information log for a program object.

Parameters

program Specifies the program object whose information log is to be queried.

maxLength Specifies the size of the character buffer for storing the returned information log.

length Returns the length of the string returned in infoLog (excluding the null terminator).

infoLog Specifies an array of characters that is used to return the information log.

5.37.2.475 static void OpenTK.Graphics.OpenGL.GL.GetQuery
(OpenTK.Graphics.OpenGL.QueryTarget target,
OpenTK.Graphics.OpenGL.GetQueryParam pname,
[OutAttribute] Int32 @[] params) [static]

Return parameters of a query object target.

Parameters

target Specifies a query object target. Must be GL_SAMPLES_PASSED.

pname Specifies the symbolic name of a query object target parameter. Accepted values are GL_CURRENT_QUERY or GL_QUERY_COUNTER_BITS.

params Returns the requested data.

5.37.2.476 static void OpenTK.Graphics.OpenGL.GL.GetQuery
(OpenTK.Graphics.OpenGL.QueryTarget target,
OpenTK.Graphics.OpenGL.GetQueryParam pname,
[OutAttribute] out Int32 @ params) [static]

Return parameters of a query object target.

Parameters

- target* Specifies a query object target. Must be GL_SAMPLES_PASSED.
- pname* Specifies the symbolic name of a query object target parameter. Accepted values are GL_CURRENT_QUERY or GL_QUERY_COUNTER_BITS.
- params* Returns the requested data.

5.37.2.477 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetQuery
(OpenTK.Graphics.OpenGL.QueryTarget target,
OpenTK.Graphics.OpenGL.GetQueryParam pname,
[OutAttribute] Int32 *@ params) [static]`

Return parameters of a query object target.

Parameters

- target* Specifies a query object target. Must be GL_SAMPLES_PASSED.
- pname* Specifies the symbolic name of a query object target parameter. Accepted values are GL_CURRENT_QUERY or GL_QUERY_COUNTER_BITS.
- params* Returns the requested data.

5.37.2.478 `static void OpenTK.Graphics.OpenGL.GL.GetQueryObject (Int32
id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,
[OutAttribute] Int32 @[] params) [static]`

Return parameters of a query object.

Parameters

- id* Specifies the name of a query object.
- pname* Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.
- params* Returns the requested data.

5.37.2.479 `static void OpenTK.Graphics.OpenGL.GL.GetQueryObject (Int32
id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname,
[OutAttribute] out Int32 @ params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.480 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetQueryObject (Int32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname, [OutAttribute] Int32 *@ params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.481 `static void OpenTK.Graphics.OpenGL.GL.GetQueryObject (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname, [OutAttribute] Int32 @[] params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.482 `static void OpenTK.Graphics.OpenGL.GL.GetQueryObject (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname, [OutAttribute] out Int32 @ params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.483 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetQueryObject (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname, [OutAttribute] Int32 *@ params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.484 `static void OpenTK.Graphics.OpenGL.GL.GetQueryObject (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname, [OutAttribute] UInt32 @[] params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.485 `static void OpenTK.Graphics.OpenGL.GL.GetQueryObject (UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam pname, [OutAttribute] out UInt32 @ params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.486 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetQueryObject
(UInt32 id, OpenTK.Graphics.OpenGL.GetQueryObjectParam
pname, [OutAttribute] UInt32 *@ params) [static]`

Return parameters of a query object.

Parameters

id Specifies the name of a query object.

pname Specifies the symbolic name of a query object parameter. Accepted values are GL_QUERY_RESULT or GL_QUERY_RESULT_AVAILABLE.

params Returns the requested data.

5.37.2.487 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter
(OpenTK.Graphics.OpenGL.SeparableTarget target,
OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
row, [OutAttribute] IntPtr column, [OutAttribute] IntPtr span)
[static]`

Get separable convolution filter kernel images.

Parameters

target The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

5.37.2.488 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T3, T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T3[] row, [InAttribute, OutAttribute] T4 column[,], [InAttribute, OutAttribute] T5 span[,]) [static]`

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.
- format** Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.
- type** Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.
- row** Pointer to storage for the row filter image.
- column** Pointer to storage for the column filter image.
- span** Pointer to storage for the span filter image (currently unused).

Type Constraints

T3 : *struct*

T4 : *struct*

T5 : *struct*

5.37.2.489 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T3, T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T3 row[,], [InAttribute, OutAttribute] T4 column[,], [InAttribute, OutAttribute] T5 span[,]) [static]`

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.
- format** Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.
- type** Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.
- row** Pointer to storage for the row filter image.
- column** Pointer to storage for the column filter image.
- span** Pointer to storage for the span filter image (currently unused).

Type Constraints

- T3** : *struct*
- T4** : *struct*
- T5** : *struct*

5.37.2.490 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T3, T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T3 row[,], [InAttribute, OutAttribute] T4 column[,], [InAttribute, OutAttribute] T5 span[,]) [static]`

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.
- format** Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T3 : struct

T4 : struct

T5 : struct

```
5.37.2.491 static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter<
    T3, T4, T5 > ( OpenTK.Graphics.OpenGL.SeparableTarget
    target, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] ref T3 row, [InAttribute, OutAttribute] T4
    column[,], [InAttribute, OutAttribute] T5 span[,]) [static]
```

Get separable convolution filter kernel images.

Parameters

target The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T3 : *struct*

T4 : *struct*

T5 : *struct*

5.37.2.492 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T4, T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr row, [InAttribute, OutAttribute] T4[] column, [InAttribute, OutAttribute] T5 span[,]) [static]`

Get separable convolution filter kernel images.

Parameters

target The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T4 : *struct*

T5 : *struct*

```

5.37.2.493 static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter<
    T4, T5 > ( OpenTK.Graphics.OpenGL.SeparableTarget
    target, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
    row, [InAttribute, OutAttribute] T4 column[,], [InAttribute,
    OutAttribute] T5 span[,]) [static]

```

Get separable convolution filter kernel images.

Parameters

target The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T4 : struct

T5 : struct

```

5.37.2.494 static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter<
    T4, T5 > ( OpenTK.Graphics.OpenGL.SeparableTarget
    target, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
    row, [InAttribute, OutAttribute] T4 column[,], [InAttribute,
    OutAttribute] T5 span[,]) [static]

```

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.
- format** Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.
- type** Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.
- row** Pointer to storage for the row filter image.
- column** Pointer to storage for the column filter image.
- span** Pointer to storage for the span filter image (currently unused).

Type Constraints

T4 : *struct*

T5 : *struct*

```
5.37.2.495 static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter<
    T4, T5 > ( OpenTK.Graphics.OpenGL.SeparableTarget
    target, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
    row, [InAttribute, OutAttribute] ref T4 column, [InAttribute,
    OutAttribute] T5 span[,]) [static]
```

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.
- format** Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.
- type** Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT,

GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T4 : *struct*

T5 : *struct*

```
5.37.2.496 static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter<
    T5 > ( OpenTK.Graphics.OpenGL.SeparableTarget
    target, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
    row, [OutAttribute] IntPtr column, [InAttribute, OutAttribute]
    T5[] span ) [static]
```

Get separable convolution filter kernel images.

Parameters

target The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T5 : *struct*

5.37.2.497 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr row, [OutAttribute] IntPtr column, [InAttribute, OutAttribute] T5 span[,]) [static]`

Get separable convolution filter kernel images.

Parameters

target The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T5 : *struct*

5.37.2.498 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr row, [OutAttribute] IntPtr column, [InAttribute, OutAttribute] T5 span[,]) [static]`

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.
- format** Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.
- type** Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.
- row** Pointer to storage for the row filter image.
- column** Pointer to storage for the column filter image.
- span** Pointer to storage for the span filter image (currently unused).

Type Constraints

T5 : struct

5.37.2.499 `static void OpenTK.Graphics.OpenGL.GL.GetSeparableFilter< T5 > (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr row, [OutAttribute] IntPtr column, [InAttribute, OutAttribute] ref T5 span) [static]`

Get separable convolution filter kernel images.

Parameters

- target** The separable filter to be retrieved. Must be GL_SEPARABLE_2D.

format Format of the output images. Must be one of GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, or GL_LUMINANCE_ALPHA.

type Data type of components in the output images. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to storage for the row filter image.

column Pointer to storage for the column filter image.

span Pointer to storage for the span filter image (currently unused).

Type Constraints

T5 : struct

5.37.2.500 `static void OpenTK.Graphics.OpenGL.GL.GetShader (Int32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname, [OutAttribute] Int32 @[] params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.37.2.501 `static void OpenTK.Graphics.OpenGL.GL.GetShader (Int32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname, [OutAttribute] out Int32 @ params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.37.2.502 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetShader (Int32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname, [OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.37.2.503 `static void OpenTK.Graphics.OpenGL.GL.GetShader (UInt32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname, [OutAttribute] Int32 @[] params) [static]`

Returns a parameter from a shader object.

Parameters

shader Specifies the shader object to be queried.

pname Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.

params Returns the requested object parameter.

5.37.2.504 `static void OpenTK.Graphics.OpenGL.GL.GetShader (UInt32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname, [OutAttribute] out Int32 @ params) [static]`

Returns a parameter from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- pname* Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.
- params* Returns the requested object parameter.

5.37.2.505 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetShader (UInt32 shader, OpenTK.Graphics.OpenGL.ShaderParameter pname, [OutAttribute] Int32 *@ params) [static]`

Returns a parameter from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- pname* Specifies the object parameter. Accepted symbolic names are GL_SHADER_TYPE, GL_DELETE_STATUS, GL_COMPILE_STATUS, GL_INFO_LOG_LENGTH, GL_SHADER_SOURCE_LENGTH.
- params* Returns the requested object parameter.

5.37.2.506 `static void OpenTK.Graphics.OpenGL.GL.GetShaderInfoLog (Int32 shader, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.37.2.507 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetShaderInfoLog (Int32 shader, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.37.2.508 `static void OpenTK.Graphics.OpenGL.GL.GetShaderInfoLog (UInt32 shader, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.37.2.509 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetShaderInfoLog (UInt32 shader, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder infoLog) [static]`

Returns the information log for a shader object.

Parameters

- shader* Specifies the shader object whose information log is to be queried.
- maxLength* Specifies the size of the character buffer for storing the returned information log.
- length* Returns the length of the string returned in infoLog (excluding the null terminator).
- infoLog* Specifies an array of characters that is used to return the information log.

5.37.2.510 `static void OpenTK.Graphics.OpenGL.GL.GetShaderSource (Int32 shader, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.37.2.511 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetShaderSource (Int32 shader, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.37.2.512 `static void OpenTK.Graphics.OpenGL.GL.GetShaderSource (UInt32 shader, Int32 bufSize, [OutAttribute] out Int32 length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.37.2.513 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetShaderSource (UInt32 shader, Int32 bufSize, [OutAttribute] Int32 * length, [OutAttribute] StringBuilder source) [static]`

Returns the source code string from a shader object.

Parameters

- shader* Specifies the shader object to be queried.
- bufSize* Specifies the size of the character buffer for storing the returned source code string.
- length* Returns the length of the string returned in source (excluding the null terminator).
- source* Specifies an array of characters that is used to return the source code string.

5.37.2.514 `static System.String OpenTK.Graphics.OpenGL.GL.GetString (OpenTK.Graphics.OpenGL.StringName name) [static]`

Return a string describing the current [GL](#) connection.

Parameters

- name* Specifies a symbolic constant, one of GL_VENDOR, GL_RENDERER, GL_VERSION, GL_SHADING_LANGUAGE_VERSION, or GL_EXTENSIONS.

5.37.2.515 `static System.String OpenTK.Graphics.OpenGL.GL.GetString (OpenTK.Graphics.OpenGL.StringName name, Int32 index) [static]`

Return a string describing the current [GL](#) connection.

Parameters

- name* Specifies a symbolic constant, one of GL_VENDOR, GL_RENDERER, GL_VERSION, GL_SHADING_LANGUAGE_VERSION, or GL_EXTENSIONS.

5.37.2.516 `static System.String OpenTK.Graphics.OpenGL.GL.GetString (OpenTK.Graphics.OpenGL.StringName name, UInt32 index) [static]`

Return a string describing the current [GL](#) connection.

Parameters

name Specifies a symbolic constant, one of GL_VENDOR, GL_RENDERER, GL_VERSION, GL_SHADING_LANGUAGE_VERSION, or GL_EXTENSIONS.

5.37.2.517 `static void OpenTK.Graphics.OpenGL.GL.GetTexEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname,
[OutAttribute] Single @[] params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.37.2.518 `static void OpenTK.Graphics.OpenGL.GL.GetTexEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname,
[OutAttribute] out Single @ params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA,

GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.37.2.519 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname,
[OutAttribute] Single *@ params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.37.2.520 `static void OpenTK.Graphics.OpenGL.GL.GetTexEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname,
[OutAttribute] Int32 @[] params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB,

GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.37.2.521 `static void OpenTK.Graphics.OpenGL.GL.GetTexEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.37.2.522 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname,
[OutAttribute] Int32 * @ params) [static]`

Return texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL, or GL_POINT_SPRITE.

pname Specifies the symbolic name of a texture environment parameter. Accepted values are GL_TEXTURE_ENV_MODE, GL_TEXTURE_ENV_COLOR, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

params Returns the requested data.

5.37.2.523 `static void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] Double @[] params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.

params Returns the requested data.

5.37.2.524 `static void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] out Double @ params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.

params Returns the requested data.

5.37.2.525 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] Double *@ params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.

params Returns the requested data.

5.37.2.526 `static void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] Single @[] params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.

params Returns the requested data.

5.37.2.527 `static void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] out Single @ params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.

params Returns the requested data.

5.37.2.528 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] Single *@ params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.
pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.
params Returns the requested data.

5.37.2.529 `static void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] Int32 @[] params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.
pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.
params Returns the requested data.

5.37.2.530 `static void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Return texture coordinate generation parameters.

Parameters

coord Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.
pname Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.
params Returns the requested data.

5.37.2.531 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname,
[OutAttribute] Int32 *@ params) [static]`

Return texture coordinate generation parameters.

Parameters

- coord* Specifies a texture coordinate. Must be GL_S, GL_T, GL_R, or GL_Q.
- pname* Specifies the symbolic name of the value(s) to be returned. Must be either GL_TEXTURE_GEN_MODE or the name of one of the texture generation plane equations: GL_OBJECT_PLANE or GL_EYE_PLANE.
- params* Returns the requested data.

5.37.2.532 `static void OpenTK.Graphics.OpenGL.GL.GetTexImage
(OpenTK.Graphics.OpenGL.TextureTarget target, Int32
level, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr
pixels) [static]`

Return a texture image.

Parameters

- target* Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.
- level* Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.
- format* Specifies a pixel format for the returned data. The supported formats are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.
- type* Specifies a pixel type for the returned data. The supported types are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, and GL_UNSIGNED_INT_8_8_8_8_REV.

GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

img Returns the texture image. Should be a pointer to an array of the type specified by type.

5.37.2.533 `static void OpenTK.Graphics.OpenGL.GL.GetTexImage<T4> (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4[] pixels) [static]`

Return a texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

format Specifies a pixel format for the returned data. The supported formats are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies a pixel type for the returned data. The supported types are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

img Returns the texture image. Should be a pointer to an array of the type specified by type.

Type Constraints

T4 : *struct*

5.37.2.534 `static void OpenTK.Graphics.OpenGL.GL.GetTexImage<T4> (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T4 pixels[,J]) [static]`

Return a texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

format Specifies a pixel format for the returned data. The supported formats are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies a pixel type for the returned data. The supported types are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

img Returns the texture image. Should be a pointer to an array of the type specified by type.

Type Constraints

T4 : *struct*

```

5.37.2.535 static void OpenTK.Graphics.OpenGL.GL.GetTexImage<
            T4 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
                  Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format,
                  OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
                  OutAttribute] T4 pixels[,]) [static]

```

Return a texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

format Specifies a pixel format for the returned data. The supported formats are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies a pixel type for the returned data. The supported types are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_REV.

img Returns the texture image. Should be a pointer to an array of the type specified by type.

Type Constraints

T4 : struct

5.37.2.536 `static void OpenTK.Graphics.OpenGL.GL.GetTexImage< T4 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T4 pixels) [static]`

Return a texture image.

Parameters

target Specifies which texture is to be obtained. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, and GL_TEXTURE_CUBE_MAP_NEGATIVE_Z are accepted.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

format Specifies a pixel format for the returned data. The supported formats are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies a pixel type for the returned data. The supported types are GL_UNSIGNED_BYTE, GL_BYTE, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

img Returns the texture image. Should be a pointer to an array of the type specified by type.

Type Constraints

T4 : struct

5.37.2.537 `static void OpenTK.Graphics.OpenGL.GL.GetTexLevelParameter (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname, [OutAttribute] Single @[] params) [static]`

Return texture parameter values for a specific level of detail.

Parameters

target Specifies the symbolic name of the target texture, either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_PROXY_TEXTURE_1D, GL_PROXY_TEXTURE_2D, GL_PROXY_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_WIDTH, GL_TEXTURE_HEIGHT, GL_TEXTURE_DEPTH, GL_TEXTURE_INTERNAL_FORMAT, GL_TEXTURE_BORDER, GL_TEXTURE_RED_SIZE, GL_TEXTURE_GREEN_SIZE, GL_TEXTURE_BLUE_SIZE, GL_TEXTURE_ALPHA_SIZE, GL_TEXTURE_LUMINANCE_SIZE, GL_TEXTURE_INTENSITY_SIZE, GL_TEXTURE_DEPTH_SIZE, GL_TEXTURE_COMPRESSED, and GL_TEXTURE_COMPRESSED_IMAGE_SIZE are accepted.

params Returns the requested data.

5.37.2.538 **static void OpenTK.Graphics.OpenGL.GL.GetTexLevelParameter (**
OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
OpenTK.Graphics.OpenGL.GetTextureParameter pname,
[OutAttribute] out Single @ params) [static]

Return texture parameter values for a specific level of detail.

Parameters

target Specifies the symbolic name of the target texture, either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_PROXY_TEXTURE_1D, GL_PROXY_TEXTURE_2D, GL_PROXY_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_WIDTH, GL_TEXTURE_HEIGHT, GL_TEXTURE_DEPTH, GL_TEXTURE_INTERNAL_FORMAT, GL_TEXTURE_BORDER, GL_TEXTURE_RED_SIZE, GL_TEXTURE_GREEN_SIZE,

GL_TEXTURE_BLUE_SIZE, GL_TEXTURE_ALPHA_SIZE, GL_TEXTURE_LUMINANCE_SIZE, GL_TEXTURE_INTENSITY_SIZE, GL_TEXTURE_DEPTH_SIZE, GL_TEXTURE_COMPRESSED, and GL_TEXTURE_COMPRESSED_IMAGE_SIZE are accepted.

params Returns the requested data.

5.37.2.539 static unsafe void

OpenTK.Graphics.OpenGL.GL.GetTexLevelParameter (
OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
OpenTK.Graphics.OpenGL.GetTextureParameter *pname*,
[OutAttribute] Single *@ *params*) [static]

Return texture parameter values for a specific level of detail.

Parameters

target Specifies the symbolic name of the target texture, either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_PROXY_TEXTURE_1D, GL_PROXY_TEXTURE_2D, GL_PROXY_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_WIDTH, GL_TEXTURE_HEIGHT, GL_TEXTURE_DEPTH, GL_TEXTURE_INTERNAL_FORMAT, GL_TEXTURE_BORDER, GL_TEXTURE_RED_SIZE, GL_TEXTURE_GREEN_SIZE, GL_TEXTURE_BLUE_SIZE, GL_TEXTURE_ALPHA_SIZE, GL_TEXTURE_LUMINANCE_SIZE, GL_TEXTURE_INTENSITY_SIZE, GL_TEXTURE_DEPTH_SIZE, GL_TEXTURE_COMPRESSED, and GL_TEXTURE_COMPRESSED_IMAGE_SIZE are accepted.

params Returns the requested data.

5.37.2.540 static void OpenTK.Graphics.OpenGL.GL.GetTexLevelParameter (

OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
OpenTK.Graphics.OpenGL.GetTextureParameter *pname*,
[OutAttribute] Int32 @[] *params*) [static]

Return texture parameter values for a specific level of detail.

Parameters

target Specifies the symbolic name of the target texture, either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_PROXY_TEXTURE_1D, GL_PROXY_TEXTURE_2D, GL_PROXY_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_WIDTH, GL_TEXTURE_HEIGHT, GL_TEXTURE_DEPTH, GL_TEXTURE_INTERNAL_FORMAT, GL_TEXTURE_BORDER, GL_TEXTURE_RED_SIZE, GL_TEXTURE_GREEN_SIZE, GL_TEXTURE_BLUE_SIZE, GL_TEXTURE_ALPHA_SIZE, GL_TEXTURE_LUMINANCE_SIZE, GL_TEXTURE_INTENSITY_SIZE, GL_TEXTURE_DEPTH_SIZE, GL_TEXTURE_COMPRESSED, and GL_TEXTURE_COMPRESSED_IMAGE_SIZE are accepted.

params Returns the requested data.

5.37.2.541 `static void OpenTK.Graphics.OpenGL.GL.GetTexLevelParameter (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.GetTextureParameter pname, [OutAttribute] out Int32 @ params) [static]`

Return texture parameter values for a specific level of detail.

Parameters

target Specifies the symbolic name of the target texture, either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_PROXY_TEXTURE_1D, GL_PROXY_TEXTURE_2D, GL_PROXY_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_WIDTH, GL_TEXTURE_HEIGHT, GL_TEXTURE_DEPTH, GL_TEXTURE_INTERNAL_FORMAT, GL_TEXTURE_BORDER, GL_TEXTURE_RED_SIZE, GL_TEXTURE_GREEN_SIZE,

GL_TEXTURE_BLUE_SIZE, GL_TEXTURE_ALPHA_SIZE, GL_TEXTURE_LUMINANCE_SIZE, GL_TEXTURE_INTENSITY_SIZE, GL_TEXTURE_DEPTH_SIZE, GL_TEXTURE_COMPRESSED, and GL_TEXTURE_COMPRESSED_IMAGE_SIZE are accepted.

params Returns the requested data.

5.37.2.542 static unsafe void

OpenTK.Graphics.OpenGL.GL.GetTexLevelParameter (
OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
OpenTK.Graphics.OpenGL.GetTextureParameter *pname*,
[OutAttribute] Int32 *@ *params*) [static]

Return texture parameter values for a specific level of detail.

Parameters

target Specifies the symbolic name of the target texture, either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, GL_PROXY_TEXTURE_1D, GL_PROXY_TEXTURE_2D, GL_PROXY_TEXTURE_3D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number of the desired image. Level 0 is the base image level. Level is the th mipmap reduction image.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_WIDTH, GL_TEXTURE_HEIGHT, GL_TEXTURE_DEPTH, GL_TEXTURE_INTERNAL_FORMAT, GL_TEXTURE_BORDER, GL_TEXTURE_RED_SIZE, GL_TEXTURE_GREEN_SIZE, GL_TEXTURE_BLUE_SIZE, GL_TEXTURE_ALPHA_SIZE, GL_TEXTURE_LUMINANCE_SIZE, GL_TEXTURE_INTENSITY_SIZE, GL_TEXTURE_DEPTH_SIZE, GL_TEXTURE_COMPRESSED, and GL_TEXTURE_COMPRESSED_IMAGE_SIZE are accepted.

params Returns the requested data.

5.37.2.543 static void OpenTK.Graphics.OpenGL.GL.GetTexParameter
(OpenTK.Graphics.OpenGL.TextureTarget *target*,
OpenTK.Graphics.OpenGL.GetTextureParameter *pname*,
[OutAttribute] Single @[] *params*) [static]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.37.2.544 **static void OpenTK.Graphics.OpenGL.GL.GetTexParameter**
 (**OpenTK.Graphics.OpenGL.TextureTarget target**,
OpenTK.Graphics.OpenGL.GetTextureParameter pname,
 [OutAttribute] out Single @ **params**) [**static**]

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.37.2.545 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexParameter
(OpenTK.Graphics.OpenGL.TextureTarget target,
OpenTK.Graphics.OpenGL.GetTextureParameter pname,
[OutAttribute] Single *@ params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.37.2.546 `static void OpenTK.Graphics.OpenGL.GL.GetTexParameter
(OpenTK.Graphics.OpenGL.TextureTarget target,
OpenTK.Graphics.OpenGL.GetTextureParameter pname,
[OutAttribute] Int32 @[] params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.37.2.547 `static void OpenTK.Graphics.OpenGL.GL.GetTexParameter
(OpenTK.Graphics.OpenGL.TextureTarget target,
OpenTK.Graphics.OpenGL.GetTextureParameter pname,
[OutAttribute] out Int32 @ params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.37.2.548 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetTexParameter
(OpenTK.Graphics.OpenGL.TextureTarget target,
OpenTK.Graphics.OpenGL.GetTextureParameter pname,
[OutAttribute] Int32 *@ params) [static]`

Return texture parameter values.

Parameters

target Specifies the symbolic name of the target texture. GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, and GL_TEXTURE_CUBE_MAP are accepted.

pname Specifies the symbolic name of a texture parameter. GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_BORDER_COLOR, GL_TEXTURE_PRIORITY, GL_TEXTURE_RESIDENT, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, and GL_GENERATE_MIPMAP are accepted.

params Returns the texture parameters.

5.37.2.549 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (Int32 program, Int32 location, [OutAttribute] Single @[] params) [static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.37.2.550 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (Int32 program, Int32 location, [OutAttribute] out Single @ params) [static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.37.2.551 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetUniform (Int32 program, Int32 location, [OutAttribute] Single *@ params) [static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.37.2.552 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] Single @[] params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.553 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] out Single @ params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.554 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] Single *@ params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.555 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (Int32 program, Int32 location, [OutAttribute] Int32 @[] params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.556 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] out Int32 @ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.37.2.557 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetUniform (Int32
program, Int32 location, [OutAttribute] Int32 *@ params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.37.2.558 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32
program, Int32 location, [OutAttribute] Int32 @[] params)
[static]`

Returns the value of a uniform variable.

Parameters

program Specifies the program object to be queried.

location Specifies the location of the uniform variable to be queried.

params Returns the value of the specified uniform variable.

5.37.2.559 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32
program, Int32 location, [OutAttribute] out Int32 @ params)
[static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.560 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] Int32 *@ params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.561 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] UInt32 @[] params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.562 `static void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] out UInt32 @ params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.563 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetUniform (UInt32 program, Int32 location, [OutAttribute] UInt32 *@params) [static]`

Returns the value of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- location* Specifies the location of the uniform variable to be queried.
- params* Returns the value of the specified uniform variable.

5.37.2.564 `static Int32 OpenTK.Graphics.OpenGL.GL.GetUniformLocation (Int32 program, String name) [static]`

Returns the location of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- name* Points to a null terminated string containing the name of the uniform variable whose location is to be queried.

5.37.2.565 `static Int32 OpenTK.Graphics.OpenGL.GL.GetUniformLocation (UInt32 program, String name) [static]`

Returns the location of a uniform variable.

Parameters

- program* Specifies the program object to be queried.
- name* Points to a null terminated string containing the name of the uniform variable whose location is to be queried.

5.37.2.566 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Double @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

- index* Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.567 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] out Double @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.568 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Double *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.569 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Double @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.570 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] out Double @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.571 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Double *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.572 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Single @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.573 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] out Single @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.574 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Single *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.575 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Single @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.576 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] out Single @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.577 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Single *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.578 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Int32 @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.579 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] out Int32 @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.580 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (Int32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Int32 *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.581 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Int32 @[] params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.582 `static void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] out Int32 @ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.583 `static unsafe void OpenTK.Graphics.OpenGL.GL.GetVertexAttrib (UInt32 index, OpenTK.Graphics.OpenGL.VertexAttribParameter pname, [OutAttribute] Int32 *@ params) [static]`

Return a generic vertex attribute parameter.

Parameters

index Specifies the generic vertex attribute parameter to be queried.

pname Specifies the symbolic name of the vertex attribute parameter to be queried. Accepted values are GL_VERTEX_ATTRIB_ARRAY_BUFFER_BINDING, GL_VERTEX_ATTRIB_ARRAY_ENABLED, GL_VERTEX_ATTRIB_ARRAY_SIZE, GL_VERTEX_ATTRIB_ARRAY_STRIDE, GL_VERTEX_ATTRIB_ARRAY_TYPE, GL_VERTEX_ATTRIB_ARRAY_NORMALIZED, or GL_CURRENT_VERTEX_ATTRIB.

params Returns the requested data.

5.37.2.584 static void OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer
 (Int32 *index*,
 OpenTK.Graphics.OpenGL.VertexAttribPointerParameter *pname*,
 [OutAttribute] IntPtr *pointer*) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

5.37.2.585 static void OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer
 (UInt32 *index*,
 OpenTK.Graphics.OpenGL.VertexAttribPointerParameter *pname*,
 [OutAttribute] IntPtr *pointer*) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

5.37.2.586 static void
 OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
 T2 > (Int32 *index*,
 OpenTK.Graphics.OpenGL.VertexAttribPointerParameter *pname*,
 [InAttribute, OutAttribute] T2[] *pointer*) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

Type Constraints

T2 : struct

5.37.2.587 **static void**
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > (Int32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] T2 pointer[,]) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

Type Constraints

T2 : struct

5.37.2.588 **static void**
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > (Int32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] T2 pointer[,]) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

index Specifies the generic vertex attribute parameter to be returned.

pname Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

Type Constraints

T2 : struct

5.37.2.589 static void
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > (Int32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] ref T2 pointer) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : struct

5.37.2.590 static void
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > (UInt32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] T2[] pointer) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : struct

5.37.2.591 static void
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > (UInt32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] T2 pointer[,]) [static]

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

```
5.37.2.592 static void
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > ( UInt32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] T2 pointer[, ] ) [static]
```

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.
- pointer* Returns the pointer value.

Type Constraints

T2 : *struct*

```
5.37.2.593 static void
OpenTK.Graphics.OpenGL.GL.GetVertexAttribPointer<
T2 > ( UInt32 index,
OpenTK.Graphics.OpenGL.VertexAttribPointerParameter pname,
[InAttribute, OutAttribute] ref T2 pointer ) [static]
```

Return the address of the specified generic vertex attribute pointer.

Parameters

- index* Specifies the generic vertex attribute parameter to be returned.
- pname* Specifies the symbolic name of the generic vertex attribute parameter to be returned. Must be GL_VERTEX_ATTRIB_ARRAY_POINTER.

pointer Returns the pointer value.

Type Constraints

T2 : *struct*

5.37.2.594 `static void OpenTK.Graphics.OpenGL.GL.Hint
(OpenTK.Graphics.OpenGL.HintTarget target,
OpenTK.Graphics.OpenGL.HintMode mode) [static]`

Specify implementation-specific hints.

Parameters

target Specifies a symbolic constant indicating the behavior to be controlled. GL_FOG_HINT, GL_GENERATE_MIPMAP_HINT, GL_LINE_SMOOTH_HINT, GL_PERSPECTIVE_CORRECTION_HINT, GL_POINT_SMOOTH_HINT, GL_POLYGON_SMOOTH_HINT, GL_TEXTURE_COMPRESSION_HINT, and GL_FRAGMENT_SHADER_DERIVATIVE_HINT are accepted.

mode Specifies a symbolic constant indicating the desired behavior. GL_FASTEST, GL_NICEST, and GL_DONT_CARE are accepted.

5.37.2.595 `static void OpenTK.Graphics.OpenGL.GL.Histogram (`
`OpenTK.Graphics.OpenGL.HistogramTarget target, Int32 width,`
`OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,`
`bool sink) [static]`

Define histogram table.

Parameters

target The histogram whose parameters are to be set. Must be one of GL_HISTOGRAM or GL_PROXY_HISTOGRAM.

width The number of entries in the histogram table. Must be a power of 2.

internalformat The format of entries in the histogram table. Must be one of GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_R3_G3_B2, GL_RGB, GL_RGBA, GL_RGBA5, GL_RGBA8, GL_RGBA10,

GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

sink If GL_TRUE, pixels will be consumed by the histogramming process and no drawing or texture loading will take place. If GL_FALSE, pixels will proceed to the minmax process after histogramming.

5.37.2.596 `static void OpenTK.Graphics.OpenGL.GL.Index (Double c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.597 `static unsafe void OpenTK.Graphics.OpenGL.GL.Index (Double *
c) [static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.598 `static void OpenTK.Graphics.OpenGL.GL.Index (Single c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.599 `static unsafe void OpenTK.Graphics.OpenGL.GL.Index (Single *
c) [static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.600 `static void OpenTK.Graphics.OpenGL.GL.Index (Int32 c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.601 `static unsafe void OpenTK.Graphics.OpenGL.GL.Index (Int32 * c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.602 `static void OpenTK.Graphics.OpenGL.GL.Index (Int16 c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.603 `static unsafe void OpenTK.Graphics.OpenGL.GL.Index (Int16 * c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.604 `static void OpenTK.Graphics.OpenGL.GL.Index (Byte c)`
`[static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.605 `static unsafe void OpenTK.Graphics.OpenGL.GL.Index (Byte * c) [static]`

Set the current color index.

Parameters

c Specifies the new value for the current color index.

5.37.2.606 `static void OpenTK.Graphics.OpenGL.GL.IndexMask (Int32 mask) [static]`

Control the writing of individual bits in the color index buffers.

Parameters

mask Specifies a bit mask to enable and disable the writing of individual bits in the color index buffers. Initially, the mask is all 1's.

5.37.2.607 `static void OpenTK.Graphics.OpenGL.GL.IndexMask (UInt32 mask) [static]`

Control the writing of individual bits in the color index buffers.

Parameters

mask Specifies a bit mask to enable and disable the writing of individual bits in the color index buffers. Initially, the mask is all 1's.

5.37.2.608 `static void OpenTK.Graphics.OpenGL.GL.IndexPointer (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, IntPtr pointer) [static]`

Define an array of color indexes.

Parameters

type Specifies the data type of each color index in the array. Symbolic constants GL_UNSIGNED_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive color indexes. If stride is 0, the color indexes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first index in the array. The initial value is 0.

5.37.2.609 `static void OpenTK.Graphics.OpenGL.GL.IndexPointer< T2 > (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer) [static]`

Define an array of color indexes.

Parameters

type Specifies the data type of each color index in the array. Symbolic constants `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

stride Specifies the byte offset between consecutive color indexes. If stride is 0, the color indexes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first index in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.610 `static void OpenTK.Graphics.OpenGL.GL.IndexPointer< T2 > (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of color indexes.

Parameters

type Specifies the data type of each color index in the array. Symbolic constants `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

stride Specifies the byte offset between consecutive color indexes. If stride is 0, the color indexes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first index in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.611 `static void OpenTK.Graphics.OpenGL.GL.IndexPointer< T2 > (OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of color indexes.

Parameters

type Specifies the data type of each color index in the array. Symbolic constants `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

stride Specifies the byte offset between consecutive color indexes. If stride is 0, the color indexes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first index in the array. The initial value is 0.

Type Constraints

T2 : struct

```
5.37.2.612 static void OpenTK.Graphics.OpenGL.GL.IndexPointer< T2 > (
    OpenTK.Graphics.OpenGL.IndexPointerType type, Int32 stride,
    [InAttribute, OutAttribute] ref T2 pointer ) [static]
```

Define an array of color indexes.

Parameters

type Specifies the data type of each color index in the array. Symbolic constants `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_INT`, `GL_FLOAT`, and `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

stride Specifies the byte offset between consecutive color indexes. If stride is 0, the color indexes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first index in the array. The initial value is 0.

Type Constraints

T2 : struct

```
5.37.2.613 static void OpenTK.Graphics.OpenGL.GL.InitNames ( )
    [static]
```

Initialize the name stack.

```
5.37.2.614 static void OpenTK.Graphics.OpenGL.GL.InterleavedArrays (
    OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32
    stride, IntPtr pointer ) [static]
```

Simultaneously specify and enable several interleaved arrays.

Parameters

format Specifies the type of array to enable. Symbolic constants GL_V2F, GL_V3F, GL_C4UB_V2F, GL_C4UB_V3F, GL_C3F_V3F, GL_N3F_V3F, GL_C4F_N3F_V3F, GL_T2F_V3F, GL_T4F_V4F, GL_T2F_C4UB_V3F, GL_T2F_C3F_V3F, GL_T2F_N3F_V3F, GL_T2F_C4F_N3F_V3F, and GL_T4F_C4F_N3F_V4F are accepted.

stride Specifies the offset in bytes between each aggregate array element.

5.37.2.615 `static void OpenTK.Graphics.OpenGL.GL.InterleavedArrays<T2> (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, [InAttribute, OutAttribute] T2[] pointer) [static]`

Simultaneously specify and enable several interleaved arrays.

Parameters

format Specifies the type of array to enable. Symbolic constants GL_V2F, GL_V3F, GL_C4UB_V2F, GL_C4UB_V3F, GL_C3F_V3F, GL_N3F_V3F, GL_C4F_N3F_V3F, GL_T2F_V3F, GL_T4F_V4F, GL_T2F_C4UB_V3F, GL_T2F_C3F_V3F, GL_T2F_N3F_V3F, GL_T2F_C4F_N3F_V3F, and GL_T4F_C4F_N3F_V4F are accepted.

stride Specifies the offset in bytes between each aggregate array element.

Type Constraints

T2 : *struct*

5.37.2.616 `static void OpenTK.Graphics.OpenGL.GL.InterleavedArrays<T2> (OpenTK.Graphics.OpenGL.InterleavedArrayFormat format, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Simultaneously specify and enable several interleaved arrays.

Parameters

format Specifies the type of array to enable. Symbolic constants GL_V2F, GL_V3F, GL_C4UB_V2F, GL_C4UB_V3F, GL_C3F_V3F, GL_N3F_V3F, GL_C4F_N3F_V3F, GL_T2F_V3F, GL_T4F_V4F, GL_T2F_C4UB_V3F, GL_T2F_C3F_V3F, GL_T2F_N3F_V3F, GL_T2F_C4F_N3F_V3F, and GL_T4F_C4F_N3F_V4F are accepted.

stride Specifies the offset in bytes between each aggregate array element.

Type Constraints*T2 : struct*

5.37.2.617 `static void OpenTK.Graphics.OpenGL.GL.InterleavedArrays<
T2 > (OpenTK.Graphics.OpenGL.InterleavedArrayFormat
format, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,])
[static]`

Simultaneously specify and enable several interleaved arrays.

Parameters

format Specifies the type of array to enable. Symbolic constants GL_V2F, GL_V3F, GL_C4UB_V2F, GL_C4UB_V3F, GL_C3F_V3F, GL_N3F_V3F, GL_C4F_N3F_V3F, GL_T2F_V3F, GL_T4F_V4F, GL_T2F_C4UB_V3F, GL_T2F_C3F_V3F, GL_T2F_N3F_V3F, GL_T2F_C4F_N3F_V3F, and GL_T4F_C4F_N3F_V4F are accepted.

stride Specifies the offset in bytes between each aggregate array element.

Type Constraints*T2 : struct*

5.37.2.618 `static void OpenTK.Graphics.OpenGL.GL.InterleavedArrays<
T2 > (OpenTK.Graphics.OpenGL.InterleavedArrayFormat
format, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer)
[static]`

Simultaneously specify and enable several interleaved arrays.

Parameters

format Specifies the type of array to enable. Symbolic constants GL_V2F, GL_V3F, GL_C4UB_V2F, GL_C4UB_V3F, GL_C3F_V3F, GL_N3F_V3F, GL_C4F_N3F_V3F, GL_T2F_V3F, GL_T4F_V4F, GL_T2F_C4UB_V3F, GL_T2F_C3F_V3F, GL_T2F_N3F_V3F, GL_T2F_C4F_N3F_V3F, and GL_T4F_C4F_N3F_V4F are accepted.

stride Specifies the offset in bytes between each aggregate array element.

Type Constraints*T2 : struct*

5.37.2.619 `static bool OpenTK.Graphics.OpenGL.GL.IsBuffer (Int32 buffer)
[static]`

Determine if a name corresponds to a buffer object.

Parameters

buffer Specifies a value that may be the name of a buffer object.

5.37.2.620 `static bool OpenTK.Graphics.OpenGL.GL.IsBuffer (UInt32 buffer
) [static]`

Determine if a name corresponds to a buffer object.

Parameters

buffer Specifies a value that may be the name of a buffer object.

5.37.2.621 `static bool OpenTK.Graphics.OpenGL.GL.IsEnabled (OpenTK.Graphics.OpenGL.EnableCap cap) [static]`

Test whether a capability is enabled.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.37.2.622 `static bool OpenTK.Graphics.OpenGL.GL.IsEnabled (OpenTK.Graphics.OpenGL.IndexedEnableCap target, Int32 index
) [static]`

Test whether a capability is enabled.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.37.2.623 `static bool OpenTK.Graphics.OpenGL.GL.IsEnabled (OpenTK.Graphics.OpenGL.IndexedEnableCap target, UInt32
index) [static]`

Test whether a capability is enabled.

Parameters

cap Specifies a symbolic constant indicating a [GL](#) capability.

5.37.2.624 `static bool OpenTK.Graphics.OpenGL.GL.IsList (Int32 list)
[static]`

Determine if a name corresponds to a display list.

Parameters

list Specifies a potential display list name.

5.37.2.625 `static bool OpenTK.Graphics.OpenGL.GL.IsList (UInt32 list)
[static]`

Determine if a name corresponds to a display list.

Parameters

list Specifies a potential display list name.

5.37.2.626 `static bool OpenTK.Graphics.OpenGL.GL.IsProgram (Int32
program) [static]`

Determines if a name corresponds to a program object.

Parameters

program Specifies a potential program object.

5.37.2.627 `static bool OpenTK.Graphics.OpenGL.GL.IsProgram (UInt32
program) [static]`

Determines if a name corresponds to a program object.

Parameters

program Specifies a potential program object.

5.37.2.628 `static bool OpenTK.Graphics.OpenGL.GL.IsQuery (Int32 id)`
`[static]`

Determine if a name corresponds to a query object.

Parameters

id Specifies a value that may be the name of a query object.

5.37.2.629 `static bool OpenTK.Graphics.OpenGL.GL.IsQuery (UInt32 id)`
`[static]`

Determine if a name corresponds to a query object.

Parameters

id Specifies a value that may be the name of a query object.

5.37.2.630 `static bool OpenTK.Graphics.OpenGL.GL.IsShader (Int32 shader)`
`[static]`

Determines if a name corresponds to a shader object.

Parameters

shader Specifies a potential shader object.

5.37.2.631 `static bool OpenTK.Graphics.OpenGL.GL.IsShader (UInt32 shader)`
`[static]`

Determines if a name corresponds to a shader object.

Parameters

shader Specifies a potential shader object.

5.37.2.632 `static bool OpenTK.Graphics.OpenGL.GL.IsTexture (Int32 texture)`
`[static]`

Determine if a name corresponds to a texture.

Parameters

texture Specifies a value that may be the name of a texture.

5.37.2.633 `static bool OpenTK.Graphics.OpenGL.GL.IsTexture (UInt32 texture) [static]`

Determine if a name corresponds to a texture.

Parameters

texture Specifies a value that may be the name of a texture.

5.37.2.634 `static void OpenTK.Graphics.OpenGL.GL.Light (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname, Single param) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form `GL_LIGHTi`, where *i* ranges from 0 to the value of `GL_MAX_LIGHTS` - 1.

pname Specifies a single-valued light source parameter for light. `GL_SPOT_EXPONENT`, `GL_SPOT_CUTOFF`, `GL_CONSTANT_ATTENUATION`, `GL_LINEAR_ATTENUATION`, and `GL_QUADRATIC_ATTENUATION` are accepted.

param Specifies the value that parameter *pname* of light source light will be set to.

5.37.2.635 `static void OpenTK.Graphics.OpenGL.GL.Light (OpenTK.Graphics.OpenGL.LightName light, OpenTK.Graphics.OpenGL.LightParameter pname, Single @[] params) [static]`

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form `GL_LIGHTi`, where *i* ranges from 0 to the value of `GL_MAX_LIGHTS` - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter *pname* of light source light will be set to.

```
5.37.2.636 static unsafe void OpenTK.Graphics.OpenGL.GL.Light
( OpenTK.Graphics.OpenGL.LightName light,
  OpenTK.Graphics.OpenGL.LightParameter pname, Single *@
  params ) [static]
```

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where *i* ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter *pname* of light source light will be set to.

```
5.37.2.637 static void OpenTK.Graphics.OpenGL.GL.Light
( OpenTK.Graphics.OpenGL.LightName light,
  OpenTK.Graphics.OpenGL.LightParameter pname, Int32 param
) [static]
```

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where *i* ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

```
5.37.2.638 static void OpenTK.Graphics.OpenGL.GL.Light
( OpenTK.Graphics.OpenGL.LightName light,
  OpenTK.Graphics.OpenGL.LightParameter pname, Int32 @[
  params ) [static]
```

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where i ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

```
5.37.2.639 static unsafe void OpenTK.Graphics.OpenGL.GL.Light
( OpenTK.Graphics.OpenGL.LightName light,
  OpenTK.Graphics.OpenGL.LightParameter pname, Int32 *@[
  params ) [static]
```

Set light source parameters.

Parameters

light Specifies a light. The number of lights depends on the implementation, but at least eight lights are supported. They are identified by symbolic names of the form GL_LIGHT_i, where i ranges from 0 to the value of GL_MAX_LIGHTS - 1.

pname Specifies a single-valued light source parameter for light. GL_SPOT_EXPONENT, GL_SPOT_CUTOFF, GL_CONSTANT_ATTENUATION, GL_LINEAR_ATTENUATION, and GL_QUADRATIC_ATTENUATION are accepted.

param Specifies the value that parameter pname of light source light will be set to.

5.37.2.640 `static void OpenTK.Graphics.OpenGL.GL.LightModel (`
`OpenTK.Graphics.OpenGL.LightModelParameter pname, Single`
`param) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.37.2.641 `static void OpenTK.Graphics.OpenGL.GL.LightModel (`
`OpenTK.Graphics.OpenGL.LightModelParameter pname, Single`
`@[] params) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.37.2.642 `static unsafe void OpenTK.Graphics.OpenGL.GL.LightModel (`
`OpenTK.Graphics.OpenGL.LightModelParameter pname, Single`
`*@ params) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.37.2.643 `static void OpenTK.Graphics.OpenGL.GL.LightModel (`
`OpenTK.Graphics.OpenGL.LightModelParameter pname, Int32`
`param) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.37.2.644 `static void OpenTK.Graphics.OpenGL.GL.LightModel (`
`OpenTK.Graphics.OpenGL.LightModelParameter pname, Int32`
`@[] params) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.37.2.645 `static unsafe void OpenTK.Graphics.OpenGL.GL.LightModel (`
`OpenTK.Graphics.OpenGL.LightModelParameter pname, Int32`
`*@ params) [static]`

Set the lighting model parameters.

Parameters

pname Specifies a single-valued lighting model parameter. GL_LIGHT_MODEL_LOCAL_VIEWER, GL_LIGHT_MODEL_COLOR_CONTROL, and GL_LIGHT_MODEL_TWO_SIDE are accepted.

param Specifies the value that param will be set to.

5.37.2.646 `static void OpenTK.Graphics.OpenGL.GL.LineStipple (Int32
factor, Int16 pattern) [static]`

Specify the line stipple pattern.

Parameters

factor Specifies a multiplier for each bit in the line stipple pattern. If factor is 3, for example, each bit in the pattern is used three times before the next bit in the pattern is used. factor is clamped to the range [1, 256] and defaults to 1.

pattern Specifies a 16-bit integer whose bit pattern determines which fragments of a line will be drawn when the line is rasterized. Bit zero is used first; the default pattern is all 1's.

5.37.2.647 `static void OpenTK.Graphics.OpenGL.GL.LineStipple (Int32
factor, UInt16 pattern) [static]`

Specify the line stipple pattern.

Parameters

factor Specifies a multiplier for each bit in the line stipple pattern. If factor is 3, for example, each bit in the pattern is used three times before the next bit in the pattern is used. factor is clamped to the range [1, 256] and defaults to 1.

pattern Specifies a 16-bit integer whose bit pattern determines which fragments of a line will be drawn when the line is rasterized. Bit zero is used first; the default pattern is all 1's.

5.37.2.648 `static void OpenTK.Graphics.OpenGL.GL.LineWidth (Single
width) [static]`

Specify the width of rasterized lines.

Parameters

width Specifies the width of rasterized lines. The initial value is 1.

5.37.2.649 `static void OpenTK.Graphics.OpenGL.GL.LinkProgram (Int32
program) [static]`

Links a program object.

Parameters

program Specifies the handle of the program object to be linked.

5.37.2.650 `static void OpenTK.Graphics.OpenGL.GL.LinkProgram (UInt32
program) [static]`

Links a program object.

Parameters

program Specifies the handle of the program object to be linked.

5.37.2.651 `static void OpenTK.Graphics.OpenGL.GL.ListBase (Int32 @ base
) [static]`

Set the display-list base for glCallLists.

Parameters

base Specifies an integer offset that will be added to glCallLists offsets to generate display-list names. The initial value is 0.

5.37.2.652 `static void OpenTK.Graphics.OpenGL.GL.ListBase (UInt32 @
base) [static]`

Set the display-list base for glCallLists.

Parameters

base Specifies an integer offset that will be added to glCallLists offsets to generate display-list names. The initial value is 0.

5.37.2.653 `static void OpenTK.Graphics.OpenGL.GL.LoadAll ()
[static]`

Loads all [OpenGL](#) entry points (core and extension). This method is provided for compatibility purposes with older OpenTK versions.

5.37.2.654 `static void OpenTK.Graphics.OpenGL.GL.LoadIdentity ()
[static]`

Replace the current matrix with the identity matrix.

5.37.2.655 `static void OpenTK.Graphics.OpenGL.GL.LoadMatrix (Double[]
m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.37.2.656 `static void OpenTK.Graphics.OpenGL.GL.LoadMatrix (ref
Double m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.37.2.657 `static unsafe void OpenTK.Graphics.OpenGL.GL.LoadMatrix (Double * m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.37.2.658 `static void OpenTK.Graphics.OpenGL.GL.LoadMatrix (Single[]
m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.37.2.659 `static void OpenTK.Graphics.OpenGL.GL.LoadMatrix (ref Single m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.37.2.660 `static unsafe void OpenTK.Graphics.OpenGL.GL.LoadMatrix (Single * m) [static]`

Replace the current matrix with the specified matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 column-major matrix.

5.37.2.661 `static void OpenTK.Graphics.OpenGL.GL.LoadName (Int32 name) [static]`

Load a name onto the name stack.

Parameters

name Specifies a name that will replace the top value on the name stack.

5.37.2.662 `static void OpenTK.Graphics.OpenGL.GL.LoadName (UInt32 name) [static]`

Load a name onto the name stack.

Parameters

name Specifies a name that will replace the top value on the name stack.

5.37.2.663 `static void OpenTK.Graphics.OpenGL.GL.LoadTransposeMatrix (Double[] m) [static]`

Replace the current matrix with the specified row-major ordered matrix.

Parameters

- m* Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 row-major matrix.

5.37.2.664 `static void OpenTK.Graphics.OpenGL.GL.LoadTransposeMatrix (Single[] m) [static]`

Replace the current matrix with the specified row-major ordered matrix.

Parameters

- m* Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 row-major matrix.

5.37.2.665 `static void OpenTK.Graphics.OpenGL.GL.LoadTransposeMatrix (ref Single m) [static]`

Replace the current matrix with the specified row-major ordered matrix.

Parameters

- m* Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 row-major matrix.

5.37.2.666 `static unsafe void OpenTK.Graphics.OpenGL.GL.LoadTransposeMatrix (Single * m) [static]`

Replace the current matrix with the specified row-major ordered matrix.

Parameters

- m* Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 row-major matrix.

5.37.2.667 `static void OpenTK.Graphics.OpenGL.GL.LoadTransposeMatrix (ref Double m) [static]`

Replace the current matrix with the specified row-major ordered matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 row-major matrix.

5.37.2.668 static unsafe void
OpenTK.Graphics.OpenGL.GL.LoadTransposeMatrix (Double *
***m*) [static]**

Replace the current matrix with the specified row-major ordered matrix.

Parameters

m Specifies a pointer to 16 consecutive values, which are used as the elements of a 4 times 4 row-major matrix.

5.37.2.669 static void OpenTK.Graphics.OpenGL.GL.LogicOp (
OpenTK.Graphics.OpenGL.LogicOp *opcode*) [static]

Specify a logical pixel operation for color index rendering.

Parameters

opcode Specifies a symbolic constant that selects a logical operation. The following symbols are accepted: GL_CLEAR, GL_SET, GL_COPY, GL_COPY_INVERTED, GL_NOOP, GL_INVERT, GL_AND, GL_NAND, GL_OR, GL_NOR, GL_XOR, GL_EQUIV, GL_AND_REVERSE, GL_AND_INVERTED, GL_OR_REVERSE, and GL_OR_INVERTED. The initial value is GL_COPY.

5.37.2.670 static unsafe void OpenTK.Graphics.OpenGL.GL.Map1 (
OpenTK.Graphics.OpenGL.MapTarget *target*, Double *u1*, Double
***u2*, Int32 *stride*, Int32 *order*, Double * *points*) [static]**

Define a one-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP1_INDEX, GL_MAP1_COLOR_4, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, and GL_MAP1_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to `glEvalCoord1`, to \hat{u} , the variable that is evaluated by the equations specified by this command.

stride Specifies the number of floats or doubles between the beginning of one control point and the beginning of the next one in the data structure referenced in points. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations.

order Specifies the number of control points. Must be positive.

points Specifies a pointer to the array of control points.

5.37.2.671 `static void OpenTK.Graphics.OpenGL.GL.Map1 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single`
`u2, Int32 stride, Int32 order, Single[] points) [static]`

Define a one-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants `GL_MAP1_VERTEX_3`, `GL_MAP1_VERTEX_4`, `GL_MAP1_INDEX`, `GL_MAP1_COLOR_4`, `GL_MAP1_NORMAL`, `GL_MAP1_TEXTURE_COORD_1`, `GL_MAP1_TEXTURE_COORD_2`, `GL_MAP1_TEXTURE_COORD_3`, and `GL_MAP1_TEXTURE_COORD_4` are accepted.

u1 Specify a linear mapping of u , as presented to `glEvalCoord1`, to \hat{u} , the variable that is evaluated by the equations specified by this command.

stride Specifies the number of floats or doubles between the beginning of one control point and the beginning of the next one in the data structure referenced in points. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations.

order Specifies the number of control points. Must be positive.

points Specifies a pointer to the array of control points.

5.37.2.672 `static void OpenTK.Graphics.OpenGL.GL.Map1 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double`
`u2, Int32 stride, Int32 order, Double[] points) [static]`

Define a one-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP1_INDEX, GL_MAP1_COLOR_4, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, and GL_MAP1_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord1, to \hat{u} , the variable that is evaluated by the equations specified by this command.

stride Specifies the number of floats or doubles between the beginning of one control point and the beginning of the next one in the data structure referenced in points. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations.

order Specifies the number of control points. Must be positive.

points Specifies a pointer to the array of control points.

5.37.2.673 `static void OpenTK.Graphics.OpenGL.GL.Map1 (OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double u2, Int32 stride, Int32 order, ref Double points) [static]`

Define a one-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP1_INDEX, GL_MAP1_COLOR_4, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, and GL_MAP1_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord1, to \hat{u} , the variable that is evaluated by the equations specified by this command.

stride Specifies the number of floats or doubles between the beginning of one control point and the beginning of the next one in the data structure referenced in points. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations.

order Specifies the number of control points. Must be positive.

points Specifies a pointer to the array of control points.

5.37.2.674 `static void OpenTK.Graphics.OpenGL.GL.Map1 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single`
`u2, Int32 stride, Int32 order, ref Single points) [static]`

Define a one-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP1_INDEX, GL_MAP1_COLOR_4, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, and GL_MAP1_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord1, to \hat{u} , the variable that is evaluated by the equations specified by this command.

stride Specifies the number of floats or doubles between the beginning of one control point and the beginning of the next one in the data structure referenced in points. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations.

order Specifies the number of control points. Must be positive.

points Specifies a pointer to the array of control points.

5.37.2.675 `static unsafe void OpenTK.Graphics.OpenGL.GL.Map1 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single`
`u2, Int32 stride, Int32 order, Single * points) [static]`

Define a one-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP1_VERTEX_3, GL_MAP1_VERTEX_4, GL_MAP1_INDEX, GL_MAP1_COLOR_4, GL_MAP1_NORMAL, GL_MAP1_TEXTURE_COORD_1, GL_MAP1_TEXTURE_COORD_2, GL_MAP1_TEXTURE_COORD_3, and GL_MAP1_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord1, to \hat{u} , the variable that is evaluated by the equations specified by this command.

stride Specifies the number of floats or doubles between the beginning of one control point and the beginning of the next one in the data structure referenced in points. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations.

order Specifies the number of control points. Must be positive.

points Specifies a pointer to the array of control points.

5.37.2.676 static unsafe void OpenTK.Graphics.OpenGL.GL.Map2 (
OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single
u2, Int32 ustride, Int32 uorder, Single v1, Single v2, Int32
vstride, Int32 vorder, Single * points) [static]

Define a two-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP2_VERTEX_3, GL_MAP2_VERTEX_4, GL_MAP2_INDEX, GL_MAP2_COLOR_4, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, and GL_MAP2_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord2, to \hat{u} , one of the two variables that are evaluated by the equations specified by this command. Initially, u_1 is 0 and u_2 is 1.

ustride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{(i+1)j}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of $ustride$ is 0.

uorder Specifies the dimension of the control point array in the u axis. Must be positive. The initial value is 1.

v1 Specify a linear mapping of v , as presented to glEvalCoord2, to \hat{v} , one of the two variables that are evaluated by the equations specified by this command. Initially, v_1 is 0 and v_2 is 1.

vstride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{i(j+1)}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of $vstride$ is 0.

vorder Specifies the dimension of the control point array in the v axis. Must be positive. The initial value is 1.

points Specifies a pointer to the array of control points.

5.37.2.677 `static void OpenTK.Graphics.OpenGL.GL.Map2 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double`
`u2, Int32 ustride, Int32 uorder, Double v1, Double v2, Int32`
`vstride, Int32 vorder, Double[] points) [static]`

Define a two-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP2_VERTEX_3, GL_MAP2_VERTEX_4, GL_MAP2_INDEX, GL_MAP2_COLOR_4, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, and GL_MAP2_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord2, to \hat{u} , one of the two variables that are evaluated by the equations specified by this command. Initially, u_1 is 0 and u_2 is 1.

ustride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{(i+1)j}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of $ustride$ is 0.

uorder Specifies the dimension of the control point array in the u axis. Must be positive. The initial value is 1.

v1 Specify a linear mapping of v , as presented to glEvalCoord2, to \hat{v} , one of the two variables that are evaluated by the equations specified by this command. Initially, v_1 is 0 and v_2 is 1.

vstride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{i(j+1)}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of $vstride$ is 0.

vorder Specifies the dimension of the control point array in the v axis. Must be positive. The initial value is 1.

points Specifies a pointer to the array of control points.

5.37.2.678 `static void OpenTK.Graphics.OpenGL.GL.Map2 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double`
`u2, Int32 ustride, Int32 uorder, Double v1, Double v2, Int32`
`vstride, Int32 vorder, ref Double points) [static]`

Define a two-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP2_VERTEX_3, GL_MAP2_VERTEX_4, GL_MAP2_INDEX, GL_MAP2_COLOR_4, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, and GL_MAP2_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to glEvalCoord2, to \hat{u} , one of the two variables that are evaluated by the equations specified by this command. Initially, u_1 is 0 and u_2 is 1.

ustride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{(i+1)j}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of $ustride$ is 0.

uorder Specifies the dimension of the control point array in the u axis. Must be positive. The initial value is 1.

v1 Specify a linear mapping of v , as presented to glEvalCoord2, to \hat{v} , one of the two variables that are evaluated by the equations specified by this command. Initially, v_1 is 0 and v_2 is 1.

vstride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{i(j+1)}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of $vstride$ is 0.

vorder Specifies the dimension of the control point array in the v axis. Must be positive. The initial value is 1.

points Specifies a pointer to the array of control points.

5.37.2.679 `static unsafe void OpenTK.Graphics.OpenGL.GL.Map2 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Double u1, Double`
`u2, Int32 ustride, Int32 uorder, Double v1, Double v2, Int32`
`vstride, Int32 vorder, Double * points) [static]`

Define a two-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP2_VERTEX_3, GL_MAP2_VERTEX_4, GL_MAP2_INDEX, GL_MAP2_COLOR_4, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, and GL_MAP2_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to `glEvalCoord2`, to \hat{u} , one of the two variables that are evaluated by the equations specified by this command. Initially, u_1 is 0 and u_2 is 1.

ustride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{(i+1)j}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of `ustride` is 0.

uorder Specifies the dimension of the control point array in the u axis. Must be positive. The initial value is 1.

v1 Specify a linear mapping of v , as presented to `glEvalCoord2`, to \hat{v} , one of the two variables that are evaluated by the equations specified by this command. Initially, v_1 is 0 and v_2 is 1.

vstride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{i(j+1)}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of `vstride` is 0.

vorder Specifies the dimension of the control point array in the v axis. Must be positive. The initial value is 1.

points Specifies a pointer to the array of control points.

5.37.2.680 `static void OpenTK.Graphics.OpenGL.GL.Map2 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single`
`u2, Int32 ustride, Int32 uorder, Single v1, Single v2, Int32`
`vstride, Int32 vorder, Single[] points) [static]`

Define a two-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP2_VERTEX_3, GL_MAP2_VERTEX_4, GL_MAP2_INDEX, GL_MAP2_COLOR_4, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, and GL_MAP2_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to `glEvalCoord2`, to \hat{u} , one of the two variables that are evaluated by the equations specified by this command. Initially, u_1 is 0 and u_2 is 1.

ustride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{(i+1)j}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of `ustride` is 0.

uorder Specifies the dimension of the control point array in the u axis. Must be positive. The initial value is 1.

v1 Specify a linear mapping of v , as presented to `glEvalCoord2`, to \hat{v} , one of the two variables that are evaluated by the equations specified by this command. Initially, v_1 is 0 and v_2 is 1.

vstride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{i(j+1)}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of `vstride` is 0.

vorder Specifies the dimension of the control point array in the v axis. Must be positive. The initial value is 1.

points Specifies a pointer to the array of control points.

5.37.2.681 `static void OpenTK.Graphics.OpenGL.GL.Map2 (`
`OpenTK.Graphics.OpenGL.MapTarget target, Single u1, Single`
`u2, Int32 ustride, Int32 uorder, Single v1, Single v2, Int32`
`vstride, Int32 vorder, ref Single points) [static]`

Define a two-dimensional evaluator.

Parameters

target Specifies the kind of values that are generated by the evaluator. Symbolic constants GL_MAP2_VERTEX_3, GL_MAP2_VERTEX_4, GL_MAP2_INDEX, GL_MAP2_COLOR_4, GL_MAP2_NORMAL, GL_MAP2_TEXTURE_COORD_1, GL_MAP2_TEXTURE_COORD_2, GL_MAP2_TEXTURE_COORD_3, and GL_MAP2_TEXTURE_COORD_4 are accepted.

u1 Specify a linear mapping of u , as presented to `glEvalCoord2`, to \hat{u} , one of the two variables that are evaluated by the equations specified by this command. Initially, u_1 is 0 and u_2 is 1.

ustride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{(i+1)j}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of `ustride` is 0.

uorder Specifies the dimension of the control point array in the u axis. Must be positive. The initial value is 1.

v1 Specify a linear mapping of v , as presented to `glEvalCoord2`, to \hat{v} , one of the two variables that are evaluated by the equations specified by this command. Initially, v_1 is 0 and v_2 is 1.

vstride Specifies the number of floats or doubles between the beginning of control point R_{ij} and the beginning of control point $R_{i(j+1)}$, where i and j are the control point indices, respectively. This allows control points to be embedded in arbitrary data structures. The only constraint is that the values for a particular control point must occupy contiguous memory locations. The initial value of `vstride` is 0.

vorder Specifies the dimension of the control point array in the v axis. Must be positive. The initial value is 1.

points Specifies a pointer to the array of control points.

5.37.2.682 `static unsafe System.IntPtr
OpenTK.Graphics.OpenGL.GL.MapBuffer (
OpenTK.Graphics.OpenGL.BufferTarget target,
OpenTK.Graphics.OpenGL.BufferAccess access) [static]`

Map a buffer object's data store.

Parameters

target Specifies the target buffer object being mapped. The symbolic constant must be GL_ARRAY_BUFFER, GL_ELEMENT_ARRAY_BUFFER, GL_PIXEL_PACK_BUFFER, or GL_PIXEL_UNPACK_BUFFER.

access Specifies the access policy, indicating whether it will be possible to read from, write to, or both read from and write to the buffer object's mapped data store. The symbolic constant must be GL_READ_ONLY, GL_WRITE_ONLY, or GL_READ_WRITE.

5.37.2.683 `static void OpenTK.Graphics.OpenGL.GL.MapGrid1 (Int32 un,
Double u1, Double u2) [static]`

Define a one- or two-dimensional mesh.

Parameters

un Specifies the number of partitions in the grid range interval [*u1*, *u2*]. Must be positive.

u1 Specify the mappings for integer grid domain values *i* = 0 and *i* = *un*.

vn Specifies the number of partitions in the grid range interval [*v1*, *v2*] (glMap-Grid2 only).

v1 Specify the mappings for integer grid domain values *j* = 0 and *j* = *vn* (glMap-Grid2 only).

5.37.2.684 `static void OpenTK.Graphics.OpenGL.GL.MapGrid1 (Int32 un,
Single u1, Single u2) [static]`

Define a one- or two-dimensional mesh.

Parameters

un Specifies the number of partitions in the grid range interval [*u1*, *u2*]. Must be positive.

u1 Specify the mappings for integer grid domain values *i* = 0 and *i* = *un*.

vn Specifies the number of partitions in the grid range interval [*v1*, *v2*] (glMap-Grid2 only).

v1 Specify the mappings for integer grid domain values *j* = 0 and *j* = *vn* (glMap-Grid2 only).

5.37.2.685 `static void OpenTK.Graphics.OpenGL.GL.MapGrid2 (Int32 un, Single u1, Single u2, Int32 vn, Single v1, Single v2)`
`[static]`

Define a one- or two-dimensional mesh.

Parameters

un Specifies the number of partitions in the grid range interval [*u1*, *u2*]. Must be positive.

u1 Specify the mappings for integer grid domain values *i* = 0 and *i* = *un*.

vn Specifies the number of partitions in the grid range interval [*v1*, *v2*] (glMap-Grid2 only).

v1 Specify the mappings for integer grid domain values *j* = 0 and *j* = *vn* (glMap-Grid2 only).

5.37.2.686 `static void OpenTK.Graphics.OpenGL.GL.MapGrid2 (Int32 un, Double u1, Double u2, Int32 vn, Double v1, Double v2)`
`[static]`

Define a one- or two-dimensional mesh.

Parameters

un Specifies the number of partitions in the grid range interval [*u1*, *u2*]. Must be positive.

u1 Specify the mappings for integer grid domain values *i* = 0 and *i* = *un*.

vn Specifies the number of partitions in the grid range interval [*v1*, *v2*] (glMap-Grid2 only).

v1 Specify the mappings for integer grid domain values *j* = 0 and *j* = *vn* (glMap-Grid2 only).

```

5.37.2.687 static void OpenTK.Graphics.OpenGL.GL.Material
( OpenTK.Graphics.OpenGL.MaterialFace face,
  OpenTK.Graphics.OpenGL.MaterialParameter pname, Int32 @[
params ) [static]

```

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

```

5.37.2.688 static unsafe void OpenTK.Graphics.OpenGL.GL.Material
( OpenTK.Graphics.OpenGL.MaterialFace face,
  OpenTK.Graphics.OpenGL.MaterialParameter pname, Int32 *@[
params ) [static]

```

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

```

5.37.2.689 static void OpenTK.Graphics.OpenGL.GL.Material
( OpenTK.Graphics.OpenGL.MaterialFace face,
  OpenTK.Graphics.OpenGL.MaterialParameter pname, Single
param ) [static]

```

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.37.2.690 `static void OpenTK.Graphics.OpenGL.GL.Material
(OpenTK.Graphics.OpenGL.MaterialFace face,
OpenTK.Graphics.OpenGL.MaterialParameter pname, Int32
param) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.37.2.691 `static unsafe void OpenTK.Graphics.OpenGL.GL.Material
(OpenTK.Graphics.OpenGL.MaterialFace face,
OpenTK.Graphics.OpenGL.MaterialParameter pname, Single *@
params) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.37.2.692 `static void OpenTK.Graphics.OpenGL.GL.Material
(OpenTK.Graphics.OpenGL.MaterialFace face,
OpenTK.Graphics.OpenGL.MaterialParameter pname, Single @[]
params) [static]`

Specify material parameters for the lighting model.

Parameters

face Specifies which face or faces are being updated. Must be one of GL_FRONT, GL_BACK, or GL_FRONT_AND_BACK.

pname Specifies the single-valued material parameter of the face or faces that is being updated. Must be GL_SHININESS.

param Specifies the value that parameter GL_SHININESS will be set to.

5.37.2.693 `static void OpenTK.Graphics.OpenGL.GL.MatrixMode (OpenTK.Graphics.OpenGL.MatrixMode mode) [static]`

Specify which matrix is the current matrix.

Parameters

mode Specifies which matrix stack is the target for subsequent matrix operations. Three values are accepted: GL_MODELVIEW, GL_PROJECTION, and GL_TEXTURE. The initial value is GL_MODELVIEW. Additionally, if the ARB_imaging extension is supported, GL_COLOR is also accepted.

5.37.2.694 `static void OpenTK.Graphics.OpenGL.GL.Minmax (OpenTK.Graphics.OpenGL.MinmaxTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, bool sink) [static]`

Define minmax table.

Parameters

target The minmax table whose parameters are to be set. Must be GL_MINMAX.

internalformat The format of entries in the minmax table. Must be one of GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

sink If GL_TRUE, pixels will be consumed by the minmax process and no drawing or texture loading will take place. If GL_FALSE, pixels will proceed to the final conversion process after minmax.

5.37.2.695 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawArrays (OpenTK.Graphics.OpenGL.BeginMode mode, [OutAttribute] Int32[] first, [OutAttribute] Int32[] count, Int32 primcount) [static]`

Render multiple sets of primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Points to an array of starting indices in the enabled arrays.

count Points to an array of the number of indices to be rendered.

primcount Specifies the size of the first and count

5.37.2.696 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawArrays (OpenTK.Graphics.OpenGL.BeginMode mode, [OutAttribute] out Int32 first, [OutAttribute] out Int32 count, Int32 primcount) [static]`

Render multiple sets of primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Points to an array of starting indices in the enabled arrays.

count Points to an array of the number of indices to be rendered.

primcount Specifies the size of the first and count

5.37.2.697 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiDrawArrays (OpenTK.Graphics.OpenGL.BeginMode mode, [OutAttribute] Int32 * first, [OutAttribute] Int32 * count, Int32 primcount) [static]`

Render multiple sets of primitives from array data.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

first Points to an array of starting indices in the enabled arrays.

count Points to an array of the number of indices to be rendered.

primcount Specifies the size of the first and count

5.37.2.698 `static unsafe void
 OpenTK.Graphics.OpenGL.GL.MultiDrawElements (
 OpenTK.Graphics.OpenGL.BeginMode mode, Int32 * count,
 OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr
indices, Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

5.37.2.699 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements (
 OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count,
 OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr
indices, Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

5.37.2.700 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, IntPtr indices, Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

5.37.2.701 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiDrawElements< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 * count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[] indices, Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : *struct*

5.37.2.702 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3 indices[,], Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : *struct*

5.37.2.703 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3 indices[,], Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints*T3 : struct*

5.37.2.704 `static unsafe void`
OpenTK.Graphics.OpenGL.GL.MultiDrawElements< **T3** > (
OpenTK.Graphics.OpenGL.BeginMode *mode*, **Int32** * *count*,
OpenTK.Graphics.OpenGL.DrawElementsType *type*, [InAttribute,
OutAttribute] **T3** *indices*[], **Int32** *primcount*) [**static**]

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants `GL_POINTS`, `GL_LINE_STRIP`, `GL_LINE_LOOP`, `GL_LINES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`, `GL_TRIANGLES`, `GL_QUAD_STRIP`, `GL_QUADS`, and `GL_POLYGON` are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of `GL_UNSIGNED_BYTE`, `GL_UNSIGNED_SHORT`, or `GL_UNSIGNED_INT`.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints*T3 : struct*

5.37.2.705 `static void` **OpenTK.Graphics.OpenGL.GL.MultiDrawElements**< **T3** > (
OpenTK.Graphics.OpenGL.BeginMode *mode*, **ref** **Int32** *count*,
OpenTK.Graphics.OpenGL.DrawElementsType *type*,
[InAttribute, OutAttribute] **ref** **T3** *indices*, **Int32** *primcount*)
[**static**]

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants `GL_POINTS`, `GL_LINE_STRIP`, `GL_LINE_LOOP`, `GL_LINES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`, `GL_TRIANGLES`, `GL_QUAD_STRIP`, `GL_QUADS`, and `GL_POLYGON` are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : struct

5.37.2.706 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements<T3> (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] ref T3 indices, Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : struct

5.37.2.707 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements<T3> (OpenTK.Graphics.OpenGL.BeginMode mode, Int32[] count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3[] indices, Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : struct

```
5.37.2.708 static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements<
T3 > ( OpenTK.Graphics.OpenGL.BeginMode mode, Int32[]
count, OpenTK.Graphics.OpenGL.DrawElementsType type,
[InAttribute, OutAttribute] T3 indices[,], Int32 primcount )
[static]
```

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : struct

5.37.2.709 `static void OpenTK.Graphics.OpenGL.GL.MultiDrawElements< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, ref Int32 count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3 indices[,], Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : *struct*

5.37.2.710 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiDrawElements< T3 > (OpenTK.Graphics.OpenGL.BeginMode mode, Int32 * count, OpenTK.Graphics.OpenGL.DrawElementsType type, [InAttribute, OutAttribute] T3 indices[,], Int32 primcount) [static]`

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants GL_POINTS, GL_LINE_STRIP, GL_LINE_LOOP, GL_LINES, GL_TRIANGLE_STRIP, GL_TRIANGLE_FAN, GL_TRIANGLES, GL_QUAD_STRIP, GL_QUADS, and GL_POLYGON are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints*T3 : struct*

5.37.2.711 `static unsafe void`
OpenTK.Graphics.OpenGL.GL.MultiDrawElements< **T3** > (
OpenTK.Graphics.OpenGL.BeginMode *mode*, **Int32** * *count*,
OpenTK.Graphics.OpenGL.DrawElementsType *type*, [InAttribute,
OutAttribute] ref **T3** *indices*, **Int32** *primcount*) [**static**]

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants `GL_POINTS`, `GL_LINE_STRIP`, `GL_LINE_LOOP`, `GL_LINES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`, `GL_TRIANGLES`, `GL_QUAD_STRIP`, `GL_QUADS`, and `GL_POLYGON` are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of `GL_UNSIGNED_BYTE`, `GL_UNSIGNED_SHORT`, or `GL_UNSIGNED_INT`.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints*T3 : struct*

5.37.2.712 `static void` **OpenTK.Graphics.OpenGL.GL.MultiDrawElements**< **T3** > (
OpenTK.Graphics.OpenGL.BeginMode *mode*, ref **Int32** *count*,
OpenTK.Graphics.OpenGL.DrawElementsType *type*,
[InAttribute, OutAttribute] **T3**[] *indices*, **Int32** *primcount*)
[**static**]

Render multiple sets of primitives by specifying indices of array data elements.

Parameters

mode Specifies what kind of primitives to render. Symbolic constants `GL_POINTS`, `GL_LINE_STRIP`, `GL_LINE_LOOP`, `GL_LINES`, `GL_TRIANGLE_STRIP`, `GL_TRIANGLE_FAN`, `GL_TRIANGLES`, `GL_QUAD_STRIP`, `GL_QUADS`, and `GL_POLYGON` are accepted.

count Points to an array of the elements counts.

type Specifies the type of the values in indices. Must be one of GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, or GL_UNSIGNED_INT.

indices Specifies a pointer to the location where the indices are stored.

primcount Specifies the size of the count array.

Type Constraints

T3 : struct

5.37.2.713 static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1 (OpenTK.Graphics.OpenGL.TextureUnit target, Single * v) [static]

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.714 static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1 (OpenTK.Graphics.OpenGL.TextureUnit target, Double s) [static]

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.715 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1 (OpenTK.Graphics.OpenGL.TextureUnit target, Double * v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.716 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1 (OpenTK.Graphics.OpenGL.TextureUnit target, Single s)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.717 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1 (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.718 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1
(OpenTK.Graphics.OpenGL.TextureUnit target, Int32 * v)
[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.719 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1
(OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s)
[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.720 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord1
(OpenTK.Graphics.OpenGL.TextureUnit target, Int16 * v)
[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.721 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 * v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.722 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Single * v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.723 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Double * v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.724 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Double s, Double t) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.725 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Double[] v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.726 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Double v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.727 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Single s, Single t) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.728 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Single[] v) [static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.729 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, ref Single v)`
`[static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.730 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s, Int32 t)`
`[static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.731 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Int32[] v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.732 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int32 v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.733 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s, Int16 t)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.734 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16[] v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.735 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int16 v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.736 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord2 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 * v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.737 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 * v)`
[**static**]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.738 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 * v)`
[**static**]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.739 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Double * v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.740 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Single * v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.741 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Double s, Double t, Double r) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.742 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, Double[] v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.743 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, ref Double v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.744 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, Single s, Single t,`
`Single r) [static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.745 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Single[] v) [static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.746 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Single v) [static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.747 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s, Int32 t,
Int32 r) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.748 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3`
`(OpenTK.Graphics.OpenGL.TextureUnit target, Int32[] v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.749 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, ref Int32 v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.750 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s, Int16 t, Int16 r) [static]`

Set the current texture coordinates.

Parameters

- target*** Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.751 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16[] v) [static]`

Set the current texture coordinates.

Parameters

- target*** Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.752 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord3 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int16 v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.753 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Int32 * v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.754 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Single * v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.755 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Double * v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE*i*, where *i* ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.756 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Double s, Double t, Double r, Double q) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE*i*, where *i* ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.757 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Double[] v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE*i*, where *i* ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.758 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, ref Double v)`
`[static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.759 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, Single s, Single t,`
`Single r, Single q) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.760 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4`
`(OpenTK.Graphics.OpenGL.TextureUnit target, Single[] v)`
`[static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.761 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, ref Single v)`
`[static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.762 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (`
`OpenTK.Graphics.OpenGL.TextureUnit target, Int32 s, Int32 t,`
`Int32 r, Int32 q) [static]`

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.763 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Int32[] v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.764 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int32 v) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.765 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 s, Int16 t, Int16 r, Int16 q) [static]`

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.766 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16[] v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.767 `static void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, ref Int16 v)`
[static]

Set the current texture coordinates.

Parameters

- target* Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.
- s* Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.768 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultiTexCoord4 (OpenTK.Graphics.OpenGL.TextureUnit target, Int16 * v)`
[static]

Set the current texture coordinates.

Parameters

target Specifies the texture unit whose coordinates should be modified. The number of texture units is implementation dependent, but must be at least two. Symbolic constant must be one of GL_TEXTURE, where i ranges from 0 to GL_MAX_TEXTURE_COORDS - 1, which is an implementation-dependent value.

s Specify s, t, r, and q texture coordinates for target texture unit. Not all parameters are present in all forms of the command.

5.37.2.769 `static void OpenTK.Graphics.OpenGL.GL.MultMatrix (Single[]
m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.37.2.770 `static void OpenTK.Graphics.OpenGL.GL.MultMatrix (Double[]
m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.37.2.771 `static void OpenTK.Graphics.OpenGL.GL.MultMatrix (ref Double
m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.37.2.772 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultMatrix (Double * m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.37.2.773 `static void OpenTK.Graphics.OpenGL.GL.MultMatrix (ref Single m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.37.2.774 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultMatrix (Single * m) [static]`

Multiply the current matrix with the specified matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 column-major matrix.

5.37.2.775 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultTransposeMatrix (Single * m) [static]`

Multiply the current matrix with the specified row-major ordered matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 row-major matrix.

5.37.2.776 `static void OpenTK.Graphics.OpenGL.GL.MultTransposeMatrix (ref Double m) [static]`

Multiply the current matrix with the specified row-major ordered matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 row-major matrix.

5.37.2.777 `static void OpenTK.Graphics.OpenGL.GL.MultTransposeMatrix (Double[] m) [static]`

Multiply the current matrix with the specified row-major ordered matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 row-major matrix.

5.37.2.778 `static unsafe void OpenTK.Graphics.OpenGL.GL.MultTransposeMatrix (Double * m) [static]`

Multiply the current matrix with the specified row-major ordered matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 row-major matrix.

5.37.2.779 `static void OpenTK.Graphics.OpenGL.GL.MultTransposeMatrix (Single[] m) [static]`

Multiply the current matrix with the specified row-major ordered matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 row-major matrix.

5.37.2.780 `static void OpenTK.Graphics.OpenGL.GL.MultTransposeMatrix (ref Single m) [static]`

Multiply the current matrix with the specified row-major ordered matrix.

Parameters

m Points to 16 consecutive values that are used as the elements of a 4 times 4 row-major matrix.

5.37.2.781 `static void OpenTK.Graphics.OpenGL.GL.NewList (Int32 list, OpenTK.Graphics.OpenGL.ListMode mode) [static]`

Create or replace a display list.

Parameters

list Specifies the display-list name.

mode Specifies the compilation mode, which can be GL_COMPILE or GL_COMPILE_AND_EXECUTE.

5.37.2.782 `static void OpenTK.Graphics.OpenGL.GL.NewList (UInt32 list, OpenTK.Graphics.OpenGL.ListMode mode) [static]`

Create or replace a display list.

Parameters

list Specifies the display-list name.

mode Specifies the compilation mode, which can be GL_COMPILE or GL_COMPILE_AND_EXECUTE.

5.37.2.783 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Byte nx, Byte ny, Byte nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.784 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (SByte nx,
SByte ny, SByte nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.785 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Byte[] v)
[static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.786 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (ref Byte v)
[static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.787 `static unsafe void OpenTK.Graphics.OpenGL.GL.Normal3 (Byte *
v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.788 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (ref SByte v)
[static]`

Set the current normal vector.

Parameters

nx Specify the , , and coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.789 `static unsafe void OpenTK.Graphics.OpenGL.GL.Normal3 (SByte
* v) [static]`

Set the current normal vector.

Parameters

nx Specify the , , and coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.790 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Double nx,
Double ny, Double nz) [static]`

Set the current normal vector.

Parameters

nx Specify the , , and coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.791 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (ref Double v
) [static]`

Set the current normal vector.

Parameters

nx Specify the , , and coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.792 `static unsafe void OpenTK.Graphics.OpenGL.GL.Normal3 (Double * v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.793 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Single nx, Single ny, Single nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.794 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (ref Single v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.795 `static unsafe void OpenTK.Graphics.OpenGL.GL.Normal3 (Single * v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.796 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Int32 nx, Int32 ny, Int32 nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.797 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Int32[] v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.798 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (ref Int32 v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.799 `static unsafe void OpenTK.Graphics.OpenGL.GL.Normal3 (Int32 * v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.800 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Int16 nx, Int16 ny, Int16 nz) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.801 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (ref Int16 v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.802 `static unsafe void OpenTK.Graphics.OpenGL.GL.Normal3 (Int16 * v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.803 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Int16[] v) [static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.804 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (SByte[] v)`
`[static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.805 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Single[] v)`
`[static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.806 `static void OpenTK.Graphics.OpenGL.GL.Normal3 (Double[] v)`
`[static]`

Set the current normal vector.

Parameters

nx Specify the *x*, *y*, and *z* coordinates of the new current normal. The initial value of the current normal is the unit vector, (0, 0, 1).

5.37.2.807 `static void OpenTK.Graphics.OpenGL.GL.NormalPointer (`
`OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride,`
`IntPtr pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

5.37.2.808 `static void OpenTK.Graphics.OpenGL.GL.NormalPointer< T2 > (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.809 `static void OpenTK.Graphics.OpenGL.GL.NormalPointer< T2 > (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] T2 pointer[,]) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : *struct*

5.37.2.810 `static void OpenTK.Graphics.OpenGL.GL.NormalPointer< T2 > (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] ref T2 pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : struct

5.37.2.811 `static void OpenTK.Graphics.OpenGL.GL.NormalPointer< T2 > (OpenTK.Graphics.OpenGL.NormalPointerType type, Int32 stride, [InAttribute, OutAttribute] T2[] pointer) [static]`

Define an array of normals.

Parameters

type Specifies the data type of each coordinate in the array. Symbolic constants GL_BYTE, GL_SHORT, GL_INT, GL_FLOAT, and GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive normals. If stride is 0, the normals are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first normal in the array. The initial value is 0.

Type Constraints

T2 : struct

5.37.2.812 `static void OpenTK.Graphics.OpenGL.GL.Ortho (Double left, Double right, Double bottom, Double top, Double zNear, Double zFar) [static]`

Multiply the current matrix with an orthographic matrix.

Parameters

left Specify the coordinates for the left and right vertical clipping planes.

bottom Specify the coordinates for the bottom and top horizontal clipping planes.

nearVal Specify the distances to the nearer and farther depth clipping planes. These values are negative if the plane is to be behind the viewer.

5.37.2.813 `static void OpenTK.Graphics.OpenGL.GL.PassThrough (Single token) [static]`

Place a marker in the feedback buffer.

Parameters

token Specifies a marker value to be placed in the feedback buffer following a GL_PASS_THROUGH_TOKEN.

5.37.2.814 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Single[] values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.815 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref`
`Single values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.816 `static unsafe void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Single`
`* values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.817 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Int32[]`
`values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.818 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref`
`Int32 values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.819 `static unsafe void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Int32 *`
`values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.820 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref`
`UInt32 values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.821 `static unsafe void OpenTK.Graphics.OpenGL.GL.PixelMap (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, UInt32 * values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.822 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Int16[] values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.823 `static unsafe void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, Int16 *`
`values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.824 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize,`
`UInt16[] values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.825 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref`
`UInt16 values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.826 `static unsafe void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, UInt16`
`* values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.827 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize, ref`
`Int16 values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.828 `static void OpenTK.Graphics.OpenGL.GL.PixelMap (`
`OpenTK.Graphics.OpenGL.PixelMap map, Int32 mapsize,`
`UInt32[] values) [static]`

Set up pixel transfer maps.

Parameters

map Specifies a symbolic map name. Must be one of the following: GL_PIXEL_MAP_I_TO_I, GL_PIXEL_MAP_S_TO_S, GL_PIXEL_MAP_I_TO_R, GL_PIXEL_MAP_I_TO_G, GL_PIXEL_MAP_I_TO_B, GL_PIXEL_MAP_I_TO_A, GL_PIXEL_MAP_R_TO_R, GL_PIXEL_MAP_G_TO_G, GL_PIXEL_MAP_B_TO_B, or GL_PIXEL_MAP_A_TO_A.

mapsize Specifies the size of the map being defined.

values Specifies an array of mapsize values.

5.37.2.829 `static void OpenTK.Graphics.OpenGL.GL.PixelStore (OpenTK.Graphics.OpenGL.PixelStoreParameter pname, Single param) [static]`

Set pixel storage modes.

Parameters

pname Specifies the symbolic name of the parameter to be set. Six values affect the packing of pixel data into memory: GL_PACK_SWAP_BYTES, GL_PACK_LSB_FIRST, GL_PACK_ROW_LENGTH, GL_PACK_IMAGE_HEIGHT, GL_PACK_SKIP_PIXELS, GL_PACK_SKIP_ROWS, GL_PACK_SKIP_IMAGES, and GL_PACK_ALIGNMENT. Six more affect the unpacking of pixel data from memory: GL_UNPACK_SWAP_BYTES, GL_UNPACK_LSB_FIRST, GL_UNPACK_ROW_LENGTH, GL_UNPACK_IMAGE_HEIGHT, GL_UNPACK_SKIP_PIXELS, GL_UNPACK_SKIP_ROWS, GL_UNPACK_SKIP_IMAGES, and GL_UNPACK_ALIGNMENT.

param Specifies the value that pname is set to.

5.37.2.830 `static void OpenTK.Graphics.OpenGL.GL.PixelStore (OpenTK.Graphics.OpenGL.PixelStoreParameter pname, Int32 param) [static]`

Set pixel storage modes.

Parameters

pname Specifies the symbolic name of the parameter to be set. Six values affect the packing of pixel data into memory: GL_PACK_SWAP_BYTES, GL_PACK_LSB_FIRST, GL_PACK_ROW_LENGTH, GL_PACK_IMAGE_HEIGHT, GL_PACK_SKIP_PIXELS, GL_PACK_SKIP_ROWS, GL_PACK_SKIP_IMAGES, and GL_PACK_ALIGNMENT. Six more

affect the unpacking of pixel data from memory: GL_UNPACK_SWAP_BYTES, GL_UNPACK_LSB_FIRST, GL_UNPACK_ROW_LENGTH, GL_UNPACK_IMAGE_HEIGHT, GL_UNPACK_SKIP_PIXELS, GL_UNPACK_SKIP_ROWS, GL_UNPACK_SKIP_IMAGES, and GL_UNPACK_ALIGNMENT.

param Specifies the value that pname is set to.

5.37.2.831 `static void OpenTK.Graphics.OpenGL.GL.PixelTransfer (OpenTK.Graphics.OpenGL.PixelTransferParameter pname, Single param) [static]`

Set pixel transfer modes.

Parameters

pname Specifies the symbolic name of the pixel transfer parameter to be set. Must be one of the following: GL_MAP_COLOR, GL_MAP_STENCIL, GL_INDEX_SHIFT, GL_INDEX_OFFSET, GL_RED_SCALE, GL_RED_BIAS, GL_GREEN_SCALE, GL_GREEN_BIAS, GL_BLUE_SCALE, GL_BLUE_BIAS, GL_ALPHA_SCALE, GL_ALPHA_BIAS, GL_DEPTH_SCALE, or GL_DEPTH_BIAS.

Additionally, if the ARB_imaging extension is supported, the following symbolic names are accepted: GL_POST_COLOR_MATRIX_RED_SCALE, GL_POST_COLOR_MATRIX_GREEN_SCALE, GL_POST_COLOR_MATRIX_BLUE_SCALE, GL_POST_COLOR_MATRIX_ALPHA_SCALE, GL_POST_COLOR_MATRIX_RED_BIAS, GL_POST_COLOR_MATRIX_GREEN_BIAS, GL_POST_COLOR_MATRIX_BLUE_BIAS, GL_POST_COLOR_MATRIX_ALPHA_BIAS, GL_POST_CONVOLUTION_RED_SCALE, GL_POST_CONVOLUTION_GREEN_SCALE, GL_POST_CONVOLUTION_BLUE_SCALE, GL_POST_CONVOLUTION_ALPHA_SCALE, GL_POST_CONVOLUTION_RED_BIAS, GL_POST_CONVOLUTION_GREEN_BIAS, GL_POST_CONVOLUTION_BLUE_BIAS, and GL_POST_CONVOLUTION_ALPHA_BIAS.

param Specifies the value that pname is set to.

5.37.2.832 `static void OpenTK.Graphics.OpenGL.GL.PixelTransfer (OpenTK.Graphics.OpenGL.PixelTransferParameter pname, Int32 param) [static]`

Set pixel transfer modes.

Parameters

pname Specifies the symbolic name of the pixel transfer parameter to be set. Must be one of the following: GL_MAP_COLOR, GL_MAP_STENCIL, GL_INDEX_SHIFT, GL_INDEX_OFFSET, GL_RED_SCALE, GL_RED_BIAS, GL_GREEN_SCALE, GL_GREEN_BIAS, GL_BLUE_SCALE, GL_BLUE_BIAS, GL_ALPHA_SCALE, GL_ALPHA_BIAS, GL_DEPTH_SCALE, or GL_DEPTH_BIAS.

Additionally, if the ARB_imaging extension is supported, the following symbolic names are accepted: GL_POST_COLOR_MATRIX_RED_SCALE, GL_POST_COLOR_MATRIX_GREEN_SCALE, GL_POST_COLOR_MATRIX_BLUE_SCALE, GL_POST_COLOR_MATRIX_ALPHA_SCALE, GL_POST_COLOR_MATRIX_RED_BIAS, GL_POST_COLOR_MATRIX_GREEN_BIAS, GL_POST_COLOR_MATRIX_BLUE_BIAS, GL_POST_COLOR_MATRIX_ALPHA_BIAS, GL_POST_CONVOLUTION_RED_SCALE, GL_POST_CONVOLUTION_GREEN_SCALE, GL_POST_CONVOLUTION_BLUE_SCALE, GL_POST_CONVOLUTION_ALPHA_SCALE, GL_POST_CONVOLUTION_RED_BIAS, GL_POST_CONVOLUTION_GREEN_BIAS, GL_POST_CONVOLUTION_BLUE_BIAS, and GL_POST_CONVOLUTION_ALPHA_BIAS.

param Specifies the value that pname is set to.

5.37.2.833 `static void OpenTK.Graphics.OpenGL.GL.PixelZoom (Single xfactor, Single yfactor) [static]`

Specify the pixel zoom factors.

Parameters

xfactor Specify the and zoom factors for pixel write operations.

5.37.2.834 `static void OpenTK.Graphics.OpenGL.GL.PointParameter (OpenTK.Graphics.OpenGL.PointParameterName pname, Single param) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.37.2.835 `static void OpenTK.Graphics.OpenGL.GL.PointParameter (`
`OpenTK.Graphics.OpenGL.PointParameterName pname, Single`
`@[] params) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.37.2.836 `static void OpenTK.Graphics.OpenGL.GL.PointParameter (`
`OpenTK.Graphics.OpenGL.PointParameterName pname, Int32`
`param) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.37.2.837 `static void OpenTK.Graphics.OpenGL.GL.PointParameter (`
`OpenTK.Graphics.OpenGL.PointParameterName pname, Int32`
`@[] params) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.37.2.838 `static unsafe void OpenTK.Graphics.OpenGL.GL.PointParameter (OpenTK.Graphics.OpenGL.PointParameterName pname, Int32 *@ params) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.37.2.839 `static unsafe void OpenTK.Graphics.OpenGL.GL.PointParameter (OpenTK.Graphics.OpenGL.PointParameterName pname, Single *@ params) [static]`

Specify point parameters.

Parameters

pname Specifies a single-valued point parameter. GL_POINT_SIZE_MIN, GL_POINT_SIZE_MAX, GL_POINT_FADE_THRESHOLD_SIZE, and GL_POINT_SPRITE_COORD_ORIGIN are accepted.

param Specifies the value that pname will be set to.

5.37.2.840 `static void OpenTK.Graphics.OpenGL.GL.PointParameter (PointSpriteCoordOriginParameter param) [static]`

Helper function that defines the coordinate origin of the Point Sprite.

Parameters

param A OpenTK.Graphics.OpenGL.GL.PointSpriteCoordOriginParameter token, denoting the origin of the Point Sprite.

5.37.2.841 `static void OpenTK.Graphics.OpenGL.GL.PointSize (Single size) [static]`

Specify the diameter of rasterized points.

Parameters

size Specifies the diameter of rasterized points. The initial value is 1.

5.37.2.842 `static void OpenTK.Graphics.OpenGL.GL.PolygonMode
(OpenTK.Graphics.OpenGL.MaterialFace face,
OpenTK.Graphics.OpenGL.PolygonMode mode) [static]`

Select a polygon rasterization mode.

Parameters

- face* Specifies the polygons that mode applies to. Must be GL_FRONT for front-facing polygons, GL_BACK for back-facing polygons, or GL_FRONT_AND_BACK for front- and back-facing polygons.
- mode* Specifies how polygons will be rasterized. Accepted values are GL_POINT, GL_LINE, and GL_FILL. The initial value is GL_FILL for both front- and back-facing polygons.

5.37.2.843 `static void OpenTK.Graphics.OpenGL.GL.PolygonOffset (Single
factor, Single units) [static]`

Set the scale and units used to calculate depth values.

Parameters

- factor* Specifies a scale factor that is used to create a variable depth offset for each polygon. The initial value is 0.
- units* Is multiplied by an implementation-specific value to create a constant depth offset. The initial value is 0.

5.37.2.844 `static void OpenTK.Graphics.OpenGL.GL.PolygonStipple (Byte[]
mask) [static]`

Set the polygon stippling pattern.

Parameters

- pattern* Specifies a pointer to a 32 times 32 stipple pattern that will be unpacked from memory in the same way that glDrawPixels unpacks pixels.

5.37.2.845 `static unsafe void OpenTK.Graphics.OpenGL.GL.PolygonStipple (Byte * mask) [static]`

Set the polygon stippling pattern.

Parameters

pattern Specifies a pointer to a 32 times 32 stipple pattern that will be unpacked from memory in the same way that `glDrawPixels` unpacks pixels.

5.37.2.846 `static void OpenTK.Graphics.OpenGL.GL.PolygonStipple (ref Byte mask) [static]`

Set the polygon stippling pattern.

Parameters

pattern Specifies a pointer to a 32 times 32 stipple pattern that will be unpacked from memory in the same way that `glDrawPixels` unpacks pixels.

5.37.2.847 `static void OpenTK.Graphics.OpenGL.GL.PrioritizeTextures (Int32 n, ref Int32 textures, ref Single priorities) [static]`

Set texture residence priority.

Parameters

n Specifies the number of textures to be prioritized.

textures Specifies an array containing the names of the textures to be prioritized.

priorities Specifies an array containing the texture priorities. A priority given in an element of *priorities* applies to the texture named by the corresponding element of *textures*.

5.37.2.848 `static unsafe void OpenTK.Graphics.OpenGL.GL.PrioritizeTextures (Int32 n, Int32 * textures, Single * priorities) [static]`

Set texture residence priority.

Parameters

n Specifies the number of textures to be prioritized.

textures Specifies an array containing the names of the textures to be prioritized.

priorities Specifies an array containing the texture priorities. A priority given in an element of *priorities* applies to the texture named by the corresponding element of *textures*.

5.37.2.849 `static void OpenTK.Graphics.OpenGL.GL.PrioritizeTextures (Int32 n, UInt32[] textures, Single[] priorities) [static]`

Set texture residence priority.

Parameters

n Specifies the number of textures to be prioritized.

textures Specifies an array containing the names of the textures to be prioritized.

priorities Specifies an array containing the texture priorities. A priority given in an element of priorities applies to the texture named by the corresponding element of textures.

5.37.2.850 `static unsafe void OpenTK.Graphics.OpenGL.GL.PrioritizeTextures (Int32 n, UInt32 * textures, Single * priorities) [static]`

Set texture residence priority.

Parameters

n Specifies the number of textures to be prioritized.

textures Specifies an array containing the names of the textures to be prioritized.

priorities Specifies an array containing the texture priorities. A priority given in an element of priorities applies to the texture named by the corresponding element of textures.

5.37.2.851 `static void OpenTK.Graphics.OpenGL.GL.PrioritizeTextures (Int32 n, Int32[] textures, Single[] priorities) [static]`

Set texture residence priority.

Parameters

n Specifies the number of textures to be prioritized.

textures Specifies an array containing the names of the textures to be prioritized.

priorities Specifies an array containing the texture priorities. A priority given in an element of priorities applies to the texture named by the corresponding element of textures.

5.37.2.852 `static void OpenTK.Graphics.OpenGL.GL.PrioritizeTextures (Int32 n, ref UInt32 textures, ref Single priorities) [static]`

Set texture residence priority.

Parameters

n Specifies the number of textures to be prioritized.

textures Specifies an array containing the names of the textures to be prioritized.

priorities Specifies an array containing the texture priorities. A priority given in an element of priorities applies to the texture named by the corresponding element of textures.

5.37.2.853 `static void OpenTK.Graphics.OpenGL.GL.PushAttrib (OpenTK.Graphics.OpenGL.AttribMask mask) [static]`

Push and pop the server attribute stack.

Parameters

mask Specifies a mask that indicates which attributes to save. Values for mask are listed below.

5.37.2.854 `static void OpenTK.Graphics.OpenGL.GL.PushClientAttrib (OpenTK.Graphics.OpenGL.ClientAttribMask mask) [static]`

Push and pop the client attribute stack.

Parameters

mask Specifies a mask that indicates which attributes to save. Values for mask are listed below.

5.37.2.855 `static void OpenTK.Graphics.OpenGL.GL.PushMatrix () [static]`

Push and pop the current matrix stack.

5.37.2.856 `static void OpenTK.Graphics.OpenGL.GL.PushName (Int32 name) [static]`

Push and pop the name stack.

Parameters

name Specifies a name that will be pushed onto the name stack.

5.37.2.857 `static void OpenTK.Graphics.OpenGL.GL.PushName (UInt32
name) [static]`

Push and pop the name stack.

Parameters

name Specifies a name that will be pushed onto the name stack.

5.37.2.858 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Double x,
Double y) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , and object coordinates (if present) for the raster position.

5.37.2.859 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Double[] v
) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , and object coordinates (if present) for the raster position.

5.37.2.860 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Double * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , and object coordinates (if present) for the raster position.

5.37.2.861 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Single x,
Single y) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.862 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Single[] v
) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.863 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Single * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.864 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Int32 x,
Int32 y) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.865 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Int32[] v)
[static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.866 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Int32 * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.867 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Int16 x, Int16 y) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.868 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Int16[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.869 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (ref Int16 v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.870 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos2 (Int16 * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.871 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (ref Int32 v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.872 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (ref Double v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.873 `static void OpenTK.Graphics.OpenGL.GL.RasterPos2 (ref Single v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.874 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Double x, Double y, Double z) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.875 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Double[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.876 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Double * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.877 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Single x, Single y, Single z) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.878 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Single[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.879 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Single * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.880 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Int32 x, Int32 y, Int32 z) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.881 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Int32[] v)
[static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.882 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Int32 * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.883 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Int16 x,
Int16 y, Int16 z) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.884 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Int16[] v)
[static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.885 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (ref Int16 v
) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.886 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos3 (Int16 * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.887 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (ref Int32 v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.888 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (ref Single v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.889 `static void OpenTK.Graphics.OpenGL.GL.RasterPos3 (ref Double v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.890 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Double x, Double y, Double z, Double w) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.891 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Double[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.892 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Double * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.893 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Single x, Single y, Single z, Single w) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.894 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Single[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.895 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Single * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.896 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Int32 x, Int32 y, Int32 z, Int32 w) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.897 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Int32[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.898 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Int32* v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.899 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Int16 x, Int16 y, Int16 z, Int16 w) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.900 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Int16[] v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.901 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (ref Int16 v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.902 `static unsafe void OpenTK.Graphics.OpenGL.GL.RasterPos4 (Int16 * v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.903 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (ref Single v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.904 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (ref Double v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.905 `static void OpenTK.Graphics.OpenGL.GL.RasterPos4 (ref Int32 v) [static]`

Specify the raster position for pixel operations.

Parameters

x Specify the , , , and object coordinates (if present) for the raster position.

5.37.2.906 `static void OpenTK.Graphics.OpenGL.GL.ReadBuffer (`
`OpenTK.Graphics.OpenGL.ReadBufferMode mode) [static]`

Select a color buffer source for pixels.

Parameters

mode Specifies a color buffer. Accepted values are GL_FRONT_LEFT, GL_FRONT_RIGHT, GL_BACK_LEFT, GL_BACK_RIGHT, GL_FRONT, GL_BACK, GL_LEFT, GL_RIGHT, and GL_AUXi, where i is between 0 and the value of GL_AUX_BUFFERS minus 1.

5.37.2.907 `static void OpenTK.Graphics.OpenGL.GL.ReadPixels`
`(Int32 x, Int32 y, Int32 width, Int32 height,`
`OpenTK.Graphics.OpenGL.PixelFormat format,`
`OpenTK.Graphics.OpenGL.PixelType type, [OutAttribute] IntPtr`
`pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

5.37.2.908 static void OpenTK.Graphics.OpenGL.GL.ReadPixels<
T6 > (Int32 *x*, Int32 *y*, Int32 *width*, Int32 *height*,
 OpenTK.Graphics.OpenGL.PixelFormat *format*,
 OpenTK.Graphics.OpenGL.PixelType *type*, [InAttribute,
 OutAttribute] **T6** *pixels*[,]) [static]

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. *width* and *height* of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : struct

5.37.2.909 static void OpenTK.Graphics.OpenGL.GL.ReadPixels<
T6 > (Int32 *x*, Int32 *y*, Int32 *width*, Int32 *height*,
 OpenTK.Graphics.OpenGL.PixelFormat *format*,
 OpenTK.Graphics.OpenGL.PixelType *type*, [InAttribute,
 OutAttribute] **T6** *pixels*[,]) [static]

Read a block of pixels from the frame buffer.

Parameters

- x** Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.
- width** Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.
- format** Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.
- type** Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.
- data** Returns the pixel data.

Type Constraints

T6 : struct

```
5.37.2.910 static void OpenTK.Graphics.OpenGL.GL.ReadPixels<
    T6 > ( Int32 x, Int32 y, Int32 width, Int32 height,
    OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] ref T6 pixels ) [static]
```

Read a block of pixels from the frame buffer.

Parameters

- x** Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.
- width** Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.
- format** Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : struct

5.37.2.911 `static void OpenTK.Graphics.OpenGL.GL.ReadPixels<T6> (Int32 x, Int32 y, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[] pixels) [static]`

Read a block of pixels from the frame buffer.

Parameters

x Specify the window coordinates of the first pixel that is read from the frame buffer. This location is the lower left corner of a rectangular block of pixels.

width Specify the dimensions of the pixel rectangle. width and height of one correspond to a single pixel.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_STENCIL_INDEX, GL_DEPTH_COMPONENT, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. Must be one of GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, or GL_UNSIGNED_INT_2_10_10_10_REV.

data Returns the pixel data.

Type Constraints

T6 : struct

5.37.2.912 `static void OpenTK.Graphics.OpenGL.GL.Rect (Double[] v1,
Double[] v2) [static]`

Draw a rectangle.

Parameters

x1 Specify one vertex of a rectangle.

x2 Specify the opposite vertex of the rectangle.

5.37.2.913 `static void OpenTK.Graphics.OpenGL.GL.Rect (ref Double v1,
ref Double v2) [static]`

Draw a rectangle.

Parameters

x1 Specify one vertex of a rectangle.

x2 Specify the opposite vertex of the rectangle.

5.37.2.914 `static unsafe void OpenTK.Graphics.OpenGL.GL.Rect (Double *
v1, Double * v2) [static]`

Draw a rectangle.

Parameters

x1 Specify one vertex of a rectangle.

x2 Specify the opposite vertex of the rectangle.

5.37.2.915 `static void OpenTK.Graphics.OpenGL.GL.Rect (Single[] v1,
Single[] v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.916 `static void OpenTK.Graphics.OpenGL.GL.Rect (ref Single v1, ref Single v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.917 `static unsafe void OpenTK.Graphics.OpenGL.GL.Rect (Single * v1, Single * v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.918 `static void OpenTK.Graphics.OpenGL.GL.Rect (Int32 x1, Int32 y1, Int32 x2, Int32 y2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.919 `static void OpenTK.Graphics.OpenGL.GL.Rect (Int32[] v1, Int32[] v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.920 `static void OpenTK.Graphics.OpenGL.GL.Rect (ref Int32 v1, ref Int32 v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.921 `static unsafe void OpenTK.Graphics.OpenGL.GL.Rect (Int32 * v1, Int32 * v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.922 `static void OpenTK.Graphics.OpenGL.GL.Rect (Int16[] v1, Int16[] v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.923 `static void OpenTK.Graphics.OpenGL.GL.Rect (ref Int16 v1, ref Int16 v2) [static]`

Draw a rectangle.

Parameters

- x1* Specify one vertex of a rectangle.
- x2* Specify the opposite vertex of the rectangle.

5.37.2.924 `static unsafe void OpenTK.Graphics.OpenGL.GL.Rect (Int16 * v1, Int16 * v2) [static]`

Draw a rectangle.

Parameters

x1 Specify one vertex of a rectangle.

x2 Specify the opposite vertex of the rectangle.

5.37.2.925 `static void OpenTK.Graphics.OpenGL.GL.Rect (Single x1, Single y1, Single x2, Single y2) [static]`

Draw a rectangle.

Parameters

x1 Specify one vertex of a rectangle.

x2 Specify the opposite vertex of the rectangle.

5.37.2.926 `static void OpenTK.Graphics.OpenGL.GL.Rect (Double x1, Double y1, Double x2, Double y2) [static]`

Draw a rectangle.

Parameters

x1 Specify one vertex of a rectangle.

x2 Specify the opposite vertex of the rectangle.

5.37.2.927 `static Int32 OpenTK.Graphics.OpenGL.GL.RenderMode (OpenTK.Graphics.OpenGL.RenderingMode mode) [static]`

Set rasterization mode.

Parameters

mode Specifies the rasterization mode. Three values are accepted: GL_RENDER, GL_SELECT, and GL_FEEDBACK. The initial value is GL_RENDER.

5.37.2.928 `static void OpenTK.Graphics.OpenGL.GL.ResetHistogram (OpenTK.Graphics.OpenGL.HistogramTarget target) [static]`

Reset histogram table entries to zero.

Parameters

target Must be GL_HISTOGRAM.

5.37.2.929 `static void OpenTK.Graphics.OpenGL.GL.ResetMinmax (OpenTK.Graphics.OpenGL.MinmaxTarget target) [static]`

Reset minmax table entries to initial values.

Parameters

target Must be GL_MINMAX.

5.37.2.930 `static void OpenTK.Graphics.OpenGL.GL.Rotate (Double angle, Double x, Double y, Double z) [static]`

Multiply the current matrix by a rotation matrix.

Parameters

angle Specifies the angle of rotation, in degrees.

x Specify the x, y, and z coordinates of a vector, respectively.

5.37.2.931 `static void OpenTK.Graphics.OpenGL.GL.Rotate (Single angle, Single x, Single y, Single z) [static]`

Multiply the current matrix by a rotation matrix.

Parameters

angle Specifies the angle of rotation, in degrees.

x Specify the x, y, and z coordinates of a vector, respectively.

5.37.2.932 `static void OpenTK.Graphics.OpenGL.GL.SampleCoverage (Single value, bool invert) [static]`

Specify multisample coverage parameters.

Parameters

value Specify a single floating-point sample coverage value. The value is clamped to the range [0,1]. The initial value is 1.0.

invert Specify a single boolean value representing if the coverage masks should be inverted. GL_TRUE and GL_FALSE are accepted. The initial value is GL_FALSE.

5.37.2.933 `static void OpenTK.Graphics.OpenGL.GL.Scale (Double x, Double y, Double z) [static]`

Multiply the current matrix by a general scaling matrix.

Parameters

x Specify scale factors along the x, y, and z axes, respectively.

5.37.2.934 `static void OpenTK.Graphics.OpenGL.GL.Scale (Single x, Single y, Single z) [static]`

Multiply the current matrix by a general scaling matrix.

Parameters

x Specify scale factors along the x, y, and z axes, respectively.

5.37.2.935 `static void OpenTK.Graphics.OpenGL.GL.Scissor (Int32 x, Int32 y, Int32 width, Int32 height) [static]`

Define the scissor box.

Parameters

x Specify the lower left corner of the scissor box. Initially (0, 0).

width Specify the width and height of the scissor box. When a GL context is first attached to a window, width and height are set to the dimensions of that window.

5.37.2.936 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (SByte red, SByte green, SByte blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.937 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (SByte[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.938 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref SByte v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.939 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Double red, Double green, Double blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.940 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Double[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.941 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref Double v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.942 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Single red, Single green, Single blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.943 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Single[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.944 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref Single v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.945 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Single * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.946 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Int32 red, Int32 green, Int32 blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.947 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref UInt16 v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.948 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Int32[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.949 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref Int32 v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.950 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Int16 red, Int16 green, Int16 blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.951 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Int16[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.952 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref Int16 v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.953 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Byte red, Byte green, Byte blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.954 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (Byte[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.955 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref Byte v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.956 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (UInt32 red, UInt32 green, UInt32 blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.957 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (UInt32[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.958 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (ref UInt32 v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.959 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (UInt16 red, UInt16 green, UInt16 blue) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.960 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColor3 (UInt16[] v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.961 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(UInt32 * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.962 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(UInt16 * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.963 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(Byte * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.964 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(Double * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.965 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(Int16 * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.966 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(Int32 * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.967 `static unsafe void OpenTK.Graphics.OpenGL.GL.SecondaryColor3
(SByte * v) [static]`

Set the current secondary color.

Parameters

red Specify new red, green, and blue values for the current secondary color.

5.37.2.968 `static void OpenTK.Graphics.OpenGL.GL.SecondaryColorPointer (
Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType type,
Int32 stride, IntPtr pointer) [static]`

Define an array of secondary colors.

Parameters

size Specifies the number of components per color. Must be 3.

type Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

5.37.2.969 `static void
OpenTK.Graphics.OpenGL.GL.SecondaryColorPointer< T3 >
(Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,])
[static]`

Define an array of secondary colors.

Parameters

- size* Specifies the number of components per color. Must be 3.
- type* Specifies the data type of each color component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.970 static void
      OpenTK.Graphics.OpenGL.GL.SecondaryColorPointer< T3 >
      ( Int32 size, OpenTK.Graphics.OpenGL.ColorPointerType
        type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer )
      [static]
```

Define an array of secondary colors.

Parameters

- size* Specifies the number of components per color. Must be 3.
- type* Specifies the data type of each color component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : struct

5.37.2.971 `static void`
OpenTK.Graphics.OpenGL.GL.SecondaryColorPointer< T3 >
 (Int32 *size*, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 *stride*, [InAttribute, OutAttribute] ref T3 *pointer*)
 [static]

Define an array of secondary colors.

Parameters

- size* Specifies the number of components per color. Must be 3.
- type* Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.37.2.972 `static void`
OpenTK.Graphics.OpenGL.GL.SecondaryColorPointer< T3 >
 (Int32 *size*, OpenTK.Graphics.OpenGL.ColorPointerType
type, Int32 *stride*, [InAttribute, OutAttribute] T3 *pointer*[,,])
 [static]

Define an array of secondary colors.

Parameters

- size* Specifies the number of components per color. Must be 3.
- type* Specifies the data type of each color component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive colors. If stride is 0, the colors are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first color element in the array. The initial value is 0.

Type Constraints

T3 : *struct*

5.37.2.973 `static void OpenTK.Graphics.OpenGL.GL.SelectBuffer (Int32 size, [OutAttribute] Int32[] buffer) [static]`

Establish a buffer for selection mode values.

Parameters

size Specifies the size of buffer.

buffer Returns the selection data.

5.37.2.974 `static void OpenTK.Graphics.OpenGL.GL.SelectBuffer (Int32 size, [OutAttribute] out Int32 buffer) [static]`

Establish a buffer for selection mode values.

Parameters

size Specifies the size of buffer.

buffer Returns the selection data.

5.37.2.975 `static void OpenTK.Graphics.OpenGL.GL.SelectBuffer (Int32 size, [OutAttribute] UInt32[] buffer) [static]`

Establish a buffer for selection mode values.

Parameters

size Specifies the size of buffer.

buffer Returns the selection data.

5.37.2.976 `static void OpenTK.Graphics.OpenGL.GL.SelectBuffer (Int32 size, [OutAttribute] out UInt32 buffer) [static]`

Establish a buffer for selection mode values.

Parameters

size Specifies the size of buffer.

buffer Returns the selection data.

5.37.2.977 `static unsafe void OpenTK.Graphics.OpenGL.GL.SelectBuffer (Int32 size, [OutAttribute] UInt32 * buffer) [static]`

Establish a buffer for selection mode values.

Parameters

size Specifies the size of buffer.
buffer Returns the selection data.

5.37.2.978 `static unsafe void OpenTK.Graphics.OpenGL.GL.SelectBuffer (Int32 size, [OutAttribute] Int32 * buffer) [static]`

Establish a buffer for selection mode values.

Parameters

size Specifies the size of buffer.
buffer Returns the selection data.

5.37.2.979 `static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D (OpenTK.Graphics.OpenGL.SeparableTarget target, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row, IntPtr column) [static]`

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.
internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

```
5.37.2.980 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
T6, T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] ref T6 row, [InAttribute, OutAttribute] T7
column[, ] ) [static]
```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_

LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T6 : *struct*

T7 : *struct*

```
5.37.2.981 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
    T6, T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T6 row[,], [InAttribute, OutAttribute] T7 column[,]
    ) [static]
```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T6 : struct

T7 : struct

```

5.37.2.982 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
            T6, T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
            OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
            Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
            format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
            OutAttribute] T6[] row, [InAttribute, OutAttribute] T7 column[],
            ) [static]

```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T6 : *struct*

T7 : *struct*

```
5.37.2.983 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
    T6, T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T6 row[], [InAttribute, OutAttribute] T7 column[],
    ) [static]
```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGBA5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT,

GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T6 : *struct*

T7 : *struct*

```
5.37.2.984 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
    T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,
    [InAttribute, OutAttribute] T7 column[,/] ) [static]
```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T7 : struct

```
5.37.2.985 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
    T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
    OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
    Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,
    [InAttribute, OutAttribute] T7 column[, ] ) [static]
```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_-

LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T7 : *struct*

```
5.37.2.986 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,
[InAttribute, OutAttribute] T7[] column ) [static]
```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T7 : struct

```

5.37.2.987 static void OpenTK.Graphics.OpenGL.GL.SeparableFilter2D<
T7 > ( OpenTK.Graphics.OpenGL.SeparableTarget target,
OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat,
Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr row,
[InAttribute, OutAttribute] ref T7 column ) [static]

```

Define a separable two-dimensional convolution filter.

Parameters

target Must be GL_SEPARABLE_2D.

internalformat The internal format of the convolution filter kernel. The allowable values are GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, or GL_RGBA16.

width The number of elements in the pixel array referenced by row. (This is the width of the separable filter kernel.)

height The number of elements in the pixel array referenced by column. (This is the height of the separable filter kernel.)

format The format of the pixel data in row and column. The allowable values are GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_INTENSITY, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type The type of the pixel data in row and column. Symbolic constants GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV are accepted.

row Pointer to a one-dimensional array of pixel data that is processed to build the row filter kernel.

column Pointer to a one-dimensional array of pixel data that is processed to build the column filter kernel.

Type Constraints

T7 : *struct*

5.37.2.988 `static void OpenTK.Graphics.OpenGL.GL.ShadeModel (OpenTK.Graphics.OpenGL.ShadingModel mode) [static]`

Select flat or smooth shading.

Parameters

mode Specifies a symbolic value representing a shading technique. Accepted values are GL_FLAT and GL_SMOOTH. The initial value is GL_SMOOTH.

5.37.2.989 `static unsafe void OpenTK.Graphics.OpenGL.GL.ShaderSource (Int32 shader, Int32 count, String @[] string, Int32 * length) [static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.37.2.990 `static void OpenTK.Graphics.OpenGL.GL.ShaderSource (Int32 shader, Int32 count, String @[] string, ref Int32 length) [static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.37.2.991 `static void OpenTK.Graphics.OpenGL.GL.ShaderSource (UInt32 shader, Int32 count, String @[] string, ref Int32 length) [static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.37.2.992 `static unsafe void OpenTK.Graphics.OpenGL.GL.ShaderSource (UInt32 shader, Int32 count, String @[] string, Int32 * length) [static]`

Replaces the source code in a shader object.

Parameters

shader Specifies the handle of the shader object whose source code is to be replaced.

count Specifies the number of elements in the string and length arrays.

string Specifies an array of pointers to strings containing the source code to be loaded into the shader.

length Specifies an array of string lengths.

5.37.2.993 `static void OpenTK.Graphics.OpenGL.GL.StencilFunc (OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ ref, UInt32 mask) [static]`

Set front and back function and reference value for stencil testing.

Parameters

- func*** Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.
- ref*** Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.
- mask*** Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.37.2.994 `static void OpenTK.Graphics.OpenGL.GL.StencilFunc (`
`OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ ref,`
`Int32 mask) [static]`

Set front and back function and reference value for stencil testing.

Parameters

- func*** Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.
- ref*** Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where *n* is the number of bitplanes in the stencil buffer. The initial value is 0.
- mask*** Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.37.2.995 `static void OpenTK.Graphics.OpenGL.GL.StencilFuncSeparate`
`(OpenTK.Graphics.OpenGL.StencilFace face,`
`OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ ref,`
`Int32 mask) [static]`

Set front and/or back function and reference value for stencil testing.

Parameters

- face*** Specifies whether front and/or back stencil state is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.
- func*** Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.

ref Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where n is the number of bitplanes in the stencil buffer. The initial value is 0.

mask Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.37.2.996 `static void OpenTK.Graphics.OpenGL.GL.StencilFuncSeparate (OpenTK.Graphics.OpenGL.StencilFace face, OpenTK.Graphics.OpenGL.StencilFunction func, Int32 @ ref, UInt32 mask) [static]`

Set front and/or back function and reference value for stencil testing.

Parameters

face Specifies whether front and/or back stencil state is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

func Specifies the test function. Eight symbolic constants are valid: GL_NEVER, GL_LESS, GL_LEQUAL, GL_GREATER, GL_GEQUAL, GL_EQUAL, GL_NOTEQUAL, and GL_ALWAYS. The initial value is GL_ALWAYS.

ref Specifies the reference value for the stencil test. *ref* is clamped to the range $[0, 2^{\sup n - 1}]$, where n is the number of bitplanes in the stencil buffer. The initial value is 0.

mask Specifies a mask that is ANDed with both the reference value and the stored stencil value when the test is done. The initial value is all 1's.

5.37.2.997 `static void OpenTK.Graphics.OpenGL.GL.StencilMask (Int32 mask) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.37.2.998 `static void OpenTK.Graphics.OpenGL.GL.StencilMask (UInt32 mask) [static]`

Control the front and back writing of individual bits in the stencil planes.

Parameters

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.37.2.999 `static void OpenTK.Graphics.OpenGL.GL.StencilMaskSeparate (OpenTK.Graphics.OpenGL.StencilFace face, Int32 mask) [static]`

Control the front and/or back writing of individual bits in the stencil planes.

Parameters

face Specifies whether the front and/or back stencil writemask is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.37.2.1000 `static void OpenTK.Graphics.OpenGL.GL.StencilMaskSeparate (OpenTK.Graphics.OpenGL.StencilFace face, UInt32 mask) [static]`

Control the front and/or back writing of individual bits in the stencil planes.

Parameters

face Specifies whether the front and/or back stencil writemask is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

mask Specifies a bit mask to enable and disable writing of individual bits in the stencil planes. Initially, the mask is all 1's.

5.37.2.1001 `static void OpenTK.Graphics.OpenGL.GL.StencilOp (OpenTK.Graphics.OpenGL.StencilOp fail, OpenTK.Graphics.OpenGL.StencilOp zfail, OpenTK.Graphics.OpenGL.StencilOp zpass) [static]`

Set front and back stencil test actions.

Parameters

sfail Specifies the action to take when the stencil test fails. Eight symbolic constants are accepted: GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR,

GL_INCR_WRAP, GL_DECAR, GL_DECAR_WRAP, and GL_INVERT. The initial value is GL_KEEP.

dpfail Specifies the stencil action when the stencil test passes, but the depth test fails. *dpfail* accepts the same symbolic constants as *sfail*. The initial value is GL_KEEP.

dppass Specifies the stencil action when both the stencil test and the depth test pass, or when the stencil test passes and either there is no depth buffer or depth testing is not enabled. *dppass* accepts the same symbolic constants as *sfail*. The initial value is GL_KEEP.

```
5.37.2.1002 static void OpenTK.Graphics.OpenGL.GL.StencilOpSeparate
( OpenTK.Graphics.OpenGL.StencilFace face,
  OpenTK.Graphics.OpenGL.StencilOp sfail,
  OpenTK.Graphics.OpenGL.StencilOp dpfail,
  OpenTK.Graphics.OpenGL.StencilOp dppass ) [static]
```

Set front and/or back stencil test actions.

Parameters

face Specifies whether front and/or back stencil state is updated. Three symbolic constants are valid: GL_FRONT, GL_BACK, and GL_FRONT_AND_BACK.

sfail Specifies the action to take when the stencil test fails. Eight symbolic constants are accepted: GL_KEEP, GL_ZERO, GL_REPLACE, GL_INCR, GL_INCR_WRAP, GL_DECAR, GL_DECAR_WRAP, and GL_INVERT. The initial value is GL_KEEP.

dpfail Specifies the stencil action when the stencil test passes, but the depth test fails. *dpfail* accepts the same symbolic constants as *sfail*. The initial value is GL_KEEP.

dppass Specifies the stencil action when both the stencil test and the depth test pass, or when the stencil test passes and either there is no depth buffer or depth testing is not enabled. *dppass* accepts the same symbolic constants as *sfail*. The initial value is GL_KEEP.

```
5.37.2.1003 static void OpenTK.Graphics.OpenGL.GL.TexCoord1 ( Double s
) [static]
```

Set the current texture coordinates.

Parameters

s Specify *s*, *t*, *r*, and *q* texture coordinates. Not all parameters are present in all forms of the command.

**5.37.2.1004 static void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Single *s*)
[static]**

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1005 static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Single * *v*) [static]

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1006 static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Int16 * *v*) [static]

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1007 static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Int32 * *v*) [static]

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1008 `static void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Int32 s)
[static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1009 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Double * v) [static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1010 `static void OpenTK.Graphics.OpenGL.GL.TexCoord1 (Int16 s)
[static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1011 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Double s,
Double t) [static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1012 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (ref Int32
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1013 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Single[] v
) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1014 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord2 (
Double * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1015 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord2 (
Single * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1016 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Int16 s, Int16 t) [static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1017 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Int32 * v) [static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1018 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Int16 * v) [static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1019 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Int32 s, Int32 t) [static]`

Set the current texture coordinates.

Parameters

s Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1020 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Int16[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1021 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Int32[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1022 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Double[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1023 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (Single s, Single t) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1024 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (ref Double v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1025 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (ref Single v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1026 `static void OpenTK.Graphics.OpenGL.GL.TexCoord2 (ref Int16 v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1027 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Int32[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1028 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (ref Int32
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1029 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (ref Single
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1030 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (ref
Double v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1031 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Single[] v
) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1032 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Int16[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1033 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Int16 * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1034 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Double * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1035 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Single * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1036 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Int32 * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1037 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Double[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1038 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Double s, Double t, Double r) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1039 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Int32 s, Int32 t, Int32 r) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1040 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (ref Int16
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1041 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Single s,
Single t, Single r) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1042 `static void OpenTK.Graphics.OpenGL.GL.TexCoord3 (Int16 s,
Int16 t, Int16 r) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1043 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (ref Int16
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1044 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (ref Int32
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1045 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Int32[] v
) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1046 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Single[] v
) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1047 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord4 (
Double * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1048 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Single * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1049 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Int32 * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1050 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Int16[] v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1051 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Int16 * v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1052 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Double s,
Double t, Double r, Double q) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1053 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (ref
Double v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1054 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Double[]
v) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1055 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Int16 s,
Int16 t, Int16 r, Int16 q) [static]`

Set the current texture coordinates.

Parameters

- s* Specify s, t, r, and q texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1056 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Int32 s,
Int32 t, Int32 r, Int32 q) [static]`

Set the current texture coordinates.

Parameters

s Specify *s*, *t*, *r*, and *q* texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1057 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (Single s,
Single t, Single r, Single q) [static]`

Set the current texture coordinates.

Parameters

s Specify *s*, *t*, *r*, and *q* texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1058 `static void OpenTK.Graphics.OpenGL.GL.TexCoord4 (ref Single
v) [static]`

Set the current texture coordinates.

Parameters

s Specify *s*, *t*, *r*, and *q* texture coordinates. Not all parameters are present in all forms of the command.

5.37.2.1059 `static void OpenTK.Graphics.OpenGL.GL.TexCoordPointer (Int32 size,
OpenTK.Graphics.OpenGL.TexCoordPointerType type, Int32 stride, IntPtr pointer) [static]`

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

```
5.37.2.1060 static void OpenTK.Graphics.OpenGL.GL.TexCoordPointer< T3
> ( Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[, ] )
[static]
```

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1061 static void OpenTK.Graphics.OpenGL.GL.TexCoordPointer< T3
> ( Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer )
[static]
```

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1062 static void OpenTK.Graphics.OpenGL.GL.TexCoordPointer< T3
> ( Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[, ] )
[static]
```

Define an array of texture coordinates.

Parameters

size Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1063 static void OpenTK.Graphics.OpenGL.GL.TexCoordPointer< T3
> ( Int32 size, OpenTK.Graphics.OpenGL.TexCoordPointerType
type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer )
[static]
```

Define an array of texture coordinates.

Parameters

- size* Specifies the number of coordinates per array element. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each texture coordinate. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- stride* Specifies the byte offset between consecutive texture coordinate sets. If stride is 0, the array elements are understood to be tightly packed. The initial value is 0.
- pointer* Specifies a pointer to the first coordinate of the first texture coordinate set in the array. The initial value is 0.

Type Constraints

T3 : struct

5.37.2.1064 `static unsafe void OpenTK.Graphics.OpenGL.GL.TextureEnv
(OpenTK.Graphics.OpenGL.TextureEnvTarget target,
OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Int32
*@ params) [static]`

Set texture environment parameters.

Parameters

- target* Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.
- pname* Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.
- param* Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.37.2.1065 `static void OpenTK.Graphics.OpenGL.GL.TexEnv (`
`OpenTK.Graphics.OpenGL.TextureEnvTarget target,`
`OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Single`
`@[] params) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.37.2.1066 `static void OpenTK.Graphics.OpenGL.GL.TexEnv (`
`OpenTK.Graphics.OpenGL.TextureEnvTarget target,`
`OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Single`
`param) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA,

GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.37.2.1067 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexEnv (OpenTK.Graphics.OpenGL.TextureEnvTarget target, OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Single *@ params) [static]`

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

```
5.37.2.1068 static void OpenTK.Graphics.OpenGL.GL.TexEnv (
    OpenTK.Graphics.OpenGL.TextureEnvTarget target,
    OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Int32
    @[] params ) [static]
```

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA, GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

```
5.37.2.1069 static void OpenTK.Graphics.OpenGL.GL.TexEnv (
    OpenTK.Graphics.OpenGL.TextureEnvTarget target,
    OpenTK.Graphics.OpenGL.TextureEnvParameter pname, Int32
    param ) [static]
```

Set texture environment parameters.

Parameters

target Specifies a texture environment. May be GL_TEXTURE_ENV, GL_TEXTURE_FILTER_CONTROL or GL_POINT_SPRITE.

pname Specifies the symbolic name of a single-valued texture environment parameter. May be either GL_TEXTURE_ENV_MODE, GL_TEXTURE_LOD_BIAS, GL_COMBINE_RGB, GL_COMBINE_ALPHA, GL_SRC0_RGB, GL_SRC1_RGB, GL_SRC2_RGB, GL_SRC0_ALPHA,

GL_SRC1_ALPHA, GL_SRC2_ALPHA, GL_OPERAND0_RGB, GL_OPERAND1_RGB, GL_OPERAND2_RGB, GL_OPERAND0_ALPHA, GL_OPERAND1_ALPHA, GL_OPERAND2_ALPHA, GL_RGB_SCALE, GL_ALPHA_SCALE, or GL_COORD_REPLACE.

param Specifies a single symbolic constant, one of GL_ADD, GL_ADD_SIGNED, GL_INTERPOLATE, GL_MODULATE, GL_DECAL, GL_BLEND, GL_REPLACE, GL_SUBTRACT, GL_COMBINE, GL_TEXTURE, GL_CONSTANT, GL_PRIMARY_COLOR, GL_PREVIOUS, GL_SRC_COLOR, GL_ONE_MINUS_SRC_COLOR, GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA, a single boolean value for the point sprite texture coordinate replacement, a single floating-point value for the texture level-of-detail bias, or 1.0, 2.0, or 4.0 when specifying the GL_RGB_SCALE or GL_ALPHA_SCALE.

5.37.2.1070 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexGen (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Single *@ params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1071 `static void OpenTK.Graphics.OpenGL.GL.TexGen (OpenTK.Graphics.OpenGL.TextureCoordName coord, OpenTK.Graphics.OpenGL.TextureGenParameter pname, Double @[] params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1072 `static void OpenTK.Graphics.OpenGL.GL.TexGen (`
`OpenTK.Graphics.OpenGL.TextureCoordName coord,`
`OpenTK.Graphics.OpenGL.TextureGenParameter pname, ref`
`Double @ params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1073 `static void OpenTK.Graphics.OpenGL.GL.TexGen (`
`OpenTK.Graphics.OpenGL.TextureCoordName coord,`
`OpenTK.Graphics.OpenGL.TextureGenParameter pname, Single`
`param) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1074 `static void OpenTK.Graphics.OpenGL.GL.TexGen (`
 `OpenTK.Graphics.OpenGL.TextureCoordName coord,`
 `OpenTK.Graphics.OpenGL.TextureGenParameter pname, Int32`
 `@[] params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1075 `static void OpenTK.Graphics.OpenGL.GL.TexGen (`
 `OpenTK.Graphics.OpenGL.TextureCoordName coord,`
 `OpenTK.Graphics.OpenGL.TextureGenParameter pname, Single`
 `@[] params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1076 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexGen`
 `(OpenTK.Graphics.OpenGL.TextureCoordName coord,`
 `OpenTK.Graphics.OpenGL.TextureGenParameter pname,`
 `Double *@ params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1077 `static unsafe void OpenTK.Graphics.OpenGL.GL.TexGen
(OpenTK.Graphics.OpenGL.TextureCoordName coord,
OpenTK.Graphics.OpenGL.TextureGenParameter pname, Int32
*@ params) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1078 `static void OpenTK.Graphics.OpenGL.GL.TexGen (`
`OpenTK.Graphics.OpenGL.TextureCoordName coord,`
`OpenTK.Graphics.OpenGL.TextureGenParameter pname, Int32`
`param) [static]`

Control the generation of texture coordinates.

Parameters

coord Specifies a texture coordinate. Must be one of GL_S, GL_T, GL_R, or GL_Q.

pname Specifies the symbolic name of the texture-coordinate generation function. Must be GL_TEXTURE_GEN_MODE.

param Specifies a single-valued texture generation parameter, one of GL_OBJECT_LINEAR, GL_EYE_LINEAR, GL_SPHERE_MAP, GL_NORMAL_MAP, or GL_REFLECTION_MAP.

5.37.2.1079 static void OpenTK.Graphics.OpenGL.GL TexImage1D (
 OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32 *level*,
 OpenTK.Graphics.OpenGL.PixelInternalFormat *internalformat*,
 Int32 *width*, Int32 *border*,
 OpenTK.Graphics.OpenGL.PixelFormat *format*,
 OpenTK.Graphics.OpenGL.PixelType *type*, IntPtr *pixels*)
 [static]

Specify a one-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n + 2 (border)}$ for some integer *n*. All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_

BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

```
5.37.2.1080 static void OpenTK.Graphics.OpenGL.GL.TexImage1D<
T7 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
internalformat, Int32 width, Int32 border,
OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T7 pixels[,]) [static]
```

Specify a one-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12,

GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T7 : struct

```
5.37.2.1081 static void OpenTK.Graphics.OpenGL.GL.TextureImage1D<
    T7 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 border,
    OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] ref T7 pixels ) [static]
```

Specify a one-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2 \text{ (border)}$ for some integer n. All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8,

GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T7 : *struct*

5.37.2.1082 `static void OpenTK.Graphics.OpenGL.GL.Texture1D< T7 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T7 pixels[,]) [static]`

Specify a one-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n + 2}$ (border) for some integer . All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T7 : struct

```
5.37.2.1083 static void OpenTK.Graphics.OpenGL.GL.Texture1D<
    T7 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 border,
    OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T7[] pixels ) [static]
```

Specify a one-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D or GL_PROXY_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{n+2} \cdot (\text{border})$ for some integer n . All implementations support texture images that are at least 64 texels wide. The height of the 1D texture image is 1.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T7 : *struct*

```

5.37.2.1084 static void OpenTK.Graphics.OpenGL.GL.TextureImage2D (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32
    level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels )
    [static]

```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2

$\text{sup } m + 2 \text{ (border)}$ for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

```
5.37.2.1085 static void OpenTK.Graphics.OpenGL.GL.TextureImage2D<
    T8 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T8 pixels[,]) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE,

GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m} + 2$ (border) for some integer . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct


```
5.37.2.1086 static void OpenTK.Graphics.OpenGL.GL TexImage2D<
    T8 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T8 pixels[, ] ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2} \cdot (\text{border})$ for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n + 2} \cdot (\text{border})$ for some integer n .

$\text{sup } m + 2 (\text{border})$ for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.37.2.1087 static void OpenTK.Graphics.OpenGL.GL.TextureImage2D<
    T8 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T8[] pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n th mipmap reduction image.

internalFormat Specifies the number of color components in the texture.

Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n} + 2 \cdot (\text{border})$ for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m} + 2 \cdot (\text{border})$ for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.37.2.1088 static void OpenTK.Graphics.OpenGL.GL.TextureImage2D<
T8 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
internalformat, Int32 width, Int32 height, Int32
border, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] ref T8 pixels ) [static]
```

Specify a two-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_PROXY_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, GL_TEXTURE_CUBE_MAP_NEGATIVE_Z, or GL_PROXY_TEXTURE_CUBE_MAP.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_DEPTH_COMPONENT, GL_DEPTH_COMPONENT16, GL_DEPTH_COMPONENT24, GL_DEPTH_COMPONENT32, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support texture images that are at least 64 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{m+2} (border) for some integer m . All implementations support texture images that are at least 64 texels high.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

```
5.37.2.1089 static void OpenTK.Graphics.OpenGL.GL.TextureImage3D (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32
    level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32 depth, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels )
    [static]
```

Specify a three-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n sup th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m} + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup k} + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_

UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8,
GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2,
and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.37.2.1090 `static void OpenTK.Graphics.OpenGL.GL TexImage3D<
T9 > (OpenTK.Graphics.OpenGL.TextureTarget target,
Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
internalformat, Int32 width, Int32 height, Int32 depth, Int32
border, OpenTK.Graphics.OpenGL.PixelFormat format,
OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T9 pixels[,]) [static]`

Specify a three-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n sup th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be 2^{n+2} (border) for some integer n . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^m + 2$ (border) for some integer m . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^k + 2$ (border) for some integer k . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T9 : *struct*

```
5.37.2.1091 static void OpenTK.Graphics.OpenGL.GL.TextureImage3D<
    T9 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32 depth, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T9[] pixels ) [static]
```

Specify a three-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n sup th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } n} + 2$ (border) for some integer n . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } m} + 2$ (border) for some integer m . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\text{sup } k} + 2$ (border) for some integer k . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_

UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T9 : *struct*

5.37.2.1092 `static void OpenTK.Graphics.OpenGL.GL.TextureImage3D< T9 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat internalformat, Int32 width, Int32 height, Int32 depth, Int32 border, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T9 pixels[,]) [static]`

Specify a three-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level is the n sup th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup n} + 2 \cdot (\text{border})$ for some integer n . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup m} + 2 \cdot (\text{border})$ for some integer m . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{\sup k} + 2 \cdot (\text{border})$ for some integer k . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T9 : *struct*

```
5.37.2.1093 static void OpenTK.Graphics.OpenGL.GL.TextureImage3D<
    T9 > ( OpenTK.Graphics.OpenGL.TextureTarget target,
    Int32 level, OpenTK.Graphics.OpenGL.PixelInternalFormat
    internalformat, Int32 width, Int32 height, Int32 depth, Int32
    border, OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] ref T9 pixels ) [static]
```

Specify a three-dimensional texture image.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D or GL_PROXY_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the n sup th mipmap reduction image.

internalFormat Specifies the number of color components in the texture. Must be 1, 2, 3, or 4, or one of the following symbolic constants: GL_ALPHA, GL_ALPHA4, GL_ALPHA8, GL_ALPHA12, GL_ALPHA16, GL_COMPRESSED_ALPHA, GL_COMPRESSED_LUMINANCE, GL_COMPRESSED_LUMINANCE_ALPHA, GL_COMPRESSED_INTENSITY, GL_COMPRESSED_RGB, GL_COMPRESSED_RGBA, GL_LUMINANCE, GL_LUMINANCE4, GL_LUMINANCE8, GL_LUMINANCE12, GL_LUMINANCE16, GL_LUMINANCE_ALPHA, GL_LUMINANCE4_ALPHA4, GL_LUMINANCE6_ALPHA2, GL_LUMINANCE8_ALPHA8, GL_LUMINANCE12_ALPHA4, GL_LUMINANCE12_ALPHA12, GL_LUMINANCE16_ALPHA16, GL_INTENSITY, GL_INTENSITY4, GL_INTENSITY8, GL_INTENSITY12, GL_INTENSITY16, GL_R3_G3_B2, GL_RGB, GL_RGB4, GL_RGB5, GL_RGB8, GL_RGB10, GL_RGB12, GL_RGB16, GL_RGBA, GL_RGBA2, GL_RGBA4, GL_RGB5_A1, GL_RGBA8, GL_RGB10_A2, GL_RGBA12, GL_RGBA16, GL_SLUMINANCE, GL_SLUMINANCE8, GL_SLUMINANCE_ALPHA, GL_SLUMINANCE8_ALPHA8, GL_SRGB, GL_SRGB8, GL_SRGB_ALPHA, or GL_SRGB8_ALPHA8.

width Specifies the width of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup n} + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels wide.

height Specifies the height of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup m} + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels high.

depth Specifies the depth of the texture image including the border if any. If the GL version does not support non-power-of-two sizes, this value must be $2^{sup k} + 2$ (border) for some integer . All implementations support 3D texture images that are at least 16 texels deep.

border Specifies the width of the border. Must be either 0 or 1.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_

FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T9 : *struct*

5.37.2.1094 static void OpenTK.Graphics.OpenGL.GL.TextureParameter
(OpenTK.Graphics.OpenGL.TextureTarget *target*,
OpenTK.Graphics.OpenGL.TextureParameterName *pname*,
Single @[] *params*) [static]

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.37.2.1095 static void OpenTK.Graphics.OpenGL.GL.TextureParameter
(OpenTK.Graphics.OpenGL.TextureTarget *target*,
OpenTK.Graphics.OpenGL.TextureParameterName *pname*,
Int32 @[] *params*) [static]

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.37.2.1096 `static unsafe void OpenTK.Graphics.OpenGL.GL.TextureParameter (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Single*@ params) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

5.37.2.1097 `static void OpenTK.Graphics.OpenGL.GL.TextureParameter (OpenTK.Graphics.OpenGL.TextureTarget target, OpenTK.Graphics.OpenGL.TextureParameterName pname, Single param) [static]`

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

```
5.37.2.1098 static unsafe void OpenTK.Graphics.OpenGL.GL.TextureParameter
( OpenTK.Graphics.OpenGL.TextureTarget target,
  OpenTK.Graphics.OpenGL.TextureParameterName pname,
  Int32 *params ) [static]
```

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

```
5.37.2.1099 static void OpenTK.Graphics.OpenGL.GL.TextureParameter
( OpenTK.Graphics.OpenGL.TextureTarget target,
  OpenTK.Graphics.OpenGL.TextureParameterName pname,
  Int32 param ) [static]
```

Set texture parameters.

Parameters

target Specifies the target texture, which must be either GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D, or GL_TEXTURE_CUBE_MAP.

pname Specifies the symbolic name of a single-valued texture parameter. *pname* can be one of the following: GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAG_FILTER, GL_TEXTURE_MIN_LOD, GL_TEXTURE_MAX_LOD, GL_TEXTURE_BASE_LEVEL, GL_TEXTURE_MAX_LEVEL, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T, GL_TEXTURE_WRAP_R, GL_TEXTURE_PRIORITY, GL_TEXTURE_COMPARE_MODE, GL_TEXTURE_COMPARE_FUNC, GL_DEPTH_TEXTURE_MODE, or GL_GENERATE_MIPMAP.

param Specifies the value of *pname*.

```
5.37.2.1100 static void OpenTK.Graphics.OpenGL.GL.TexSubImage1D
( OpenTK.Graphics.OpenGL.TextureTarget
  target, Int32 level, Int32 xoffset, Int32 width,
  OpenTK.Graphics.OpenGL.PixelFormat format,
  OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels )
[static]
```

Specify a one-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level *n* is the *n*th mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

width Specifies the width of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.37.2.1101 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage1D< T6 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6[] pixels) [static]`

Specify a one-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

width Specifies the width of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T6 : *struct*

5.37.2.1102 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage1D< T6 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 width, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T6 pixels[,]) [static]`

Specify a one-dimensional texture subimage.

Parameters

- target** Specifies the target texture. Must be GL_TEXTURE_1D.
- level** Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
- xoffset** Specifies a texel offset in the x direction within the texture array.
- width** Specifies the width of the texture subimage.
- format** Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.
- type** Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.
- data** Specifies a pointer to the image data in memory.

Type Constraints

T6 : *struct*

```
5.37.2.1103 static void OpenTK.Graphics.OpenGL.GL.TexSubImage1D<
    T6 > ( OpenTK.Graphics.OpenGL.TextureTarget
    target, Int32 level, Int32 xoffset, Int32 width,
    OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] T6 pixels[,]) [static]
```

Specify a one-dimensional texture subimage.

Parameters

- target** Specifies the target texture. Must be GL_TEXTURE_1D.
- level** Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.
- xoffset** Specifies a texel offset in the x direction within the texture array.
- width** Specifies the width of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T6 : struct

```
5.37.2.1104 static void OpenTK.Graphics.OpenGL.GL.TexSubImage1D<
    T6 > ( OpenTK.Graphics.OpenGL.TextureTarget
    target, Int32 level, Int32 xoffset, Int32 width,
    OpenTK.Graphics.OpenGL.PixelFormat format,
    OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
    OutAttribute] ref T6 pixels ) [static]
```

Specify a one-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_1D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

width Specifies the width of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_

FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T6 : struct

5.37.2.1105 static void OpenTK.Graphics.OpenGL.GL.TexSubImage2D
(OpenTK.Graphics.OpenGL.TextureTarget *target*, Int32
***level*, Int32 *xoffset*, Int32 *yoffset*, Int32 *width*, Int32**
***height*, OpenTK.Graphics.OpenGL.PixelFormat *format*,**
OpenTK.Graphics.OpenGL.PixelType *type*, IntPtr *pixels*)
[static]

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_

FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

5.37.2.1106 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage2D<T8> (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] ref T8 pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8,

GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.37.2.1107 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage2D<T8 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T8 pixels[,/]) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : *struct*

5.37.2.1108 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage2D< T8 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T8[] pixels) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct

5.37.2.1109 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage2D< T8 > (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level, Int32 xoffset, Int32 yoffset, Int32 width, Int32 height, OpenTK.Graphics.OpenGL.PixelFormat format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute, OutAttribute] T8 pixels[,]) [static]`

Specify a two-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_2D, GL_TEXTURE_CUBE_MAP_POSITIVE_X, GL_TEXTURE_CUBE_MAP_NEGATIVE_X, GL_TEXTURE_CUBE_MAP_POSITIVE_Y, GL_TEXTURE_CUBE_MAP_NEGATIVE_Y, GL_TEXTURE_CUBE_MAP_POSITIVE_Z, or GL_TEXTURE_CUBE_MAP_NEGATIVE_Z.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T8 : struct


```

5.37.2.1110 static void OpenTK.Graphics.OpenGL.GL.TexSubImage3D (
    OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
    Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32
    height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat
    format, OpenTK.Graphics.OpenGL.PixelType type, IntPtr pixels
    ) [static]

```

Specify a three-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

zoffset Specifies a texel offset in the z direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

```

5.37.2.1111 static void OpenTK.Graphics.OpenGL.GL.TexSubImage3D< T10
> ( OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32
height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] ref T10 pixels ) [static]

```

Specify a three-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

zoffset Specifies a texel offset in the z direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T10 : struct

5.37.2.1112 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage3D< T10
> (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32
height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T10 pixels[,/]) [static]`

Specify a three-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

zoffset Specifies a texel offset in the z direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T10 : *struct*

```
5.37.2.1113 static void OpenTK.Graphics.OpenGL.GL.TexSubImage3D< T10
> ( OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32
height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T10 pixels[,] ) [static]
```

Specify a three-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

zoffset Specifies a texel offset in the z direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T10 : struct

5.37.2.1114 `static void OpenTK.Graphics.OpenGL.GL.TexSubImage3D< T10
> (OpenTK.Graphics.OpenGL.TextureTarget target, Int32 level,
Int32 xoffset, Int32 yoffset, Int32 zoffset, Int32 width, Int32
height, Int32 depth, OpenTK.Graphics.OpenGL.PixelFormat
format, OpenTK.Graphics.OpenGL.PixelType type, [InAttribute,
OutAttribute] T10[] pixels) [static]`

Specify a three-dimensional texture subimage.

Parameters

target Specifies the target texture. Must be GL_TEXTURE_3D.

level Specifies the level-of-detail number. Level 0 is the base image level. Level n is the nth mipmap reduction image.

xoffset Specifies a texel offset in the x direction within the texture array.

yoffset Specifies a texel offset in the y direction within the texture array.

zoffset Specifies a texel offset in the z direction within the texture array.

width Specifies the width of the texture subimage.

height Specifies the height of the texture subimage.

depth Specifies the depth of the texture subimage.

format Specifies the format of the pixel data. The following symbolic values are accepted: GL_COLOR_INDEX, GL_RED, GL_GREEN, GL_BLUE, GL_ALPHA, GL_RGB, GL_BGR, GL_RGBA, GL_BGRA, GL_LUMINANCE, and GL_LUMINANCE_ALPHA.

type Specifies the data type of the pixel data. The following symbolic values are accepted: GL_UNSIGNED_BYTE, GL_BYTE, GL_BITMAP, GL_UNSIGNED_SHORT, GL_SHORT, GL_UNSIGNED_INT, GL_INT, GL_FLOAT, GL_UNSIGNED_BYTE_3_3_2, GL_UNSIGNED_BYTE_2_3_3_REV, GL_UNSIGNED_SHORT_5_6_5, GL_UNSIGNED_SHORT_5_6_5_REV, GL_UNSIGNED_SHORT_4_4_4_4, GL_UNSIGNED_SHORT_4_4_4_4_REV, GL_UNSIGNED_SHORT_5_5_5_1, GL_UNSIGNED_SHORT_1_5_5_5_REV, GL_UNSIGNED_INT_8_8_8_8, GL_UNSIGNED_INT_8_8_8_8_REV, GL_UNSIGNED_INT_10_10_10_2, and GL_UNSIGNED_INT_2_10_10_10_REV.

data Specifies a pointer to the image data in memory.

Type Constraints

T10 : *struct*

5.37.2.1115 `static void OpenTK.Graphics.OpenGL.GL.Translate (Double x, Double y, Double z) [static]`

Multiply the current matrix by a translation matrix.

Parameters

x Specify the x, y, and z coordinates of a translation vector.

5.37.2.1116 `static void OpenTK.Graphics.OpenGL.GL.Translate (Single x, Single y, Single z) [static]`

Multiply the current matrix by a translation matrix.

Parameters

x Specify the x, y, and z coordinates of a translation vector.

5.37.2.1117 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Single v0) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1118 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, ref Int32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1119 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, ref Single value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1120 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, ref UInt32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1121 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, Single[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1122 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, UInt32 v0) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1123 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 v0) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1124 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, Single * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1125 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, Int32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1126 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, UInt32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1127 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, UInt32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1128 `static void OpenTK.Graphics.OpenGL.GL.Uniform1 (Int32 location, Int32 count, Int32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1129 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Single v0, Single v1) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1130 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 v0, Int32 v1) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1131 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, ref Single value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1132 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, UInt32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1133 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, Int32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1134 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, Int32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1135 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, Single * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1136 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, Single[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1137 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, UInt32 v0, UInt32 v1) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1138 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, UInt32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1139 `static void OpenTK.Graphics.OpenGL.GL.Uniform2 (Int32 location, Int32 count, ref UInt32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1140 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, ref Single value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1141 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Single v0, Single v1, Single v2) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1142 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, Single * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1143 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, Int32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1144 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, UInt32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1145 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, UInt32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1146 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, Single[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1147 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32
location, Int32 count, Int32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1148 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32
location, Int32 v0, Int32 v1, Int32 v2) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1149 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32
location, UInt32 v0, UInt32 v1, UInt32 v2) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1150 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32
location, Int32 count, ref Int32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1151 `static void OpenTK.Graphics.OpenGL.GL.Uniform3 (Int32 location, Int32 count, ref UInt32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1152 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, Int32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1153 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, Single[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1154 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, Single * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1155 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 v0, Int32 v1, Int32 v2, Int32 v3) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1156 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, Int32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1157 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, UInt32 v0, UInt32 v1, UInt32 v2, UInt32 v3) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1158 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, UInt32[] value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1159 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, ref UInt32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1160 `static unsafe void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, UInt32 * value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1161 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, ref Single value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1162 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32 location, Int32 count, ref Int32 value) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1163 `static void OpenTK.Graphics.OpenGL.GL.Uniform4 (Int32
location, Single v0, Single v1, Single v2, Single v3) [static]`

Specify the value of a uniform variable for the current program object.

Parameters

location Specifies the location of the uniform variable to be modified.

v0 Specifies the new values to be used for the specified uniform variable.

5.37.2.1164 `static void OpenTK.Graphics.OpenGL.GL.UseProgram (UInt32
program) [static]`

Installs a program object as part of current rendering state.

Parameters

program Specifies the handle of the program object whose executables are to be used as part of current rendering state.

5.37.2.1165 `static void OpenTK.Graphics.OpenGL.GL.UseProgram (Int32
program) [static]`

Installs a program object as part of current rendering state.

Parameters

program Specifies the handle of the program object whose executables are to be used as part of current rendering state.

5.37.2.1166 `static void OpenTK.Graphics.OpenGL.GL.ValidateProgram (Int32
program) [static]`

Validates a program object.

Parameters

program Specifies the handle of the program object to be validated.

5.37.2.1167 `static void OpenTK.Graphics.OpenGL.GL.ValidateProgram (UInt32 program) [static]`

Validates a program object.

Parameters

program Specifies the handle of the program object to be validated.

5.37.2.1168 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex2 (Int32 * v) [static]`

Specify a vertex.

Parameters

x Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1169 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex2 (Single * v) [static]`

Specify a vertex.

Parameters

x Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1170 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Single[] v) [static]`

Specify a vertex.

Parameters

x Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1171 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Int32[] v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1172 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex2 (Int16
* v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1173 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Int32 x,
Int32 y) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1174 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Int16 x,
Int16 y) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1175 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (ref Int32 v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1176 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (ref Int16 v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1177 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Int16[] v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1178 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (ref Double v
) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1179 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex2 (Double * v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1180 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Single x, Single y) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1181 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Double[] v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1182 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (ref Single v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1183 `static void OpenTK.Graphics.OpenGL.GL.Vertex2 (Double x, Double y) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1184 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Single x, Single y, Single z) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1185 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex3 (Int16 * v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1186 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex3 (Single * v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1187 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex3 (Int32 * v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1188 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Double[] v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1189 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Single[] v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1190 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Int32[] v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1191 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Int16[] v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1192 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (ref Int16 v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1193 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (ref Single v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1194 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex3 (Double * v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1195 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (ref Double v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1196 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Double x, Double y, Double z) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1197 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Int16 x, Int16 y, Int16 z) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1198 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (Int32 x, Int32 y, Int32 z) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1199 `static void OpenTK.Graphics.OpenGL.GL.Vertex3 (ref Int32 v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1200 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Int16 x,
Int16 y, Int16 z, Int16 w) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1201 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (ref Single v)
[static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1202 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex4 (Single
* v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1203 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Int32[] v)`
`[static]`

Specify a vertex.

Parameters

- x* Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1204 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex4 (Int16 * v)`
`[static]`

Specify a vertex.

Parameters

- x* Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1205 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (ref Int16 v)`
`[static]`

Specify a vertex.

Parameters

- x* Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1206 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Double[] v)`
`[static]`

Specify a vertex.

Parameters

- x* Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1207 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex4 (Double * v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1208 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Single[] v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1209 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Single x, Single y, Single z, Single w) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1210 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Int32 x, Int32 y, Int32 z, Int32 w) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1211 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (ref Double v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1212 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Double x, Double y, Double z, Double w) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1213 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (ref Int32 v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1214 `static void OpenTK.Graphics.OpenGL.GL.Vertex4 (Int16[] v) [static]`

Specify a vertex.

Parameters

- x** Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1215 `static unsafe void OpenTK.Graphics.OpenGL.GL.Vertex4 (Int32 * v) [static]`

Specify a vertex.

Parameters

x Specify x, y, z, and w coordinates of a vertex. Not all parameters are present in all forms of the command.

5.37.2.1216 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (UInt32 index, Int16 x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1217 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (UInt32 index, Single x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1218 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (Int32 index, Double x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1219 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (UInt32 index, Double x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1220 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (UInt32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1221 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (Int32 index, Single x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1222 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (Int32 index, Int16 x) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1223 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (Int32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1224 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (UInt32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1225 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (Int32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1226 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (UInt32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1227 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib1 (Int32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1228 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1229 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, ref Int16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1230 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Int16 x, Int16 y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1231 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32
index, ref Single v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1232 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32
index, Int16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1233 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32
index, Double x, Double y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1234 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32
index, Single[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1235 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32
index, Double[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1236 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32
index, Double[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1237 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32
index, Int16 x, Int16 y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1238 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32
index, ref Int16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1239 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Single x, Single y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1240 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Int16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1241 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1242 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Single[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1243 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1244 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, ref Double v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1245 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, Double x, Double y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1246 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, ref Double v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1247 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1248 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1249 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, Single x, Single y) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1250 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (UInt32 index, ref Single v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1251 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib2 (Int32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1252 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32 index, Int16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1253 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32 index, Double x, Double y, Double z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1254 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1255 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, ref Int16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1256 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, Int16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1257 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1258 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1259 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32 index, Single[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1260 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1261 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32 index, ref Single v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1262 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1263 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, ref Single v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1264 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32
index, ref Int16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1265 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32
index, ref Double v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1266 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, ref Double v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1267 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32
index, Double[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1268 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32
index, Single x, Single y, Single z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1269 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, Int16 x, Int16 y, Int16 z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1270 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, Double[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1271 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32
index, Int16 x, Int16 y, Int16 z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1272 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, Double x, Double y, Double z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1273 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (Int32
index, Single x, Single y, Single z) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1274 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32
index, Single[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1275 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib3 (UInt32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1276 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, ref Int16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1277 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Byte * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1278 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Int32[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1279 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, ref Int32 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1280 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Int32 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1281 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, UInt16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1282 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, UInt32[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1283 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, ref Single v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1284 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, Double[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1285 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1286 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, SByte * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1287 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, ref SByte v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1288 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, UInt32 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1289 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, ref Double v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1290 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, UInt16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1291 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1292 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Single x, Single y, Single z, Single w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1293 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Single * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1294 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, ref Int32 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1295 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, Double x, Double y, Double z, Double w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1296 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, SByte[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1297 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, Int16 x, Int16 y, Int16 z, Int16 w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1298 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, Double[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1299 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, Int16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1300 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, Int32 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1301 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, Single x, Single y, Single z, Single w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1302 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, Single[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1303 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1304 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, ref Double v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1305 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Int32[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1306 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, ref Int16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1307 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, Single[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1308 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, ref Single v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1309 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, Int16 x, Int16 y, Int16 z, Int16 w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1310 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, ref Byte v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1311 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Int16[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1312 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, Int16 * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1313 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Byte * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1314 `static unsafe void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Double * v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1315 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, Double x, Double y, Double z, Double w) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1316 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, ref UInt16 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1317 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32
index, ref Byte v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1318 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32
index, Byte[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1319 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (UInt32 index, ref UInt32 v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1320 `static void OpenTK.Graphics.OpenGL.GL.VertexAttrib4 (Int32 index, Byte[] v) [static]`

Specifies the value of a generic vertex attribute.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

v0 Specifies the new values to be used for the specified vertex attribute.

5.37.2.1321 `static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer (UInt32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool normalized, Int32 stride, IntPtr pointer) [static]`

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

```

5.37.2.1322 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer
( Int32 index, Int32 size,
  OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool
  normalized, Int32 stride, IntPtr pointer ) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index* Specifies the index of the generic vertex attribute to be modified.
- size* Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized* Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.
- stride* Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer* Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

```

5.37.2.1323 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer<
T5 > ( UInt32 index, Int32 size,
  OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool
  normalized, Int32 stride, [InAttribute, OutAttribute] T5[] pointer
) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index* Specifies the index of the generic vertex attribute to be modified.
- size* Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type* Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```
5.37.2.1324 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer<
    T5 > ( Int32 index, Int32 size,
          OpenTK.Graphics.OpenGL.VertexAttribPointerType type,
          bool normalized, Int32 stride, [InAttribute, OutAttribute] T5
          pointer[, ] ) [static]
```

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```

5.37.2.1325 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer<
              T5 > ( UInt32 index, Int32 size,
                  OpenTK.Graphics.OpenGL.VertexAttribPointerType type,
                  bool normalized, Int32 stride, [InAttribute, OutAttribute] T5
                  pointer[, ] ) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index*** Specifies the index of the generic vertex attribute to be modified.
- size*** Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type*** Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.
- normalized*** Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.
- stride*** Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer*** Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : *struct*

```

5.37.2.1326 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer<
              T5 > ( UInt32 index, Int32 size,
                  OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool
                  normalized, Int32 stride, [InAttribute, OutAttribute] T5 pointer[, ]
                  ) [static]

```

Define an array of generic vertex attribute data.

Parameters

- index*** Specifies the index of the generic vertex attribute to be modified.
- size*** Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```
5.37.2.1327 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer<
    T5 > ( UInt32 index, Int32 size,
    OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool
    normalized, Int32 stride, [InAttribute, OutAttribute] ref T5
    pointer ) [static]
```

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints*T5 : struct*

5.37.2.1328 `static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer< T5 > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool normalized, Int32 stride, [InAttribute, OutAttribute] ref T5 pointer) [static]`

Define an array of generic vertex attribute data.

Parameters

index Specifies the index of the generic vertex attribute to be modified.

size Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.

type Specifies the data type of each component in the array. Symbolic constants GL_BYTE, GL_UNSIGNED_BYTE, GL_SHORT, GL_UNSIGNED_SHORT, GL_INT, GL_UNSIGNED_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

normalized Specifies whether fixed-point data values should be normalized (GL_TRUE) or converted directly as fixed-point values (GL_FALSE) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints*T5 : struct*

5.37.2.1329 `static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer< T5 > (Int32 index, Int32 size, OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool normalized, Int32 stride, [InAttribute, OutAttribute] T5 pointer[,]) [static]`

Define an array of generic vertex attribute data.

Parameters

- index*** Specifies the index of the generic vertex attribute to be modified.
- size*** Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type*** Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized*** Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.
- stride*** Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.
- pointer*** Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : struct

```
5.37.2.1330 static void OpenTK.Graphics.OpenGL.GL.VertexAttribPointer<
    T5 > ( Int32 index, Int32 size,
    OpenTK.Graphics.OpenGL.VertexAttribPointerType type, bool
    normalized, Int32 stride, [InAttribute, OutAttribute] T5[] pointer
    ) [static]
```

Define an array of generic vertex attribute data.

Parameters

- index*** Specifies the index of the generic vertex attribute to be modified.
- size*** Specifies the number of components per generic vertex attribute. Must be 1, 2, 3, or 4. The initial value is 4.
- type*** Specifies the data type of each component in the array. Symbolic constants `GL_BYTE`, `GL_UNSIGNED_BYTE`, `GL_SHORT`, `GL_UNSIGNED_SHORT`, `GL_INT`, `GL_UNSIGNED_INT`, `GL_FLOAT`, or `GL_DOUBLE` are accepted. The initial value is `GL_FLOAT`.
- normalized*** Specifies whether fixed-point data values should be normalized (`GL_TRUE`) or converted directly as fixed-point values (`GL_FALSE`) when they are accessed.

stride Specifies the byte offset between consecutive generic vertex attributes. If stride is 0, the generic vertex attributes are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first component of the first generic vertex attribute in the array. The initial value is 0.

Type Constraints

T5 : *struct*

5.37.2.1331 `static void OpenTK.Graphics.OpenGL.GL.VertexPointer (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, Int32 stride, IntPtr pointer) [static]`

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

5.37.2.1332 `static void OpenTK.Graphics.OpenGL.GL.VertexPointer< T3 > (Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[,]) [static]`

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1333 static void OpenTK.Graphics.OpenGL.GL.VertexPointer< T3 >
( Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType
type, Int32 stride, [InAttribute, OutAttribute] ref T3 pointer )
[static]
```

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1334 static void OpenTK.Graphics.OpenGL.GL.VertexPointer< T3 >
( Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType
type, Int32 stride, [InAttribute, OutAttribute] T3[] pointer )
[static]
```

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1335 static void OpenTK.Graphics.OpenGL.GL.VertexPointer< T3 >
              ( Int32 size, OpenTK.Graphics.OpenGL.VertexPointerType
                type, Int32 stride, [InAttribute, OutAttribute] T3 pointer[, ] )
              [static]
```

Define an array of vertex data.

Parameters

size Specifies the number of coordinates per vertex. Must be 2, 3, or 4. The initial value is 4.

type Specifies the data type of each coordinate in the array. Symbolic constants GL_SHORT, GL_INT, GL_FLOAT, or GL_DOUBLE are accepted. The initial value is GL_FLOAT.

stride Specifies the byte offset between consecutive vertices. If stride is 0, the vertices are understood to be tightly packed in the array. The initial value is 0.

pointer Specifies a pointer to the first coordinate of the first vertex in the array. The initial value is 0.

Type Constraints

T3 : struct

```
5.37.2.1336 static void OpenTK.Graphics.OpenGL.GL.Viewport ( Int32 x,
                    Int32 y, Int32 width, Int32 height ) [static]
```

Set the viewport.

Parameters

- x* Specify the lower left corner of the viewport rectangle, in pixels. The initial value is (0,0).
- width* Specify the width and height of the viewport. When a [GL](#) context is first attached to a window, width and height are set to the dimensions of that window.

5.37.2.1337 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Single * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

- x* Specify the , , coordinates for the raster position.

5.37.2.1338 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Single x, Single y) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

- x* Specify the , , coordinates for the raster position.

5.37.2.1339 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (ref Int16 v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

- x* Specify the , , coordinates for the raster position.

5.37.2.1340 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Int32 * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

- x* Specify the , , coordinates for the raster position.

5.37.2.1341 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Double[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1342 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (ref Double v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1343 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Single[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1344 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Int32 x, Int32 y) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1345 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (ref Int32 v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1346 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Int32[]
v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1347 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Int16 x,
Int16 y) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1348 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (ref
Single v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1349 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Double
x, Double y) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1350 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Int16 * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1351 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Double * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1352 `static void OpenTK.Graphics.OpenGL.GL.WindowPos2 (Int16[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1353 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Int16 x, Int16 y, Int16 z) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1354 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (ref Single v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1355 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Int16[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1356 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Single * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1357 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (ref Int32 v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1358 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Double x, Double y, Double z) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1359 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Double * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1360 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Int32 x, Int32 y, Int32 z) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1361 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (ref Int16 v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1362 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Single[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1363 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (ref Double v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1364 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Int32 * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1365 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Double[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the , , coordinates for the raster position.

5.37.2.1366 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Single x, Single y, Single z) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the *x*, *y*, *z* coordinates for the raster position.

5.37.2.1367 `static unsafe void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Int16 * v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the *x*, *y*, *z* coordinates for the raster position.

5.37.2.1368 `static void OpenTK.Graphics.OpenGL.GL.WindowPos3 (Int32[] v) [static]`

Specify the raster position in window coordinates for pixel operations.

Parameters

x Specify the *x*, *y*, *z* coordinates for the raster position.

5.37.3 Property Documentation

5.37.3.1 `override object OpenTK.Graphics.OpenGL.GL.SyncRoot [get, protected]`

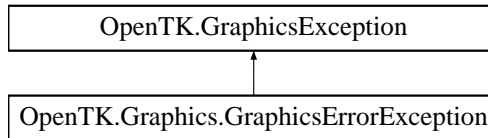
Returns a synchronization token unique for the [GL](#) class.

Reimplemented from [OpenTK.BindingsBase](#).

5.38 OpenTK.GraphicsException Class Reference

Represents errors related to [Graphics](#) operations.

Inheritance diagram for OpenTK.GraphicsException:



Public Member Functions

- [GraphicsException](#) ()
Constructs a new [GraphicsException](#).
- [GraphicsException](#) (string message)
Constructs a new [GraphicsException](#) with the specified excpetion message.

5.38.1 Detailed Description

Represents errors related to [Graphics](#) operations.

5.38.2 Constructor & Destructor Documentation

5.38.2.1 OpenTK.GraphicsException.GraphicsException ()

Constructs a new [GraphicsException](#).

5.38.2.2 OpenTK.GraphicsException.GraphicsException (string message)

Constructs a new [GraphicsException](#) with the specified excpetion message.

Parameters

message

5.39 OpenTK.Half Struct Reference

The name [Half](#) is derived from half-precision floating-point number. It occupies only 16 bits, which are split into 1 Sign bit, 5 Exponent bits and 10 Mantissa bits.

Public Member Functions

- [Half](#) (Single f)
The new [Half](#) instance will convert the parameter into 16-bit half-precision floating-point.
- [Half](#) (Single f, bool throwOnError)
The new [Half](#) instance will convert the parameter into 16-bit half-precision floating-point.
- [Half](#) (Double d)
The new [Half](#) instance will convert the parameter into 16-bit half-precision floating-point.
- [Half](#) (Double d, bool throwOnError)
The new [Half](#) instance will convert the parameter into 16-bit half-precision floating-point.
- Single [ToSingle](#) ()
Converts the 16-bit half to 32-bit floating-point.
- [Half](#) (SerializationInfo info, StreamingContext context)
*Constructor used by *ISerializable* to deserialize the object.*
- void [GetObjectData](#) (SerializationInfo info, StreamingContext context)
*Used by *ISerialize* to serialize the object.*
- void [FromBinaryStream](#) (BinaryReader bin)
Updates the [Half](#) by reading from a Stream.
- void [ToBinaryStream](#) (BinaryWriter bin)
Writes the [Half](#) into a Stream.
- bool [Equals](#) ([Half](#) other)
Returns a value indicating whether this instance is equal to a specified [OpenTK.Half](#) value.
- int [CompareTo](#) ([Half](#) other)
Compares this instance to a specified half-precision floating-point number and returns an integer that indicates whether the value of this instance is less than, equal to, or greater than the value of the specified half-precision floating-point number.
- override string [ToString](#) ()
Converts this [Half](#) into a human-legible string representation.

- string [ToString](#) (string format, IFormatProvider formatProvider)
Converts this [Half](#) into a human-legible string representation.

Static Public Member Functions

- static [operator Half](#) (float f)
Converts a System.Single to a [OpenTK.Half](#).
- static [operator Half](#) (double d)
Converts a System.Double to a [OpenTK.Half](#).
- static implicit [operator float](#) ([Half](#) h)
Converts a [OpenTK.Half](#) to a System.Single.
- static implicit [operator double](#) ([Half](#) h)
Converts a [OpenTK.Half](#) to a System.Double.
- static [Half Parse](#) (string s)
Converts the string representation of a number to a half-precision floating-point equivalent.
- static [Half Parse](#) (string s, System.Globalization.NumberStyles style, IFormatProvider provider)
Converts the string representation of a number to a half-precision floating-point equivalent.
- static bool [TryParse](#) (string s, out [Half](#) result)
Converts the string representation of a number to a half-precision floating-point equivalent. Returns success.
- static bool [TryParse](#) (string s, System.Globalization.NumberStyles style, IFormatProvider provider, out [Half](#) result)
Converts the string representation of a number to a half-precision floating-point equivalent. Returns success.
- static byte[] [GetBytes](#) ([Half](#) h)
Returns the [Half](#) as an array of bytes.
- static [Half FromBytes](#) (byte[] value, int startIndex)
Converts an array of bytes into [Half](#).

Public Attributes

- UInt16 **bits**
- const int **maxUlp** = 1

Static Public Attributes

- static readonly Int32 **SizeInBytes** = 2
The size in bytes for an instance of the [Half](#) struct.
- static readonly Single **MinValue** = 5.96046448e-08f
Smallest positive half.
- static readonly Single **MinNormalizedValue** = 6.10351562e-05f
Smallest positive normalized half.
- static readonly Single **MaxValue** = 65504.0f
Largest positive half.
- static readonly Single **Epsilon** = 0.00097656f
Smallest positive e for which $\text{half}(1.0 + e) \neq \text{half}(1.0)$.

Properties

- bool **IsZero** [get]
Returns true if the [Half](#) is zero.
- bool **IsNaN** [get]
Returns true if the [Half](#) represents Not A Number (NaN).
- bool **IsPositiveInfinity** [get]
Returns true if the [Half](#) represents positive infinity.
- bool **IsNegativeInfinity** [get]
Returns true if the [Half](#) represents negative infinity.

5.39.1 Detailed Description

The name `Half` is derived from half-precision floating-point number. It occupies only 16 bits, which are split into 1 Sign bit, 5 Exponent bits and 10 Mantissa bits. Quote from ARB_half_float_pixel specification: Any representable 16-bit floating-point value is legal as input to a GL command that accepts 16-bit floating-point data. The result of providing a value that is not a floating-point number (such as infinity or NaN) to such a command is unspecified, but must not lead to GL interruption or termination. Providing a denormalized number or negative zero to GL must yield predictable results.

5.39.2 Constructor & Destructor Documentation

5.39.2.1 `OpenTK.Half.Half (Single f)`

The new `Half` instance will convert the parameter into 16-bit half-precision floating-point.

Parameters

f 32-bit single-precision floating-point number.

5.39.2.2 `OpenTK.Half.Half (Single f, bool throwOnError)`

The new `Half` instance will convert the parameter into 16-bit half-precision floating-point.

Parameters

f 32-bit single-precision floating-point number.

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.39.2.3 `OpenTK.Half.Half (Double d)`

The new `Half` instance will convert the parameter into 16-bit half-precision floating-point.

Parameters

d 64-bit double-precision floating-point number.

5.39.2.4 OpenTK.Half.Half (Double *d*, bool *throwOnError*)

The new [Half](#) instance will convert the parameter into 16-bit half-precision floating-point.

Parameters

d 64-bit double-precision floating-point number.

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.39.2.5 OpenTK.Half.Half (SerializationInfo *info*, StreamingContext *context*)

Constructor used by ISerializable to deserialize the object.

Parameters

info

context

5.39.3 Member Function Documentation

5.39.3.1 int OpenTK.Half.CompareTo (Half *other*)

Compares this instance to a specified half-precision floating-point number and returns an integer that indicates whether the value of this instance is less than, equal to, or greater than the value of the specified half-precision floating-point number.

Parameters

other A half-precision floating-point number to compare.

Returns

A signed number indicating the relative values of this instance and value. If the number is:

Less than zero, then this instance is less than other, or this instance is not a number (OpenTK.Half.NaN) and other is a number.

Zero: this instance is equal to value, or both this instance and other are not a number (OpenTK.Half.NaN), OpenTK.Half.PositiveInfinity, or OpenTK.Half.NegativeInfinity.

Greater than zero: this instance is greater than others, or this instance is a number and other is not a number (OpenTK.Half.NaN).

5.39.3.2 bool OpenTK.Half.Equals (Half *other*)

Returns a value indicating whether this instance is equal to a specified [OpenTK.Half](#) value.

Parameters

other [OpenTK.Half](#) object to compare to this instance..

Returns

True, if other is equal to this instance; false otherwise.

5.39.3.3 void OpenTK.Half.FromBinaryStream (BinaryReader *bin*)

Updates the [Half](#) by reading from a Stream.

Parameters

bin A BinaryReader instance associated with an open Stream.

**5.39.3.4 static Half OpenTK.Half.FromBytes (byte[] *value*, int *startIndex*)
[static]**

Converts an array of bytes into [Half](#).

Parameters

value A [Half](#) in it's byte[] representation.

startIndex The starting position within value.

Returns

A new [Half](#) instance.

5.39.3.5 static byte [] OpenTK.Half.GetBytes (Half *h*) [static]

Returns the [Half](#) as an array of bytes.

Parameters

h The [Half](#) to convert.

Returns

The input as byte array.

5.39.3.6 void OpenTK.Half.GetObjectData (SerializationInfo *info*, StreamingContext *context*)

Used by ISerialize to serialize the object.

Parameters

info
context

5.39.3.7 static implicit OpenTK.Half.operator double (Half *h*) [static]

Converts a [OpenTK.Half](#) to a System.Double.

Parameters

h The value to convert. A [Half](#)

Returns

The result of the conversion. A System.Double

5.39.3.8 static implicit OpenTK.Half.operator float (Half *h*) [static]

Converts a [OpenTK.Half](#) to a System.Single.

Parameters

h The value to convert. A [Half](#)

Returns

The result of the conversion. A System.Single

5.39.3.9 static OpenTK.Half.operator Half (float *f*) [explicit, static]

Converts a System.Single to a [OpenTK.Half](#).

Parameters

f The value to convert. A System.Single

Returns

The result of the conversion. A [Half](#)

5.39.3.10 static OpenTK.Half.operator Half (double *d*) [explicit, static]

Converts a System.Double to a [OpenTK.Half](#).

Parameters

d The value to convert. A System.Double

Returns

The result of the conversion. A [Half](#)

5.39.3.11 static Half OpenTK.Half.Parse (string *s*) [static]

Converts the string representation of a number to a half-precision floating-point equivalent.

Parameters

s String representation of the number to convert.

Returns

A new [Half](#) instance.

5.39.3.12 static Half OpenTK.Half.Parse (string *s*, System.Globalization.NumberStyles *style*, IFormatProvider *provider*) [static]

Converts the string representation of a number to a half-precision floating-point equivalent.

Parameters

s String representation of the number to convert.

style Specifies the format of *s*.

provider Culture-specific formatting information.

Returns

A new [Half](#) instance.

5.39.3.13 void OpenTK.Half.ToBinaryStream (BinaryWriter *bin*)

Writes the [Half](#) into a Stream.

Parameters

bin A BinaryWriter instance associated with an open Stream.

5.39.3.14 Single OpenTK.Half.ToSingle ()

Converts the 16-bit half to 32-bit floating-point.

Returns

A single-precision floating-point number.

5.39.3.15 override string OpenTK.Half.ToString ()

Converts this [Half](#) into a human-legible string representation.

Returns

The string representation of this instance.

5.39.3.16 string OpenTK.Half.ToString (string *format*, IFormatProvider *formatProvider*)

Converts this [Half](#) into a human-legible string representation.

Parameters

format Formatting for the output string.

formatProvider Culture-specific formatting information.

Returns

The string representation of this instance.

5.39.3.17 static bool OpenTK.Half.TryParse (string *s*, out Half *result*) [static]

Converts the string representation of a number to a half-precision floating-point equivalent. Returns success.

Parameters

s String representation of the number to convert.

result The [Half](#) instance to write to.

Returns

Success.

5.39.3.18 static bool OpenTK.Half.TryParse (string *s*, System.Globalization.NumberStyles *style*, IFormatProvider *provider*, out Half *result*) [static]

Converts the string representation of a number to a half-precision floating-point equivalent. Returns success.

Parameters

s String representation of the number to convert.

style Specifies the format of *s*.

provider Culture-specific formatting information.

result The [Half](#) instance to write to.

Returns

Success.

5.39.4 Member Data Documentation

5.39.4.1 readonly Single OpenTK.Half.Epsilon = 0.00097656f [static]

Smallest positive *e* for which half (1.0 + *e*) != half (1.0).

5.39.4.2 readonly Single OpenTK.Half.MaxValue = 65504.0f [static]

Largest positive half.

**5.39.4.3 readonly Single OpenTK.Half.MinNormalizedValue = 6.10351562e-05f
[static]**

Smallest positive normalized half.

**5.39.4.4 readonly Single OpenTK.Half.MinValue = 5.96046448e-08f
[static]**

Smallest positive half.

5.39.4.5 readonly Int32 OpenTK.Half.SizeInBytes = 2 [static]

The size in bytes for an instance of the [Half](#) struct.

5.39.5 Property Documentation**5.39.5.1 bool OpenTK.Half.IsNaN [get]**

Returns true if the [Half](#) represents Not A Number (NaN).

5.39.5.2 bool OpenTK.Half.IsNegativeInfinity [get]

Returns true if the [Half](#) represents negative infinity.

5.39.5.3 bool OpenTK.Half.IsPositiveInfinity [get]

Returns true if the [Half](#) represents positive infinity.

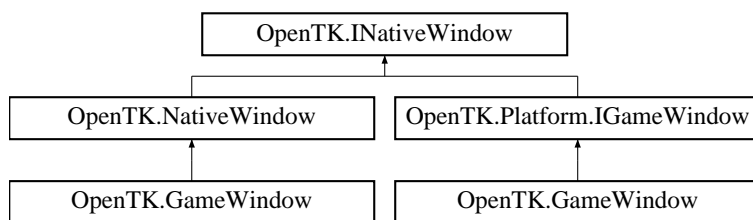
5.39.5.4 bool OpenTK.Half.IsZero [get]

Returns true if the [Half](#) is zero.

5.40 OpenTK.INativeWindow Interface Reference

Defines the interface for a native window.

Inheritance diagram for OpenTK.INativeWindow:



Public Member Functions

- void `Close ()`
Closes this window.
- void `ProcessEvents ()`
Processes pending window events.
- Point `PointToClient` (Point point)
Transforms the specified point from screen to client coordinates.
- Point `PointToScreen` (Point point)
Transforms the specified point from client to screen coordinates.

Properties

- Icon `Icon` [get, set]
Gets or sets the System.Drawing.Icon of the window.
- string `Title` [get, set]
Gets or sets the title of the window.
- bool `Focused` [get]
Gets a System.Boolean that indicates whether this window has input focus.
- bool `Visible` [get, set]
Gets or sets a System.Boolean that indicates whether the window is visible.
- bool `Exists` [get]
Gets a System.Boolean that indicates whether the window has been created and has not been destroyed.

- [IWindowInfo WindowInfo](#) [get]
Gets the [OpenTK.Platform.IWindowInfo](#) for this window.
- [WindowState WindowState](#) [get, set]
Gets or sets the [OpenTK.WindowState](#) for this window.
- [WindowBorder WindowBorder](#) [get, set]
Gets or sets the [OpenTK.WindowBorder](#) for this window.
- [Rectangle Bounds](#) [get, set]
Gets or sets a [System.Drawing.Rectangle](#) structure the contains the external bounds of this window, in screen coordinates. External bounds include the title bar, borders and drawing area of the window.
- [Point Location](#) [get, set]
Gets or sets a [System.Drawing.Point](#) structure that contains the location of this window on the desktop.
- [Size Size](#) [get, set]
Gets or sets a [System.Drawing.Size](#) structure that contains the external size of this window.
- [int X](#) [get, set]
Gets or sets the horizontal location of this window on the desktop.
- [int Y](#) [get, set]
Gets or sets the vertical location of this window on the desktop.
- [int Width](#) [get, set]
Gets or sets the external width of this window.
- [int Height](#) [get, set]
Gets or sets the external height of this window.
- [Rectangle ClientRectangle](#) [get, set]
Gets or sets a [System.Drawing.Rectangle](#) structure that contains the internal bounds of this window, in client coordinates. The internal bounds include the drawing area of the window, but exclude the titlebar and window borders.
- [Size ClientSize](#) [get, set]
Gets or sets a [System.Drawing.Size](#) structure that contains the internal size this window.
- [OpenTK.Input.IInputDriver InputDriver](#) [get]

This property is deprecated and should not be used.

Events

- EventHandler< EventArgs > [Move](#)
Occurs whenever the window is moved.
- EventHandler< EventArgs > [Resize](#)
Occurs whenever the window is resized.
- EventHandler< CancelEventArgs > [Closing](#)
Occurs when the window is about to close.
- EventHandler< EventArgs > [Closed](#)
Occurs after the window has closed.
- EventHandler< EventArgs > [Disposed](#)
Occurs when the window is disposed.
- EventHandler< EventArgs > [IconChanged](#)
Occurs when the [Icon](#) property of the window changes.
- EventHandler< EventArgs > [TitleChanged](#)
Occurs when the [Title](#) property of the window changes.
- EventHandler< EventArgs > [VisibleChanged](#)
Occurs when the [Visible](#) property of the window changes.
- EventHandler< EventArgs > [FocusedChanged](#)
Occurs when the [Focused](#) property of the window changes.
- EventHandler< EventArgs > [WindowBorderChanged](#)
Occurs when the [WindowBorder](#) property of the window changes.
- EventHandler< EventArgs > [WindowStateChanged](#)
Occurs when the [WindowState](#) property of the window changes.
- EventHandler< [KeyPressEventArgs](#) > [KeyPress](#)
Occurs whenever a character is typed.
- EventHandler< EventArgs > [MouseLeave](#)

Occurs whenever the mouse cursor leaves the window [Bounds](#).

- EventHandler< EventArgs > [MouseEnter](#)

Occurs whenever the mouse cursor enters the window [Bounds](#).

5.40.1 Detailed Description

Defines the interface for a native window.

5.40.2 Member Function Documentation

5.40.2.1 void OpenTK.INativeWindow.Close ()

Closes this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.2.2 Point OpenTK.INativeWindow.PointToClient (Point *point*)

Transforms the specified point from screen to client coordinates.

Parameters

point A System.Drawing.Point to transform.

Returns

The point transformed to client coordinates.

Implemented in [OpenTK.NativeWindow](#).

5.40.2.3 Point OpenTK.INativeWindow.PointToScreen (Point *point*)

Transforms the specified point from client to screen coordinates.

Parameters

point A System.Drawing.Point to transform.

Returns

The point transformed to screen coordinates.

Implemented in [OpenTK.NativeWindow](#).

5.40.2.4 void OpenTK.INativeWindow.ProcessEvents ()

Processes pending window events.

Implemented in [OpenTK.NativeWindow](#).

5.40.3 Property Documentation**5.40.3.1 Rectangle OpenTK.INativeWindow.Bounds [get, set]**

Gets or sets a System.Drawing.Rectangle structure the contains the external bounds of this window, in screen coordinates. External bounds include the title bar, borders and drawing area of the window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.2 Rectangle OpenTK.INativeWindow.ClientRectangle [get, set]

Gets or sets a System.Drawing.Rectangle structure that contains the internal bounds of this window, in client coordinates. The internal bounds include the drawing area of the window, but exclude the titlebar and window borders.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.3 Size OpenTK.INativeWindow.ClientSize [get, set]

Gets or sets a System.Drawing.Size structure that contains the internal size this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.4 bool OpenTK.INativeWindow.Exists [get]

Gets a System.Boolean that indicates whether the window has been created and has not been destroyed.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.5 bool OpenTK.INativeWindow.Focused [get]

Gets a System.Boolean that indicates whether this window has input focus.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.6 int OpenTK.INativeWindow.Height [get, set]

Gets or sets the external height of this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.7 Icon OpenTK.INativeWindow.Icon [get, set]

Gets or sets the System.Drawing.Icon of the window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.8 OpenTK.Input.IInputDriver OpenTK.INativeWindow.InputDriver [get]

This property is deprecated and should not be used.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.9 Point OpenTK.INativeWindow.Location [get, set]

Gets or sets a System.Drawing.Point structure that contains the location of this window on the desktop.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.10 Size OpenTK.INativeWindow.Size [get, set]

Gets or sets a System.Drawing.Size structure that contains the external size of this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.11 string OpenTK.INativeWindow.Title [get, set]

Gets or sets the title of the window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.12 bool OpenTK.INativeWindow.Visible [get, set]

Gets or sets a System.Boolean that indicates whether the window is visible.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.13 int OpenTK.INativeWindow.Width [get, set]

Gets or sets the external width of this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.14 WindowBorder OpenTK.INativeWindow.WindowBorder [get, set]

Gets or sets the OpenTK.WindowBorder for this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.15 IWindowInfo OpenTK.INativeWindow.WindowInfo [get]

Gets the [OpenTK.Platform.IWindowInfo](#) for this window.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.16 WindowState OpenTK.INativeWindow.WindowState [get, set]

Gets or sets the OpenTK.WindowState for this window.

Implemented in [OpenTK.GameWindow](#), and [OpenTK.NativeWindow](#).

5.40.3.17 int OpenTK.INativeWindow.X [get, set]

Gets or sets the horizontal location of this window on the desktop.

Implemented in [OpenTK.NativeWindow](#).

5.40.3.18 int OpenTK.INativeWindow.Y [get, set]

Gets or sets the vertical location of this window on the desktop.

Implemented in [OpenTK.NativeWindow](#).

5.40.4 Event Documentation**5.40.4.1 EventHandler<EventArgs> OpenTK.INativeWindow.Closed**

Occurs after the window has closed.

Implemented in [OpenTK.NativeWindow](#).

5.40.4.2 EventHandler<CancelEventArgs> OpenTK.INativeWindow.Closing

Occurs when the window is about to close.

Implemented in [OpenTK.NativeWindow](#).

5.40.4.3 EventHandler<EventArgs> OpenTK.INativeWindow.Disposed

Occurs when the window is disposed.

Implemented in [OpenTK.NativeWindow](#).

**5.40.4.4 EventHandler<EventArgs>
OpenTK.INativeWindow.FocusedChanged**

Occurs when the [Focused](#) property of the window changes.

Implemented in [OpenTK.NativeWindow](#).

5.40.4.5 EventHandler<EventArgs> OpenTK.INativeWindow.IconChanged

Occurs when the [Icon](#) property of the window changes.

Implemented in [OpenTK.NativeWindow](#).

**5.40.4.6 EventHandler<KeyPressEventArgs>
OpenTK.INativeWindow.KeyPress**

Occurs whenever a character is typed.

Implemented in [OpenTK.NativeWindow](#).

5.40.4.7 EventHandler<EventArgs> OpenTK.INativeWindow.MouseEnter

Occurs whenever the mouse cursor enters the window [Bounds](#).

Implemented in [OpenTK.NativeWindow](#).

5.40.4.8 EventHandler<EventArgs> OpenTK.INativeWindow.MouseLeave

Occurs whenever the mouse cursor leaves the window [Bounds](#).

Implemented in [OpenTK.NativeWindow](#).

5.40.4.9 EventHandler<EventArgs> OpenTK.INativeWindow.Move

Occurs whenever the window is moved.

Implemented in [OpenTK.NativeWindow](#).

5.40.4.10 EventHandler<EventArgs> OpenTK.INativeWindow.Resize

Occurs whenever the window is resized.

Implemented in [OpenTK.NativeWindow](#).

5.40.4.11 EventHandler<EventArgs> OpenTK.INativeWindow.TitleChanged

Occurs when the [Title](#) property of the window changes.

Implemented in [OpenTK.NativeWindow](#).

**5.40.4.12 EventHandler<EventArgs>
OpenTK.INativeWindow.VisibleChanged**

Occurs when the [Visible](#) property of the window changes.

Implemented in [OpenTK.NativeWindow](#).

**5.40.4.13 EventHandler<EventArgs>
OpenTK.INativeWindow.WindowBorderChanged**

Occurs when the [WindowBorder](#) property of the window changes.

Implemented in [OpenTK.NativeWindow](#).

**5.40.4.14 EventHandler<EventArgs>
OpenTK.INativeWindow.WindowStateChanged**

Occurs when the [WindowState](#) property of the window changes.

Implemented in [OpenTK.NativeWindow](#).

5.41 OpenTK.Input.GamePad Class Reference

Provides access to [GamePad](#) devices. Note: this API is not implemented yet.

5.41.1 Detailed Description

Provides access to [GamePad](#) devices. Note: this API is not implemented yet.

5.42 OpenTK.Input.GamePadState Struct Reference

Encapsulates the state of a [GamePad](#) device.

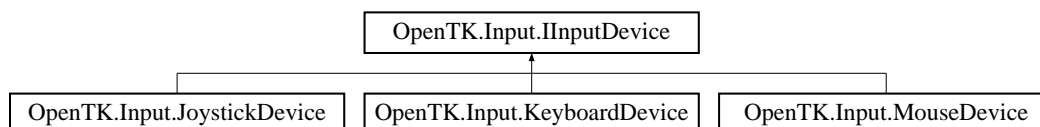
5.42.1 Detailed Description

Encapsulates the state of a [GamePad](#) device.

5.43 OpenTK.Input.IInputDevice Interface Reference

Defines a common interface for all input devices.

Inheritance diagram for OpenTK.Input.IInputDevice:



Properties

- string [Description](#) [get]
Gets a System.String with a unique description of this [IInputDevice](#) instance.
- [InputDeviceType DeviceType](#) [get]
Gets an [OpenTK.Input.InputDeviceType](#) value, representing the device type of this [IInputDevice](#) instance.

5.43.1 Detailed Description

Defines a common interface for all input devices.

5.43.2 Property Documentation

5.43.2.1 string OpenTK.Input.IInputDevice.Description [get]

Gets a System.String with a unique description of this [IInputDevice](#) instance.

Implemented in [OpenTK.Input.JoystickDevice](#), [OpenTK.Input.KeyboardDevice](#), and [OpenTK.Input.MouseDevice](#).

5.43.2.2 InputDeviceType OpenTK.Input.IInputDevice.DeviceType [get]

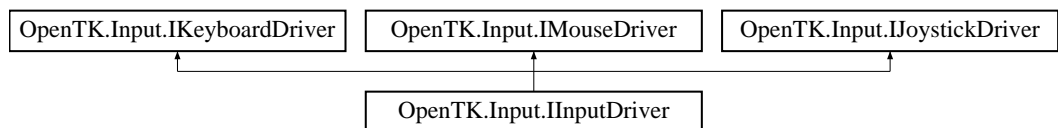
Gets an [OpenTK.Input.InputDeviceType](#) value, representing the device type of this [IInputDevice](#) instance.

Implemented in [OpenTK.Input.JoystickDevice](#), [OpenTK.Input.KeyboardDevice](#), and [OpenTK.Input.MouseDevice](#).

5.44 OpenTK.Input.IInputDriver Interface Reference

Defines the interface for an input driver.

Inheritance diagram for OpenTK.Input.IInputDriver:



Public Member Functions

- void [Poll](#) ()

Updates the state of the driver.

5.44.1 Detailed Description

Defines the interface for an input driver.

5.44.2 Member Function Documentation

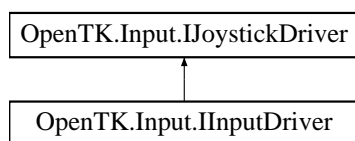
5.44.2.1 void OpenTK.Input.IInputDriver.Poll ()

Updates the state of the driver.

5.45 OpenTK.Input.IJoystickDriver Interface Reference

Defines the interface for [JoystickDevice](#) drivers.

Inheritance diagram for OpenTK.Input.IJoystickDriver:



Properties

- `IList< JoystickDevice > Joysticks` [get]

Gets the list of available JoystickDevices.

5.45.1 Detailed Description

Defines the interface for [JoystickDevice](#) drivers.

5.45.2 Property Documentation

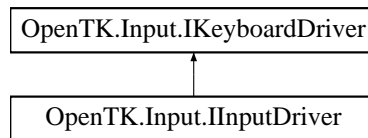
5.45.2.1 `IList<JoystickDevice> OpenTK.Input.IJoystickDriver.Joysticks` [get]

Gets the list of available JoystickDevices.

5.46 OpenTK.Input.IKeyboardDriver Interface Reference

Defines the interface for [KeyboardDevice](#) drivers.

Inheritance diagram for OpenTK.Input.IKeyboardDriver:



Properties

- `IList< KeyboardDevice > Keyboard` [get]

Gets the list of available KeyboardDevices.

5.46.1 Detailed Description

Defines the interface for [KeyboardDevice](#) drivers.

5.46.2 Property Documentation

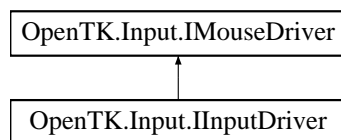
5.46.2.1 `IList<KeyboardDevice> OpenTK.Input.IKeyboardDriver.Keyboard` [get]

Gets the list of available KeyboardDevices.

5.47 OpenTK.Input.IMouseDriver Interface Reference

Defines the interface for [MouseDevice](#) drivers.

Inheritance diagram for OpenTK.Input.IMouseDriver:



Properties

- `ICollection< MouseDevice> Mouse` [get]
Gets the list of available MouseDevices.

5.47.1 Detailed Description

Defines the interface for [MouseDevice](#) drivers.

5.47.2 Property Documentation

5.47.2.1 `ICollection<MouseDevice> OpenTK.Input.IMouseDriver.Mouse` [get]

Gets the list of available MouseDevices.

5.48 OpenTK.Input.JoystickAxisCollection Class Reference

Defines a collection of JoystickAxes.

Properties

- `float this[int index]` [get, set]
Gets a System.Single indicating the absolute position of the JoystickAxis with the specified index.
- `int Count` [get]
Gets a System.Int32 indicating the available amount of JoystickAxes.

5.48.1 Detailed Description

Defines a collection of JoystickAxes.

5.48.2 Property Documentation

5.48.2.1 `int OpenTK.Input.JoystickAxisCollection.Count` `[get]`

Gets a System.Int32 indicating the available amount of JoystickAxes.

5.48.2.2 `float OpenTK::Input.JoystickAxisCollection::this` `[get, set]`

Gets a System.Single indicating the absolute position of the JoystickAxis with the specified index.

Gets a System.Single indicating the absolute position of the JoystickAxis.

Parameters

index The index of the JoystickAxis to check.

Returns

A System.Single in the range [-1, 1].

Parameters

axis The JoystickAxis to check.

Returns

A System.Single in the range [-1, 1].

5.49 OpenTK.Input.JoystickButtonCollection Class Reference

Defines a collection of JoystickButtons.

Properties

- `bool this [int index]` `[get, set]`

Gets a System.Boolean indicating whether the JoystickButton with the specified index is pressed.

- `int Count` [get]

Gets a System.Int32 indicating the available amount of JoystickButtons.

5.49.1 Detailed Description

Defines a collection of JoystickButtons.

5.49.2 Property Documentation

5.49.2.1 `int OpenTK.Input.JoystickButtonCollection.Count` [get]

Gets a System.Int32 indicating the available amount of JoystickButtons.

5.49.2.2 `bool OpenTK::Input.JoystickButtonCollection::this` [get, set]

Gets a System.Boolean indicating whether the JoystickButton with the specified index is pressed.

Gets a System.Boolean indicating whether the specified JoystickButton is pressed.

Parameters

index The index of the JoystickButton to check.

Returns

True if the JoystickButton is pressed; false otherwise.

Parameters

button The JoystickButton to check.

Returns

True if the JoystickButton is pressed; false otherwise.

5.50 OpenTK.Input.JoystickButtonEventArgs Class Reference

Provides data for the `JoystickDevice.ButtonDown` and `JoystickDevice.ButtonUp` events. This class is cached for performance reasons - avoid storing references outside the scope of the event.

Properties

- [JoystickButton Button](#) [get, set]
The index of the joystick button for the event.
- bool [Pressed](#) [get, set]
Gets a System.Boolean representing the state of the button for the event.

5.50.1 Detailed Description

Provides data for the [JoystickDevice.ButtonDown](#) and [JoystickDevice.ButtonUp](#) events. This class is cached for performance reasons - avoid storing references outside the scope of the event.

5.50.2 Property Documentation

5.50.2.1 JoystickButton OpenTK.Input.JoystickButtonEventArgs.Button [get, set]

The index of the joystick button for the event.

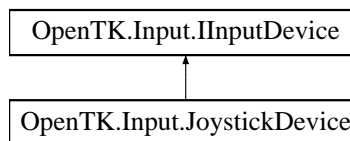
5.50.2.2 bool OpenTK.Input.JoystickButtonEventArgs.Pressed [get, set]

Gets a System.Boolean representing the state of the button for the event.

5.51 OpenTK.Input.JoystickDevice Class Reference

Represents a joystick device and provides methods to query its status.

Inheritance diagram for OpenTK.Input.JoystickDevice:



Public Attributes

- EventHandler< [JoystickMoveEventArgs](#) > [Move](#)

Occurs when an axis of this [JoystickDevice](#) instance is moved.

- EventHandler< [JoystickButtonEventArgs](#) > [ButtonDown](#)
Occurs when a button of this [JoystickDevice](#) instance is pressed.
- EventHandler< [JoystickButtonEventArgs](#) > [ButtonUp](#)
Occurs when a button of this [JoystickDevice](#) is released.

Properties

- [JoystickAxisCollection Axis](#) [get]
Gets a [JoystickAxisCollection](#) containing the state of each axis on this instance. Values are normalized in the [-1, 1] range.
- [JoystickButtonCollection Button](#) [get]
Gets [JoystickButtonCollection](#) containing the state of each button on this instance. True indicates that the button is pressed.
- string [Description](#) [get, set]
Gets a System.String containing a unique description for this instance.
- [InputDeviceType DeviceType](#) [get]
Gets a value indicating the InputDeviceType of this InputDevice.

5.51.1 Detailed Description

Represents a joystick device and provides methods to query its status.

5.51.2 Member Data Documentation

5.51.2.1 EventHandler<JoystickButtonEventArgs> OpenTK.Input.JoystickDevice.ButtonDown

Initial value:

```
delegate(object sender, JoystickButtonEventArgs e) { }
```

Occurs when a button of this [JoystickDevice](#) instance is pressed.

5.51.2.2 EventHandler<JoystickButtonEventArgs> OpenTK.Input.JoystickDevice.ButtonUp

Initial value:

```
delegate(object sender, JoystickButtonEventArgs e) { }
```

Occurs when a button of this [JoystickDevice](#) is released.

5.51.2.3 EventHandler<JoystickMoveEventArgs> OpenTK.Input.JoystickDevice.Move

Initial value:

```
delegate(object sender, JoystickMoveEventArgs e) { }
```

Occurs when an axis of this [JoystickDevice](#) instance is moved.

5.51.3 Property Documentation

5.51.3.1 JoystickAxisCollection OpenTK.Input.JoystickDevice.Axis [get]

Gets a [JoystickAxisCollection](#) containing the state of each axis on this instance. Values are normalized in the [-1, 1] range.

5.51.3.2 JoystickButtonCollection OpenTK.Input.JoystickDevice.Button [get]

Gets [JoystickButtonCollection](#) containing the state of each button on this instance. True indicates that the button is pressed.

5.51.3.3 string OpenTK.Input.JoystickDevice.Description [get, set]

Gets a System.String containing a unique description for this instance.

Implements [OpenTK.Input.IInputDevice](#).

5.51.3.4 InputDeviceType OpenTK.Input.JoystickDevice.DeviceType [get]

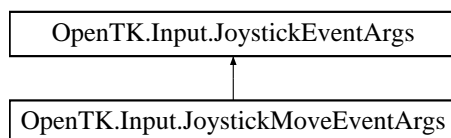
Gets a value indicating the InputDeviceType of this InputDevice.

Implements [OpenTK.Input.IInputDevice](#).

5.52 OpenTK.Input.JoystickEventArgs Class Reference

The base class for [JoystickDevice](#) event arguments.

Inheritance diagram for OpenTK.Input.JoystickEventArgs:



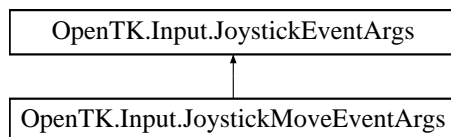
5.52.1 Detailed Description

The base class for [JoystickDevice](#) event arguments.

5.53 OpenTK.Input.JoystickMoveEventArgs Class Reference

Provides data for the [JoystickDevice.Move](#) event. This class is cached for performance reasons - avoid storing references outside the scope of the event.

Inheritance diagram for OpenTK.Input.JoystickMoveEventArgs:



Public Member Functions

- [JoystickMoveEventArgs](#) ([JoystickAxis](#) axis, float value, float delta)
Initializes a new instance of the [JoystickMoveEventArgs](#) class.

Properties

- [JoystickAxis](#) [Axis](#) [get, set]

Gets a System.Int32 representing the index of the axis that was moved.

- float [Value](#) [get, set]

Gets a System.Single representing the absolute position of the axis.

- float [Delta](#) [get, set]

Gets a System.Single representing the relative change in the position of the axis.

5.53.1 Detailed Description

Provides data for the [JoystickDevice.Move](#) event. This class is cached for performance reasons - avoid storing references outside the scope of the event.

5.53.2 Constructor & Destructor Documentation

5.53.2.1 [OpenTK.Input.JoystickMoveEventArgs.JoystickMoveEventArgs](#) ([JoystickAxis](#) *axis*, float *value*, float *delta*)

Initializes a new instance of the [JoystickMoveEventArgs](#) class.

Parameters

axis The index of the joystick axis that was moved.

value The absolute value of the joystick axis.

delta The relative change in value of the joystick axis.

5.53.3 Property Documentation

5.53.3.1 [JoystickAxis](#) [OpenTK.Input.JoystickMoveEventArgs.Axis](#) [get, set]

Gets a System.Int32 representing the index of the axis that was moved.

5.53.3.2 float [OpenTK.Input.JoystickMoveEventArgs.Delta](#) [get, set]

Gets a System.Single representing the relative change in the position of the axis.

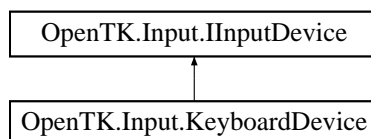
5.53.3.3 float [OpenTK.Input.JoystickMoveEventArgs.Value](#) [get, set]

Gets a System.Single representing the absolute position of the axis.

5.54 OpenTK.Input.KeyboardDevice Class Reference

Represents a keyboard device and provides methods to query its status.

Inheritance diagram for OpenTK.Input.KeyboardDevice:



Public Member Functions

- override int [GetHashCode](#) ()
Returns the hash code for this [KeyboardDevice](#).
- override string [ToString](#) ()
Returns a [System.String](#) representing this [KeyboardDevice](#).

Properties

- bool [this](#) [[Key](#) key] [get, set]
Gets a value indicating the status of the specified [Key](#).
- int [NumberOfKeys](#) [get, set]
Gets an integer representing the number of keys on this [KeyboardDevice](#).
- int [NumberOfFunctionKeys](#) [get, set]
Gets an integer representing the number of function keys (F-keys) on this [KeyboardDevice](#).
- int [NumberOfLeds](#) [get, set]
Gets a value indicating the number of led indicators on this [KeyboardDevice](#).
- IntPtr [DeviceID](#) [get, set]
Gets an [IntPtr](#) representing a device dependent ID.
- bool [KeyRepeat](#) [get, set]
Gets or sets a [System.Boolean](#) indicating key repeat status.

- string [Description](#) [get, set]
Gets a System.String which describes this instance.
- [InputDeviceType](#) DeviceType [get]
Gets the [InputDeviceType](#) for this instance.

Events

- EventHandler< [KeyboardKeyEventArgs](#) > [KeyDown](#)
Occurs when a key is pressed.
- EventHandler< [KeyboardKeyEventArgs](#) > [KeyUp](#)
Occurs when a key is released.

5.54.1 Detailed Description

Represents a keyboard device and provides methods to query its status.

5.54.2 Member Function Documentation

5.54.2.1 override int OpenTK.Input.KeyboardDevice.GetHashCode ()

Returns the hash code for this [KeyboardDevice](#).

Returns

A 32-bit signed integer hash code.

5.54.2.2 override string OpenTK.Input.KeyboardDevice.ToString ()

Returns a System.String representing this [KeyboardDevice](#).

Returns

A System.String representing this [KeyboardDevice](#).

5.54.3 Property Documentation

5.54.3.1 `string OpenTK.Input.KeyboardDevice.Description` `[get, set]`

Gets a System.String which describes this instance.

Implements [OpenTK.Input.IInputDevice](#).

5.54.3.2 `IntPtr OpenTK.Input.KeyboardDevice.DeviceID` `[get, set]`

Gets an IntPtr representing a device dependent ID.

5.54.3.3 `InputDeviceType OpenTK.Input.KeyboardDevice.DeviceType` `[get]`

Gets the [InputDeviceType](#) for this instance.

Implements [OpenTK.Input.IInputDevice](#).

5.54.3.4 `bool OpenTK.Input.KeyboardDevice.KeyRepeat` `[get, set]`

Gets or sets a System.Boolean indicating key repeat status.

If KeyRepeat is true, multiple KeyDown events will be generated while a key is being held. Otherwise only one KeyDown event will be reported.

The rate of the generated KeyDown events is controlled by the Operating System. Usually, one KeyDown event will be reported, followed by a small (250-1000ms) pause and several more KeyDown events (6-30 events per second).

Set to true to handle text input (where keyboard repeat is desirable), but set to false for game input.

5.54.3.5 `int OpenTK.Input.KeyboardDevice.NumberOfFunctionKeys` `[get, set]`

Gets an integer representing the number of function keys (F-keys) on this [KeyboardDevice](#).

5.54.3.6 `int OpenTK.Input.KeyboardDevice.NumberOfKeys` `[get, set]`

Gets an integer representing the number of keys on this [KeyboardDevice](#).

5.54.3.7 int OpenTK.Input.KeyboardDevice.NumberOfLeds [get, set]

Gets a value indicating the number of led indicators on this [KeyboardDevice](#).

5.54.3.8 bool OpenTK.Input.KeyboardDevice.this[Key key] [get, set]

Gets a value indicating the status of the specified Key.

Parameters

key The Key to check.

Returns

True if the Key is pressed, false otherwise.

5.54.4 Event Documentation**5.54.4.1 EventHandler<KeyboardKeyEventArgs>
OpenTK.Input.KeyboardDevice.KeyDown**

Occurs when a key is pressed.

**5.54.4.2 EventHandler<KeyboardKeyEventArgs>
OpenTK.Input.KeyboardDevice.KeyUp**

Occurs when a key is released.

5.55 OpenTK.Input.KeyboardKeyEventArgs Class Reference

Defines the event data for [KeyboardDevice](#) events.

Public Member Functions

- [KeyboardKeyEventArgs](#) ()
Constructs a new KeyboardEventArgs instance.
- [KeyboardKeyEventArgs](#) ([KeyboardKeyEventArgs](#) args)
Constructs a new KeyboardEventArgs instance.

Properties

- [Key](#) [Key](#) [get, set]
Gets the [Key](#) that generated this event.

5.55.1 Detailed Description

Defines the event data for [KeyboardDevice](#) events. Do not cache instances of this type outside their event handler. If necessary, you can clone a [KeyboardEventArgs](#) instance using the [KeyboardKeyEventArgs\(KeyboardKeyEventArgs\)](#) constructor.

5.55.2 Constructor & Destructor Documentation

5.55.2.1 OpenTK.Input.KeyboardKeyEventArgs.KeyboardKeyEventArgs ()

Constructs a new [KeyboardEventArgs](#) instance.

5.55.2.2 OpenTK.Input.KeyboardKeyEventArgs.KeyboardKeyEventArgs ([KeyboardKeyEventArgs](#) args)

Constructs a new [KeyboardEventArgs](#) instance.

Parameters

args An existing [KeyboardEventArgs](#) instance to clone.

5.55.3 Property Documentation

5.55.3.1 Key OpenTK.Input.KeyboardKeyEventArgs.Key [get, set]

Gets the [Key](#) that generated this event.

5.56 OpenTK.Input.KeyboardState Struct Reference

Encapsulates the state of a Keyboard device.

Public Types

- enum [BitValue](#) { [Zero](#) = 0, [One](#) = 1 }

Public Member Functions

- bool [IsKeyDown](#) ([Key](#) key)
Gets a System.Boolean indicating whether this key is down.
- bool [IsKeyUp](#) ([Key](#) key)
Gets a System.Boolean indicating whether this key is up.
- internal int **ReadBit** (int offset)
- internal void **WriteBit** (int offset, BitValue bit)
- bool [Equals](#) ([KeyboardState](#) other)
Compares two [KeyboardState](#) instances.

Public Attributes

- const int **NumKeys** = ((int)Key.LastKey + 16) / 32

5.56.1 Detailed Description

Encapsulates the state of a Keyboard device.

5.56.2 Member Function Documentation

5.56.2.1 bool OpenTK.Input.KeyboardState.Equals ([KeyboardState](#) *other*)

Compares two [KeyboardState](#) instances.

Parameters

other The instance to compare two.

Returns

True, if both instances are equal; false otherwise.

5.56.2.2 bool OpenTK.Input.KeyboardState.IsKeyDown ([Key](#) *key*)

Gets a System.Boolean indicating whether this key is down.

Parameters

key The [OpenTK.Input.Key](#) to check.

5.56.2.3 bool OpenTK.Input.KeyboardState.IsKeyUp (Key key)

Gets a System.Boolean indicating whether this key is up.

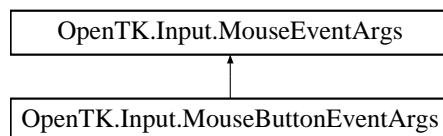
Parameters

key The [OpenTK.Input.Key](#) to check.

5.57 OpenTK.Input.MouseButtonEventArgs Class Reference

Defines the event data for [MouseDevice.ButtonDown](#) and [MouseDevice.ButtonUp](#) events.

Inheritance diagram for OpenTK.Input.MouseButtonEventArgs:



Public Member Functions

- [MouseButtonEventArgs](#) ()
Constructs a new [MouseButtonEventArgs](#) instance.
- [MouseButtonEventArgs](#) (int x, int y, [MouseButton](#) button, bool pressed)
Constructs a new [MouseButtonEventArgs](#) instance.
- [MouseButtonEventArgs](#) ([MouseButtonEventArgs](#) args)
Constructs a new [MouseButtonEventArgs](#) instance.

Properties

- [MouseButton](#) [Button](#) [get, set]
The mouse button for the event.
- bool [IsPressed](#) [get, set]
Gets a System.Boolean representing the state of the mouse button for the event.

5.57.1 Detailed Description

Defines the event data for [MouseDevice.ButtonDown](#) and [MouseDevice.ButtonUp](#) events. Do not cache instances of this type outside their event handler. If necessary, you can clone an instance using the [MouseButtonEventArgs\(MouseButtonEventArgs\)](#) constructor.

5.57.2 Constructor & Destructor Documentation

5.57.2.1 [OpenTK.Input.MouseButtonEventArgs.MouseButtonEventArgs \(\)](#)

Constructs a new [MouseButtonEventArgs](#) instance.

5.57.2.2 [OpenTK.Input.MouseButtonEventArgs.MouseButtonEventArgs \(int x, int y, MouseButton *button*, bool *pressed* \)](#)

Constructs a new [MouseButtonEventArgs](#) instance.

Parameters

- x* The X position.
- y* The Y position.
- button* The mouse button for the event.
- pressed* The current state of the button.

5.57.2.3 [OpenTK.Input.MouseButtonEventArgs.MouseButtonEventArgs \(MouseButtonEventArgs *args* \)](#)

Constructs a new [MouseButtonEventArgs](#) instance.

Parameters

- args* The [MouseButtonEventArgs](#) instance to clone.

5.57.3 Property Documentation

5.57.3.1 [MouseButton](#) [OpenTK.Input.MouseButtonEventArgs.Button](#) [get, set]

The mouse button for the event.

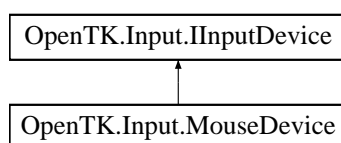
5.57.3.2 bool OpenTK.Input.MouseButtonEventArgs.IsPressed [get, set]

Gets a System.Boolean representing the state of the mouse button for the event.

5.58 OpenTK.Input.MouseDevice Class Reference

Represents a mouse device and provides methods to query its status.

Inheritance diagram for OpenTK.Input.MouseDevice:



Public Member Functions

- override int [GetHashCode](#) ()
Calculates the hash code for this instance.
- override string [ToString](#) ()
Returns a System.String that describes this instance.

Properties

- string [Description](#) [get, set]
Gets a string describing this [MouseDevice](#).
- [InputDeviceType](#) [DeviceType](#) [get]
Gets a value indicating the [InputDeviceType](#) of this [InputDevice](#).
- int [NumberOfButtons](#) [get, set]
Gets an integer representing the number of buttons on this [MouseDevice](#).
- int [NumberOfWheels](#) [get, set]
Gets an integer representing the number of wheels on this [MouseDevice](#).
- IntPtr [DeviceID](#) [get, set]
Gets an IntPtr representing a device dependent ID.

- int [Wheel](#) [get, set]

Gets the absolute wheel position in integer units. To support high-precision mice, it is recommended to use [WheelPrecise](#) instead.

- float [WheelPrecise](#) [get, set]

Gets the absolute wheel position in floating-point units.

- int [X](#) [get]

Gets an integer representing the absolute x position of the pointer, in window pixel coordinates.

- int [Y](#) [get]

Gets an integer representing the absolute y position of the pointer, in window pixel coordinates.

- bool [this](#) [[MouseButton](#) button] [get, set]

Gets a System.Boolean indicating the state of the specified MouseButton.

Events

- EventHandler< [MouseMoveEventArgs](#) > [Move](#) = delegate { }

Occurs when the mouse's position is moved.

- EventHandler< [MouseButtonEventArgs](#) > [ButtonDown](#) = delegate { }

Occurs when a button is pressed.

- EventHandler< [MouseButtonEventArgs](#) > [ButtonUp](#) = delegate { }

Occurs when a button is released.

- EventHandler< [MouseWheelEventArgs](#) > [WheelChanged](#) = delegate { }

Occurs when one of the mouse wheels is moved.

5.58.1 Detailed Description

Represents a mouse device and provides methods to query its status.

5.58.2 Member Function Documentation

5.58.2.1 override int OpenTK.Input.MouseDevice.GetHashCode ()

Calculates the hash code for this instance.

Returns

5.58.2.2 override string OpenTK.Input.MouseDevice.ToString ()

Returns a System.String that describes this instance.

Returns

A System.String that describes this instance.

5.58.3 Property Documentation

5.58.3.1 string OpenTK.Input.MouseDevice.Description [get, set]

Gets a string describing this [MouseDevice](#).

Implements [OpenTK.Input.IInputDevice](#).

5.58.3.2 IntPtr OpenTK.Input.MouseDevice.DeviceID [get, set]

Gets an IntPtr representing a device dependent ID.

5.58.3.3 InputDeviceType OpenTK.Input.MouseDevice.DeviceType [get]

Gets a value indicating the InputDeviceType of this InputDevice.

Implements [OpenTK.Input.IInputDevice](#).

5.58.3.4 int OpenTK.Input.MouseDevice.NumberOfButtons [get, set]

Gets an integer representing the number of buttons on this [MouseDevice](#).

5.58.3.5 int OpenTK.Input.MouseDevice.NumberOfWheels [get, set]

Gets an integer representing the number of wheels on this [MouseDevice](#).

5.58.3.6 `bool OpenTK.Input.MouseDevice.this[MouseButton button] [get, set]`

Gets a System.Boolean indicating the state of the specified MouseButton.

Parameters

button The MouseButton to check.

Returns

True if the MouseButton is pressed, false otherwise.

5.58.3.7 `int OpenTK.Input.MouseDevice.Wheel [get, set]`

Gets the absolute wheel position in integer units. To support high-precision mice, it is recommended to use [WheelPrecise](#) instead.

5.58.3.8 `float OpenTK.Input.MouseDevice.WheelPrecise [get, set]`

Gets the absolute wheel position in floating-point units.

5.58.3.9 `int OpenTK.Input.MouseDevice.X [get]`

Gets an integer representing the absolute x position of the pointer, in window pixel coordinates.

5.58.3.10 `int OpenTK.Input.MouseDevice.Y [get]`

Gets an integer representing the absolute y position of the pointer, in window pixel coordinates.

5.58.4 Event Documentation

5.58.4.1 `EventHandler<MouseButtonEventArgs> OpenTK.Input.MouseDevice.ButtonDown = delegate { }`

Occurs when a button is pressed.

5.58.4.2 `EventHandler<MouseButtonEventArgs> OpenTK.Input.MouseDevice.ButtonUp = delegate { }`

Occurs when a button is released.

5.58.4.3 EventHandler<MouseMoveEventArgs> OpenTK.Input.MouseDevice.Move = delegate { }

Occurs when the mouse's position is moved.

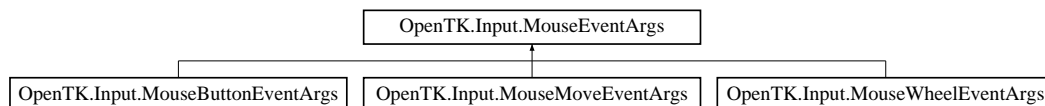
5.58.4.4 EventHandler<MouseWheelEventArgs> OpenTK.Input.MouseDevice.WheelChanged = delegate { }

Occurs when one of the mouse wheels is moved.

5.59 OpenTK.Input.MouseEventArgs Class Reference

Defines the event data for [MouseDevice](#) events.

Inheritance diagram for OpenTK.Input.MouseEventArgs:



Public Member Functions

- [MouseEventArgs](#) ()
Constructs a new instance.
- [MouseEventArgs](#) (int x, int y)
Constructs a new instance.
- [MouseEventArgs](#) ([MouseEventArgs](#) args)
Constructs a new instance.

Properties

- int [X](#) [get, set]
Gets the X position of the mouse for the event.
- int [Y](#) [get, set]
Gets the Y position of the mouse for the event.

- Point [Position](#) [get]

Gets a System.Drawing.Points representing the location of the mouse for the event.

5.59.1 Detailed Description

Defines the event data for [MouseDevice](#) events. Do not cache instances of this type outside their event handler. If necessary, you can clone an instance using the [MouseEventArgs\(MouseEventArgs\)](#) constructor.

5.59.2 Constructor & Destructor Documentation

5.59.2.1 OpenTK.Input.MouseEventArgs.MouseEventArgs ()

Constructs a new instance.

5.59.2.2 OpenTK.Input.MouseEventArgs.MouseEventArgs (int x, int y)

Constructs a new instance.

Parameters

- x* The X position.
- y* The Y position.

5.59.2.3 OpenTK.Input.MouseEventArgs.MouseEventArgs (MouseEventArgs args)

Constructs a new instance.

Parameters

- args* The [MouseEventArgs](#) instance to clone.

5.59.3 Property Documentation

5.59.3.1 Point OpenTK.Input.MouseEventArgs.Position [get]

Gets a System.Drawing.Points representing the location of the mouse for the event.

5.59.3.2 int OpenTK.Input.MouseEventArgs.X [get, set]

Gets the X position of the mouse for the event.

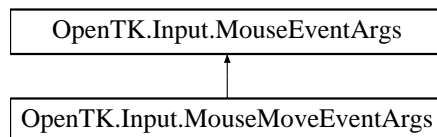
5.59.3.3 int OpenTK.Input.MouseEventArgs.Y [get, set]

Gets the Y position of the mouse for the event.

5.60 OpenTK.Input.MouseMoveEventArgs Class Reference

Defines the event data for [MouseDevice.Move](#) events.

Inheritance diagram for OpenTK.Input.MouseMoveEventArgs:



Public Member Functions

- [MouseMoveEventArgs](#) ()
Constructs a new [MouseMoveEventArgs](#) instance.
- [MouseMoveEventArgs](#) (int x, int y, int xDelta, int yDelta)
Constructs a new [MouseMoveEventArgs](#) instance.
- [MouseMoveEventArgs](#) ([MouseMoveEventArgs](#) args)
Constructs a new [MouseMoveEventArgs](#) instance.

Properties

- int [XDelta](#) [get, set]
Gets the change in X position produced by this event.
- int [YDelta](#) [get, set]
Gets the change in Y position produced by this event.

5.60.1 Detailed Description

Defines the event data for [MouseDevice.Move](#) events. Do not cache instances of this type outside their event handler. If necessary, you can clone an instance using the [MouseMoveEventArgs\(MouseMoveEventArgs\)](#) constructor.

5.60.2 Constructor & Destructor Documentation

5.60.2.1 `OpenTK.Input.MouseMoveEventArgs.MouseMoveEventArgs ()`

Constructs a new [MouseMoveEventArgs](#) instance.

5.60.2.2 `OpenTK.Input.MouseMoveEventArgs.MouseMoveEventArgs (int x, int y, int xDelta, int yDelta)`

Constructs a new [MouseMoveEventArgs](#) instance.

Parameters

- x* The X position.
- y* The Y position.
- xDelta* The change in X position produced by this event.
- yDelta* The change in Y position produced by this event.

5.60.2.3 `OpenTK.Input.MouseMoveEventArgs.MouseMoveEventArgs (MouseMoveEventArgs args)`

Constructs a new [MouseMoveEventArgs](#) instance.

Parameters

- args* The [MouseMoveEventArgs](#) instance to clone.

5.60.3 Property Documentation

5.60.3.1 `int OpenTK.Input.MouseMoveEventArgs.XDelta [get, set]`

Gets the change in X position produced by this event.

5.60.3.2 `int OpenTK.Input.MouseMoveEventArgs.YDelta [get, set]`

Gets the change in Y position produced by this event.

5.61 OpenTK.Input.MouseState Struct Reference

Encapsulates the state of a mouse device.

Public Member Functions

- internal **MouseState** ([MouseButton](#)[] buttons)
- bool [Equals](#) ([MouseState](#) other)

Compares two [MouseState](#) instances for equality.

5.61.1 Detailed Description

Encapsulates the state of a mouse device.

5.61.2 Member Function Documentation

5.61.2.1 bool OpenTK.Input.MouseState.Equals ([MouseState](#) other)

Compares two [MouseState](#) instances for equality.

Parameters

other The instance to compare to.

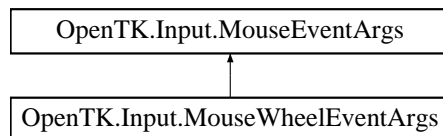
Returns

True, if both instances are equal; false otherwise.

5.62 OpenTK.Input.MouseWheelEventArgs Class Reference

Defines the event data for [MouseDevice.WheelChanged](#) events.

Inheritance diagram for OpenTK.Input.MouseWheelEventArgs:



Public Member Functions

- [MouseWheelEventArgs](#) ()
Constructs a new [MouseWheelEventArgs](#) instance.
- [MouseWheelEventArgs](#) (int x, int y, int value, int delta)
Constructs a new [MouseWheelEventArgs](#) instance.
- [MouseWheelEventArgs](#) ([MouseWheelEventArgs](#) args)
Constructs a new [MouseWheelEventArgs](#) instance.

Properties

- int [Value](#) [get]
Gets the value of the wheel in integer units. To support high-precision mice, it is recommended to use [ValuePrecise](#) instead.
- int [Delta](#) [get]
Gets the change in value of the wheel for this event in integer units. To support high-precision mice, it is recommended to use [DeltaPrecise](#) instead.
- float [ValuePrecise](#) [get, set]
Gets the precise value of the wheel in floating-point units.
- float [DeltaPrecise](#) [get, set]
Gets the precise change in value of the wheel for this event in floating-point units.

5.62.1 Detailed Description

Defines the event data for [MouseDevice.WheelChanged](#) events. Do not cache instances of this type outside their event handler. If necessary, you can clone an instance using the [MouseWheelEventArgs\(MouseWheelEventArgs\)](#) constructor.

5.62.2 Constructor & Destructor Documentation

5.62.2.1 OpenTK.Input.MouseWheelEventArgs.MouseWheelEventArgs ()

Constructs a new [MouseWheelEventArgs](#) instance.

5.62.2.2 OpenTK.Input.MouseWheelEventArgs.MouseWheelEventArgs (int *x*, int *y*, int *value*, int *delta*)

Constructs a new [MouseWheelEventArgs](#) instance.

Parameters

- x* The X position.
- y* The Y position.
- value* The value of the wheel.
- delta* The change in value of the wheel for this event.

5.62.2.3 OpenTK.Input.MouseWheelEventArgs.MouseWheelEventArgs (MouseWheelEventArgs *args*)

Constructs a new [MouseWheelEventArgs](#) instance.

Parameters

- args* The [MouseWheelEventArgs](#) instance to clone.

5.62.3 Property Documentation

5.62.3.1 int OpenTK.Input.MouseWheelEventArgs.Delta [get]

Gets the change in value of the wheel for this event in integer units. To support high-precision mice, it is recommended to use [DeltaPrecise](#) instead.

5.62.3.2 float OpenTK.Input.MouseWheelEventArgs.DeltaPrecise [get, set]

Gets the precise change in value of the wheel for this event in floating-point units.

5.62.3.3 int OpenTK.Input.MouseWheelEventArgs.Value [get]

Gets the value of the wheel in integer units. To support high-precision mice, it is recommended to use [ValuePrecise](#) instead.

5.62.3.4 float OpenTK.Input.MouseWheelEventArgs.ValuePrecise [get, set]

Gets the precise value of the wheel in floating-point units.

5.63 OpenTK.KeyPressEventArgs Class Reference

Defines the event arguments for KeyPress events. Instances of this class are cached: [KeyPressEventArgs](#) should only be used inside the relevant event, unless manually cloned.

Public Member Functions

- [KeyPressEventArgs](#) (char keyChar)

Constructs a new instance.

Properties

- char [KeyChar](#) [get, set]

Gets a System.Char that defines the ASCII character that was typed.

5.63.1 Detailed Description

Defines the event arguments for KeyPress events. Instances of this class are cached: [KeyPressEventArgs](#) should only be used inside the relevant event, unless manually cloned.

5.63.2 Constructor & Destructor Documentation

5.63.2.1 OpenTK.KeyPressEventArgs.KeyPressEventArgs (char keyChar)

Constructs a new instance.

Parameters

keyChar The ASCII character that was typed.

5.63.3 Property Documentation

5.63.3.1 char OpenTK.KeyPressEventArgs.KeyChar [get, set]

Gets a System.Char that defines the ASCII character that was typed.

5.64 OpenTK.Matrix4 Struct Reference

Represents a 4x4 Matrix.

Public Member Functions

- [Matrix4](#) ([Vector4](#) row0, [Vector4](#) row1, [Vector4](#) row2, [Vector4](#) row3)
Constructs a new instance.
- [Matrix4](#) (float m00, float m01, float m02, float m03, float m10, float m11, float m12, float m13, float m20, float m21, float m22, float m23, float m30, float m31, float m32, float m33)
Constructs a new instance.
- void [Invert](#) ()
Converts this instance into its inverse.
- void [Transpose](#) ()
Converts this instance into its transpose.
- override string [ToString](#) ()
Returns a System.String that represents the current Matrix44.
- override int [GetHashCode](#) ()
Returns the hashcode for this instance.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Matrix4](#) other)
Indicates whether the current matrix is equal to another matrix.

Static Public Member Functions

- static void [CreateFromAxisAngle](#) ([Vector3](#) axis, float angle, out [Matrix4](#) result)
Build a rotation matrix from the specified axis/angle rotation.
- static [Matrix4](#) [CreateFromAxisAngle](#) ([Vector3](#) axis, float angle)
Build a rotation matrix from the specified axis/angle rotation.

- static void [CreateRotationX](#) (float angle, out [Matrix4](#) result)
Builds a rotation matrix for a rotation around the x-axis.
- static [Matrix4](#) [CreateRotationX](#) (float angle)
Builds a rotation matrix for a rotation around the x-axis.
- static void [CreateRotationY](#) (float angle, out [Matrix4](#) result)
Builds a rotation matrix for a rotation around the y-axis.
- static [Matrix4](#) [CreateRotationY](#) (float angle)
Builds a rotation matrix for a rotation around the y-axis.
- static void [CreateRotationZ](#) (float angle, out [Matrix4](#) result)
Builds a rotation matrix for a rotation around the z-axis.
- static [Matrix4](#) [CreateRotationZ](#) (float angle)
Builds a rotation matrix for a rotation around the z-axis.
- static void [CreateTranslation](#) (float x, float y, float z, out [Matrix4](#) result)
Creates a translation matrix.
- static void [CreateTranslation](#) (ref [Vector3](#) vector, out [Matrix4](#) result)
Creates a translation matrix.
- static [Matrix4](#) [CreateTranslation](#) (float x, float y, float z)
Creates a translation matrix.
- static [Matrix4](#) [CreateTranslation](#) ([Vector3](#) vector)
Creates a translation matrix.
- static void [CreateOrthographic](#) (float width, float height, float zNear, float zFar, out [Matrix4](#) result)
Creates an orthographic projection matrix.
- static [Matrix4](#) [CreateOrthographic](#) (float width, float height, float zNear, float zFar)
Creates an orthographic projection matrix.
- static void [CreateOrthographicOffCenter](#) (float left, float right, float bottom, float top, float zNear, float zFar, out [Matrix4](#) result)
Creates an orthographic projection matrix.

- static [Matrix4 CreateOrthographicOffCenter](#) (float left, float right, float bottom, float top, float zNear, float zFar)
Creates an orthographic projection matrix.
- static void [CreatePerspectiveFieldOfView](#) (float fovy, float aspect, float zNear, float zFar, out [Matrix4](#) result)
Creates a perspective projection matrix.
- static [Matrix4 CreatePerspectiveFieldOfView](#) (float fovy, float aspect, float zNear, float zFar)
Creates a perspective projection matrix.
- static void [CreatePerspectiveOffCenter](#) (float left, float right, float bottom, float top, float zNear, float zFar, out [Matrix4](#) result)
Creates an perspective projection matrix.
- static [Matrix4 CreatePerspectiveOffCenter](#) (float left, float right, float bottom, float top, float zNear, float zFar)
Creates an perspective projection matrix.
- static [Matrix4 Translation](#) ([Vector3](#) trans)
Builds a translation matrix.
- static [Matrix4 Translation](#) (float x, float y, float z)
Build a translation matrix with the given translation.
- static [Matrix4 Scale](#) (float scale)
Build a scaling matrix.
- static [Matrix4 Scale](#) ([Vector3](#) scale)
Build a scaling matrix.
- static [Matrix4 Scale](#) (float x, float y, float z)
Build a scaling matrix.
- static [Matrix4 RotateX](#) (float angle)
Build a rotation matrix that rotates about the x-axis.
- static [Matrix4 RotateY](#) (float angle)
Build a rotation matrix that rotates about the y-axis.
- static [Matrix4 RotateZ](#) (float angle)
Build a rotation matrix that rotates about the z-axis.

- static [Matrix4 Rotate](#) ([Vector3](#) axis, float angle)
Build a rotation matrix to rotate about the given axis.
- static [Matrix4 Rotate](#) ([Quaternion](#) q)
Build a rotation matrix from a quaternion.
- static [Matrix4 LookAt](#) ([Vector3](#) eye, [Vector3](#) target, [Vector3](#) up)
Build a world space to camera space matrix.
- static [Matrix4 LookAt](#) (float eyeX, float eyeY, float eyeZ, float targetX, float targetY, float targetZ, float upX, float upY, float upZ)
Build a world space to camera space matrix.
- static [Matrix4 Frustum](#) (float left, float right, float bottom, float top, float near, float far)
Build a projection matrix.
- static [Matrix4 Perspective](#) (float fovy, float aspect, float near, float far)
Build a projection matrix.
- static [Matrix4 Mult](#) ([Matrix4](#) left, [Matrix4](#) right)
Multiplies two instances.
- static void [Mult](#) (ref [Matrix4](#) left, ref [Matrix4](#) right, out [Matrix4](#) result)
Multiplies two instances.
- static [Matrix4 Invert](#) ([Matrix4](#) mat)
Calculate the inverse of the given matrix.
- static [Matrix4 Transpose](#) ([Matrix4](#) mat)
Calculate the transpose of the given matrix.
- static void [Transpose](#) (ref [Matrix4](#) mat, out [Matrix4](#) result)
Calculate the transpose of the given matrix.
- static [Matrix4 operator*](#) ([Matrix4](#) left, [Matrix4](#) right)
Matrix multiplication.
- static bool [operator==](#) ([Matrix4](#) left, [Matrix4](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([Matrix4](#) left, [Matrix4](#) right)

Compares two instances for inequality.

Public Attributes

- [Vector4 Row0](#)
Top row of the matrix.
- [Vector4 Row1](#)
2nd row of the matrix
- [Vector4 Row2](#)
3rd row of the matrix
- [Vector4 Row3](#)
Bottom row of the matrix.

Static Public Attributes

- static [Matrix4 Identity](#) = new [Matrix4](#)([Vector4.UnitX](#), [Vector4.UnitY](#), [Vector4.UnitZ](#), [Vector4.UnitW](#))
The identity matrix.

Properties

- float [Determinant](#) [get]
The determinant of this matrix.
- [Vector4 Column0](#) [get]
The first column of this matrix.
- [Vector4 Column1](#) [get]
The second column of this matrix.
- [Vector4 Column2](#) [get]
The third column of this matrix.
- [Vector4 Column3](#) [get]
The fourth column of this matrix.

- float [M11](#) [get, set]
Gets or sets the value at row 1, column 1 of this instance.
- float [M12](#) [get, set]
Gets or sets the value at row 1, column 2 of this instance.
- float [M13](#) [get, set]
Gets or sets the value at row 1, column 3 of this instance.
- float [M14](#) [get, set]
Gets or sets the value at row 1, column 4 of this instance.
- float [M21](#) [get, set]
Gets or sets the value at row 2, column 1 of this instance.
- float [M22](#) [get, set]
Gets or sets the value at row 2, column 2 of this instance.
- float [M23](#) [get, set]
Gets or sets the value at row 2, column 3 of this instance.
- float [M24](#) [get, set]
Gets or sets the value at row 2, column 4 of this instance.
- float [M31](#) [get, set]
Gets or sets the value at row 3, column 1 of this instance.
- float [M32](#) [get, set]
Gets or sets the value at row 3, column 2 of this instance.
- float [M33](#) [get, set]
Gets or sets the value at row 3, column 3 of this instance.
- float [M34](#) [get, set]
Gets or sets the value at row 3, column 4 of this instance.
- float [M41](#) [get, set]
Gets or sets the value at row 4, column 1 of this instance.
- float [M42](#) [get, set]
Gets or sets the value at row 4, column 2 of this instance.

- float [M43](#) [get, set]
Gets or sets the value at row 4, column 3 of this instance.
- float [M44](#) [get, set]
Gets or sets the value at row 4, column 4 of this instance.

5.64.1 Detailed Description

Represents a 4x4 Matrix.

5.64.2 Constructor & Destructor Documentation

5.64.2.1 OpenTK.Matrix4.Matrix4 (Vector4 row0, Vector4 row1, Vector4 row2, Vector4 row3)

Constructs a new instance.

Parameters

- row0* Top row of the matrix
- row1* Second row of the matrix
- row2* Third row of the matrix
- row3* Bottom row of the matrix

5.64.2.2 OpenTK.Matrix4.Matrix4 (float m00, float m01, float m02, float m03, float m10, float m11, float m12, float m13, float m20, float m21, float m22, float m23, float m30, float m31, float m32, float m33)

Constructs a new instance.

Parameters

- m00* First item of the first row of the matrix.
- m01* Second item of the first row of the matrix.
- m02* Third item of the first row of the matrix.
- m03* Fourth item of the first row of the matrix.
- m10* First item of the second row of the matrix.
- m11* Second item of the second row of the matrix.

m12 Third item of the second row of the matrix.

m13 Fourth item of the second row of the matrix.

m20 First item of the third row of the matrix.

m21 Second item of the third row of the matrix.

m22 Third item of the third row of the matrix.

m23 First item of the third row of the matrix.

m30 Fourth item of the fourth row of the matrix.

m31 Second item of the fourth row of the matrix.

m32 Third item of the fourth row of the matrix.

m33 Fourth item of the fourth row of the matrix.

5.64.3 Member Function Documentation

5.64.3.1 `static void OpenTK.Matrix4.CreateFromAxisAngle (Vector3 axis, float angle, out Matrix4 result) [static]`

Build a rotation matrix from the specified axis/angle rotation.

Parameters

axis The axis to rotate about.

angle Angle in radians to rotate counter-clockwise (looking in the direction of the given axis).

result A matrix instance.

5.64.3.2 `static Matrix4 OpenTK.Matrix4.CreateFromAxisAngle (Vector3 axis, float angle) [static]`

Build a rotation matrix from the specified axis/angle rotation.

Parameters

axis The axis to rotate about.

angle Angle in radians to rotate counter-clockwise (looking in the direction of the given axis).

Returns

A matrix instance.

5.64.3.3 static Matrix4 OpenTK.Matrix4.CreateOrthographic (float *width*, float *height*, float *zNear*, float *zFar*) [static]

Creates an orthographic projection matrix.

Parameters

- width* The width of the projection volume.
- height* The height of the projection volume.
- zNear* The near edge of the projection volume.
- zFar* The far edge of the projection volume.

<rereturns>The resulting [Matrix4](#) instance.</rereturns>

5.64.3.4 static void OpenTK.Matrix4.CreateOrthographic (float *width*, float *height*, float *zNear*, float *zFar*, out Matrix4 *result*) [static]

Creates an orthographic projection matrix.

Parameters

- width* The width of the projection volume.
- height* The height of the projection volume.
- zNear* The near edge of the projection volume.
- zFar* The far edge of the projection volume.
- result* The resulting [Matrix4](#) instance.

5.64.3.5 static void OpenTK.Matrix4.CreateOrthographicOffCenter (float *left*, float *right*, float *bottom*, float *top*, float *zNear*, float *zFar*, out Matrix4 *result*) [static]

Creates an orthographic projection matrix.

Parameters

- left* The left edge of the projection volume.
- right* The right edge of the projection volume.
- bottom* The bottom edge of the projection volume.
- top* The top edge of the projection volume.
- zNear* The near edge of the projection volume.
- zFar* The far edge of the projection volume.
- result* The resulting [Matrix4](#) instance.

5.64.3.6 `static Matrix4 OpenTK.Matrix4.CreateOrthographicOffCenter (float left, float right, float bottom, float top, float zNear, float zFar)`
`[static]`

Creates an orthographic projection matrix.

Parameters

left The left edge of the projection volume.
right The right edge of the projection volume.
bottom The bottom edge of the projection volume.
top The top edge of the projection volume.
zNear The near edge of the projection volume.
zFar The far edge of the projection volume.

Returns

The resulting [Matrix4](#) instance.

5.64.3.7 `static void OpenTK.Matrix4.CreatePerspectiveFieldOfView (float fovy, float aspect, float zNear, float zFar, out Matrix4 result)`
`[static]`

Creates a perspective projection matrix.

Parameters

fovy Angle of the field of view in the y direction (in radians)
aspect Aspect ratio of the view (width / height)
zNear Distance to the near clip plane
zFar Distance to the far clip plane
result A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- *fovy* is zero, less than zero or larger than Math.PI
- *aspect* is negative or zero
- *zNear* is negative or zero
- *zFar* is negative or zero
- *zNear* is larger than *zFar*

5.64.3.8 static Matrix4 OpenTK.Matrix4.CreatePerspectiveFieldOfView (float *fovy*, float *aspect*, float *zNear*, float *zFar*) [static]

Creates a perspective projection matrix.

Parameters

fovy Angle of the field of view in the y direction (in radians)

aspect Aspect ratio of the view (width / height)

zNear Distance to the near clip plane

zFar Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- *fovy* is zero, less than zero or larger than Math.PI
- *aspect* is negative or zero
- *zNear* is negative or zero
- *zFar* is negative or zero
- *zNear* is larger than *zFar*

5.64.3.9 static void OpenTK.Matrix4.CreatePerspectiveOffCenter (float *left*, float *right*, float *bottom*, float *top*, float *zNear*, float *zFar*, out Matrix4 *result*) [static]

Creates an perspective projection matrix.

Parameters

left Left edge of the view frustum

right Right edge of the view frustum

bottom Bottom edge of the view frustum

top Top edge of the view frustum

zNear Distance to the near clip plane

zFar Distance to the far clip plane

result A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- *zNear* is negative or zero
- *zFar* is negative or zero
- *zNear* is larger than *zFar*

5.64.3.10 **static Matrix4 OpenTK.Matrix4.CreatePerspectiveOffCenter (float *left*, float *right*, float *bottom*, float *top*, float *zNear*, float *zFar*) [static]**

Creates an perspective projection matrix.

Parameters

left Left edge of the view frustum
right Right edge of the view frustum
bottom Bottom edge of the view frustum
top Top edge of the view frustum
zNear Distance to the near clip plane
zFar Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- *zNear* is negative or zero
- *zFar* is negative or zero
- *zNear* is larger than *zFar*

5.64.3.11 **static void OpenTK.Matrix4.CreateRotationX (float *angle*, out Matrix4 *result*) [static]**

Builds a rotation matrix for a rotation around the x-axis.

Parameters

angle The counter-clockwise angle in radians.
result The resulting [Matrix4](#) instance.

**5.64.3.12 static Matrix4 OpenTK.Matrix4.CreateRotationX (float *angle*)
[static]**

Builds a rotation matrix for a rotation around the x-axis.

Parameters

angle The counter-clockwise angle in radians.

Returns

The resulting [Matrix4](#) instance.

**5.64.3.13 static Matrix4 OpenTK.Matrix4.CreateRotationY (float *angle*)
[static]**

Builds a rotation matrix for a rotation around the y-axis.

Parameters

angle The counter-clockwise angle in radians.

Returns

The resulting [Matrix4](#) instance.

**5.64.3.14 static void OpenTK.Matrix4.CreateRotationY (float *angle*, out
Matrix4 *result*) [static]**

Builds a rotation matrix for a rotation around the y-axis.

Parameters

angle The counter-clockwise angle in radians.

result The resulting [Matrix4](#) instance.

**5.64.3.15 static void OpenTK.Matrix4.CreateRotationZ (float *angle*, out
Matrix4 *result*) [static]**

Builds a rotation matrix for a rotation around the z-axis.

Parameters

angle The counter-clockwise angle in radians.

result The resulting [Matrix4](#) instance.

**5.64.3.16 static Matrix4 OpenTK.Matrix4.CreateRotationZ (float *angle*)
[static]**

Builds a rotation matrix for a rotation around the z-axis.

Parameters

angle The counter-clockwise angle in radians.

Returns

The resulting [Matrix4](#) instance.

**5.64.3.17 static void OpenTK.Matrix4.CreateTranslation (float *x*, float *y*,
float *z*, out Matrix4 *result*) [static]**

Creates a translation matrix.

Parameters

x X translation.

y Y translation.

z Z translation.

result The resulting [Matrix4](#) instance.

**5.64.3.18 static void OpenTK.Matrix4.CreateTranslation (ref Vector3 *vector*,
out Matrix4 *result*) [static]**

Creates a translation matrix.

Parameters

vector The translation vector.

result The resulting [Matrix4](#) instance.

**5.64.3.19 static Matrix4 OpenTK.Matrix4.CreateTranslation (float *x*, float *y*,
float *z*) [static]**

Creates a translation matrix.

Parameters

x X translation.

y Y translation.

z Z translation.

Returns

The resulting [Matrix4](#) instance.

5.64.3.20 static Matrix4 OpenTK.Matrix4.CreateTranslation (Vector3 *vector*) [static]

Creates a translation matrix.

Parameters

vector The translation vector.

Returns

The resulting [Matrix4](#) instance.

5.64.3.21 override bool OpenTK.Matrix4.Equals (object *obj*)

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare result.

Returns

True if the instances are equal; false otherwise.

5.64.3.22 bool OpenTK.Matrix4.Equals (Matrix4 *other*)

Indicates whether the current matrix is equal to another matrix.

Parameters

other An matrix to compare with this matrix.

Returns

true if the current matrix is equal to the matrix parameter; otherwise, false.

5.64.3.23 static `Matrix4` `OpenTK.Matrix4.Frustum` (`float left`, `float right`, `float bottom`, `float top`, `float near`, `float far`) [`static`]

Build a projection matrix.

Parameters

left Left edge of the view frustum
right Right edge of the view frustum
bottom Bottom edge of the view frustum
top Top edge of the view frustum
near Distance to the near clip plane
far Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

5.64.3.24 override `int` `OpenTK.Matrix4.GetHashCode` ()

Returns the hashcode for this instance.

Returns

A `System.Int32` containing the unique hashcode for this instance.

5.64.3.25 static `Matrix4` `OpenTK.Matrix4.Invert` (`Matrix4 mat`) [`static`]

Calculate the inverse of the given matrix.

Parameters

mat The matrix to invert

Returns

The inverse of the given matrix if it has one, or the input if it is singular

Exceptions

InvalidOperationException Thrown if the [Matrix4](#) is singular.

5.64.3.26 void OpenTK.Matrix4.Invert ()

Converts this instance into its inverse.

5.64.3.27 static Matrix4 OpenTK.Matrix4.LookAt (Vector3 *eye*, Vector3 *target*, Vector3 *up*) [static]

Build a world space to camera space matrix.

Parameters

eye Eye (camera) position in world space

target Target position in world space

up Up vector in world space (should not be parallel to the camera direction, that is target - eye)

Returns

A [Matrix4](#) that transforms world space to camera space

5.64.3.28 static Matrix4 OpenTK.Matrix4.LookAt (float *eyeX*, float *eyeY*, float *eyeZ*, float *targetX*, float *targetY*, float *targetZ*, float *upX*, float *upY*, float *upZ*) [static]

Build a world space to camera space matrix.

Parameters

eyeX Eye (camera) position in world space

eyeY Eye (camera) position in world space

eyeZ Eye (camera) position in world space

targetX Target position in world space

targetY Target position in world space

targetZ Target position in world space

upX Up vector in world space (should not be parallel to the camera direction, that is target - eye)

upY Up vector in world space (should not be parallel to the camera direction, that is target - eye)

upZ Up vector in world space (should not be parallel to the camera direction, that is target - eye)

Returns

A [Matrix4](#) that transforms world space to camera space

5.64.3.29 `static Matrix4 OpenTK.Matrix4.Mult (Matrix4 left, Matrix4 right) [static]`

Multiplies two instances.

Parameters

left The left operand of the multiplication.

right The right operand of the multiplication.

Returns

A new instance that is the result of the multiplication

5.64.3.30 `static void OpenTK.Matrix4.Mult (ref Matrix4 left, ref Matrix4 right, out Matrix4 result) [static]`

Multiplies two instances.

Parameters

left The left operand of the multiplication.

right The right operand of the multiplication.

result A new instance that is the result of the multiplication

5.64.3.31 `static bool OpenTK.Matrix4.operator!= (Matrix4 left, Matrix4 right) [static]`

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.64.3.32 static Matrix4 OpenTK.Matrix4.operator* (Matrix4 *left*, Matrix4 *right*) [static]

Matrix multiplication.

Parameters

left left-hand operand

right right-hand operand

Returns

A new Matrix44 which holds the result of the multiplication

5.64.3.33 static bool OpenTK.Matrix4.operator== (Matrix4 *left*, Matrix4 *right*) [static]

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.64.3.34 static Matrix4 OpenTK.Matrix4.Perspective (float *fovy*, float *aspect*, float *near*, float *far*) [static]

Build a projection matrix.

Parameters

fovy Angle of the field of view in the y direction (in radians)

aspect Aspect ratio of the view (width / height)

near Distance to the near clip plane

far Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

5.64.3.35 static Matrix4 OpenTK.Matrix4.Rotate (Quaternion *q*) [static]

Build a rotation matrix from a quaternion.

Parameters

q the quaternion

Returns

A rotation matrix

5.64.3.36 static Matrix4 OpenTK.Matrix4.Rotate (Vector3 *axis*, float *angle*) [static]

Build a rotation matrix to rotate about the given axis.

Parameters

axis the axis to rotate about

angle angle in radians to rotate counter-clockwise (looking in the direction of the given axis)

Returns

A rotation matrix

5.64.3.37 static Matrix4 OpenTK.Matrix4.RotateX (float *angle*) [static]

Build a rotation matrix that rotates about the x-axis.

Parameters

angle angle in radians to rotate counter-clockwise around the x-axis

Returns

A rotation matrix

5.64.3.38 static Matrix4 OpenTK.Matrix4.RotateY (float *angle*) [static]

Build a rotation matrix that rotates about the y-axis.

Parameters

angle angle in radians to rotate counter-clockwise around the y-axis

Returns

A rotation matrix

5.64.3.39 static Matrix4 OpenTK.Matrix4.RotateZ (float *angle*) [static]

Build a rotation matrix that rotates about the z-axis.

Parameters

angle angle in radians to rotate counter-clockwise around the z-axis

Returns

A rotation matrix

5.64.3.40 static Matrix4 OpenTK.Matrix4.Scale (float *scale*) [static]

Build a scaling matrix.

Parameters

scale Single scale factor for x,y and z axes

Returns

A scaling matrix

5.64.3.41 static Matrix4 OpenTK.Matrix4.Scale (float *x*, float *y*, float *z*) [static]

Build a scaling matrix.

Parameters

x Scale factor for x-axis

y Scale factor for y-axis

z Scale factor for z-axis

Returns

A scaling matrix

5.64.3.42 static Matrix4 OpenTK.Matrix4.Scale (Vector3 *scale*) [static]

Build a scaling matrix.

Parameters

scale Scale factors for x,y and z axes

Returns

A scaling matrix

5.64.3.43 override string OpenTK.Matrix4.ToString ()

Returns a System.String that represents the current Matrix44.

Returns**5.64.3.44 static Matrix4 OpenTK.Matrix4.Translation (Vector3 *trans*) [static]**

Builds a translation matrix.

Parameters

trans The translation vector.

Returns

A new [Matrix4](#) instance.

5.64.3.45 static Matrix4 OpenTK.Matrix4.Translation (float *x*, float *y*, float *z*) [static]

Build a translation matrix with the given translation.

Parameters

x X translation

y Y translation

z Z translation

Returns

A Translation matrix

5.64.3.46 `static void OpenTK.Matrix4.Transpose (ref Matrix4 mat, out Matrix4 result) [static]`

Calculate the transpose of the given matrix.

Parameters

mat The matrix to transpose

result The result of the calculation

5.64.3.47 `static Matrix4 OpenTK.Matrix4.Transpose (Matrix4 mat) [static]`

Calculate the transpose of the given matrix.

Parameters

mat The matrix to transpose

Returns

The transpose of the given matrix

5.64.3.48 `void OpenTK.Matrix4.Transpose ()`

Converts this instance into its transpose.

5.64.4 Member Data Documentation

5.64.4.1 `Matrix4 OpenTK.Matrix4.Identity = new Matrix4(Vector4.UnitX, Vector4.UnitY, Vector4.UnitZ, Vector4.UnitW) [static]`

The identity matrix.

5.64.4.2 `Vector4 OpenTK.Matrix4.Row0`

Top row of the matrix.

5.64.4.3 `Vector4 OpenTK.Matrix4.Row1`

2nd row of the matrix

5.64.4.4 Vector4 OpenTK.Matrix4.Row2

3rd row of the matrix

5.64.4.5 Vector4 OpenTK.Matrix4.Row3

Bottom row of the matrix.

5.64.5 Property Documentation**5.64.5.1 Vector4 OpenTK.Matrix4.Column0 [get]**

The first column of this matrix.

5.64.5.2 Vector4 OpenTK.Matrix4.Column1 [get]

The second column of this matrix.

5.64.5.3 Vector4 OpenTK.Matrix4.Column2 [get]

The third column of this matrix.

5.64.5.4 Vector4 OpenTK.Matrix4.Column3 [get]

The fourth column of this matrix.

5.64.5.5 float OpenTK.Matrix4.Determinant [get]

The determinant of this matrix.

5.64.5.6 float OpenTK.Matrix4.M11 [get, set]

Gets or sets the value at row 1, column 1 of this instance.

5.64.5.7 float OpenTK.Matrix4.M12 [get, set]

Gets or sets the value at row 1, column 2 of this instance.

5.64.5.8 float OpenTK.Matrix4.M13 [get, set]

Gets or sets the value at row 1, column 3 of this instance.

5.64.5.9 float OpenTK.Matrix4.M14 [get, set]

Gets or sets the value at row 1, column 4 of this instance.

5.64.5.10 float OpenTK.Matrix4.M21 [get, set]

Gets or sets the value at row 2, column 1 of this instance.

5.64.5.11 float OpenTK.Matrix4.M22 [get, set]

Gets or sets the value at row 2, column 2 of this instance.

5.64.5.12 float OpenTK.Matrix4.M23 [get, set]

Gets or sets the value at row 2, column 3 of this instance.

5.64.5.13 float OpenTK.Matrix4.M24 [get, set]

Gets or sets the value at row 2, column 4 of this instance.

5.64.5.14 float OpenTK.Matrix4.M31 [get, set]

Gets or sets the value at row 3, column 1 of this instance.

5.64.5.15 float OpenTK.Matrix4.M32 [get, set]

Gets or sets the value at row 3, column 2 of this instance.

5.64.5.16 float OpenTK.Matrix4.M33 [get, set]

Gets or sets the value at row 3, column 3 of this instance.

5.64.5.17 float OpenTK.Matrix4.M34 [get, set]

Gets or sets the value at row 3, column 4 of this instance.

5.64.5.18 float OpenTK.Matrix4.M41 [get, set]

Gets or sets the value at row 4, column 1 of this instance.

5.64.5.19 float OpenTK.Matrix4.M42 [get, set]

Gets or sets the value at row 4, column 2 of this instance.

5.64.5.20 float OpenTK.Matrix4.M43 [get, set]

Gets or sets the value at row 4, column 3 of this instance.

5.64.5.21 float OpenTK.Matrix4.M44 [get, set]

Gets or sets the value at row 4, column 4 of this instance.

5.65 OpenTK.Matrix4d Struct Reference

Represents a 4x4 Matrix with double-precision components.

Public Member Functions

- [Matrix4d](#) ([Vector4d](#) row0, [Vector4d](#) row1, [Vector4d](#) row2, [Vector4d](#) row3)
Constructs a new instance.
- [Matrix4d](#) (double m00, double m01, double m02, double m03, double m10, double m11, double m12, double m13, double m20, double m21, double m22, double m23, double m30, double m31, double m32, double m33)
Constructs a new instance.
- void [Invert](#) ()
Converts this instance into its inverse.
- void [Transpose](#) ()
Converts this instance into its transpose.
- override string [ToString](#) ()
Returns a System.String that represents the current Matrix4d.
- override int [GetHashCode](#) ()

Returns the hashcode for this instance.

- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Matrix4d](#) other)
Indicates whether the current matrix is equal to another matrix.

Static Public Member Functions

- static void [CreateFromAxisAngle](#) ([Vector3d](#) axis, double angle, out [Matrix4d](#) result)
Build a rotation matrix from the specified axis/angle rotation.
- static [Matrix4d](#) [CreateFromAxisAngle](#) ([Vector3d](#) axis, double angle)
Build a rotation matrix from the specified axis/angle rotation.
- static void [CreateRotationX](#) (double angle, out [Matrix4d](#) result)
Builds a rotation matrix for a rotation around the x-axis.
- static [Matrix4d](#) [CreateRotationX](#) (double angle)
Builds a rotation matrix for a rotation around the x-axis.
- static void [CreateRotationY](#) (double angle, out [Matrix4d](#) result)
Builds a rotation matrix for a rotation around the y-axis.
- static [Matrix4d](#) [CreateRotationY](#) (double angle)
Builds a rotation matrix for a rotation around the y-axis.
- static void [CreateRotationZ](#) (double angle, out [Matrix4d](#) result)
Builds a rotation matrix for a rotation around the z-axis.
- static [Matrix4d](#) [CreateRotationZ](#) (double angle)
Builds a rotation matrix for a rotation around the z-axis.
- static void [CreateTranslation](#) (double x, double y, double z, out [Matrix4d](#) result)
Creates a translation matrix.
- static void [CreateTranslation](#) (ref [Vector3d](#) vector, out [Matrix4d](#) result)
Creates a translation matrix.

- static [Matrix4d CreateTranslation](#) (double x, double y, double z)
Creates a translation matrix.
- static [Matrix4d CreateTranslation](#) ([Vector3d](#) vector)
Creates a translation matrix.
- static void [CreateOrthographic](#) (double width, double height, double zNear, double zFar, out [Matrix4d](#) result)
Creates an orthographic projection matrix.
- static [Matrix4d CreateOrthographic](#) (double width, double height, double zNear, double zFar)
Creates an orthographic projection matrix.
- static void [CreateOrthographicOffCenter](#) (double left, double right, double bottom, double top, double zNear, double zFar, out [Matrix4d](#) result)
Creates an orthographic projection matrix.
- static [Matrix4d CreateOrthographicOffCenter](#) (double left, double right, double bottom, double top, double zNear, double zFar)
Creates an orthographic projection matrix.
- static void [CreatePerspectiveFieldOfView](#) (double fovy, double aspect, double zNear, double zFar, out [Matrix4d](#) result)
Creates a perspective projection matrix.
- static [Matrix4d CreatePerspectiveFieldOfView](#) (double fovy, double aspect, double zNear, double zFar)
Creates a perspective projection matrix.
- static void [CreatePerspectiveOffCenter](#) (double left, double right, double bottom, double top, double zNear, double zFar, out [Matrix4d](#) result)
Creates an perspective projection matrix.
- static [Matrix4d CreatePerspectiveOffCenter](#) (double left, double right, double bottom, double top, double zNear, double zFar)
Creates an perspective projection matrix.
- static [Matrix4d Translation](#) ([Vector3d](#) trans)
Build a translation matrix with the given translation.
- static [Matrix4d Translation](#) (double x, double y, double z)

Build a translation matrix with the given translation.

- static [Matrix4d Scale](#) (double scale)

Build a scaling matrix.

- static [Matrix4d Scale](#) ([Vector3d](#) scale)

Build a scaling matrix.

- static [Matrix4d Scale](#) (double x, double y, double z)

Build a scaling matrix.

- static [Matrix4d RotateX](#) (double angle)

Build a rotation matrix that rotates about the x-axis.

- static [Matrix4d RotateY](#) (double angle)

Build a rotation matrix that rotates about the y-axis.

- static [Matrix4d RotateZ](#) (double angle)

Build a rotation matrix that rotates about the z-axis.

- static [Matrix4d Rotate](#) ([Vector3d](#) axis, double angle)

Build a rotation matrix to rotate about the given axis.

- static [Matrix4d Rotate](#) ([Quaterniond](#) q)

Build a rotation matrix from a quaternion.

- static [Matrix4d LookAt](#) ([Vector3d](#) eye, [Vector3d](#) target, [Vector3d](#) up)

Build a world space to camera space matrix.

- static [Matrix4d LookAt](#) (double eyeX, double eyeY, double eyeZ, double targetX, double targetY, double targetZ, double upX, double upY, double upZ)

Build a world space to camera space matrix.

- static [Matrix4d Frustum](#) (double left, double right, double bottom, double top, double near, double far)

Build a projection matrix.

- static [Matrix4d Perspective](#) (double fovy, double aspect, double near, double far)

Build a projection matrix.

- static [Matrix4d Mult](#) ([Matrix4d](#) left, [Matrix4d](#) right)

Multiplies two instances.

- static void [Mult](#) (ref [Matrix4d](#) left, ref [Matrix4d](#) right, out [Matrix4d](#) result)
Multiplies two instances.
- static [Matrix4d Invert](#) ([Matrix4d](#) mat)
Calculate the inverse of the given matrix.
- static [Matrix4d Transpose](#) ([Matrix4d](#) mat)
Calculate the transpose of the given matrix.
- static void [Transpose](#) (ref [Matrix4d](#) mat, out [Matrix4d](#) result)
Calculate the transpose of the given matrix.
- static [Matrix4d operator*](#) ([Matrix4d](#) left, [Matrix4d](#) right)
Matrix multiplication.
- static bool [operator==](#) ([Matrix4d](#) left, [Matrix4d](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([Matrix4d](#) left, [Matrix4d](#) right)
Compares two instances for inequality.

Public Attributes

- [Vector4d Row0](#)
Top row of the matrix.
- [Vector4d Row1](#)
2nd row of the matrix
- [Vector4d Row2](#)
3rd row of the matrix
- [Vector4d Row3](#)
Bottom row of the matrix.

Static Public Attributes

- static [Matrix4d Identity](#) = new [Matrix4d](#)([Vector4d](#) .UnitX, [Vector4d](#) .UnitY, [Vector4d](#) .UnitZ, [Vector4d](#) .UnitW)

The identity matrix.

Properties

- double [Determinant](#) [get]
The determinant of this matrix.
- [Vector4d Column0](#) [get]
The first column of this matrix.
- [Vector4d Column1](#) [get]
The second column of this matrix.
- [Vector4d Column2](#) [get]
The third column of this matrix.
- [Vector4d Column3](#) [get]
The fourth column of this matrix.
- double [M11](#) [get, set]
Gets or sets the value at row 1, column 1 of this instance.
- double [M12](#) [get, set]
Gets or sets the value at row 1, column 2 of this instance.
- double [M13](#) [get, set]
Gets or sets the value at row 1, column 3 of this instance.
- double [M14](#) [get, set]
Gets or sets the value at row 1, column 4 of this instance.
- double [M21](#) [get, set]
Gets or sets the value at row 2, column 1 of this instance.
- double [M22](#) [get, set]
Gets or sets the value at row 2, column 2 of this instance.

- double [M23](#) [get, set]
Gets or sets the value at row 2, column 3 of this instance.
- double [M24](#) [get, set]
Gets or sets the value at row 2, column 4 of this instance.
- double [M31](#) [get, set]
Gets or sets the value at row 3, column 1 of this instance.
- double [M32](#) [get, set]
Gets or sets the value at row 3, column 2 of this instance.
- double [M33](#) [get, set]
Gets or sets the value at row 3, column 3 of this instance.
- double [M34](#) [get, set]
Gets or sets the value at row 3, column 4 of this instance.
- double [M41](#) [get, set]
Gets or sets the value at row 4, column 1 of this instance.
- double [M42](#) [get, set]
Gets or sets the value at row 4, column 2 of this instance.
- double [M43](#) [get, set]
Gets or sets the value at row 4, column 3 of this instance.
- double [M44](#) [get, set]
Gets or sets the value at row 4, column 4 of this instance.

5.65.1 Detailed Description

Represents a 4x4 Matrix with double-precision components.

5.65.2 Constructor & Destructor Documentation

5.65.2.1 `OpenTK.Matrix4d.Matrix4d (Vector4d row0, Vector4d row1, Vector4d row2, Vector4d row3)`

Constructs a new instance.

Parameters

row0 Top row of the matrix
row1 Second row of the matrix
row2 Third row of the matrix
row3 Bottom row of the matrix

5.65.2.2 OpenTK.Matrix4d.Matrix4d (double *m00*, double *m01*, double *m02*, double *m03*, double *m10*, double *m11*, double *m12*, double *m13*, double *m20*, double *m21*, double *m22*, double *m23*, double *m30*, double *m31*, double *m32*, double *m33*)

Constructs a new instance.

Parameters

m00 First item of the first row.
m01 Second item of the first row.
m02 Third item of the first row.
m03 Fourth item of the first row.
m10 First item of the second row.
m11 Second item of the second row.
m12 Third item of the second row.
m13 Fourth item of the second row.
m20 First item of the third row.
m21 Second item of the third row.
m22 Third item of the third row.
m23 Fourth item of the third row.
m30 First item of the fourth row.
m31 Second item of the fourth row.
m32 Third item of the fourth row.
m33 Fourth item of the fourth row.

5.65.3 Member Function Documentation

5.65.3.1 static void OpenTK.Matrix4d.CreateFromAxisAngle (Vector3d *axis*, double *angle*, out Matrix4d *result*) [static]

Build a rotation matrix from the specified axis/angle rotation.

Parameters

- axis* The axis to rotate about.
- angle* Angle in radians to rotate counter-clockwise (looking in the direction of the given axis).
- result* A matrix instance.

5.65.3.2 `static Matrix4d OpenTK.Matrix4d.CreateFromAxisAngle (Vector3d axis, double angle) [static]`

Build a rotation matrix from the specified axis/angle rotation.

Parameters

- axis* The axis to rotate about.
- angle* Angle in radians to rotate counter-clockwise (looking in the direction of the given axis).

Returns

A matrix instance.

5.65.3.3 `static Matrix4d OpenTK.Matrix4d.CreateOrthographic (double width, double height, double zNear, double zFar) [static]`

Creates an orthographic projection matrix.

Parameters

- width* The width of the projection volume.
- height* The height of the projection volume.
- zNear* The near edge of the projection volume.
- zFar* The far edge of the projection volume.

<rereturns>The resulting [Matrix4d](#) instance.</rereturns>

5.65.3.4 `static void OpenTK.Matrix4d.CreateOrthographic (double width, double height, double zNear, double zFar, out Matrix4d result) [static]`

Creates an orthographic projection matrix.

Parameters

width The width of the projection volume.
height The height of the projection volume.
zNear The near edge of the projection volume.
zFar The far edge of the projection volume.
result The resulting [Matrix4d](#) instance.

5.65.3.5 `static void OpenTK.Matrix4d.CreateOrthographicOffCenter (double left, double right, double bottom, double top, double zNear, double zFar, out Matrix4d result) [static]`

Creates an orthographic projection matrix.

Parameters

left The left edge of the projection volume.
right The right edge of the projection volume.
bottom The bottom edge of the projection volume.
top The top edge of the projection volume.
zNear The near edge of the projection volume.
zFar The far edge of the projection volume.
result The resulting [Matrix4d](#) instance.

5.65.3.6 `static Matrix4d OpenTK.Matrix4d.CreateOrthographicOffCenter (double left, double right, double bottom, double top, double zNear, double zFar) [static]`

Creates an orthographic projection matrix.

Parameters

left The left edge of the projection volume.
right The right edge of the projection volume.
bottom The bottom edge of the projection volume.
top The top edge of the projection volume.
zNear The near edge of the projection volume.
zFar The far edge of the projection volume.

Returns

The resulting [Matrix4d](#) instance.

5.65.3.7 static void **OpenTK.Matrix4d.CreatePerspectiveFieldOfView** (double *fovy*, double *aspect*, double *zNear*, double *zFar*, out Matrix4d *result*) [**static**]

Creates a perspective projection matrix.

Parameters

fovy Angle of the field of view in the y direction (in radians)
aspect Aspect ratio of the view (width / height)
zNear Distance to the near clip plane
zFar Distance to the far clip plane
result A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- *fovy* is zero, less than zero or larger than Math.PI
- *aspect* is negative or zero
- *zNear* is negative or zero
- *zFar* is negative or zero
- *zNear* is larger than *zFar*

5.65.3.8 static Matrix4d **OpenTK.Matrix4d.CreatePerspectiveFieldOfView** (double *fovy*, double *aspect*, double *zNear*, double *zFar*) [**static**]

Creates a perspective projection matrix.

Parameters

fovy Angle of the field of view in the y direction (in radians)
aspect Aspect ratio of the view (width / height)
zNear Distance to the near clip plane
zFar Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- fovy is zero, less than zero or larger than Math.PI
- aspect is negative or zero
- zNear is negative or zero
- zFar is negative or zero
- zNear is larger than zFar

5.65.3.9 `static void OpenTK.Matrix4d.CreatePerspectiveOffCenter (double left, double right, double bottom, double top, double zNear, double zFar, out Matrix4d result) [static]`

Creates an perspective projection matrix.

Parameters

left Left edge of the view frustum

right Right edge of the view frustum

bottom Bottom edge of the view frustum

top Top edge of the view frustum

zNear Distance to the near clip plane

zFar Distance to the far clip plane

result A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- zNear is negative or zero
- zFar is negative or zero
- zNear is larger than zFar

5.65.3.10 `static Matrix4d OpenTK.Matrix4d.CreatePerspectiveOffCenter (double left, double right, double bottom, double top, double zNear, double zFar) [static]`

Creates an perspective projection matrix.

Parameters

left Left edge of the view frustum

right Right edge of the view frustum

bottom Bottom edge of the view frustum

top Top edge of the view frustum

zNear Distance to the near clip plane

zFar Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

Exceptions

System.ArgumentOutOfRangeException Thrown under the following conditions:

- *zNear* is negative or zero
- *zFar* is negative or zero
- *zNear* is larger than *zFar*

5.65.3.11 static void OpenTK.Matrix4d.CreateRotationX (double *angle*, out Matrix4d *result*) [static]

Builds a rotation matrix for a rotation around the x-axis.

Parameters

angle The counter-clockwise angle in radians.

result The resulting [Matrix4](#) instance.

5.65.3.12 static Matrix4d OpenTK.Matrix4d.CreateRotationX (double *angle*) [static]

Builds a rotation matrix for a rotation around the x-axis.

Parameters

angle The counter-clockwise angle in radians.

Returns

The resulting [Matrix4](#) instance.

5.65.3.13 static Matrix4d OpenTK.Matrix4d.CreateRotationY (double *angle*) [static]

Builds a rotation matrix for a rotation around the y-axis.

Parameters

angle The counter-clockwise angle in radians.

Returns

The resulting [Matrix4](#) instance.

5.65.3.14 static void OpenTK.Matrix4d.CreateRotationY (double *angle*, out Matrix4d *result*) [static]

Builds a rotation matrix for a rotation around the y-axis.

Parameters

angle The counter-clockwise angle in radians.

result The resulting [Matrix4](#) instance.

5.65.3.15 static void OpenTK.Matrix4d.CreateRotationZ (double *angle*, out Matrix4d *result*) [static]

Builds a rotation matrix for a rotation around the z-axis.

Parameters

angle The counter-clockwise angle in radians.

result The resulting [Matrix4](#) instance.

5.65.3.16 static Matrix4d OpenTK.Matrix4d.CreateRotationZ (double *angle*) [static]

Builds a rotation matrix for a rotation around the z-axis.

Parameters

angle The counter-clockwise angle in radians.

Returns

The resulting [Matrix4](#) instance.

5.65.3.17 `static void OpenTK.Matrix4d.CreateTranslation (double x, double y, double z, out Matrix4d result) [static]`

Creates a translation matrix.

Parameters

x X translation.
y Y translation.
z Z translation.
result The resulting [Matrix4d](#) instance.

5.65.3.18 `static void OpenTK.Matrix4d.CreateTranslation (ref Vector3d vector, out Matrix4d result) [static]`

Creates a translation matrix.

Parameters

vector The translation vector.
result The resulting [Matrix4d](#) instance.

5.65.3.19 `static Matrix4d OpenTK.Matrix4d.CreateTranslation (double x, double y, double z) [static]`

Creates a translation matrix.

Parameters

x X translation.
y Y translation.
z Z translation.

Returns

The resulting [Matrix4d](#) instance.

5.65.3.20 `static Matrix4d OpenTK.Matrix4d.CreateTranslation (Vector3d vector) [static]`

Creates a translation matrix.

Parameters

vector The translation vector.

Returns

The resulting [Matrix4d](#) instance.

5.65.3.21 override bool OpenTK.Matrix4d.Equals (object *obj*)

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.65.3.22 bool OpenTK.Matrix4d.Equals (Matrix4d *other*)

Indicates whether the current matrix is equal to another matrix.

Parameters

other An matrix to compare with this matrix.

Returns

true if the current matrix is equal to the matrix parameter; otherwise, false.

**5.65.3.23 static Matrix4d OpenTK.Matrix4d.Frustum (double *left*, double *right*, double *bottom*, double *top*, double *near*, double *far*)
[static]**

Build a projection matrix.

Parameters

left Left edge of the view frustum

right Right edge of the view frustum

bottom Bottom edge of the view frustum

top Top edge of the view frustum

near Distance to the near clip plane

far Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

5.65.3.24 `override int OpenTK.Matrix4d.GetHashCode ()`

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.65.3.25 `static Matrix4d OpenTK.Matrix4d.Invert (Matrix4d mat)` `[static]`

Calculate the inverse of the given matrix.

Parameters

mat The matrix to invert

Returns

The inverse of the given matrix if it has one, or the input if it is singular

Exceptions

InvalidOperationException Thrown if the [Matrix4d](#) is singular.

5.65.3.26 `void OpenTK.Matrix4d.Invert ()`

Converts this instance into its inverse.

5.65.3.27 `static Matrix4d OpenTK.Matrix4d.LookAt (Vector3d eye, Vector3d target, Vector3d up)` `[static]`

Build a world space to camera space matrix.

Parameters

eye Eye (camera) position in world space

target Target position in world space

up Up vector in world space (should not be parallel to the camera direction, that is target - eye)

Returns

A Matrix that transforms world space to camera space

5.65.3.28 `static Matrix4d OpenTK.Matrix4d.LookAt (double eyeX, double eyeY, double eyeZ, double targetX, double targetY, double targetZ, double upX, double upY, double upZ) [static]`

Build a world space to camera space matrix.

Parameters

eyeX Eye (camera) position in world space

eyeY Eye (camera) position in world space

eyeZ Eye (camera) position in world space

targetX Target position in world space

targetY Target position in world space

targetZ Target position in world space

upX Up vector in world space (should not be parallel to the camera direction, that is target - eye)

upY Up vector in world space (should not be parallel to the camera direction, that is target - eye)

upZ Up vector in world space (should not be parallel to the camera direction, that is target - eye)

Returns

A [Matrix4](#) that transforms world space to camera space

5.65.3.29 `static Matrix4d OpenTK.Matrix4d.Mult (Matrix4d left, Matrix4d right) [static]`

Multiplies two instances.

Parameters

left The left operand of the multiplication.

right The right operand of the multiplication.

Returns

A new instance that is the result of the multiplication

5.65.3.30 `static void OpenTK.Matrix4d.Mult (ref Matrix4d left, ref Matrix4d right, out Matrix4d result) [static]`

Multiplies two instances.

Parameters

left The left operand of the multiplication.

right The right operand of the multiplication.

result A new instance that is the result of the multiplication

5.65.3.31 `static bool OpenTK.Matrix4d.operator!= (Matrix4d left, Matrix4d right) [static]`

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.65.3.32 `static Matrix4d OpenTK.Matrix4d.operator* (Matrix4d left, Matrix4d right) [static]`

Matrix multiplication.

Parameters

left left-hand operand

right right-hand operand

Returns

A new Matrix44 which holds the result of the multiplication

5.65.3.33 `static bool OpenTK.Matrix4d.operator==(Matrix4d left, Matrix4d right) [static]`

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.65.3.34 `static Matrix4d OpenTK.Matrix4d.Perspective (double fovy, double aspect, double near, double far) [static]`

Build a projection matrix.

Parameters

fovy Angle of the field of view in the y direction (in radians)

aspect Aspect ratio of the view (width / height)

near Distance to the near clip plane

far Distance to the far clip plane

Returns

A projection matrix that transforms camera space to raster space

5.65.3.35 `static Matrix4d OpenTK.Matrix4d.Rotate (Quaterniond q) [static]`

Build a rotation matrix from a quaternion.

Parameters

q the quaternion

Returns

A rotation matrix

5.65.3.36 static Matrix4d OpenTK.Matrix4d.Rotate (Vector3d *axis*, double *angle*) [static]

Build a rotation matrix to rotate about the given axis.

Parameters

axis the axis to rotate about

angle angle in radians to rotate counter-clockwise (looking in the direction of the given axis)

Returns

A rotation matrix

5.65.3.37 static Matrix4d OpenTK.Matrix4d.RotateX (double *angle*) [static]

Build a rotation matrix that rotates about the x-axis.

Parameters

angle angle in radians to rotate counter-clockwise around the x-axis

Returns

A rotation matrix

5.65.3.38 static Matrix4d OpenTK.Matrix4d.RotateY (double *angle*) [static]

Build a rotation matrix that rotates about the y-axis.

Parameters

angle angle in radians to rotate counter-clockwise around the y-axis

Returns

A rotation matrix

5.65.3.39 static Matrix4d OpenTK.Matrix4d.RotateZ (double *angle*) [static]

Build a rotation matrix that rotates about the z-axis.

Parameters

angle angle in radians to rotate counter-clockwise around the z-axis

Returns

A rotation matrix

5.65.3.40 static Matrix4d OpenTK.Matrix4d.Scale (double *scale*) [static]

Build a scaling matrix.

Parameters

scale Single scale factor for x,y and z axes

Returns

A scaling matrix

5.65.3.41 static Matrix4d OpenTK.Matrix4d.Scale (double *x*, double *y*, double *z*) [static]

Build a scaling matrix.

Parameters

x Scale factor for x-axis

y Scale factor for y-axis

z Scale factor for z-axis

Returns

A scaling matrix

**5.65.3.42 static Matrix4d OpenTK.Matrix4d.Scale (Vector3d *scale*)
[static]**

Build a scaling matrix.

Parameters

scale Scale factors for x,y and z axes

Returns

A scaling matrix

5.65.3.43 override string OpenTK.Matrix4d.ToString ()

Returns a System.String that represents the current Matrix44.

Returns**5.65.3.44 static Matrix4d OpenTK.Matrix4d.Translation (Vector3d *trans*)
[static]**

Build a translation matrix with the given translation.

Parameters

trans The vector to translate along

Returns

A Translation matrix

**5.65.3.45 static Matrix4d OpenTK.Matrix4d.Translation (double *x*, double
y, double *z*) [static]**

Build a translation matrix with the given translation.

Parameters

x X translation

y Y translation

z Z translation

Returns

A Translation matrix

5.65.3.46 `static void OpenTK.Matrix4d.Transpose (ref Matrix4d mat, out Matrix4d result) [static]`

Calculate the transpose of the given matrix.

Parameters

mat The matrix to transpose

result The result of the calculation

5.65.3.47 `static Matrix4d OpenTK.Matrix4d.Transpose (Matrix4d mat) [static]`

Calculate the transpose of the given matrix.

Parameters

mat The matrix to transpose

Returns

The transpose of the given matrix

5.65.3.48 `void OpenTK.Matrix4d.Transpose ()`

Converts this instance into its transpose.

5.65.4 Member Data Documentation

5.65.4.1 `Matrix4d OpenTK.Matrix4d.Identity = new Matrix4d(Vector4d .UnitX, Vector4d .UnitY, Vector4d .UnitZ, Vector4d .UnitW) [static]`

The identity matrix.

5.65.4.2 Vector4d OpenTK.Matrix4d.Row0

Top row of the matrix.

5.65.4.3 Vector4d OpenTK.Matrix4d.Row1

2nd row of the matrix

5.65.4.4 Vector4d OpenTK.Matrix4d.Row2

3rd row of the matrix

5.65.4.5 Vector4d OpenTK.Matrix4d.Row3

Bottom row of the matrix.

5.65.5 Property Documentation**5.65.5.1 Vector4d OpenTK.Matrix4d.Column0 [get]**

The first column of this matrix.

5.65.5.2 Vector4d OpenTK.Matrix4d.Column1 [get]

The second column of this matrix.

5.65.5.3 Vector4d OpenTK.Matrix4d.Column2 [get]

The third column of this matrix.

5.65.5.4 Vector4d OpenTK.Matrix4d.Column3 [get]

The fourth column of this matrix.

5.65.5.5 double OpenTK.Matrix4d.Determinant [get]

The determinant of this matrix.

5.65.5.6 double OpenTK.Matrix4d.M11 [get, set]

Gets or sets the value at row 1, column 1 of this instance.

5.65.5.7 double OpenTK.Matrix4d.M12 [get, set]

Gets or sets the value at row 1, column 2 of this instance.

5.65.5.8 double OpenTK.Matrix4d.M13 [get, set]

Gets or sets the value at row 1, column 3 of this instance.

5.65.5.9 double OpenTK.Matrix4d.M14 [get, set]

Gets or sets the value at row 1, column 4 of this instance.

5.65.5.10 double OpenTK.Matrix4d.M21 [get, set]

Gets or sets the value at row 2, column 1 of this instance.

5.65.5.11 double OpenTK.Matrix4d.M22 [get, set]

Gets or sets the value at row 2, column 2 of this instance.

5.65.5.12 double OpenTK.Matrix4d.M23 [get, set]

Gets or sets the value at row 2, column 3 of this instance.

5.65.5.13 double OpenTK.Matrix4d.M24 [get, set]

Gets or sets the value at row 2, column 4 of this instance.

5.65.5.14 double OpenTK.Matrix4d.M31 [get, set]

Gets or sets the value at row 3, column 1 of this instance.

5.65.5.15 double OpenTK.Matrix4d.M32 [get, set]

Gets or sets the value at row 3, column 2 of this instance.

5.65.5.16 double OpenTK.Matrix4d.M33 [get, set]

Gets or sets the value at row 3, column 3 of this instance.

5.65.5.17 double OpenTK.Matrix4d.M34 [get, set]

Gets or sets the value at row 3, column 4 of this instance.

5.65.5.18 double OpenTK.Matrix4d.M41 [get, set]

Gets or sets the value at row 4, column 1 of this instance.

5.65.5.19 double OpenTK.Matrix4d.M42 [get, set]

Gets or sets the value at row 4, column 2 of this instance.

5.65.5.20 double OpenTK.Matrix4d.M43 [get, set]

Gets or sets the value at row 4, column 3 of this instance.

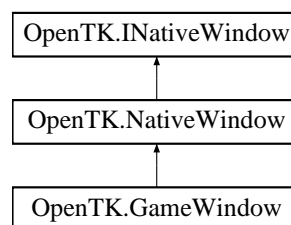
5.65.5.21 double OpenTK.Matrix4d.M44 [get, set]

Gets or sets the value at row 4, column 4 of this instance.

5.66 OpenTK.NativeWindow Class Reference

Instances of this class implement the [OpenTK.INativeWindow](#) interface on the current platform.

Inheritance diagram for OpenTK.NativeWindow:



Public Member Functions

- [NativeWindow](#) ()
Constructs a new [NativeWindow](#) with default attributes without enabling events.
- [NativeWindow](#) (int width, int height, string title, GameWindowFlags options, [GraphicsMode](#) mode, [DisplayDevice](#) device)
Constructs a new centered [NativeWindow](#) with the specified attributes.
- [NativeWindow](#) (int x, int y, int width, int height, string title, GameWindowFlags options, [GraphicsMode](#) mode, [DisplayDevice](#) device)
Constructs a new [NativeWindow](#) with the specified attributes.
- void [Close](#) ()
Closes the [NativeWindow](#).
- Point [PointToClient](#) (Point point)
Transforms the specified point from screen to client coordinates.
- Point [PointToScreen](#) (Point point)
Transforms the specified point from client to screen coordinates.
- void [ProcessEvents](#) ()
Processes operating system events until the [NativeWindow](#) becomes idle.
- virtual void [Dispose](#) ()
Releases all non-managed resources belonging to this [NativeWindow](#).

Protected Member Functions

- void [EnsureUndisposed](#) ()
Ensures that this [NativeWindow](#) has not been disposed.
- virtual void [OnClosed](#) (EventArgs e)
Called when the [NativeWindow](#) has closed.
- virtual void [OnClosing](#) (CancelEventArgs e)
Called when the [NativeWindow](#) is about to close.
- virtual void [OnDisposed](#) (EventArgs e)
Called when the [NativeWindow](#) is disposed.

- virtual void [OnFocusedChanged](#) (EventArgs e)
Called when the [OpenTK.NativeWindow.Focused](#) property of the [NativeWindow](#) has changed.
- virtual void [OnIconChanged](#) (EventArgs e)
Called when the [OpenTK.NativeWindow.Icon](#) property of the [NativeWindow](#) has changed.
- virtual void [OnKeyPress](#) ([KeyPressEventArgs](#) e)
Called when a character is typed.
- virtual void [OnMove](#) (EventArgs e)
Called when the [NativeWindow](#) is moved.
- virtual void [OnMouseEnter](#) (EventArgs e)
Called whenever the mouse cursor reenters the window [Bounds](#).
- virtual void [OnMouseLeave](#) (EventArgs e)
Called whenever the mouse cursor leaves the window [Bounds](#).
- virtual void [OnResize](#) (EventArgs e)
Called when the [NativeWindow](#) is resized.
- virtual void [OnTitleChanged](#) (EventArgs e)
Called when the [OpenTK.NativeWindow.Title](#) property of the [NativeWindow](#) has changed.
- virtual void [OnVisibleChanged](#) (EventArgs e)
Called when the [OpenTK.NativeWindow.Visible](#) property of the [NativeWindow](#) has changed.
- virtual void [OnWindowBorderChanged](#) (EventArgs e)
Called when the [WindowBorder](#) of this [NativeWindow](#) has changed.
- virtual void [OnWindowStateChanged](#) (EventArgs e)
Called when the [WindowState](#) of this [NativeWindow](#) has changed.
- void [ProcessEvents](#) (bool retainEvents)
Processes operating system events until the [NativeWindow](#) becomes idle.

Properties

- Rectangle **Bounds** [get, set]
Gets or sets a System.Drawing.Rectangle structure that contains the external bounds of this window, in screen coordinates. External bounds include the title bar, borders and drawing area of the window.
- Rectangle **ClientRectangle** [get, set]
Gets or sets a System.Drawing.Rectangle structure that contains the internal bounds of this window, in client coordinates. The internal bounds include the drawing area of the window, but exclude the titlebar and window borders.
- Size **ClientSize** [get, set]
Gets or sets a System.Drawing.Size structure that contains the internal size this window.
- bool **Exists** [get]
Gets a value indicating whether a render window exists.
- bool **Focused** [get]
Gets a System.Boolean that indicates whether this [NativeWindow](#) has input focus.
- int **Height** [get, set]
Gets or sets the external height of this window.
- Icon **Icon** [get, set]
Gets or sets the System.Drawing.Icon for this [GameWindow](#).
- **IInputDriver InputDriver** [get]
This property is deprecated.
- Point **Location** [get, set]
Gets or sets a System.Drawing.Point structure that contains the location of this window on the desktop.
- Size **Size** [get, set]
Gets or sets a System.Drawing.Size structure that contains the external size of this window.
- string **Title** [get, set]
Gets or sets the [NativeWindow](#) title.
- bool **Visible** [get, set]
Gets or sets a System.Boolean that indicates whether this [NativeWindow](#) is visible.

- `int Width` [get, set]
Gets or sets the external width of this window.
- `WindowBorder WindowBorder` [get, set]
Gets or states the border of the [NativeWindow](#).
- `IWindowInfo WindowInfo` [get]
Gets the [OpenTK.Platform.IWindowInfo](#) of this window.
- `virtual WindowState WindowState` [get, set]
Gets or states the state of the [NativeWindow](#).
- `int X` [get, set]
Gets or sets the horizontal location of this window on the desktop.
- `int Y` [get, set]
Gets or sets the vertical location of this window on the desktop.
- `bool IsDisposed` [get, set]
Gets or sets a [System.Boolean](#), which indicates whether this instance has been disposed.

Events

- `EventHandler< EventArgs > Closed`
Occurs after the window has closed.
- `EventHandler< CancelEventArgs > Closing`
Occurs when the window is about to close.
- `EventHandler< EventArgs > Disposed`
Occurs when the window is disposed.
- `EventHandler< EventArgs > FocusedChanged`
Occurs when the [Focused](#) property of the window changes.
- `EventHandler< EventArgs > IconChanged`
Occurs when the [Icon](#) property of the window changes.
- `EventHandler< KeyPressEventArgs > KeyPress`

Occurs whenever a character is typed.

- EventHandler< EventArgs > [Move](#)

Occurs whenever the window is moved.

- EventHandler< EventArgs > [MouseEnter](#)

Occurs whenever the mouse cursor enters the window [Bounds](#).

- EventHandler< EventArgs > [MouseLeave](#)

Occurs whenever the mouse cursor leaves the window [Bounds](#).

- EventHandler< EventArgs > [Resize](#)

Occurs whenever the window is resized.

- EventHandler< EventArgs > [TitleChanged](#)

Occurs when the [Title](#) property of the window changes.

- EventHandler< EventArgs > [VisibleChanged](#)

Occurs when the [Visible](#) property of the window changes.

- EventHandler< EventArgs > [WindowBorderChanged](#)

Occurs when the [WindowBorder](#) property of the window changes.

- EventHandler< EventArgs > [WindowStateChanged](#)

Occurs when the [WindowState](#) property of the window changes.

5.66.1 Detailed Description

Instances of this class implement the [OpenTK.INativeWindow](#) interface on the current platform.

5.66.2 Constructor & Destructor Documentation

5.66.2.1 OpenTK.NativeWindow.NativeWindow ()

Constructs a new [NativeWindow](#) with default attributes without enabling events.

5.66.2.2 `OpenTK.NativeWindow.NativeWindow (int width, int height, string title, GameWindowFlags options, GraphicsMode mode, DisplayDevice device)`

Constructs a new centered [NativeWindow](#) with the specified attributes.

Parameters

- width* The width of the [NativeWindow](#) in pixels.
- height* The height of the [NativeWindow](#) in pixels.
- title* The title of the [NativeWindow](#).
- options* [GameWindow](#) options specifying window appearance and behavior.
- mode* The [OpenTK.Graphics.GraphicsMode](#) of the [NativeWindow](#).
- device* The [OpenTK.Graphics.DisplayDevice](#) to construct the [NativeWindow](#) in.

Exceptions

- System.ArgumentOutOfRangeException* If width or height is less than 1.
- System.ArgumentNullException* If mode or device is null.

5.66.2.3 `OpenTK.NativeWindow.NativeWindow (int x, int y, int width, int height, string title, GameWindowFlags options, GraphicsMode mode, DisplayDevice device)`

Constructs a new [NativeWindow](#) with the specified attributes.

Parameters

- x* Horizontal screen space coordinate of the [NativeWindow](#)'s origin.
- y* Vertical screen space coordinate of the [NativeWindow](#)'s origin.
- width* The width of the [NativeWindow](#) in pixels.
- height* The height of the [NativeWindow](#) in pixels.
- title* The title of the [NativeWindow](#).
- options* [GameWindow](#) options specifying window appearance and behavior.
- mode* The [OpenTK.Graphics.GraphicsMode](#) of the [NativeWindow](#).
- device* The [OpenTK.Graphics.DisplayDevice](#) to construct the [NativeWindow](#) in.

Exceptions

- System.ArgumentOutOfRangeException* If width or height is less than 1.
- System.ArgumentNullException* If mode or device is null.

5.66.3 Member Function Documentation

5.66.3.1 void OpenTK.NativeWindow.Close ()

Closes the [NativeWindow](#).

Implements [OpenTK.INativeWindow](#).

5.66.3.2 virtual void OpenTK.NativeWindow.Dispose () [virtual]

Releases all non-managed resources belonging to this [NativeWindow](#).

5.66.3.3 void OpenTK.NativeWindow.EnsureUndisposed () [protected]

Ensures that this [NativeWindow](#) has not been disposed.

Exceptions

System.ObjectDisposedException If this [NativeWindow](#) has been disposed.

5.66.3.4 virtual void OpenTK.NativeWindow.OnClosed (EventArgs *e*) [protected, virtual]

Called when the [NativeWindow](#) has closed.

Parameters

e Not used.

5.66.3.5 virtual void OpenTK.NativeWindow.OnClosing (CancelEventArgs *e*) [protected, virtual]

Called when the [NativeWindow](#) is about to close.

Parameters

e The System.ComponentModel.CancelEventArgs for this event. Set e.Cancel to true in order to stop the [NativeWindow](#) from closing.

5.66.3.6 virtual void OpenTK.NativeWindow.OnDisposed (EventArgs *e*) [protected, virtual]

Called when the [NativeWindow](#) is disposed.

Parameters

e Not used.

5.66.3.7 virtual void OpenTK.NativeWindow.OnFocusedChanged (EventArgs *e*) [protected, virtual]

Called when the [OpenTK.INativeWindow.Focused](#) property of the [NativeWindow](#) has changed.

Parameters

e Not used.

5.66.3.8 virtual void OpenTK.NativeWindow.OnIconChanged (EventArgs *e*) [protected, virtual]

Called when the [OpenTK.INativeWindow.Icon](#) property of the [NativeWindow](#) has changed.

Parameters

e Not used.

5.66.3.9 virtual void OpenTK.NativeWindow.OnKeyPress (KeyPressEventArgs *e*) [protected, virtual]

Called when a character is typed.

Parameters

e The [OpenTK.KeyPressEventArgs](#) for this event.

**5.66.3.10 virtual void OpenTK.NativeWindow.OnMouseEnter (EventArgs *e*)
[protected, virtual]**

Called whenever the mouse cursor reenters the window [Bounds](#).

Parameters

e Not used.

**5.66.3.11 virtual void OpenTK.NativeWindow.OnMouseLeave (EventArgs *e*)
[protected, virtual]**

Called whenever the mouse cursor leaves the window [Bounds](#).

Parameters

e Not used.

**5.66.3.12 virtual void OpenTK.NativeWindow.OnMove (EventArgs *e*)
[protected, virtual]**

Called when the [NativeWindow](#) is moved.

Parameters

e Not used.

**5.66.3.13 virtual void OpenTK.NativeWindow.OnResize (EventArgs *e*)
[protected, virtual]**

Called when the [NativeWindow](#) is resized.

Parameters

e Not used.

**5.66.3.14 virtual void OpenTK.NativeWindow.OnTitleChanged (EventArgs *e*)
[protected, virtual]**

Called when the [OpenTK.INativeWindow.Title](#) property of the [NativeWindow](#) has changed.

Parameters

e Not used.

5.66.3.15 `virtual void OpenTK.NativeWindow.OnVisibleChanged (EventArgs e) [protected, virtual]`

Called when the [OpenTK.INativeWindow.Visible](#) property of the [NativeWindow](#) has changed.

Parameters

e Not used.

5.66.3.16 `virtual void OpenTK.NativeWindow.OnWindowBorderChanged (EventArgs e) [protected, virtual]`

Called when the WindowBorder of this [NativeWindow](#) has changed.

Parameters

e Not used.

5.66.3.17 `virtual void OpenTK.NativeWindow.OnWindowStateChanged (EventArgs e) [protected, virtual]`

Called when the WindowState of this [NativeWindow](#) has changed.

Parameters

e Not used.

5.66.3.18 `Point OpenTK.NativeWindow.PointToClient (Point point)`

Transforms the specified point from screen to client coordinates.

Parameters

point A System.Drawing.Point to transform.

Returns

The point transformed to client coordinates.

Implements [OpenTK.INativeWindow](#).

5.66.3.19 Point OpenTK.NativeWindow.PointToScreen (Point *point*)

Transforms the specified point from client to screen coordinates.

Parameters

point A System.Drawing.Point to transform.

Returns

The point transformed to screen coordinates.

Implements [OpenTK.INativeWindow](#).

5.66.3.20 void OpenTK.NativeWindow.ProcessEvents ()

Processes operating system events until the [NativeWindow](#) becomes idle.

Implements [OpenTK.INativeWindow](#).

**5.66.3.21 void OpenTK.NativeWindow.ProcessEvents (bool *retainEvents*)
[protected]**

Processes operating system events until the [NativeWindow](#) becomes idle.

Parameters

retainEvents If true, the state of underlying system event propagation will be preserved, otherwise event propagation will be enabled if it has not been already.

5.66.4 Property Documentation**5.66.4.1 Rectangle OpenTK.NativeWindow.Bounds [get, set]**

Gets or sets a System.Drawing.Rectangle structure that contains the external bounds of this window, in screen coordinates. External bounds include the title bar, borders and drawing area of the window.

Implements [OpenTK.INativeWindow](#).

5.66.4.2 Rectangle OpenTK.NativeWindow.ClientRectangle [get, set]

Gets or sets a System.Drawing.Rectangle structure that contains the internal bounds of this window, in client coordinates. The internal bounds include the drawing area of the window, but exclude the titlebar and window borders.

Implements [OpenTK.INativeWindow](#).

5.66.4.3 Size `OpenTK.NativeWindow.ClientSize` [get, set]

Gets or sets a `System.Drawing.Size` structure that contains the internal size this window.

Implements [OpenTK.INativeWindow](#).

5.66.4.4 bool `OpenTK.NativeWindow.Exists` [get]

Gets a value indicating whether a render window exists.

Implements [OpenTK.INativeWindow](#).

5.66.4.5 bool `OpenTK.NativeWindow.Focused` [get]

Gets a `System.Boolean` that indicates whether this [NativeWindow](#) has input focus.

Implements [OpenTK.INativeWindow](#).

5.66.4.6 int `OpenTK.NativeWindow.Height` [get, set]

Gets or sets the external height of this window.

Implements [OpenTK.INativeWindow](#).

5.66.4.7 Icon `OpenTK.NativeWindow.Icon` [get, set]

Gets or sets the `System.Drawing.Icon` for this [GameWindow](#).

Implements [OpenTK.INativeWindow](#).

5.66.4.8 `IInputDriver` `OpenTK.NativeWindow.InputDriver` [get]

This property is deprecated.

Implements [OpenTK.INativeWindow](#).

5.66.4.9 bool `OpenTK.NativeWindow.IsDisposed` [get, set, protected]

Gets or sets a `System.Boolean`, which indicates whether this instance has been disposed.

5.66.4.10 Point OpenTK.NativeWindow.Location [get, set]

Gets or sets a System.Drawing.Point structure that contains the location of this window on the desktop.

Implements [OpenTK.INativeWindow](#).

5.66.4.11 Size OpenTK.NativeWindow.Size [get, set]

Gets or sets a System.Drawing.Size structure that contains the external size of this window.

Implements [OpenTK.INativeWindow](#).

5.66.4.12 string OpenTK.NativeWindow.Title [get, set]

Gets or sets the [NativeWindow](#) title.

Implements [OpenTK.INativeWindow](#).

5.66.4.13 bool OpenTK.NativeWindow.Visible [get, set]

Gets or sets a System.Boolean that indicates whether this [NativeWindow](#) is visible.

Implements [OpenTK.INativeWindow](#).

5.66.4.14 int OpenTK.NativeWindow.Width [get, set]

Gets or sets the external width of this window.

Implements [OpenTK.INativeWindow](#).

5.66.4.15 WindowBorder OpenTK.NativeWindow.WindowBorder [get, set]

Gets or states the border of the [NativeWindow](#).

Implements [OpenTK.INativeWindow](#).

5.66.4.16 IWindowInfo OpenTK.NativeWindow.WindowInfo [get]

Gets the [OpenTK.Platform.IWindowInfo](#) of this window.

Implements [OpenTK.INativeWindow](#).

5.66.4.17 virtual WindowState OpenTK.NativeWindow.WindowState [get, set]

Gets or states the state of the [NativeWindow](#).

Implements [OpenTK.INativeWindow](#).

5.66.4.18 int OpenTK.NativeWindow.X [get, set]

Gets or sets the horizontal location of this window on the desktop.

Implements [OpenTK.INativeWindow](#).

5.66.4.19 int OpenTK.NativeWindow.Y [get, set]

Gets or sets the vertical location of this window on the desktop.

Implements [OpenTK.INativeWindow](#).

5.66.5 Event Documentation**5.66.5.1 EventHandler<EventArgs> OpenTK.NativeWindow.Closed**

Occurs after the window has closed.

Implements [OpenTK.INativeWindow](#).

5.66.5.2 EventHandler<CancelEventArgs> OpenTK.NativeWindow.Closing

Occurs when the window is about to close.

Implements [OpenTK.INativeWindow](#).

5.66.5.3 EventHandler<EventArgs> OpenTK.NativeWindow.Disposed

Occurs when the window is disposed.

Implements [OpenTK.INativeWindow](#).

5.66.5.4 EventHandler<EventArgs> OpenTK.NativeWindow.FocusedChanged

Occurs when the [Focused](#) property of the window changes.

Implements [OpenTK.INativeWindow](#).

5.66.5.5 EventHandler<EventArgs> OpenTK.NativeWindow.IconChanged

Occurs when the [Icon](#) property of the window changes.

Implements [OpenTK.INativeWindow](#).

**5.66.5.6 EventHandler<KeyPressEventArgs>
OpenTK.NativeWindow.KeyPress**

Occurs whenever a character is typed.

Implements [OpenTK.INativeWindow](#).

5.66.5.7 EventHandler<EventArgs> OpenTK.NativeWindow.MouseEnter

Occurs whenever the mouse cursor enters the window [Bounds](#).

Implements [OpenTK.INativeWindow](#).

5.66.5.8 EventHandler<EventArgs> OpenTK.NativeWindow.MouseLeave

Occurs whenever the mouse cursor leaves the window [Bounds](#).

Implements [OpenTK.INativeWindow](#).

5.66.5.9 EventHandler<EventArgs> OpenTK.NativeWindow.Move

Occurs whenever the window is moved.

Implements [OpenTK.INativeWindow](#).

5.66.5.10 EventHandler<EventArgs> OpenTK.NativeWindow.Resize

Occurs whenever the window is resized.

Implements [OpenTK.INativeWindow](#).

5.66.5.11 EventHandler<EventArgs> OpenTK.NativeWindow.TitleChanged

Occurs when the [Title](#) property of the window changes.

Implements [OpenTK.INativeWindow](#).

5.66.5.12 EventHandler<EventArgs> OpenTK.NativeWindow.VisibleChanged

Occurs when the [Visible](#) property of the window changes.

Implements [OpenTK.INativeWindow](#).

**5.66.5.13 EventHandler<EventArgs>
OpenTK.NativeWindow.WindowBorderChanged**

Occurs when the [WindowBorder](#) property of the window changes.

Implements [OpenTK.INativeWindow](#).

**5.66.5.14 EventHandler<EventArgs>
OpenTK.NativeWindow.WindowStateChanged**

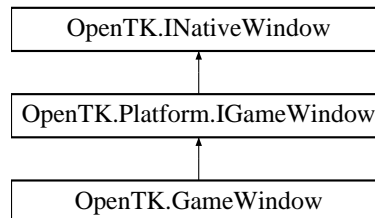
Occurs when the [WindowState](#) property of the window changes.

Implements [OpenTK.INativeWindow](#).

5.67 OpenTK.Platform.IGameWindow Interface Reference

Defines the interface for a [GameWindow](#).

Inheritance diagram for OpenTK.Platform.IGameWindow:



Public Member Functions

- void [Run](#) ()
Enters the game loop of the [GameWindow](#) using the maximum update rate.
- void [Run](#) (double updateRate)
Enters the game loop of the [GameWindow](#) using the specified update rate.

- void [MakeCurrent](#) ()
Makes the GraphicsContext current on the calling thread.
- void [SwapBuffers](#) ()
Swaps the front and back buffers of the current GraphicsContext, presenting the rendered scene to the user.

Events

- EventHandler< EventArgs > [Load](#)
Occurs before the window is displayed for the first time.
- EventHandler< EventArgs > [Unload](#)
Occurs before the window is destroyed.
- EventHandler< [FrameEventArgs](#) > [UpdateFrame](#)
Occurs when it is time to update a frame.
- EventHandler< [FrameEventArgs](#) > [RenderFrame](#)
Occurs when it is time to render a frame.

5.67.1 Detailed Description

Defines the interface for a [GameWindow](#).

5.67.2 Member Function Documentation

5.67.2.1 void OpenTK.Platform.IGameWindow.MakeCurrent ()

Makes the GraphicsContext current on the calling thread.

Implemented in [OpenTK.GameWindow](#).

5.67.2.2 void OpenTK.Platform.IGameWindow.Run (double *updateRate*)

Enters the game loop of the [GameWindow](#) using the specified update rate.

Implemented in [OpenTK.GameWindow](#).

5.67.2.3 void OpenTK.Platform.IGameWindow.Run ()

Enters the game loop of the [GameWindow](#) using the maximum update rate.

See also

[Run\(double\)](#)

Implemented in [OpenTK.GameWindow](#).

5.67.2.4 void OpenTK.Platform.IGameWindow.SwapBuffers ()

Swaps the front and back buffers of the current GraphicsContext, presenting the rendered scene to the user.

Implemented in [OpenTK.GameWindow](#).

5.67.3 Event Documentation**5.67.3.1 EventHandler<EventArgs> OpenTK.Platform.IGameWindow.Load**

Occurs before the window is displayed for the first time.

Implemented in [OpenTK.GameWindow](#).

**5.67.3.2 EventHandler<FrameEventArgs>
OpenTK.Platform.IGameWindow.RenderFrame**

Occurs when it is time to render a frame.

Implemented in [OpenTK.GameWindow](#).

5.67.3.3 EventHandler<EventArgs> OpenTK.Platform.IGameWindow.Unload

Occurs before the window is destroyed.

Implemented in [OpenTK.GameWindow](#).

**5.67.3.4 EventHandler<FrameEventArgs>
OpenTK.Platform.IGameWindow.UpdateFrame**

Occurs when it is time to update a frame.

Implemented in [OpenTK.GameWindow](#).

5.68 OpenTK.Platform.IWindowInfo Interface Reference

Describes an OS window.

Inherited by OpenTK.Platform.Dummy.DummyWindowInfo, OpenTK.Platform.Egl.EglWindowInfo, OpenTK.Platform.MacOS.CarbonWindowInfo, OpenTK.Platform.Windows.WinWindowInfo, and OpenTK.Platform.X11.X11WindowInfo.

5.68.1 Detailed Description

Describes an OS window.

5.69 OpenTK.PlatformException Class Reference

Defines a platform specific exception.

Public Member Functions

- [PlatformException](#) (string s)
Constructs a new [PlatformException](#).

5.69.1 Detailed Description

Defines a platform specific exception.

5.69.2 Constructor & Destructor Documentation

5.69.2.1 OpenTK.PlatformException.PlatformException (string s)

Constructs a new [PlatformException](#).

5.70 OpenTK.Properties.Resources Class Reference

A strongly-typed resource class, for looking up localized strings, etc.

5.70.1 Detailed Description

A strongly-typed resource class, for looking up localized strings, etc.

5.71 OpenTK.Quaternion Struct Reference

Represents a [Quaternion](#).

Public Member Functions

- [Quaternion](#) ([Vector3](#) v, float w)
Construct a new [Quaternion](#) from vector and w components.
- [Quaternion](#) (float x, float y, float z, float w)
Construct a new [Quaternion](#).
- void [ToAxisAngle](#) (out [Vector3](#) axis, out float angle)
Convert the current quaternion to axis angle representation.
- [Vector4 ToAxisAngle](#) ()
Convert this instance to an axis-angle representation.
- void [Normalize](#) ()
Scales the [Quaternion](#) to unit length.
- void [Conjugate](#) ()
Convert this quaternion to its conjugate.
- override string [ToString](#) ()
Returns a System.String that represents the current [Quaternion](#).
- override bool [Equals](#) (object other)
Compares this object instance to another object for equality.
- override int [GetHashCode](#) ()
Provides the hash code for this object.
- bool [Equals](#) ([Quaternion](#) other)
Compares this [Quaternion](#) instance to another [Quaternion](#) for equality.

Static Public Member Functions

- static [Quaternion Add](#) ([Quaternion](#) left, [Quaternion](#) right)
Add two quaternions.
- static void [Add](#) (ref [Quaternion](#) left, ref [Quaternion](#) right, out [Quaternion](#) result)
Add two quaternions.
- static [Quaternion Sub](#) ([Quaternion](#) left, [Quaternion](#) right)
Subtracts two instances.
- static void [Sub](#) (ref [Quaternion](#) left, ref [Quaternion](#) right, out [Quaternion](#) result)
Subtracts two instances.
- static [Quaternion Mult](#) ([Quaternion](#) left, [Quaternion](#) right)
Multiplies two instances.
- static void [Mult](#) (ref [Quaternion](#) left, ref [Quaternion](#) right, out [Quaternion](#) result)
Multiplies two instances.
- static [Quaternion Multiply](#) ([Quaternion](#) left, [Quaternion](#) right)
Multiplies two instances.
- static void [Multiply](#) (ref [Quaternion](#) left, ref [Quaternion](#) right, out [Quaternion](#) result)
Multiplies two instances.
- static void [Multiply](#) (ref [Quaternion](#) quaternion, float scale, out [Quaternion](#) result)
Multiplies an instance by a scalar.
- static [Quaternion Multiply](#) ([Quaternion](#) quaternion, float scale)
Multiplies an instance by a scalar.
- static [Quaternion Conjugate](#) ([Quaternion](#) q)
Get the conjugate of the given quaternion.
- static void [Conjugate](#) (ref [Quaternion](#) q, out [Quaternion](#) result)
Get the conjugate of the given quaternion.
- static [Quaternion Invert](#) ([Quaternion](#) q)

Get the inverse of the given quaternion.

- static void **Invert** (ref **Quaternion** q, out **Quaternion** result)
Get the inverse of the given quaternion.
- static **Quaternion Normalize** (**Quaternion** q)
Scale the given quaternion to unit length.
- static void **Normalize** (ref **Quaternion** q, out **Quaternion** result)
Scale the given quaternion to unit length.
- static **Quaternion FromAxisAngle** (**Vector3** axis, float angle)
Build a quaternion from the given axis and angle.
- static **Quaternion Slerp** (**Quaternion** q1, **Quaternion** q2, float blend)
Do Spherical linear interpolation between two quaternions.
- static **Quaternion operator+** (**Quaternion** left, **Quaternion** right)
Adds two instances.
- static **Quaternion operator-** (**Quaternion** left, **Quaternion** right)
Subtracts two instances.
- static **Quaternion operator*** (**Quaternion** left, **Quaternion** right)
Multiplies two instances.
- static **Quaternion operator*** (**Quaternion** quaternion, float scale)
Multiplies an instance by a scalar.
- static **Quaternion operator*** (float scale, **Quaternion** quaternion)
Multiplies an instance by a scalar.
- static bool **operator==** (**Quaternion** left, **Quaternion** right)
Compares two instances for equality.
- static bool **operator!=** (**Quaternion** left, **Quaternion** right)
Compares two instances for inequality.

Public Attributes

- **Vector3 xyz**
- float **w**

Static Public Attributes

- static [Quaternion Identity](#) = new [Quaternion](#)(0, 0, 0, 1)
Defines the identity quaternion.

Properties

- [Vector3 XYZ](#) [get, set]
Gets or sets an [OpenTK.Vector3](#) with the X, Y and Z components of this instance.
- [Vector3 Xyz](#) [get, set]
Gets or sets an [OpenTK.Vector3](#) with the X, Y and Z components of this instance.
- float [X](#) [get, set]
Gets or sets the X component of this instance.
- float [Y](#) [get, set]
Gets or sets the Y component of this instance.
- float [Z](#) [get, set]
Gets or sets the Z component of this instance.
- float [W](#) [get, set]
Gets or sets the W component of this instance.
- float [Length](#) [get]
Gets the length (magnitude) of the quaternion.
- float [LengthSquared](#) [get]
Gets the square of the quaternion length (magnitude).

5.71.1 Detailed Description

Represents a [Quaternion](#).

5.71.2 Constructor & Destructor Documentation

5.71.2.1 OpenTK.Quaternion.Quaternion (Vector3 v, float w)

Construct a new [Quaternion](#) from vector and w components.

Parameters

- v* The vector part
- w* The w part

5.71.2.2 OpenTK.Quaternion.Quaternion (float *x*, float *y*, float *z*, float *w*)

Construct a new [Quaternion](#).

Parameters

- x* The x component
- y* The y component
- z* The z component
- w* The w component

5.71.3 Member Function Documentation**5.71.3.1 static Quaternion OpenTK.Quaternion.Add (Quaternion *left*, Quaternion *right*) [static]**

Add two quaternions.

Parameters

- left* The first operand
- right* The second operand

Returns

The result of the addition

5.71.3.2 static void OpenTK.Quaternion.Add (ref Quaternion *left*, ref Quaternion *right*, out Quaternion *result*) [static]

Add two quaternions.

Parameters

- left* The first operand
- right* The second operand
- result* The result of the addition

5.71.3.3 void OpenTK.Quaternion.Conjugate ()

Convert this quaternion to its conjugate.

5.71.3.4 static void OpenTK.Quaternion.Conjugate (ref Quaternion *q*, out Quaternion *result*) [static]

Get the conjugate of the given quaternion.

Parameters

q The quaternion

result The conjugate of the given quaternion

5.71.3.5 static Quaternion OpenTK.Quaternion.Conjugate (Quaternion *q*) [static]

Get the conjugate of the given quaternion.

Parameters

q The quaternion

Returns

The conjugate of the given quaternion

5.71.3.6 bool OpenTK.Quaternion.Equals (Quaternion *other*)

Compares this [Quaternion](#) instance to another [Quaternion](#) for equality.

Parameters

other The other [Quaternion](#) to be used in the comparison.

Returns

True if both instances are equal; false otherwise.

5.71.3.7 override bool OpenTK.Quaternion.Equals (object *other*)

Compares this object instance to another object for equality.

Parameters

other The other object to be used in the comparison.

Returns

True if both objects are Quaternions of equal value. Otherwise it returns false.

5.71.3.8 static Quaternion OpenTK.Quaternion.FromAxisAngle (Vector3 *axis*, float *angle*) [static]

Build a quaternion from the given axis and angle.

Parameters

axis The axis to rotate about

angle The rotation angle in radians

Returns**5.71.3.9 override int OpenTK.Quaternion.GetHashCode ()**

Provides the hash code for this object.

Returns

A hash code formed from the bitwise XOR of this objects members.

5.71.3.10 static void OpenTK.Quaternion.Invert (ref Quaternion *q*, out Quaternion *result*) [static]

Get the inverse of the given quaternion.

Parameters

q The quaternion to invert

result The inverse of the given quaternion

**5.71.3.11 static Quaternion OpenTK.Quaternion.Invert (Quaternion *q*)
[static]**

Get the inverse of the given quaternion.

Parameters

q The quaternion to invert

Returns

The inverse of the given quaternion

**5.71.3.12 static Quaternion OpenTK.Quaternion.Mult (Quaternion *left*,
Quaternion *right*) [static]**

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

Returns

A new instance containing the result of the calculation.

**5.71.3.13 static void OpenTK.Quaternion.Mult (ref Quaternion *left*, ref
Quaternion *right*, out Quaternion *result*) [static]**

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

result A new instance containing the result of the calculation.

**5.71.3.14 static Quaternion OpenTK.Quaternion.Multiply (Quaternion
quaternion, float *scale*) [static]**

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

Returns

A new instance containing the result of the calculation.

5.71.3.15 static Quaternion OpenTK.Quaternion.Multiply (Quaternion *left*, Quaternion *right*) [static]

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

Returns

A new instance containing the result of the calculation.

5.71.3.16 static void OpenTK.Quaternion.Multiply (ref Quaternion *left*, ref Quaternion *right*, out Quaternion *result*) [static]

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

result A new instance containing the result of the calculation.

5.71.3.17 static void OpenTK.Quaternion.Multiply (ref Quaternion *quaternion*, float *scale*, out Quaternion *result*) [static]

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

result A new instance containing the result of the calculation.

**5.71.3.18 static Quaternion OpenTK.Quaternion.Normalize (Quaternion *q*)
[static]**

Scale the given quaternion to unit length.

Parameters

q The quaternion to normalize

Returns

The normalized quaternion

5.71.3.19 void OpenTK.Quaternion.Normalize ()

Scales the [Quaternion](#) to unit length.

5.71.3.20 static void OpenTK.Quaternion.Normalize (ref Quaternion *q*, out Quaternion *result*) [static]

Scale the given quaternion to unit length.

Parameters

q The quaternion to normalize

result The normalized quaternion

5.71.3.21 static bool OpenTK.Quaternion.operator!= (Quaternion *left*, Quaternion *right*) [static]

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.71.3.22 static Quaternion OpenTK.Quaternion.operator* (Quaternion *quaternion*, float *scale*) [static]

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

Returns

A new instance containing the result of the calculation.

5.71.3.23 static Quaternion OpenTK.Quaternion.operator* (float *scale*, Quaternion *quaternion*) [static]

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

Returns

A new instance containing the result of the calculation.

5.71.3.24 static Quaternion OpenTK.Quaternion.operator* (Quaternion *left*, Quaternion *right*) [static]

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.71.3.25 static Quaternion OpenTK.Quaternion.operator+ (Quaternion *left*, Quaternion *right*) [static]

Adds two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.71.3.26 static Quaternion OpenTK.Quaternion.operator- (Quaternion *left*, Quaternion *right*) [static]

Subtracts two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.71.3.27 static bool OpenTK.Quaternion.operator== (Quaternion *left*, Quaternion *right*) [static]

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.71.3.28 static Quaternion OpenTK.Quaternion.Slerp (Quaternion *q1*, Quaternion *q2*, float *blend*) [static]

Do Spherical linear interpolation between two quaternions.

Parameters

q1 The first quaternion

q2 The second quaternion

blend The blend factor

Returns

A smooth blend between the given quaternions

5.71.3.29 static Quaternion OpenTK.Quaternion.Sub (Quaternion *left*, Quaternion *right*) [static]

Subtracts two instances.

Parameters

left The left instance.

right The right instance.

Returns

The result of the operation.

5.71.3.30 static void OpenTK.Quaternion.Sub (ref Quaternion *left*, ref Quaternion *right*, out Quaternion *result*) [static]

Subtracts two instances.

Parameters

left The left instance.

right The right instance.

result The result of the operation.

5.71.3.31 `void OpenTK.Quaternion.ToAxisAngle (out Vector3 axis, out float angle)`

Convert the current quaternion to axis angle representation.

Parameters

axis The resultant axis

angle The resultant angle

5.71.3.32 `Vector4 OpenTK.Quaternion.ToAxisAngle ()`

Convert this instance to an axis-angle representation.

Returns

A [Vector4](#) that is the axis-angle representation of this quaternion.

5.71.3.33 `override string OpenTK.Quaternion.ToString ()`

Returns a System.String that represents the current [Quaternion](#).

Returns

5.71.4 Member Data Documentation

5.71.4.1 `Quaternion OpenTK.Quaternion.Identity = new Quaternion(0, 0, 0, 1)`
[static]

Defines the identity quaternion.

5.71.5 Property Documentation

5.71.5.1 `float OpenTK.Quaternion.Length` [get]

Gets the length (magnitude) of the quaternion.

See also

[LengthSquared](#)

5.71.5.2 float OpenTK.Quaternion.LengthSquared [get]

Gets the square of the quaternion length (magnitude).

5.71.5.3 float OpenTK.Quaternion.W [get, set]

Gets or sets the W component of this instance.

5.71.5.4 float OpenTK.Quaternion.X [get, set]

Gets or sets the X component of this instance.

5.71.5.5 Vector3 OpenTK.Quaternion.Xyz [get, set]

Gets or sets an [OpenTK.Vector3](#) with the X, Y and Z components of this instance.

5.71.5.6 Vector3 OpenTK.Quaternion.XYZ [get, set]

Gets or sets an [OpenTK.Vector3](#) with the X, Y and Z components of this instance.

5.71.5.7 float OpenTK.Quaternion.Y [get, set]

Gets or sets the Y component of this instance.

5.71.5.8 float OpenTK.Quaternion.Z [get, set]

Gets or sets the Z component of this instance.

5.72 OpenTK.Quaterniond Struct Reference

Represents a double-precision [Quaternion](#).

Public Member Functions

- [Quaterniond](#) ([Vector3d](#) v, double w)
Construct a new [Quaterniond](#) from vector and w components.
- [Quaterniond](#) (double x, double y, double z, double w)

Construct a new *Quaterniond*.

- void **ToAxisAngle** (out **Vector3d** axis, out double angle)
Convert the current quaternion to axis angle representation.
- **Vector4d ToAxisAngle** ()
Convert this instance to an axis-angle representation.
- void **Normalize** ()
*Scales the *Quaterniond* to unit length.*
- void **Conjugate** ()
*Convert this *Quaterniond* to its conjugate.*
- override string **ToString** ()
*Returns a *System.String* that represents the current *Quaterniond*.*
- override bool **Equals** (object other)
Compares this object instance to another object for equality.
- override int **GetHashCode** ()
Provides the hash code for this object.
- bool **Equals** (*Quaterniond* other)
*Compares this *Quaterniond* instance to another *Quaterniond* for equality.*

Static Public Member Functions

- static *Quaterniond* **Add** (*Quaterniond* left, *Quaterniond* right)
Add two quaternions.
- static void **Add** (ref *Quaterniond* left, ref *Quaterniond* right, out *Quaterniond* result)
Add two quaternions.
- static *Quaterniond* **Sub** (*Quaterniond* left, *Quaterniond* right)
Subtracts two instances.
- static void **Sub** (ref *Quaterniond* left, ref *Quaterniond* right, out *Quaterniond* result)
Subtracts two instances.

- static [Quaterniond Mult](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Multiplies two instances.
- static void [Mult](#) (ref [Quaterniond](#) left, ref [Quaterniond](#) right, out [Quaterniond](#) result)
Multiplies two instances.
- static [Quaterniond Multiply](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Multiplies two instances.
- static void [Multiply](#) (ref [Quaterniond](#) left, ref [Quaterniond](#) right, out [Quaterniond](#) result)
Multiplies two instances.
- static void [Multiply](#) (ref [Quaterniond](#) quaternion, double scale, out [Quaterniond](#) result)
Multiplies an instance by a scalar.
- static [Quaterniond Multiply](#) ([Quaterniond](#) quaternion, double scale)
Multiplies an instance by a scalar.
- static [Quaterniond Conjugate](#) ([Quaterniond](#) q)
Get the conjugate of the given [Quaterniond](#).
- static void [Conjugate](#) (ref [Quaterniond](#) q, out [Quaterniond](#) result)
Get the conjugate of the given [Quaterniond](#).
- static [Quaterniond Invert](#) ([Quaterniond](#) q)
Get the inverse of the given [Quaterniond](#).
- static void [Invert](#) (ref [Quaterniond](#) q, out [Quaterniond](#) result)
Get the inverse of the given [Quaterniond](#).
- static [Quaterniond Normalize](#) ([Quaterniond](#) q)
Scale the given [Quaterniond](#) to unit length.
- static void [Normalize](#) (ref [Quaterniond](#) q, out [Quaterniond](#) result)
Scale the given [Quaterniond](#) to unit length.
- static [Quaterniond FromAxisAngle](#) ([Vector3d](#) axis, double angle)
Build a [Quaterniond](#) from the given axis and angle.

- static [Quaterniond Slerp](#) ([Quaterniond](#) q1, [Quaterniond](#) q2, double blend)
Do Spherical linear interpolation between two quaternions.
- static [Quaterniond operator+](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Adds two instances.
- static [Quaterniond operator-](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Subtracts two instances.
- static [Quaterniond operator*](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Multiplies two instances.
- static [Quaterniond operator*](#) ([Quaterniond](#) quaternion, double scale)
Multiplies an instance by a scalar.
- static [Quaterniond operator*](#) (double scale, [Quaterniond](#) quaternion)
Multiplies an instance by a scalar.
- static bool [operator==](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([Quaterniond](#) left, [Quaterniond](#) right)
Compares two instances for inequality.

Public Attributes

- [Vector3d xyz](#)
- double w

Static Public Attributes

- static readonly [Quaterniond Identity](#) = new [Quaterniond](#)(0, 0, 0, 1)
Defines the identity quaternion.

Properties

- [Vector3d XYZ](#) [get, set]
Gets or sets an [OpenTK.Vector3d](#) with the X, Y and Z components of this instance.

- [Vector3d Xyz](#) [get, set]
Gets or sets an [OpenTK.Vector3d](#) with the X, Y and Z components of this instance.
- double [X](#) [get, set]
Gets or sets the X component of this instance.
- double [Y](#) [get, set]
Gets or sets the Y component of this instance.
- double [Z](#) [get, set]
Gets or sets the Z component of this instance.
- double [W](#) [get, set]
Gets or sets the W component of this instance.
- double [Length](#) [get]
Gets the length (magnitude) of the [Quaterniond](#).
- double [LengthSquared](#) [get]
Gets the square of the [Quaterniond](#) length (magnitude).

5.72.1 Detailed Description

Represents a double-precision [Quaternion](#).

5.72.2 Constructor & Destructor Documentation

5.72.2.1 [OpenTK.Quaterniond.Quaterniond](#) ([Vector3d](#) *v*, double *w*)

Construct a new [Quaterniond](#) from vector and w components.

Parameters

- v* The vector part
- w* The w part

5.72.2.2 [OpenTK.Quaterniond.Quaterniond](#) (double *x*, double *y*, double *z*, double *w*)

Construct a new [Quaterniond](#).

Parameters

- x* The x component
- y* The y component
- z* The z component
- w* The w component

5.72.3 Member Function Documentation**5.72.3.1 static Quaterniond OpenTK.Quaterniond.Add (Quaterniond *left*, Quaterniond *right*) [static]**

Add two quaternions.

Parameters

- left* The first operand
- right* The second operand

Returns

The result of the addition

5.72.3.2 static void OpenTK.Quaterniond.Add (ref Quaterniond *left*, ref Quaterniond *right*, out Quaterniond *result*) [static]

Add two quaternions.

Parameters

- left* The first operand
- right* The second operand
- result* The result of the addition

5.72.3.3 void OpenTK.Quaterniond.Conjugate ()

Convert this [Quaterniond](#) to its conjugate.

5.72.3.4 static void OpenTK.Quaterniond.Conjugate (ref Quaterniond *q*, out Quaterniond *result*) [static]

Get the conjugate of the given [Quaterniond](#).

Parameters

q The [Quaterniond](#)

result The conjugate of the given [Quaterniond](#)

5.72.3.5 static Quaterniond OpenTK.Quaterniond.Conjugate (Quaterniond *q*) [static]

Get the conjugate of the given [Quaterniond](#).

Parameters

q The [Quaterniond](#)

Returns

The conjugate of the given [Quaterniond](#)

5.72.3.6 bool OpenTK.Quaterniond.Equals (Quaterniond *other*)

Compares this [Quaterniond](#) instance to another [Quaterniond](#) for equality.

Parameters

other The other [Quaterniond](#) to be used in the comparison.

Returns

True if both instances are equal; false otherwise.

5.72.3.7 override bool OpenTK.Quaterniond.Equals (object *other*)

Compares this object instance to another object for equality.

Parameters

other The other object to be used in the comparison.

Returns

True if both objects are Quaternions of equal value. Otherwise it returns false.

5.72.3.8 static Quaterniond OpenTK.Quaterniond.FromAxisAngle (Vector3d *axis*, double *angle*) [static]

Build a [Quaterniond](#) from the given axis and angle.

Parameters

axis The axis to rotate about

angle The rotation angle in radians

Returns**5.72.3.9 override int OpenTK.Quaterniond.GetHashCode ()**

Provides the hash code for this object.

Returns

A hash code formed from the bitwise XOR of this objects members.

5.72.3.10 static void OpenTK.Quaterniond.Invert (ref Quaterniond *q*, out Quaterniond *result*) [static]

Get the inverse of the given [Quaterniond](#).

Parameters

q The [Quaterniond](#) to invert

result The inverse of the given [Quaterniond](#)

5.72.3.11 static Quaterniond OpenTK.Quaterniond.Invert (Quaterniond *q*) [static]

Get the inverse of the given [Quaterniond](#).

Parameters

q The [Quaterniond](#) to invert

Returns

The inverse of the given [Quaterniond](#)

5.72.3.12 static Quaterniond OpenTK.Quaterniond.Mult (Quaterniond *left*, Quaterniond *right*) [static]

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

Returns

A new instance containing the result of the calculation.

5.72.3.13 static void OpenTK.Quaterniond.Mult (ref Quaterniond *left*, ref Quaterniond *right*, out Quaterniond *result*) [static]

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

result A new instance containing the result of the calculation.

5.72.3.14 static Quaterniond OpenTK.Quaterniond.Multiply (Quaterniond *quaternion*, double *scale*) [static]

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

Returns

A new instance containing the result of the calculation.

5.72.3.15 `static Quaterniond OpenTK.Quaterniond.Multiply (Quaterniond left, Quaterniond right) [static]`

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

Returns

A new instance containing the result of the calculation.

5.72.3.16 `static void OpenTK.Quaterniond.Multiply (ref Quaterniond left, ref Quaterniond right, out Quaterniond result) [static]`

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

result A new instance containing the result of the calculation.

5.72.3.17 `static void OpenTK.Quaterniond.Multiply (ref Quaterniond quaternion, double scale, out Quaterniond result) [static]`

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

result A new instance containing the result of the calculation.

5.72.3.18 `static Quaterniond OpenTK.Quaterniond.Normalize (Quaterniond q) [static]`

Scale the given [Quaterniond](#) to unit length.

Parameters

q The [Quaterniond](#) to normalize

Returns

The normalized [Quaterniond](#)

5.72.3.19 void OpenTK.Quaterniond.Normalize ()

Scales the [Quaterniond](#) to unit length.

5.72.3.20 static void OpenTK.Quaterniond.Normalize (ref Quaterniond *q*, out Quaterniond *result*) [static]

Scale the given [Quaterniond](#) to unit length.

Parameters

q The [Quaterniond](#) to normalize

result The normalized [Quaterniond](#)

5.72.3.21 static bool OpenTK.Quaterniond.operator!= (Quaterniond *left*, Quaterniond *right*) [static]

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.72.3.22 static Quaterniond OpenTK.Quaterniond.operator* (Quaterniond *quaternion*, double *scale*) [static]

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

Returns

A new instance containing the result of the calculation.

5.72.3.23 `static Quaterniond OpenTK.Quaterniond.operator* (double scale, Quaterniond quaternion) [static]`

Multiplies an instance by a scalar.

Parameters

quaternion The instance.

scale The scalar.

Returns

A new instance containing the result of the calculation.

5.72.3.24 `static Quaterniond OpenTK.Quaterniond.operator* (Quaterniond left, Quaterniond right) [static]`

Multiplies two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.72.3.25 `static Quaterniond OpenTK.Quaterniond.operator+ (Quaterniond left, Quaterniond right) [static]`

Adds two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.72.3.26 static Quaterniond OpenTK.Quaterniond.operator- (Quaterniond *left*, Quaterniond *right*) [static]

Subtracts two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.72.3.27 static bool OpenTK.Quaterniond.operator== (Quaterniond *left*, Quaterniond *right*) [static]

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.72.3.28 static Quaterniond OpenTK.Quaterniond.Slerp (Quaterniond *q1*, Quaterniond *q2*, double *blend*) [static]

Do Spherical linear interpolation between two quaternions.

Parameters

q1 The first [Quaterniond](#)

q2 The second [Quaterniond](#)

blend The blend factor

Returns

A smooth blend between the given quaternions

5.72.3.29 static Quaterniond OpenTK.Quaterniond.Sub (Quaterniond *left*, Quaterniond *right*) [static]

Subtracts two instances.

Parameters

left The left instance.

right The right instance.

Returns

The result of the operation.

5.72.3.30 static void OpenTK.Quaterniond.Sub (ref Quaterniond *left*, ref Quaterniond *right*, out Quaterniond *result*) [static]

Subtracts two instances.

Parameters

left The left instance.

right The right instance.

result The result of the operation.

5.72.3.31 void OpenTK.Quaterniond.ToAxisAngle (out Vector3d *axis*, out double *angle*)

Convert the current quaternion to axis angle representation.

Parameters

axis The resultant axis

angle The resultant angle

5.72.3.32 Vector4d OpenTK.Quaterniond.ToAxisAngle ()

Convert this instance to an axis-angle representation.

Returns

A [Vector4](#) that is the axis-angle representation of this quaternion.

5.72.3.33 override string OpenTK.Quaterniond.ToString ()

Returns a System.String that represents the current [Quaterniond](#).

Returns

5.72.4 Member Data Documentation

5.72.4.1 readonly Quaterniond OpenTK.Quaterniond.Identity = new Quaterniond(0, 0, 0, 1) [static]

Defines the identity quaternion.

5.72.5 Property Documentation

5.72.5.1 double OpenTK.Quaterniond.Length [get]

Gets the length (magnitude) of the [Quaterniond](#).

See also

[LengthSquared](#)

5.72.5.2 double OpenTK.Quaterniond.LengthSquared [get]

Gets the square of the [Quaterniond](#) length (magnitude).

5.72.5.3 double OpenTK.Quaterniond.W [get, set]

Gets or sets the W component of this instance.

5.72.5.4 double OpenTK.Quaterniond.X [get, set]

Gets or sets the X component of this instance.

5.72.5.5 Vector3d OpenTK.Quaterniond.Xyz [get, set]

Gets or sets an [OpenTK.Vector3d](#) with the X, Y and Z components of this instance.

5.72.5.6 Vector3d OpenTK.Quaterniond.XYZ [get, set]

Gets or sets an [OpenTK.Vector3d](#) with the X, Y and Z components of this instance.

5.72.5.7 double OpenTK.Quaterniond.Y [get, set]

Gets or sets the Y component of this instance.

5.72.5.8 double OpenTK.Quaterniond.Z [get, set]

Gets or sets the Z component of this instance.

5.73 OpenTK.Toolkit Class Reference

Provides static methods to manage an OpenTK application.

Static Public Member Functions

- static void [Init](#) ()

Initializes OpenTK. This method is necessary only if you are using OpenTK alongside a different windowing toolkit (e.g. GTK#) and should be the very first method called by your application (i.e. calling this method should be the very first statement executed by the "Main" method).

5.73.1 Detailed Description

Provides static methods to manage an OpenTK application.

5.73.2 Member Function Documentation

5.73.2.1 static void OpenTK.Toolkit.Init () [static]

Initializes OpenTK. This method is necessary only if you are using OpenTK alongside a different windowing toolkit (e.g. GTK#) and should be the very first method called by your application (i.e. calling this method should be the very first statement executed by the "Main" method).

Some windowing toolkits do not configure the underlying platform correctly or configure it in a way that is incompatible with OpenTK. Calling this method first ensures that OpenTK is given the chance to initialize itself and configure the platform correctly.

5.74 OpenTK.Vector2 Struct Reference

Represents a 2D vector using two single-precision floating-point numbers.

Public Member Functions

- [Vector2](#) (float x, float y)
Constructs a new [Vector2](#).
- [Vector2](#) ([Vector2](#) v)
Constructs a new [Vector2](#) from the given [Vector2](#).
- [Vector2](#) ([Vector3](#) v)
Constructs a new [Vector2](#) from the given [Vector3](#).
- [Vector2](#) ([Vector4](#) v)
Constructs a new [Vector2](#) from the given [Vector4](#).
- void [Add](#) ([Vector2](#) right)
Add the Vector passed as parameter to this instance.
- void [Add](#) (ref [Vector2](#) right)
Add the Vector passed as parameter to this instance.
- void [Sub](#) ([Vector2](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Sub](#) (ref [Vector2](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Mult](#) (float f)
Multiply this instance by a scalar.
- void [Div](#) (float f)
Divide this instance by a scalar.
- void [Normalize](#) ()
Scales the [Vector2](#) to unit length.
- void [NormalizeFast](#) ()
Scales the [Vector2](#) to approximately unit length.

- void [Scale](#) (float sx, float sy)
Scales the current [Vector2](#) by the given amounts.
- void [Scale](#) ([Vector2](#) scale)
Scales this instance by the given parameter.
- void [Scale](#) (ref [Vector2](#) scale)
Scales this instance by the given parameter.
- override string [ToString](#) ()
Returns a System.String that represents the current [Vector2](#).
- override int [GetHashCode](#) ()
Returns the hashcode for this instance.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Vector2](#) other)
Indicates whether the current vector is equal to another vector.

Static Public Member Functions

- static [Vector2](#) [Sub](#) ([Vector2](#) a, [Vector2](#) b)
Subtract one Vector from another.
- static void [Sub](#) (ref [Vector2](#) a, ref [Vector2](#) b, out [Vector2](#) result)
Subtract one Vector from another.
- static [Vector2](#) [Mult](#) ([Vector2](#) a, float f)
Multiply a vector and a scalar.
- static void [Mult](#) (ref [Vector2](#) a, float f, out [Vector2](#) result)
Multiply a vector and a scalar.
- static [Vector2](#) [Div](#) ([Vector2](#) a, float f)
Divide a vector by a scalar.
- static void [Div](#) (ref [Vector2](#) a, float f, out [Vector2](#) result)
Divide a vector by a scalar.

- static [Vector2 Add](#) ([Vector2](#) a, [Vector2](#) b)
Adds two vectors.
- static void [Add](#) (ref [Vector2](#) a, ref [Vector2](#) b, out [Vector2](#) result)
Adds two vectors.
- static [Vector2 Subtract](#) ([Vector2](#) a, [Vector2](#) b)
Subtract one Vector from another.
- static void [Subtract](#) (ref [Vector2](#) a, ref [Vector2](#) b, out [Vector2](#) result)
Subtract one Vector from another.
- static [Vector2 Multiply](#) ([Vector2](#) vector, float scale)
Multiplies a vector by a scalar.
- static void [Multiply](#) (ref [Vector2](#) vector, float scale, out [Vector2](#) result)
Multiplies a vector by a scalar.
- static [Vector2 Multiply](#) ([Vector2](#) vector, [Vector2](#) scale)
Multiplies a vector by the components a vector (scale).
- static void [Multiply](#) (ref [Vector2](#) vector, ref [Vector2](#) scale, out [Vector2](#) result)
Multiplies a vector by the components of a vector (scale).
- static [Vector2 Divide](#) ([Vector2](#) vector, float scale)
Divides a vector by a scalar.
- static void [Divide](#) (ref [Vector2](#) vector, float scale, out [Vector2](#) result)
Divides a vector by a scalar.
- static [Vector2 Divide](#) ([Vector2](#) vector, [Vector2](#) scale)
Divides a vector by the components of a vector (scale).
- static void [Divide](#) (ref [Vector2](#) vector, ref [Vector2](#) scale, out [Vector2](#) result)
Divide a vector by the components of a vector (scale).
- static [Vector2 ComponentMin](#) ([Vector2](#) a, [Vector2](#) b)
Calculate the component-wise minimum of two vectors.
- static void [ComponentMin](#) (ref [Vector2](#) a, ref [Vector2](#) b, out [Vector2](#) result)
Calculate the component-wise minimum of two vectors.

- static [Vector2 ComponentMax](#) ([Vector2](#) a, [Vector2](#) b)
Calculate the component-wise maximum of two vectors.
- static void [ComponentMax](#) (ref [Vector2](#) a, ref [Vector2](#) b, out [Vector2](#) result)
Calculate the component-wise maximum of two vectors.
- static [Vector2 Min](#) ([Vector2](#) left, [Vector2](#) right)
Returns the [Vector3](#) with the minimum magnitude.
- static [Vector2 Max](#) ([Vector2](#) left, [Vector2](#) right)
Returns the [Vector3](#) with the minimum magnitude.
- static [Vector2 Clamp](#) ([Vector2](#) vec, [Vector2](#) min, [Vector2](#) max)
Clamp a vector to the given minimum and maximum vectors.
- static void [Clamp](#) (ref [Vector2](#) vec, ref [Vector2](#) min, ref [Vector2](#) max, out [Vector2](#) result)
Clamp a vector to the given minimum and maximum vectors.
- static [Vector2 Normalize](#) ([Vector2](#) vec)
Scale a vector to unit length.
- static void [Normalize](#) (ref [Vector2](#) vec, out [Vector2](#) result)
Scale a vector to unit length.
- static [Vector2 NormalizeFast](#) ([Vector2](#) vec)
Scale a vector to approximately unit length.
- static void [NormalizeFast](#) (ref [Vector2](#) vec, out [Vector2](#) result)
Scale a vector to approximately unit length.
- static float [Dot](#) ([Vector2](#) left, [Vector2](#) right)
Calculate the dot (scalar) product of two vectors.
- static void [Dot](#) (ref [Vector2](#) left, ref [Vector2](#) right, out float result)
Calculate the dot (scalar) product of two vectors.
- static [Vector2 Lerp](#) ([Vector2](#) a, [Vector2](#) b, float blend)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static void [Lerp](#) (ref [Vector2](#) a, ref [Vector2](#) b, float blend, out [Vector2](#) result)
Returns a new Vector that is the linear blend of the 2 given Vectors.

- static [Vector2 BaryCentric](#) ([Vector2](#) a, [Vector2](#) b, [Vector2](#) c, float u, float v)
Interpolate 3 Vectors using Barycentric coordinates.
- static void [BaryCentric](#) (ref [Vector2](#) a, ref [Vector2](#) b, ref [Vector2](#) c, float u, float v, out [Vector2](#) result)
Interpolate 3 Vectors using Barycentric coordinates.
- static [Vector2 Transform](#) ([Vector2](#) vec, [Quaternion](#) quat)
Transforms a vector by a quaternion rotation.
- static void [Transform](#) (ref [Vector2](#) vec, ref [Quaternion](#) quat, out [Vector2](#) result)
Transforms a vector by a quaternion rotation.
- static [Vector2 operator+](#) ([Vector2](#) left, [Vector2](#) right)
Adds the specified instances.
- static [Vector2 operator-](#) ([Vector2](#) left, [Vector2](#) right)
Subtracts the specified instances.
- static [Vector2 operator-](#) ([Vector2](#) vec)
Negates the specified instance.
- static [Vector2 operator*](#) ([Vector2](#) vec, float scale)
Multiplies the specified instance by a scalar.
- static [Vector2 operator*](#) (float scale, [Vector2](#) vec)
Multiplies the specified instance by a scalar.
- static [Vector2 operator/](#) ([Vector2](#) vec, float scale)
Divides the specified instance by a scalar.
- static bool [operator==](#) ([Vector2](#) left, [Vector2](#) right)
Compares the specified instances for equality.
- static bool [operator!=](#) ([Vector2](#) left, [Vector2](#) right)
Compares the specified instances for inequality.

Public Attributes

- float [X](#)
The X component of the [Vector2](#).
- float [Y](#)
The Y component of the [Vector2](#).

Static Public Attributes

- static readonly [Vector2](#) [UnitX](#) = new [Vector2](#)(1, 0)
Defines a unit-length [Vector2](#) that points towards the X-axis.
- static readonly [Vector2](#) [UnitY](#) = new [Vector2](#)(0, 1)
Defines a unit-length [Vector2](#) that points towards the Y-axis.
- static readonly [Vector2](#) [Zero](#) = new [Vector2](#)(0, 0)
Defines a zero-length [Vector2](#).
- static readonly [Vector2](#) [One](#) = new [Vector2](#)(1, 1)
Defines an instance with all components set to 1.
- static readonly int [SizeInBytes](#) = Marshal.SizeOf(new [Vector2](#)())
Defines the size of the [Vector2](#) struct in bytes.

Properties

- float [Length](#) [get]
Gets the length (magnitude) of the vector.
- float [LengthFast](#) [get]
Gets an approximation of the vector length (magnitude).
- float [LengthSquared](#) [get]
Gets the square of the vector length (magnitude).
- [Vector2](#) [PerpendicularRight](#) [get]
Gets the perpendicular vector on the right side of this vector.
- [Vector2](#) [PerpendicularLeft](#) [get]

Gets the perpendicular vector on the left side of this vector.

5.74.1 Detailed Description

Represents a 2D vector using two single-precision floating-point numbers. The [Vector2](#) structure is suitable for interoperation with unmanaged code requiring two consecutive floats.

5.74.2 Constructor & Destructor Documentation

5.74.2.1 `OpenTK.Vector2.Vector2 (float x, float y)`

Constructs a new [Vector2](#).

Parameters

- x* The x coordinate of the net [Vector2](#).
- y* The y coordinate of the net [Vector2](#).

5.74.2.2 `OpenTK.Vector2.Vector2 (Vector2 v)`

Constructs a new [Vector2](#) from the given [Vector2](#).

Parameters

- v* The [Vector2](#) to copy components from.

5.74.2.3 `OpenTK.Vector2.Vector2 (Vector3 v)`

Constructs a new [Vector2](#) from the given [Vector3](#).

Parameters

- v* The [Vector3](#) to copy components from. Z is discarded.

5.74.2.4 `OpenTK.Vector2.Vector2 (Vector4 v)`

Constructs a new [Vector2](#) from the given [Vector4](#).

Parameters

- v* The [Vector4](#) to copy components from. Z and W are discarded.

5.74.3 Member Function Documentation

5.74.3.1 void OpenTK.Vector2.Add (Vector2 *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

5.74.3.2 void OpenTK.Vector2.Add (ref Vector2 *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

5.74.3.3 static Vector2 OpenTK.Vector2.Add (Vector2 *a*, Vector2 *b*) [static]

Adds two vectors.

Parameters

a Left operand.

b Right operand.

Returns

Result of operation.

5.74.3.4 static void OpenTK.Vector2.Add (ref Vector2 *a*, ref Vector2 *b*, out Vector2 *result*) [static]

Adds two vectors.

Parameters

a Left operand.

b Right operand.

result Result of operation.

5.74.3.5 static Vector2 OpenTK.Vector2.BaryCentric (Vector2 *a*, Vector2 *b*, Vector2 *c*, float *u*, float *v*) [static]

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector
- b* Second input Vector
- c* Third input Vector
- u* First Barycentric Coordinate
- v* Second Barycentric Coordinate

Returns

a when $u=v=0$, *b* when $u=1,v=0$, *c* when $u=0,v=1$, and a linear combination of *a*,*b*,*c* otherwise

5.74.3.6 static void OpenTK.Vector2.BaryCentric (ref Vector2 *a*, ref Vector2 *b*, ref Vector2 *c*, float *u*, float *v*, out Vector2 *result*) [static]

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector.
- b* Second input Vector.
- c* Third input Vector.
- u* First Barycentric Coordinate.
- v* Second Barycentric Coordinate.
- result* Output Vector. *a* when $u=v=0$, *b* when $u=1,v=0$, *c* when $u=0,v=1$, and a linear combination of *a*,*b*,*c* otherwise

5.74.3.7 static Vector2 OpenTK.Vector2.Clamp (Vector2 *vec*, Vector2 *min*, Vector2 *max*) [static]

Clamp a vector to the given minimum and maximum vectors.

Parameters

- vec* [Input](#) vector

min Minimum vector
max Maximum vector

Returns

The clamped vector

5.74.3.8 `static void OpenTK.Vector2.Clamp (ref Vector2 vec, ref Vector2 min, ref Vector2 max, out Vector2 result) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector
min Minimum vector
max Maximum vector
result The clamped vector

5.74.3.9 `static Vector2 OpenTK.Vector2.ComponentMax (Vector2 a, Vector2 b) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand
b Second operand

Returns

The component-wise maximum

5.74.3.10 `static void OpenTK.Vector2.ComponentMax (ref Vector2 a, ref Vector2 b, out Vector2 result) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand
b Second operand
result The component-wise maximum

5.74.3.11 `static Vector2 OpenTK.Vector2.ComponentMin (Vector2 a, Vector2 b) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

- a* First operand
- b* Second operand

Returns

The component-wise minimum

5.74.3.12 `static void OpenTK.Vector2.ComponentMin (ref Vector2 a, ref Vector2 b, out Vector2 result) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

- a* First operand
- b* Second operand
- result* The component-wise minimum

5.74.3.13 `static void OpenTK.Vector2.Div (ref Vector2 a, float f, out Vector2 result) [static]`

Divide a vector by a scalar.

Parameters

- a* Vector operand
- f* Scalar operand
- result* Result of the division

5.74.3.14 `static Vector2 OpenTK.Vector2.Div (Vector2 a, float f) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand

f Scalar operand

Returns

Result of the division

5.74.3.15 void OpenTK.Vector2.Div (float *f*)

Divide this instance by a scalar.

Parameters

f Scalar operand.

**5.74.3.16 static Vector2 OpenTK.Vector2.Divide (Vector2 *vector*, float *scale*)
[static]**

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

**5.74.3.17 static void OpenTK.Vector2.Divide (ref Vector2 *vector*, float *scale*,
out Vector2 *result*) [static]**

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.74.3.18 `static void OpenTK.Vector2.Divide (ref Vector2 vector, ref Vector2 scale, out Vector2 result) [static]`

Divide a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.74.3.19 `static Vector2 OpenTK.Vector2.Divide (Vector2 vector, Vector2 scale) [static]`

Divides a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.74.3.20 `static void OpenTK.Vector2.Dot (ref Vector2 left, ref Vector2 right, out float result) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand
result The dot product of the two inputs

5.74.3.21 `static float OpenTK.Vector2.Dot (Vector2 left, Vector2 right) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The dot product of the two inputs

5.74.3.22 bool OpenTK.Vector2.Equals (Vector2 *other*)

Indicates whether the current vector is equal to another vector.

Parameters

other A vector to compare with this vector.

Returns

true if the current vector is equal to the vector parameter; otherwise, false.

5.74.3.23 override bool OpenTK.Vector2.Equals (object *obj*)

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.74.3.24 override int OpenTK.Vector2.GetHashCode ()

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.74.3.25 `static void OpenTK.Vector2.Lerp (ref Vector2 a, ref Vector2 b, float blend, out Vector2 result) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

result a when blend=0, b when blend=1, and a linear combination otherwise

5.74.3.26 `static Vector2 OpenTK.Vector2.Lerp (Vector2 a, Vector2 b, float blend) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

Returns

a when blend=0, b when blend=1, and a linear combination otherwise

5.74.3.27 `static Vector2 OpenTK.Vector2.Max (Vector2 left, Vector2 right) [static]`

Returns the [Vector3](#) with the minimum magnitude.

Parameters

left Left operand

right Right operand

Returns

The minimum [Vector3](#)

**5.74.3.28 static Vector2 OpenTK.Vector2.Min (Vector2 *left*, Vector2 *right*)
[static]**

Returns the [Vector3](#) with the minimum magnitude.

Parameters

left Left operand
right Right operand

Returns

The minimum [Vector3](#)

5.74.3.29 static void OpenTK.Vector2.Mult (ref Vector2 *a*, float *f*, out Vector2 *result*) [static]

Multiply a vector and a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the multiplication

5.74.3.30 void OpenTK.Vector2.Mult (float *f*)

Multiply this instance by a scalar.

Parameters

f Scalar operand.

**5.74.3.31 static Vector2 OpenTK.Vector2.Mult (Vector2 *a*, float *f*)
[static]**

Multiply a vector and a scalar.

Parameters

a Vector operand
f Scalar operand

Returns

Result of the multiplication

5.74.3.32 `static Vector2 OpenTK.Vector2.Multiply (Vector2 vector, float scale) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.74.3.33 `static void OpenTK.Vector2.Multiply (ref Vector2 vector, ref Vector2 scale, out Vector2 result) [static]`

Multiplies a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.74.3.34 `static Vector2 OpenTK.Vector2.Multiply (Vector2 vector, Vector2 scale) [static]`

Multiplies a vector by the components a vector (scale).

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.74.3.35 `static void OpenTK.Vector2.Multiply (ref Vector2 vector, float scale, out Vector2 result) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.74.3.36 `static Vector2 OpenTK.Vector2.Normalize (Vector2 vec) [static]`

Scale a vector to unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.74.3.37 `static void OpenTK.Vector2.Normalize (ref Vector2 vec, out Vector2 result) [static]`

Scale a vector to unit length.

Parameters

vec The input vector

result The normalized vector

5.74.3.38 `void OpenTK.Vector2.Normalize ()`

Scales the [Vector2](#) to unit length.

5.74.3.39 `void OpenTK.Vector2.NormalizeFast ()`

Scales the [Vector2](#) to approximately unit length.

5.74.3.40 `static void OpenTK.Vector2.NormalizeFast (ref Vector2 vec, out Vector2 result) [static]`

Scale a vector to approximately unit length.

Parameters

vec The input vector
result The normalized vector

5.74.3.41 `static Vector2 OpenTK.Vector2.NormalizeFast (Vector2 vec) [static]`

Scale a vector to approximately unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.74.3.42 `static bool OpenTK.Vector2.operator!= (Vector2 left, Vector2 right) [static]`

Compares the specified instances for inequality.

Parameters

left Left operand.
right Right operand.

Returns

True if both instances are not equal; false otherwise.

5.74.3.43 `static Vector2 OpenTK.Vector2.operator* (float scale, Vector2 vec) [static]`

Multiplies the specified instance by a scalar.

Parameters

scale Left operand.

vec Right operand.

Returns

Result of multiplication.

5.74.3.44 `static Vector2 OpenTK.Vector2.operator* (Vector2 vec, float scale) [static]`

Multiplies the specified instance by a scalar.

Parameters

vec Left operand.

scale Right operand.

Returns

Result of multiplication.

5.74.3.45 `static Vector2 OpenTK.Vector2.operator+ (Vector2 left, Vector2 right) [static]`

Adds the specified instances.

Parameters

left Left operand.

right Right operand.

Returns

Result of addition.

5.74.3.46 `static Vector2 OpenTK.Vector2.operator- (Vector2 vec) [static]`

Negates the specified instance.

Parameters

vec Operand.

Returns

Result of negation.

5.74.3.47 `static Vector2 OpenTK.Vector2.operator- (Vector2 left, Vector2 right) [static]`

Subtracts the specified instances.

Parameters

left Left operand.

right Right operand.

Returns

Result of subtraction.

5.74.3.48 `static Vector2 OpenTK.Vector2.operator/ (Vector2 vec, float scale) [static]`

Divides the specified instance by a scalar.

Parameters

vec Left operand

scale Right operand

Returns

Result of the division.

5.74.3.49 `static bool OpenTK.Vector2.operator==(Vector2 left, Vector2 right) [static]`

Compares the specified instances for equality.

Parameters

left Left operand.

right Right operand.

Returns

True if both instances are equal; false otherwise.

5.74.3.50 void OpenTK.Vector2.Scale (float *sx*, float *sy*)

Scales the current [Vector2](#) by the given amounts.

Parameters

sx The scale of the X component.

sy The scale of the Y component.

5.74.3.51 void OpenTK.Vector2.Scale (Vector2 *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.74.3.52 void OpenTK.Vector2.Scale (ref Vector2 *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.74.3.53 void OpenTK.Vector2.Sub (ref Vector2 *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.74.3.54 static void OpenTK.Vector2.Sub (ref Vector2 *a*, ref Vector2 *b*, out Vector2 *result*) [static]

Subtract one Vector from another.

Parameters

a First operand

b Second operand

result Result of subtraction

**5.74.3.55 static Vector2 OpenTK.Vector2.Sub (Vector2 *a*, Vector2 *b*)
[static]**

Subtract one Vector from another.

Parameters

- a* First operand
- b* Second operand

Returns

Result of subtraction

5.74.3.56 void OpenTK.Vector2.Sub (Vector2 *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

- right* Right operand. This parameter is only read from.

**5.74.3.57 static Vector2 OpenTK.Vector2.Subtract (Vector2 *a*, Vector2 *b*)
[static]**

Subtract one Vector from another.

Parameters

- a* First operand
- b* Second operand

Returns

Result of subtraction

**5.74.3.58 static void OpenTK.Vector2.Subtract (ref Vector2 *a*, ref Vector2 *b*,
out Vector2 *result*) [static]**

Subtract one Vector from another.

Parameters

- a* First operand
- b* Second operand
- result* Result of subtraction

5.74.3.59 override string OpenTK.Vector2.ToString ()

Returns a System.String that represents the current [Vector2](#).

Returns

5.74.3.60 static void OpenTK.Vector2.Transform (ref Vector2 *vec*, ref Quaternion *quat*, out Vector2 *result*) [static]

Transforms a vector by a quaternion rotation.

Parameters

- vec* The vector to transform.
- quat* The quaternion to rotate the vector by.
- result* The result of the operation.

5.74.3.61 static Vector2 OpenTK.Vector2.Transform (Vector2 *vec*, Quaternion *quat*) [static]

Transforms a vector by a quaternion rotation.

Parameters

- vec* The vector to transform.
- quat* The quaternion to rotate the vector by.

Returns

- The result of the operation.

5.74.4 Member Data Documentation**5.74.4.1** readonly Vector2 OpenTK.Vector2.One = new Vector2(1, 1) [static]

Defines an instance with all components set to 1.

5.74.4.2 readonly int OpenTK.Vector2.SizeInBytes = Marshal.SizeOf(new Vector2()) [static]

Defines the size of the [Vector2](#) struct in bytes.

5.74.4.3 readonly Vector2 OpenTK.Vector2.UnitX = new Vector2(1, 0) [static]

Defines a unit-length [Vector2](#) that points towards the X-axis.

5.74.4.4 readonly Vector2 OpenTK.Vector2.UnitY = new Vector2(0, 1) [static]

Defines a unit-length [Vector2](#) that points towards the Y-axis.

5.74.4.5 float OpenTK.Vector2.X

The X component of the [Vector2](#).

5.74.4.6 float OpenTK.Vector2.Y

The Y component of the [Vector2](#).

5.74.4.7 readonly Vector2 OpenTK.Vector2.Zero = new Vector2(0, 0) [static]

Defines a zero-length [Vector2](#).

5.74.5 Property Documentation

5.74.5.1 float OpenTK.Vector2.Length [get]

Gets the length (magnitude) of the vector.

[LengthFast](#)

See also

[LengthSquared](#)

5.74.5.2 float OpenTK.Vector2.LengthFast [get]

Gets an approximation of the vector length (magnitude).

This property uses an approximation of the square root function to calculate vector magnitude, with an upper error bound of 0.001.

[Length](#)

See also

[LengthSquared](#)

5.74.5.3 float OpenTK.Vector2.LengthSquared [get]

Gets the square of the vector length (magnitude).

This property avoids the costly square root operation required by the Length property. This makes it more suitable for comparisons.

[Length](#)

See also

[LengthFast](#)

5.74.5.4 Vector2 OpenTK.Vector2.PerpendicularLeft [get]

Gets the perpendicular vector on the left side of this vector.

5.74.5.5 Vector2 OpenTK.Vector2.PerpendicularRight [get]

Gets the perpendicular vector on the right side of this vector.

5.75 OpenTK.Vector2d Struct Reference

Represents a 2D vector using two double-precision floating-point numbers.

Public Member Functions

- [Vector2d](#) (double x, double y)
Constructs left vector with the given coordinates.
- void [Add](#) ([Vector2d](#) right)
Add the Vector passed as parameter to this instance.
- void [Add](#) (ref [Vector2d](#) right)
Add the Vector passed as parameter to this instance.
- void [Sub](#) ([Vector2d](#) right)

Subtract the Vector passed as parameter from this instance.

- void [Sub](#) (ref [Vector2d](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Mult](#) (double f)
Multiply this instance by a scalar.
- void [Div](#) (double f)
Divide this instance by a scalar.
- void [Normalize](#) ()
Scales the [Vector2](#) to unit length.
- void [Scale](#) (double sx, double sy)
Scales the current [Vector2](#) by the given amounts.
- void [Scale](#) ([Vector2d](#) scale)
Scales this instance by the given parameter.
- void [Scale](#) (ref [Vector2d](#) scale)
Scales this instance by the given parameter.
- override string [ToString](#) ()
Returns a System.String that represents the current instance.
- override int [GetHashCode](#) ()
Returns the hashcode for this instance.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Vector2d](#) other)
Indicates whether the current vector is equal to another vector.

Static Public Member Functions

- static [Vector2d](#) [Sub](#) ([Vector2d](#) a, [Vector2d](#) b)
Subtract one Vector from another.
- static void [Sub](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, out [Vector2d](#) result)

Subtract one Vector from another.

- static [Vector2d Mult](#) ([Vector2d](#) a, double d)
Multiply a vector and a scalar.
- static void [Mult](#) (ref [Vector2d](#) a, double d, out [Vector2d](#) result)
Multiply a vector and a scalar.
- static [Vector2d Div](#) ([Vector2d](#) a, double d)
Divide a vector by a scalar.
- static void [Div](#) (ref [Vector2d](#) a, double d, out [Vector2d](#) result)
Divide a vector by a scalar.
- static [Vector2d Add](#) ([Vector2d](#) a, [Vector2d](#) b)
Adds two vectors.
- static void [Add](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, out [Vector2d](#) result)
Adds two vectors.
- static [Vector2d Subtract](#) ([Vector2d](#) a, [Vector2d](#) b)
Subtract one Vector from another.
- static void [Subtract](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, out [Vector2d](#) result)
Subtract one Vector from another.
- static [Vector2d Multiply](#) ([Vector2d](#) vector, double scale)
Multiplies a vector by a scalar.
- static void [Multiply](#) (ref [Vector2d](#) vector, double scale, out [Vector2d](#) result)
Multiplies a vector by a scalar.
- static [Vector2d Multiply](#) ([Vector2d](#) vector, [Vector2d](#) scale)
Multiplies a vector by the components a vector (scale).
- static void [Multiply](#) (ref [Vector2d](#) vector, ref [Vector2d](#) scale, out [Vector2d](#) result)
Multiplies a vector by the components of a vector (scale).
- static [Vector2d Divide](#) ([Vector2d](#) vector, double scale)
Divides a vector by a scalar.
- static void [Divide](#) (ref [Vector2d](#) vector, double scale, out [Vector2d](#) result)

Divides a vector by a scalar.

- static [Vector2d Divide](#) ([Vector2d](#) vector, [Vector2d](#) scale)
Divides a vector by the components of a vector (scale).
- static void [Divide](#) (ref [Vector2d](#) vector, ref [Vector2d](#) scale, out [Vector2d](#) result)
Divide a vector by the components of a vector (scale).
- static [Vector2d Min](#) ([Vector2d](#) a, [Vector2d](#) b)
Calculate the component-wise minimum of two vectors.
- static void [Min](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, out [Vector2d](#) result)
Calculate the component-wise minimum of two vectors.
- static [Vector2d Max](#) ([Vector2d](#) a, [Vector2d](#) b)
Calculate the component-wise maximum of two vectors.
- static void [Max](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, out [Vector2d](#) result)
Calculate the component-wise maximum of two vectors.
- static [Vector2d Clamp](#) ([Vector2d](#) vec, [Vector2d](#) min, [Vector2d](#) max)
Clamp a vector to the given minimum and maximum vectors.
- static void [Clamp](#) (ref [Vector2d](#) vec, ref [Vector2d](#) min, ref [Vector2d](#) max, out [Vector2d](#) result)
Clamp a vector to the given minimum and maximum vectors.
- static [Vector2d Normalize](#) ([Vector2d](#) vec)
Scale a vector to unit length.
- static void [Normalize](#) (ref [Vector2d](#) vec, out [Vector2d](#) result)
Scale a vector to unit length.
- static [Vector2d NormalizeFast](#) ([Vector2d](#) vec)
Scale a vector to approximately unit length.
- static void [NormalizeFast](#) (ref [Vector2d](#) vec, out [Vector2d](#) result)
Scale a vector to approximately unit length.
- static double [Dot](#) ([Vector2d](#) left, [Vector2d](#) right)
Calculate the dot (scalar) product of two vectors.
- static void [Dot](#) (ref [Vector2d](#) left, ref [Vector2d](#) right, out double result)

Calculate the dot (scalar) product of two vectors.

- static [Vector2d Lerp](#) ([Vector2d](#) a, [Vector2d](#) b, double blend)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static void [Lerp](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, double blend, out [Vector2d](#) result)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static [Vector2d BaryCentric](#) ([Vector2d](#) a, [Vector2d](#) b, [Vector2d](#) c, double u, double v)
Interpolate 3 Vectors using Barycentric coordinates.
- static void [BaryCentric](#) (ref [Vector2d](#) a, ref [Vector2d](#) b, ref [Vector2d](#) c, double u, double v, out [Vector2d](#) result)
Interpolate 3 Vectors using Barycentric coordinates.
- static [Vector2d Transform](#) ([Vector2d](#) vec, [Quaterniond](#) quat)
Transforms a vector by a quaternion rotation.
- static void [Transform](#) (ref [Vector2d](#) vec, ref [Quaterniond](#) quat, out [Vector2d](#) result)
Transforms a vector by a quaternion rotation.
- static [Vector2d operator+](#) ([Vector2d](#) left, [Vector2d](#) right)
Adds two instances.
- static [Vector2d operator-](#) ([Vector2d](#) left, [Vector2d](#) right)
Subtracts two instances.
- static [Vector2d operator-](#) ([Vector2d](#) vec)
Negates an instance.
- static [Vector2d operator*](#) ([Vector2d](#) vec, double f)
Multiplies an instance by a scalar.
- static [Vector2d operator*](#) (double f, [Vector2d](#) vec)
Multiply an instance by a scalar.
- static [Vector2d operator/](#) ([Vector2d](#) vec, double f)
Divides an instance by a scalar.
- static bool [operator==](#) ([Vector2d](#) left, [Vector2d](#) right)

Compares two instances for equality.

- static bool `operator!=` (`Vector2d` left, `Vector2d` right)

Compares two instances for inequality.

- static `operator Vector2d` (`Vector2` v2)

Converts `OpenTK.Vector2` to `OpenTK.Vector2d`.

- static `operator Vector2` (`Vector2d` v2d)

Converts `OpenTK.Vector2d` to `OpenTK.Vector2`.

Public Attributes

- double `X`

The X coordinate of this instance.

- double `Y`

The Y coordinate of this instance.

Static Public Attributes

- static `Vector2d UnitX` = new `Vector2d`(1, 0)

Defines a unit-length `Vector2d` that points towards the X-axis.

- static `Vector2d UnitY` = new `Vector2d`(0, 1)

Defines a unit-length `Vector2d` that points towards the Y-axis.

- static `Vector2d Zero` = new `Vector2d`(0, 0)

Defines a zero-length `Vector2d`.

- static readonly `Vector2d One` = new `Vector2d`(1, 1)

Defines an instance with all components set to 1.

- static readonly int `SizeInBytes` = Marshal.SizeOf(new `Vector2d`())

Defines the size of the `Vector2d` struct in bytes.

Properties

- double [Length](#) [get]
Gets the length (magnitude) of the vector.
- double [LengthSquared](#) [get]
Gets the square of the vector length (magnitude).
- [Vector2d PerpendicularRight](#) [get]
Gets the perpendicular vector on the right side of this vector.
- [Vector2d PerpendicularLeft](#) [get]
Gets the perpendicular vector on the left side of this vector.

5.75.1 Detailed Description

Represents a 2D vector using two double-precision floating-point numbers.

5.75.2 Constructor & Destructor Documentation

5.75.2.1 OpenTK.Vector2d.Vector2d (double *x*, double *y*)

Constructs left vector with the given coordinates.

Parameters

- x* The X coordinate.
- y* The Y coordinate.

5.75.3 Member Function Documentation

5.75.3.1 void OpenTK.Vector2d.Add (Vector2d *right*)

Add the Vector passed as parameter to this instance.

Parameters

- right* Right operand. This parameter is only read from.

5.75.3.2 void OpenTK.Vector2d.Add (ref Vector2d *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

**5.75.3.3 static Vector2d OpenTK.Vector2d.Add (Vector2d *a*, Vector2d *b*)
[static]**

Adds two vectors.

Parameters

a Left operand.

b Right operand.

Returns

Result of operation.

**5.75.3.4 static void OpenTK.Vector2d.Add (ref Vector2d *a*, ref Vector2d *b*,
out Vector2d *result*) [static]**

Adds two vectors.

Parameters

a Left operand.

b Right operand.

result Result of operation.

**5.75.3.5 static Vector2d OpenTK.Vector2d.BaryCentric (Vector2d *a*,
Vector2d *b*, Vector2d *c*, double *u*, double *v*) [static]**

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

a First input Vector

b Second input Vector

- c* Third input Vector
- u* First Barycentric Coordinate
- v* Second Barycentric Coordinate

Returns

a when $u=v=0$, b when $u=1,v=0$, c when $u=0,v=1$, and a linear combination of a,b,c otherwise

5.75.3.6 `static void OpenTK.Vector2d.BaryCentric (ref Vector2d a, ref Vector2d b, ref Vector2d c, double u, double v, out Vector2d result) [static]`

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector.
- b* Second input Vector.
- c* Third input Vector.
- u* First Barycentric Coordinate.
- v* Second Barycentric Coordinate.
- result* Output Vector. a when $u=v=0$, b when $u=1,v=0$, c when $u=0,v=1$, and a linear combination of a,b,c otherwise

5.75.3.7 `static Vector2d OpenTK.Vector2d.Clamp (Vector2d vec, Vector2d min, Vector2d max) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

- vec* [Input](#) vector
- min* Minimum vector
- max* Maximum vector

Returns

The clamped vector

5.75.3.8 `static void OpenTK.Vector2d.Clamp (ref Vector2d vec, ref Vector2d min, ref Vector2d max, out Vector2d result) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector
min Minimum vector
max Maximum vector
result The clamped vector

5.75.3.9 `static void OpenTK.Vector2d.Div (ref Vector2d a, double d, out Vector2d result) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
d Scalar operand
result Result of the division

5.75.3.10 `void OpenTK.Vector2d.Div (double f)`

Divide this instance by a scalar.

Parameters

f Scalar operand.

5.75.3.11 `static Vector2d OpenTK.Vector2d.Div (Vector2d a, double d) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
d Scalar operand

Returns

Result of the division

5.75.3.12 static Vector2d OpenTK.Vector2d.Divide (Vector2d *vector*, double *scale*) [static]

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.75.3.13 static void OpenTK.Vector2d.Divide (ref Vector2d *vector*, double *scale*, out Vector2d *result*) [static]

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.75.3.14 static Vector2d OpenTK.Vector2d.Divide (Vector2d *vector*, Vector2d *scale*) [static]

Divides a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.75.3.15 `static void OpenTK.Vector2d.Divide (ref Vector2d vector, ref Vector2d scale, out Vector2d result) [static]`

Divide a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.75.3.16 `static double OpenTK.Vector2d.Dot (Vector2d left, Vector2d right) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The dot product of the two inputs

5.75.3.17 `static void OpenTK.Vector2d.Dot (ref Vector2d left, ref Vector2d right, out double result) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand
result The dot product of the two inputs

5.75.3.18 `override bool OpenTK.Vector2d.Equals (object obj)`

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.75.3.19 bool OpenTK.Vector2d.Equals (Vector2d *other*)

Indicates whether the current vector is equal to another vector.

Parameters

other A vector to compare with this vector.

Returns

true if the current vector is equal to the vector parameter; otherwise, false.

5.75.3.20 override int OpenTK.Vector2d.GetHashCode ()

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.75.3.21 static Vector2d OpenTK.Vector2d.Lerp (Vector2d *a*, Vector2d *b*, double *blend*) [static]

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

Returns

a when blend=0, b when blend=1, and a linear combination otherwise

5.75.3.22 `static void OpenTK.Vector2d.Lerp (ref Vector2d a, ref Vector2d b, double blend, out Vector2d result) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

result a when blend=0, b when blend=1, and a linear combination otherwise

5.75.3.23 `static void OpenTK.Vector2d.Max (ref Vector2d a, ref Vector2d b, out Vector2d result) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

result The component-wise maximum

5.75.3.24 `static Vector2d OpenTK.Vector2d.Max (Vector2d a, Vector2d b) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

Returns

The component-wise maximum

5.75.3.25 `static Vector2d OpenTK.Vector2d.Min (Vector2d a, Vector2d b) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

- a* First operand
- b* Second operand

Returns

The component-wise minimum

5.75.3.26 `static void OpenTK.Vector2d.Min (ref Vector2d a, ref Vector2d b,
out Vector2d result) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

- a* First operand
- b* Second operand
- result* The component-wise minimum

5.75.3.27 `static void OpenTK.Vector2d.Mult (ref Vector2d a, double d, out
Vector2d result) [static]`

Multiply a vector and a scalar.

Parameters

- a* Vector operand
- d* Scalar operand
- result* Result of the multiplication

5.75.3.28 `void OpenTK.Vector2d.Mult (double f)`

Multiply this instance by a scalar.

Parameters

- f* Scalar operand.

**5.75.3.29 static Vector2d OpenTK.Vector2d.Mult (Vector2d *a*, double *d*)
[static]**

Multiply a vector and a scalar.

Parameters

a Vector operand

d Scalar operand

Returns

Result of the multiplication

**5.75.3.30 static Vector2d OpenTK.Vector2d.Multiply (Vector2d *vector*,
double *scale*) [static]**

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

**5.75.3.31 static void OpenTK.Vector2d.Multiply (ref Vector2d *vector*, double
scale, out Vector2d *result*) [static]**

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.75.3.32 `static Vector2d OpenTK.Vector2d.Multiply (Vector2d vector,
Vector2d scale) [static]`

Multiplies a vector by the components a vector (scale).

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.75.3.33 `static void OpenTK.Vector2d.Multiply (ref Vector2d vector, ref
Vector2d scale, out Vector2d result) [static]`

Multiplies a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.75.3.34 `static Vector2d OpenTK.Vector2d.Normalize (Vector2d vec)
[static]`

Scale a vector to unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.75.3.35 `static void OpenTK.Vector2d.Normalize (ref Vector2d vec, out
Vector2d result) [static]`

Scale a vector to unit length.

Parameters

vec The input vector
result The normalized vector

5.75.3.36 void OpenTK.Vector2d.Normalize ()

Scales the [Vector2](#) to unit length.

**5.75.3.37 static Vector2d OpenTK.Vector2d.NormalizeFast (Vector2d vec)
[static]**

Scale a vector to approximately unit length.

Parameters

vec The input vector

Returns

The normalized vector

**5.75.3.38 static void OpenTK.Vector2d.NormalizeFast (ref Vector2d vec, out
Vector2d result) [static]**

Scale a vector to approximately unit length.

Parameters

vec The input vector
result The normalized vector

**5.75.3.39 static OpenTK.Vector2d.operator Vector2 (Vector2d v2d)
[explicit, static]**

Converts [OpenTK.Vector2d](#) to [OpenTK.Vector2](#).

Parameters

v2d The [Vector2d](#) to convert.

Returns

The resulting [Vector2](#).

5.75.3.40 `static OpenTK.Vector2d.operator Vector2d (Vector2 v2)
[explicit, static]`

Converts [OpenTK.Vector2](#) to [OpenTK.Vector2d](#).

Parameters

v2 The [Vector2](#) to convert.

Returns

The resulting [Vector2d](#).

5.75.3.41 `static bool OpenTK.Vector2d.operator!= (Vector2d left, Vector2d
right) [static]`

Compares two instances for inequality.

Parameters

left The left instance.

right The right instance.

Returns

True, if the instances are not equal; false otherwise.

5.75.3.42 `static Vector2d OpenTK.Vector2d.operator* (Vector2d vec, double
f) [static]`

Multiplies an instance by a scalar.

Parameters

vec The instance.

f The scalar.

Returns

The result of the operation.

5.75.3.43 `static Vector2d OpenTK.Vector2d.operator* (double f, Vector2d vec) [static]`

Multiply an instance by a scalar.

Parameters

f The scalar.

vec The instance.

Returns

The result of the operation.

5.75.3.44 `static Vector2d OpenTK.Vector2d.operator+ (Vector2d left, Vector2d right) [static]`

Adds two instances.

Parameters

left The left instance.

right The right instance.

Returns

The result of the operation.

5.75.3.45 `static Vector2d OpenTK.Vector2d.operator- (Vector2d vec) [static]`

Negates an instance.

Parameters

vec The instance.

Returns

The result of the operation.

5.75.3.46 static Vector2d OpenTK.Vector2d.operator- (Vector2d *left*, Vector2d *right*) [static]

Subtracts two instances.

Parameters

left The left instance.

right The right instance.

Returns

The result of the operation.

5.75.3.47 static Vector2d OpenTK.Vector2d.operator/ (Vector2d *vec*, double *f*) [static]

Divides an instance by a scalar.

Parameters

vec The instance.

f The scalar.

Returns

The result of the operation.

5.75.3.48 static bool OpenTK.Vector2d.operator== (Vector2d *left*, Vector2d *right*) [static]

Compares two instances for equality.

Parameters

left The left instance.

right The right instance.

Returns

True, if both instances are equal; false otherwise.

5.75.3.49 void OpenTK.Vector2d.Scale (Vector2d *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.75.3.50 void OpenTK.Vector2d.Scale (ref Vector2d *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.75.3.51 void OpenTK.Vector2d.Scale (double *sx*, double *sy*)

Scales the current [Vector2](#) by the given amounts.

Parameters

sx The scale of the X component.

sy The scale of the Y component.

5.75.3.52 void OpenTK.Vector2d.Sub (ref Vector2d *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

**5.75.3.53 static Vector2d OpenTK.Vector2d.Sub (Vector2d *a*, Vector2d *b*)
[static]**

Subtract one Vector from another.

Parameters

a First operand

b Second operand

Returns

Result of subtraction

5.75.3.54 void OpenTK.Vector2d.Sub (Vector2d *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.75.3.55 static void OpenTK.Vector2d.Sub (ref Vector2d *a*, ref Vector2d *b*, out Vector2d *result*) [static]

Subtract one Vector from another.

Parameters

a First operand

b Second operand

result Result of subtraction

5.75.3.56 static void OpenTK.Vector2d.Subtract (ref Vector2d *a*, ref Vector2d *b*, out Vector2d *result*) [static]

Subtract one Vector from another.

Parameters

a First operand

b Second operand

result Result of subtraction

5.75.3.57 static Vector2d OpenTK.Vector2d.Subtract (Vector2d *a*, Vector2d *b*) [static]

Subtract one Vector from another.

Parameters

- a* First operand
- b* Second operand

Returns

Result of subtraction

5.75.3.58 override string OpenTK.Vector2d.ToString ()

Returns a System.String that represents the current instance.

Returns**5.75.3.59 static void OpenTK.Vector2d.Transform (ref Vector2d *vec*, ref Quaterniond *quat*, out Vector2d *result*) [static]**

Transforms a vector by a quaternion rotation.

Parameters

- vec* The vector to transform.
- quat* The quaternion to rotate the vector by.
- result* The result of the operation.

5.75.3.60 static Vector2d OpenTK.Vector2d.Transform (Vector2d *vec*, Quaterniond *quat*) [static]

Transforms a vector by a quaternion rotation.

Parameters

- vec* The vector to transform.
- quat* The quaternion to rotate the vector by.

Returns

The result of the operation.

5.75.4 Member Data Documentation

5.75.4.1 `readonly Vector2d OpenTK.Vector2d.One = new Vector2d(1, 1)`
`[static]`

Defines an instance with all components set to 1.

5.75.4.2 `readonly int OpenTK.Vector2d.SizeInBytes = Marshal.SizeOf(new Vector2d())` `[static]`

Defines the size of the [Vector2d](#) struct in bytes.

5.75.4.3 `Vector2d OpenTK.Vector2d.UnitX = new Vector2d(1, 0)` `[static]`

Defines a unit-length [Vector2d](#) that points towards the X-axis.

5.75.4.4 `Vector2d OpenTK.Vector2d.UnitY = new Vector2d(0, 1)` `[static]`

Defines a unit-length [Vector2d](#) that points towards the Y-axis.

5.75.4.5 `double OpenTK.Vector2d.X`

The X coordinate of this instance.

5.75.4.6 `double OpenTK.Vector2d.Y`

The Y coordinate of this instance.

5.75.4.7 `Vector2d OpenTK.Vector2d.Zero = new Vector2d(0, 0)` `[static]`

Defines a zero-length [Vector2d](#).

5.75.5 Property Documentation

5.75.5.1 `double OpenTK.Vector2d.Length` `[get]`

Gets the length (magnitude) of the vector.

See also

[LengthSquared](#)

5.75.5.2 double OpenTK.Vector2d.LengthSquared [get]

Gets the square of the vector length (magnitude).

This property avoids the costly square root operation required by the Length property. This makes it more suitable for comparisons.

[Length](#)

5.75.5.3 Vector2d OpenTK.Vector2d.PerpendicularLeft [get]

Gets the perpendicular vector on the left side of this vector.

5.75.5.4 Vector2d OpenTK.Vector2d.PerpendicularRight [get]

Gets the perpendicular vector on the right side of this vector.

5.76 OpenTK.Vector2h Struct Reference

2-component Vector of the [Half](#) type. Occupies 4 Byte total.

Public Member Functions

- [Vector2h](#) ([Half](#) x, [Half](#) y)

The new Half2 instance will avoid conversion and copy directly from the [Half](#) parameters.

- [Vector2h](#) (Single x, Single y)

The new Half2 instance will convert the 2 parameters into 16-bit half-precision floating-point.

- [Vector2h](#) (Single x, Single y, bool throwOnError)

The new Half2 instance will convert the 2 parameters into 16-bit half-precision floating-point.

- [Vector2h](#) ([Vector2](#) v)

The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point.

- [Vector2h](#) ([Vector2](#) v, bool throwOnError)

The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point.

- [Vector2h](#) (ref [Vector2](#) v)
The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point. This is the fastest constructor.
- [Vector2h](#) (ref [Vector2](#) v, bool throwOnError)
The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point.
- [Vector2h](#) ([Vector2d](#) v)
The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point.
- [Vector2h](#) ([Vector2d](#) v, bool throwOnError)
The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point.
- [Vector2h](#) (ref [Vector2d](#) v)
The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point. This is the faster constructor.
- [Vector2h](#) (ref [Vector2d](#) v, bool throwOnError)
The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point.
- [Vector2 ToVector2](#) ()
Returns this Half2 instance's contents as [Vector2](#).
- [Vector2d ToVector2d](#) ()
Returns this Half2 instance's contents as [Vector2d](#).
- [Vector2h](#) (SerializationInfo info, StreamingContext context)
Constructor used by ISerializable to deserialize the object.
- void [GetObjectData](#) (SerializationInfo info, StreamingContext context)
Used by ISerialize to serialize the object.
- void [FromBinaryStream](#) (BinaryReader bin)
Updates the X and Y components of this instance by reading from a Stream.
- void [ToBinaryStream](#) (BinaryWriter bin)
Writes the X and Y components of this instance into a Stream.

- bool [Equals](#) ([Vector2h](#) other)
Returns a value indicating whether this instance is equal to a specified `OpenTK.Half2` vector.
- override string [ToString](#) ()
Returns a string that contains this `Half2`'s numbers in human-legible form.

Static Public Member Functions

- static [operator Vector2h](#) ([Vector2](#) v)
Converts `OpenTK.Vector2` to `OpenTK.Half2`.
- static [operator Vector2h](#) ([Vector2d](#) v)
Converts `OpenTK.Vector2d` to `OpenTK.Half2`.
- static [operator Vector2](#) ([Vector2h](#) h)
Converts `OpenTK.Half2` to `OpenTK.Vector2`.
- static [operator Vector2d](#) ([Vector2h](#) h)
Converts `OpenTK.Half2` to `OpenTK.Vector2d`.
- static byte[] [GetBytes](#) ([Vector2h](#) h)
Returns the `Half2` as an array of bytes.
- static [Vector2h FromBytes](#) (byte[] value, int startIndex)
Converts an array of bytes into `Half2`.

Public Attributes

- [Half X](#)
The X component of the `Half2`.
- [Half Y](#)
The Y component of the `Half2`.

Static Public Attributes

- static readonly int [SizeInBytes](#) = 4
The size in bytes for an instance of the `Half2` struct is 4.

5.76.1 Detailed Description

2-component Vector of the [Half](#) type. Occupies 4 Byte total.

5.76.2 Constructor & Destructor Documentation

5.76.2.1 OpenTK.Vector2h.Vector2h (Half x, Half y)

The new Half2 instance will avoid conversion and copy directly from the [Half](#) parameters.

Parameters

- x* An [Half](#) instance of a 16-bit half-precision floating-point number.
- y* An [Half](#) instance of a 16-bit half-precision floating-point number.

5.76.2.2 OpenTK.Vector2h.Vector2h (Single x, Single y)

The new Half2 instance will convert the 2 parameters into 16-bit half-precision floating-point.

Parameters

- x* 32-bit single-precision floating-point number.
- y* 32-bit single-precision floating-point number.

5.76.2.3 OpenTK.Vector2h.Vector2h (Single x, Single y, bool throwOnError)

The new Half2 instance will convert the 2 parameters into 16-bit half-precision floating-point.

Parameters

- x* 32-bit single-precision floating-point number.
- y* 32-bit single-precision floating-point number.
- throwOnError* Enable checks that will throw if the conversion result is not meaningful.

5.76.2.4 OpenTK.Vector2h.Vector2h (Vector2 v)

The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector2](#)

5.76.2.5 OpenTK.Vector2h.Vector2h (Vector2 v, bool throwOnError)

The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector2](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.76.2.6 OpenTK.Vector2h.Vector2h (ref Vector2 v)

The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point. This is the fastest constructor.

Parameters

v [OpenTK.Vector2](#)

5.76.2.7 OpenTK.Vector2h.Vector2h (ref Vector2 v, bool throwOnError)

The new Half2 instance will convert the [Vector2](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector2](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.76.2.8 OpenTK.Vector2h.Vector2h (Vector2d v)

The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector2d](#)

5.76.2.9 OpenTK.Vector2h.Vector2h (Vector2d v, bool throwOnError)

The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector2d](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.76.2.10 OpenTK.Vector2h.Vector2h (ref Vector2d v)

The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point. This is the faster constructor.

Parameters

v [OpenTK.Vector2d](#)

5.76.2.11 OpenTK.Vector2h.Vector2h (ref Vector2d v, bool throwOnError)

The new Half2 instance will convert the [Vector2d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector2d](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.76.2.12 **OpenTK.Vector2h.Vector2h** (**SerializationInfo** *info*, **StreamingContext** *context*)

Constructor used by ISerializable to deserialize the object.

Parameters

info
context

5.76.3 Member Function Documentation

5.76.3.1 **bool** **OpenTK.Vector2h.Equals** (**Vector2h** *other*)

Returns a value indicating whether this instance is equal to a specified OpenTK.Half2 vector.

Parameters

other OpenTK.Half2 to compare to this instance..

Returns

True, if other is equal to this instance; false otherwise.

5.76.3.2 **void** **OpenTK.Vector2h.FromBinaryStream** (**BinaryReader** *bin*)

Updates the X and Y components of this instance by reading from a Stream.

Parameters

bin A BinaryReader instance associated with an open Stream.

5.76.3.3 **static Vector2h** **OpenTK.Vector2h.FromBytes** (**byte[]** *value*, **int** *startIndex*) **[static]**

Converts an array of bytes into Half2.

Parameters

value A Half2 in it's byte[] representation.
startIndex The starting position within value.

Returns

A new Half2 instance.

5.76.3.4 static byte [] OpenTK.Vector2h.GetBytes (Vector2h *h*) [static]

Returns the Half2 as an array of bytes.

Parameters

h The Half2 to convert.

Returns

The input as byte array.

5.76.3.5 void OpenTK.Vector2h.GetObjectData (SerializationInfo *info*, StreamingContext *context*)

Used by ISerialize to serialize the object.

Parameters

info

context

5.76.3.6 static OpenTK.Vector2h.operator Vector2 (Vector2h *h*) [explicit, static]

Converts OpenTK.Half2 to [OpenTK.Vector2](#).

Parameters

h The Half2 to convert.

Returns

The resulting [Vector2](#).

5.76.3.7 static OpenTK.Vector2h.operator Vector2d (Vector2h *h*) [explicit, static]

Converts OpenTK.Half2 to [OpenTK.Vector2d](#).

Parameters

h The Half2 to convert.

Returns

The resulting [Vector2d](#).

5.76.3.8 static `OpenTK.Vector2h.operator Vector2h (Vector2 v)` [explicit, static]

Converts [OpenTK.Vector2](#) to `OpenTK.Half2`.

Parameters

v The [Vector2](#) to convert.

Returns

The resulting [Half](#) vector.

5.76.3.9 static `OpenTK.Vector2h.operator Vector2h (Vector2d v)` [explicit, static]

Converts [OpenTK.Vector2d](#) to `OpenTK.Half2`.

Parameters

v The [Vector2d](#) to convert.

Returns

The resulting [Half](#) vector.

5.76.3.10 void `OpenTK.Vector2h.ToBinaryStream (BinaryWriter bin)`

Writes the X and Y components of this instance into a Stream.

Parameters

bin A `BinaryWriter` instance associated with an open Stream.

5.76.3.11 override string `OpenTK.Vector2h.ToString ()`

Returns a string that contains this `Half2`'s numbers in human-legible form.

5.76.3.12 `Vector2 OpenTK.Vector2h.ToVector2 ()`

Returns this `Half2` instance's contents as [Vector2](#).

Returns

[OpenTK.Vector2](#)

5.76.3.13 Vector2d OpenTK.Vector2h.ToVector2d ()

Returns this Half2 instance's contents as [Vector2d](#).

5.76.4 Member Data Documentation

5.76.4.1 readonly int OpenTK.Vector2h.SizeInBytes = 4 [static]

The size in bytes for an instance of the Half2 struct is 4.

5.76.4.2 Half OpenTK.Vector2h.X

The X component of the Half2.

5.76.4.3 Half OpenTK.Vector2h.Y

The Y component of the Half2.

5.77 OpenTK.Vector3 Struct Reference

Represents a 3D vector using three single-precision floating-point numbers.

Public Member Functions

- [Vector3](#) (float x, float y, float z)
Constructs a new [Vector3](#).
- [Vector3](#) ([Vector2](#) v)
Constructs a new [Vector3](#) from the given [Vector2](#).
- [Vector3](#) ([Vector3](#) v)
Constructs a new [Vector3](#) from the given [Vector3](#).
- [Vector3](#) ([Vector4](#) v)
Constructs a new [Vector3](#) from the given [Vector4](#).
- void [Add](#) ([Vector3](#) right)
Add the Vector passed as parameter to this instance.
- void [Add](#) (ref [Vector3](#) right)

Add the Vector passed as parameter to this instance.

- void **Sub** (**Vector3** right)

Subtract the Vector passed as parameter from this instance.

- void **Sub** (ref **Vector3** right)

Subtract the Vector passed as parameter from this instance.

- void **Mult** (float f)

Multiply this instance by a scalar.

- void **Div** (float f)

Divide this instance by a scalar.

- void **Normalize** ()

*Scales the **Vector3** to unit length.*

- void **NormalizeFast** ()

*Scales the **Vector3** to approximately unit length.*

- void **Scale** (float sx, float sy, float sz)

*Scales the current **Vector3** by the given amounts.*

- void **Scale** (**Vector3** scale)

Scales this instance by the given parameter.

- void **Scale** (ref **Vector3** scale)

Scales this instance by the given parameter.

- override string **ToString** ()

*Returns a System.String that represents the current **Vector3**.*

- override int **GetHashCode** ()

Returns the hashcode for this instance.

- override bool **Equals** (object obj)

Indicates whether this instance and a specified object are equal.

- bool **Equals** (**Vector3** other)

Indicates whether the current vector is equal to another vector.

Static Public Member Functions

- static [Vector3 Sub](#) ([Vector3](#) a, [Vector3](#) b)
Subtract one Vector from another.
- static void [Sub](#) (ref [Vector3](#) a, ref [Vector3](#) b, out [Vector3](#) result)
Subtract one Vector from another.
- static [Vector3 Mult](#) ([Vector3](#) a, float f)
Multiply a vector and a scalar.
- static void [Mult](#) (ref [Vector3](#) a, float f, out [Vector3](#) result)
Multiply a vector and a scalar.
- static [Vector3 Div](#) ([Vector3](#) a, float f)
Divide a vector by a scalar.
- static void [Div](#) (ref [Vector3](#) a, float f, out [Vector3](#) result)
Divide a vector by a scalar.
- static [Vector3 Add](#) ([Vector3](#) a, [Vector3](#) b)
Adds two vectors.
- static void [Add](#) (ref [Vector3](#) a, ref [Vector3](#) b, out [Vector3](#) result)
Adds two vectors.
- static [Vector3 Subtract](#) ([Vector3](#) a, [Vector3](#) b)
Subtract one Vector from another.
- static void [Subtract](#) (ref [Vector3](#) a, ref [Vector3](#) b, out [Vector3](#) result)
Subtract one Vector from another.
- static [Vector3 Multiply](#) ([Vector3](#) vector, float scale)
Multiplies a vector by a scalar.
- static void [Multiply](#) (ref [Vector3](#) vector, float scale, out [Vector3](#) result)
Multiplies a vector by a scalar.
- static [Vector3 Multiply](#) ([Vector3](#) vector, [Vector3](#) scale)
Multiplies a vector by the components a vector (scale).
- static void [Multiply](#) (ref [Vector3](#) vector, ref [Vector3](#) scale, out [Vector3](#) result)
Multiplies a vector by the components of a vector (scale).

- static [Vector3 Divide](#) ([Vector3](#) vector, float scale)
Divides a vector by a scalar.
- static void [Divide](#) (ref [Vector3](#) vector, float scale, out [Vector3](#) result)
Divides a vector by a scalar.
- static [Vector3 Divide](#) ([Vector3](#) vector, [Vector3](#) scale)
Divides a vector by the components of a vector (scale).
- static void [Divide](#) (ref [Vector3](#) vector, ref [Vector3](#) scale, out [Vector3](#) result)
Divide a vector by the components of a vector (scale).
- static [Vector3 ComponentMin](#) ([Vector3](#) a, [Vector3](#) b)
Calculate the component-wise minimum of two vectors.
- static void [ComponentMin](#) (ref [Vector3](#) a, ref [Vector3](#) b, out [Vector3](#) result)
Calculate the component-wise minimum of two vectors.
- static [Vector3 ComponentMax](#) ([Vector3](#) a, [Vector3](#) b)
Calculate the component-wise maximum of two vectors.
- static void [ComponentMax](#) (ref [Vector3](#) a, ref [Vector3](#) b, out [Vector3](#) result)
Calculate the component-wise maximum of two vectors.
- static [Vector3 Min](#) ([Vector3](#) left, [Vector3](#) right)
Returns the [Vector3](#) with the minimum magnitude.
- static [Vector3 Max](#) ([Vector3](#) left, [Vector3](#) right)
Returns the [Vector3](#) with the minimum magnitude.
- static [Vector3 Clamp](#) ([Vector3](#) vec, [Vector3](#) min, [Vector3](#) max)
Clamp a vector to the given minimum and maximum vectors.
- static void [Clamp](#) (ref [Vector3](#) vec, ref [Vector3](#) min, ref [Vector3](#) max, out [Vector3](#) result)
Clamp a vector to the given minimum and maximum vectors.
- static [Vector3 Normalize](#) ([Vector3](#) vec)
Scale a vector to unit length.
- static void [Normalize](#) (ref [Vector3](#) vec, out [Vector3](#) result)

Scale a vector to unit length.

- static [Vector3 NormalizeFast](#) ([Vector3](#) vec)
Scale a vector to approximately unit length.
- static void [NormalizeFast](#) (ref [Vector3](#) vec, out [Vector3](#) result)
Scale a vector to approximately unit length.
- static float [Dot](#) ([Vector3](#) left, [Vector3](#) right)
Calculate the dot (scalar) product of two vectors.
- static void [Dot](#) (ref [Vector3](#) left, ref [Vector3](#) right, out float result)
Calculate the dot (scalar) product of two vectors.
- static [Vector3 Cross](#) ([Vector3](#) left, [Vector3](#) right)
Calculate the cross (vector) product of two vectors.
- static void [Cross](#) (ref [Vector3](#) left, ref [Vector3](#) right, out [Vector3](#) result)
Calculate the cross (vector) product of two vectors.
- static [Vector3 Lerp](#) ([Vector3](#) a, [Vector3](#) b, float blend)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static void [Lerp](#) (ref [Vector3](#) a, ref [Vector3](#) b, float blend, out [Vector3](#) result)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static [Vector3 BaryCentric](#) ([Vector3](#) a, [Vector3](#) b, [Vector3](#) c, float u, float v)
Interpolate 3 Vectors using Barycentric coordinates.
- static void [BaryCentric](#) (ref [Vector3](#) a, ref [Vector3](#) b, ref [Vector3](#) c, float u, float v, out [Vector3](#) result)
Interpolate 3 Vectors using Barycentric coordinates.
- static [Vector3 TransformVector](#) ([Vector3](#) vec, [Matrix4](#) mat)
Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.
- static void [TransformVector](#) (ref [Vector3](#) vec, ref [Matrix4](#) mat, out [Vector3](#) result)
Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.
- static [Vector3 TransformNormal](#) ([Vector3](#) norm, [Matrix4](#) mat)

Transform a Normal by the given Matrix.

- static void [TransformNormal](#) (ref [Vector3](#) norm, ref [Matrix4](#) mat, out [Vector3](#) result)

Transform a Normal by the given Matrix.

- static [Vector3](#) [TransformNormalInverse](#) ([Vector3](#) norm, [Matrix4](#) invMat)

Transform a Normal by the (transpose of the) given Matrix.

- static void [TransformNormalInverse](#) (ref [Vector3](#) norm, ref [Matrix4](#) invMat, out [Vector3](#) result)

Transform a Normal by the (transpose of the) given Matrix.

- static [Vector3](#) [TransformPosition](#) ([Vector3](#) pos, [Matrix4](#) mat)

Transform a Position by the given Matrix.

- static void [TransformPosition](#) (ref [Vector3](#) pos, ref [Matrix4](#) mat, out [Vector3](#) result)

Transform a Position by the given Matrix.

- static [Vector3](#) [Transform](#) ([Vector3](#) vec, [Matrix4](#) mat)

Transform a Vector by the given Matrix.

- static void [Transform](#) (ref [Vector3](#) vec, ref [Matrix4](#) mat, out [Vector3](#) result)

Transform a Vector by the given Matrix.

- static [Vector3](#) [Transform](#) ([Vector3](#) vec, [Quaternion](#) quat)

Transforms a vector by a quaternion rotation.

- static void [Transform](#) (ref [Vector3](#) vec, ref [Quaternion](#) quat, out [Vector3](#) result)

Transforms a vector by a quaternion rotation.

- static [Vector3](#) [TransformPerspective](#) ([Vector3](#) vec, [Matrix4](#) mat)

Transform a [Vector3](#) by the given Matrix, and project the resulting [Vector4](#) back to a [Vector3](#).

- static void [TransformPerspective](#) (ref [Vector3](#) vec, ref [Matrix4](#) mat, out [Vector3](#) result)

Transform a [Vector3](#) by the given Matrix, and project the resulting [Vector4](#) back to a [Vector3](#).

- static float [CalculateAngle](#) ([Vector3](#) first, [Vector3](#) second)

Calculates the angle (in radians) between two vectors.

- static void [CalculateAngle](#) (ref [Vector3](#) first, ref [Vector3](#) second, out float result)
Calculates the angle (in radians) between two vectors.
- static [Vector3](#) [operator+](#) ([Vector3](#) left, [Vector3](#) right)
Adds two instances.
- static [Vector3](#) [operator-](#) ([Vector3](#) left, [Vector3](#) right)
Subtracts two instances.
- static [Vector3](#) [operator-](#) ([Vector3](#) vec)
Negates an instance.
- static [Vector3](#) [operator*](#) ([Vector3](#) vec, float scale)
Multiplies an instance by a scalar.
- static [Vector3](#) [operator*](#) (float scale, [Vector3](#) vec)
Multiplies an instance by a scalar.
- static [Vector3](#) [operator/](#) ([Vector3](#) vec, float scale)
Divides an instance by a scalar.
- static bool [operator==](#) ([Vector3](#) left, [Vector3](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([Vector3](#) left, [Vector3](#) right)
Compares two instances for inequality.

Public Attributes

- float [X](#)
The X component of the [Vector3](#).
- float [Y](#)
The Y component of the [Vector3](#).
- float [Z](#)
The Z component of the [Vector3](#).

Static Public Attributes

- static readonly [Vector3 UnitX](#) = new [Vector3](#)(1, 0, 0)
Defines a unit-length [Vector3](#) that points towards the X-axis.
- static readonly [Vector3 UnitY](#) = new [Vector3](#)(0, 1, 0)
Defines a unit-length [Vector3](#) that points towards the Y-axis.
- static readonly [Vector3 UnitZ](#) = new [Vector3](#)(0, 0, 1)
/ Defines a unit-length [Vector3](#) that points towards the Z-axis.
- static readonly [Vector3 Zero](#) = new [Vector3](#)(0, 0, 0)
Defines a zero-length [Vector3](#).
- static readonly [Vector3 One](#) = new [Vector3](#)(1, 1, 1)
Defines an instance with all components set to 1.
- static readonly int [SizeInBytes](#) = Marshal.SizeOf(new [Vector3](#)())
Defines the size of the [Vector3](#) struct in bytes.

Properties

- float [Length](#) [get]
Gets the length (magnitude) of the vector.
- float [LengthFast](#) [get]
Gets an approximation of the vector length (magnitude).
- float [LengthSquared](#) [get]
Gets the square of the vector length (magnitude).
- [Vector2 Xy](#) [get, set]
Gets or sets an [OpenTK.Vector2](#) with the X and Y components of this instance.

5.77.1 Detailed Description

Represents a 3D vector using three single-precision floating-point numbers. The [Vector3](#) structure is suitable for interoperation with unmanaged code requiring three consecutive floats.

5.77.2 Constructor & Destructor Documentation

5.77.2.1 OpenTK.Vector3.Vector3 (float *x*, float *y*, float *z*)

Constructs a new [Vector3](#).

Parameters

- x* The x component of the [Vector3](#).
- y* The y component of the [Vector3](#).
- z* The z component of the [Vector3](#).

5.77.2.2 OpenTK.Vector3.Vector3 (Vector2 *v*)

Constructs a new [Vector3](#) from the given [Vector2](#).

Parameters

- v* The [Vector2](#) to copy components from.

5.77.2.3 OpenTK.Vector3.Vector3 (Vector3 *v*)

Constructs a new [Vector3](#) from the given [Vector3](#).

Parameters

- v* The [Vector3](#) to copy components from.

5.77.2.4 OpenTK.Vector3.Vector3 (Vector4 *v*)

Constructs a new [Vector3](#) from the given [Vector4](#).

Parameters

- v* The [Vector4](#) to copy components from.

5.77.3 Member Function Documentation

5.77.3.1 void OpenTK.Vector3.Add (Vector3 *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

5.77.3.2 void OpenTK.Vector3.Add (ref Vector3 *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

**5.77.3.3 static Vector3 OpenTK.Vector3.Add (Vector3 *a*, Vector3 *b*)
[static]**

Adds two vectors.

Parameters

a Left operand.

b Right operand.

Returns

Result of operation.

**5.77.3.4 static void OpenTK.Vector3.Add (ref Vector3 *a*, ref Vector3 *b*, out
Vector3 *result*) [static]**

Adds two vectors.

Parameters

a Left operand.

b Right operand.

result Result of operation.

**5.77.3.5 static Vector3 OpenTK.Vector3.BaryCentric (Vector3 *a*, Vector3 *b*,
Vector3 *c*, float *u*, float *v*) [static]**

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector
- b* Second input Vector
- c* Third input Vector
- u* First Barycentric Coordinate
- v* Second Barycentric Coordinate

Returns

a when $u=v=0$, b when $u=1,v=0$, c when $u=0,v=1$, and a linear combination of a,b,c otherwise

5.77.3.6 `static void OpenTK.Vector3.BaryCentric (ref Vector3 a, ref Vector3 b, ref Vector3 c, float u, float v, out Vector3 result) [static]`

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector.
- b* Second input Vector.
- c* Third input Vector.
- u* First Barycentric Coordinate.
- v* Second Barycentric Coordinate.
- result* Output Vector. a when $u=v=0$, b when $u=1,v=0$, c when $u=0,v=1$, and a linear combination of a,b,c otherwise

5.77.3.7 `static float OpenTK.Vector3.CalculateAngle (Vector3 first, Vector3 second) [static]`

Calculates the angle (in radians) between two vectors.

Parameters

- first* The first vector.
- second* The second vector.

Returns

Angle (in radians) between the vectors.

Note that the returned angle is never bigger than the constant Pi.

5.77.3.8 `static void OpenTK.Vector3.CalculateAngle (ref Vector3 first, ref Vector3 second, out float result) [static]`

Calculates the angle (in radians) between two vectors.

Parameters

first The first vector.

second The second vector.

result Angle (in radians) between the vectors.

Note that the returned angle is never bigger than the constant Pi.

5.77.3.9 `static Vector3 OpenTK.Vector3.Clamp (Vector3 vec, Vector3 min, Vector3 max) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector

min Minimum vector

max Maximum vector

Returns

The clamped vector

5.77.3.10 `static void OpenTK.Vector3.Clamp (ref Vector3 vec, ref Vector3 min, ref Vector3 max, out Vector3 result) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector

min Minimum vector

max Maximum vector

result The clamped vector

5.77.3.11 `static Vector3 OpenTK.Vector3.ComponentMax (Vector3 a,
Vector3 b) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

Returns

The component-wise maximum

5.77.3.12 `static void OpenTK.Vector3.ComponentMax (ref Vector3 a, ref
Vector3 b, out Vector3 result) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

result The component-wise maximum

5.77.3.13 `static Vector3 OpenTK.Vector3.ComponentMin (Vector3 a, Vector3
b) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

a First operand

b Second operand

Returns

The component-wise minimum

5.77.3.14 `static void OpenTK.Vector3.ComponentMin (ref Vector3 a, ref Vector3 b, out Vector3 result) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

a First operand
b Second operand
result The component-wise minimum

5.77.3.15 `static Vector3 OpenTK.Vector3.Cross (Vector3 left, Vector3 right) [static]`

Calculate the cross (vector) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The cross product of the two inputs

5.77.3.16 `static void OpenTK.Vector3.Cross (ref Vector3 left, ref Vector3 right, out Vector3 result) [static]`

Calculate the cross (vector) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The cross product of the two inputs

Parameters

result The cross product of the two inputs

5.77.3.17 `static void OpenTK.Vector3.Div (ref Vector3 a, float f, out Vector3 result) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the division

5.77.3.18 `static Vector3 OpenTK.Vector3.Div (Vector3 a, float f) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand

Returns

Result of the division

5.77.3.19 `void OpenTK.Vector3.Div (float f)`

Divide this instance by a scalar.

Parameters

f Scalar operand.

5.77.3.20 `static Vector3 OpenTK.Vector3.Divide (Vector3 vector, float scale) [static]`

Divides a vector by a scalar.

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.77.3.21 `static void OpenTK.Vector3.Divide (ref Vector3 vector, float scale, out Vector3 result) [static]`

Divides a vector by a scalar.

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.77.3.22 `static void OpenTK.Vector3.Divide (ref Vector3 vector, ref Vector3 scale, out Vector3 result) [static]`

Divide a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.77.3.23 `static Vector3 OpenTK.Vector3.Divide (Vector3 vector, Vector3 scale) [static]`

Divides a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.77.3.24 `static void OpenTK.Vector3.Dot (ref Vector3 left, ref Vector3 right, out float result) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand
result The dot product of the two inputs

**5.77.3.25 static float OpenTK.Vector3.Dot (Vector3 *left*, Vector3 *right*)
[static]**

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The dot product of the two inputs

5.77.3.26 bool OpenTK.Vector3.Equals (Vector3 *other*)

Indicates whether the current vector is equal to another vector.

Parameters

other A vector to compare with this vector.

Returns

true if the current vector is equal to the vector parameter; otherwise, false.

5.77.3.27 override bool OpenTK.Vector3.Equals (object *obj*)

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.77.3.28 override int OpenTK.Vector3.GetHashCode ()

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.77.3.29 static Vector3 OpenTK.Vector3.Lerp (Vector3 *a*, Vector3 *b*, float *blend*) [static]

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

Returns

a when blend=0, b when blend=1, and a linear combination otherwise

5.77.3.30 static void OpenTK.Vector3.Lerp (ref Vector3 *a*, ref Vector3 *b*, float *blend*, out Vector3 *result*) [static]

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

result a when blend=0, b when blend=1, and a linear combination otherwise

5.77.3.31 static Vector3 OpenTK.Vector3.Min (Vector3 *left*, Vector3 *right*) [static]

Returns the [Vector3](#) with the minimum magnitude.

Parameters

left Left operand

right Right operand

Returns

The minimum [Vector3](#)

5.77.3.32 `static Vector3 OpenTK.Vector3.Min (Vector3 left, Vector3 right)`
`[static]`

Returns the [Vector3](#) with the minimum magnitude.

Parameters

left Left operand

right Right operand

Returns

The minimum [Vector3](#)

5.77.3.33 `void OpenTK.Vector3.Mult (float f)`

Multiply this instance by a scalar.

Parameters

f Scalar operand.

5.77.3.34 `static Vector3 OpenTK.Vector3.Mult (Vector3 a, float f)`
`[static]`

Multiply a vector and a scalar.

Parameters

a Vector operand

f Scalar operand

Returns

Result of the multiplication

5.77.3.35 `static void OpenTK.Vector3.Mult (ref Vector3 a, float f, out Vector3 result) [static]`

Multiply a vector and a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the multiplication

5.77.3.36 `static Vector3 OpenTK.Vector3.Multiply (Vector3 vector, float scale) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.77.3.37 `static void OpenTK.Vector3.Multiply (ref Vector3 vector, float scale, out Vector3 result) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.77.3.38 `static void OpenTK.Vector3.Multiply (ref Vector3 vector, ref Vector3 scale, out Vector3 result) [static]`

Multiplies a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.77.3.39 `static Vector3 OpenTK.Vector3.Multiply (Vector3 vector, Vector3 scale) [static]`

Multiplies a vector by the components a vector (scale).

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.77.3.40 `static Vector3 OpenTK.Vector3.Normalize (Vector3 vec) [static]`

Scale a vector to unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.77.3.41 `static void OpenTK.Vector3.Normalize (ref Vector3 vec, out Vector3 result) [static]`

Scale a vector to unit length.

Parameters

vec The input vector
result The normalized vector

5.77.3.42 void OpenTK.Vector3.Normalize ()

Scales the [Vector3](#) to unit length.

5.77.3.43 void OpenTK.Vector3.NormalizeFast ()

Scales the [Vector3](#) to approximately unit length.

5.77.3.44 static void OpenTK.Vector3.NormalizeFast (ref Vector3 *vec*, out Vector3 *result*) [static]

Scale a vector to approximately unit length.

Parameters

vec The input vector
result The normalized vector

5.77.3.45 static Vector3 OpenTK.Vector3.NormalizeFast (Vector3 *vec*) [static]

Scale a vector to approximately unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.77.3.46 static bool OpenTK.Vector3.operator!= (Vector3 *left*, Vector3 *right*) [static]

Compares two instances for inequality.

Parameters

left The first instance.
right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.77.3.47 `static Vector3 OpenTK.Vector3.operator* (Vector3 vec, float scale) [static]`

Multiplies an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.77.3.48 `static Vector3 OpenTK.Vector3.operator* (float scale, Vector3 vec) [static]`

Multiplies an instance by a scalar.

Parameters

scale The scalar.

vec The instance.

Returns

The result of the calculation.

5.77.3.49 `static Vector3 OpenTK.Vector3.operator+ (Vector3 left, Vector3 right) [static]`

Adds two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.77.3.50 static Vector3 OpenTK.Vector3.operator- (Vector3 *left*, Vector3 *right*) [static]

Subtracts two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.77.3.51 static Vector3 OpenTK.Vector3.operator- (Vector3 *vec*) [static]

Negates an instance.

Parameters

vec The instance.

Returns

The result of the calculation.

5.77.3.52 static Vector3 OpenTK.Vector3.operator/ (Vector3 *vec*, float *scale*) [static]

Divides an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.77.3.53 `static bool OpenTK.Vector3.operator==(Vector3 left, Vector3 right) [static]`

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.77.3.54 `void OpenTK.Vector3.Scale (float sx, float sy, float sz)`

Scales the current [Vector3](#) by the given amounts.

Parameters

sx The scale of the X component.

sy The scale of the Y component.

sz The scale of the Z component.

5.77.3.55 `void OpenTK.Vector3.Scale (Vector3 scale)`

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.77.3.56 `void OpenTK.Vector3.Scale (ref Vector3 scale)`

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.77.3.57 void OpenTK.Vector3.Sub (ref Vector3 *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

**5.77.3.58 static Vector3 OpenTK.Vector3.Sub (Vector3 *a*, Vector3 *b*)
[static]**

Subtract one Vector from another.

Parameters

a First operand

b Second operand

Returns

Result of subtraction

5.77.3.59 void OpenTK.Vector3.Sub (Vector3 *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

**5.77.3.60 static void OpenTK.Vector3.Sub (ref Vector3 *a*, ref Vector3 *b*, out
Vector3 *result*) [static]**

Subtract one Vector from another.

Parameters

a First operand

b Second operand

result Result of subtraction

5.77.3.61 `static void OpenTK.Vector3.Subtract (ref Vector3 a, ref Vector3 b,
out Vector3 result) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand
result Result of subtraction

5.77.3.62 `static Vector3 OpenTK.Vector3.Subtract (Vector3 a, Vector3 b)
[static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand

Returns

Result of subtraction

5.77.3.63 `override string OpenTK.Vector3.ToString ()`

Returns a System.String that represents the current [Vector3](#).

Returns

5.77.3.64 `static Vector3 OpenTK.Vector3.Transform (Vector3 vec,
Quaternion quat) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.
quat The quaternion to rotate the vector by.

Returns

The result of the operation.

5.77.3.65 `static void OpenTK.Vector3.Transform (ref Vector3 vec, ref Matrix4 mat, out Vector3 result) [static]`

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform
mat The desired transformation
result The transformed vector

5.77.3.66 `static void OpenTK.Vector3.Transform (ref Vector3 vec, ref Quaternion quat, out Vector3 result) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.
quat The quaternion to rotate the vector by.
result The result of the operation.

5.77.3.67 `static Vector3 OpenTK.Vector3.Transform (Vector3 vec, Matrix4 mat) [static]`

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform
mat The desired transformation

Returns

The transformed vector

5.77.3.68 `static Vector3 OpenTK.Vector3.TransformNormal (Vector3 norm, Matrix4 mat) [static]`

Transform a Normal by the given Matrix.

This calculates the inverse of the given matrix, use TransformNormalInverse if you already have the inverse to avoid this extra calculation

Parameters

norm The normal to transform
mat The desired transformation

Returns

The transformed normal

5.77.3.69 `static void OpenTK.Vector3.TransformNormal (ref Vector3 norm,
ref Matrix4 mat, out Vector3 result) [static]`

Transform a Normal by the given Matrix.

This calculates the inverse of the given matrix, use TransformNormalInverse if you already have the inverse to avoid this extra calculation

Parameters

norm The normal to transform
mat The desired transformation
result The transformed normal

5.77.3.70 `static Vector3 OpenTK.Vector3.TransformNormalInverse (Vector3
norm, Matrix4 invMat) [static]`

Transform a Normal by the (transpose of the) given Matrix.

This version doesn't calculate the inverse matrix. Use this version if you already have the inverse of the desired transform to hand

Parameters

norm The normal to transform
invMat The inverse of the desired transformation

Returns

The transformed normal

5.77.3.71 `static void OpenTK.Vector3.TransformNormalInverse (ref Vector3
norm, ref Matrix4 invMat, out Vector3 result) [static]`

Transform a Normal by the (transpose of the) given Matrix.

This version doesn't calculate the inverse matrix. Use this version if you already have the inverse of the desired transform to hand

Parameters

norm The normal to transform
invMat The inverse of the desired transformation
result The transformed normal

5.77.3.72 `static void OpenTK.Vector3.TransformPerspective (ref Vector3 vec,
ref Matrix4 mat, out Vector3 result) [static]`

Transform a [Vector3](#) by the given Matrix, and project the resulting [Vector4](#) back to a [Vector3](#).

Parameters

vec The vector to transform
mat The desired transformation
result The transformed vector

5.77.3.73 `static Vector3 OpenTK.Vector3.TransformPerspective (Vector3 vec,
Matrix4 mat) [static]`

Transform a [Vector3](#) by the given Matrix, and project the resulting [Vector4](#) back to a [Vector3](#).

Parameters

vec The vector to transform
mat The desired transformation

Returns

The transformed vector

5.77.3.74 `static Vector3 OpenTK.Vector3.TransformPosition (Vector3 pos,
Matrix4 mat) [static]`

Transform a Position by the given Matrix.

Parameters

pos The position to transform
mat The desired transformation

Returns

The transformed position

5.77.3.75 `static void OpenTK.Vector3.TransformPosition (ref Vector3 pos, ref Matrix4 mat, out Vector3 result) [static]`

Transform a Position by the given Matrix.

Parameters

pos The position to transform
mat The desired transformation
result The transformed position

5.77.3.76 `static Vector3 OpenTK.Vector3.TransformVector (Vector3 vec, Matrix4 mat) [static]`

Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.

Parameters

vec The vector to transform
mat The desired transformation

Returns

The transformed vector

5.77.3.77 `static void OpenTK.Vector3.TransformVector (ref Vector3 vec, ref Matrix4 mat, out Vector3 result) [static]`

Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.

Parameters

vec The vector to transform
mat The desired transformation
result The transformed vector

5.77.4 Member Data Documentation

5.77.4.1 `readonly Vector3 OpenTK.Vector3.One = new Vector3(1, 1, 1) [static]`

Defines an instance with all components set to 1.

5.77.4.2 `readonly int OpenTK.Vector3.SizeInBytes = Marshal.SizeOf(new Vector3()) [static]`

Defines the size of the [Vector3](#) struct in bytes.

5.77.4.3 `readonly Vector3 OpenTK.Vector3.UnitX = new Vector3(1, 0, 0) [static]`

Defines a unit-length [Vector3](#) that points towards the X-axis.

5.77.4.4 `readonly Vector3 OpenTK.Vector3.UnitY = new Vector3(0, 1, 0) [static]`

Defines a unit-length [Vector3](#) that points towards the Y-axis.

5.77.4.5 `readonly Vector3 OpenTK.Vector3.UnitZ = new Vector3(0, 0, 1) [static]`

/ Defines a unit-length [Vector3](#) that points towards the Z-axis.

5.77.4.6 `float OpenTK.Vector3.X`

The X component of the [Vector3](#).

5.77.4.7 `float OpenTK.Vector3.Y`

The Y component of the [Vector3](#).

5.77.4.8 `float OpenTK.Vector3.Z`

The Z component of the [Vector3](#).

5.77.4.9 `readonly Vector3 OpenTK.Vector3.Zero = new Vector3(0, 0, 0) [static]`

Defines a zero-length [Vector3](#).

5.77.5 Property Documentation

5.77.5.1 float OpenTK.Vector3.Length [get]

Gets the length (magnitude) of the vector.

[LengthFast](#)

See also

[LengthSquared](#)

5.77.5.2 float OpenTK.Vector3.LengthFast [get]

Gets an approximation of the vector length (magnitude).

This property uses an approximation of the square root function to calculate vector magnitude, with an upper error bound of 0.001.

[Length](#)

See also

[LengthSquared](#)

5.77.5.3 float OpenTK.Vector3.LengthSquared [get]

Gets the square of the vector length (magnitude).

This property avoids the costly square root operation required by the Length property. This makes it more suitable for comparisons.

[Length](#)

See also

[LengthFast](#)

5.77.5.4 Vector2 OpenTK.Vector3.Xy [get, set]

Gets or sets an [OpenTK.Vector2](#) with the X and Y components of this instance.

5.78 OpenTK.Vector3d Struct Reference

Represents a 3D vector using three double-precision floating-point numbers.

Public Member Functions

- [Vector3d](#) (double x, double y, double z)
Constructs a new [Vector3](#).
- [Vector3d](#) ([Vector2d](#) v)
Constructs a new instance from the given [Vector2d](#).
- [Vector3d](#) ([Vector3d](#) v)
Constructs a new instance from the given [Vector3d](#).
- [Vector3d](#) ([Vector4d](#) v)
Constructs a new instance from the given [Vector4d](#).
- void [Add](#) ([Vector3d](#) right)
Add the Vector passed as parameter to this instance.
- void [Add](#) (ref [Vector3d](#) right)
Add the Vector passed as parameter to this instance.
- void [Sub](#) ([Vector3d](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Sub](#) (ref [Vector3d](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Mult](#) (double f)
Multiply this instance by a scalar.
- void [Div](#) (double f)
Divide this instance by a scalar.
- void [Normalize](#) ()
Scales the [Vector3d](#) to unit length.
- void [NormalizeFast](#) ()
Scales the [Vector3d](#) to approximately unit length.
- void [Scale](#) (double sx, double sy, double sz)
Scales the current [Vector3d](#) by the given amounts.
- void [Scale](#) ([Vector3d](#) scale)
Scales this instance by the given parameter.

- void [Scale](#) (ref [Vector3d](#) scale)
Scales this instance by the given parameter.
- override string [ToString](#) ()
Returns a System.String that represents the current [Vector3](#).
- override int [GetHashCode](#) ()
Returns the hashcode for this instance.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Vector3d](#) other)
Indicates whether the current vector is equal to another vector.

Static Public Member Functions

- static [Vector3d Sub](#) ([Vector3d](#) a, [Vector3d](#) b)
Subtract one Vector from another.
- static void [Sub](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, out [Vector3d](#) result)
Subtract one Vector from another.
- static [Vector3d Mult](#) ([Vector3d](#) a, double f)
Multiply a vector and a scalar.
- static void [Mult](#) (ref [Vector3d](#) a, double f, out [Vector3d](#) result)
Multiply a vector and a scalar.
- static [Vector3d Div](#) ([Vector3d](#) a, double f)
Divide a vector by a scalar.
- static void [Div](#) (ref [Vector3d](#) a, double f, out [Vector3d](#) result)
Divide a vector by a scalar.
- static [Vector3d Add](#) ([Vector3d](#) a, [Vector3d](#) b)
Adds two vectors.
- static void [Add](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, out [Vector3d](#) result)
Adds two vectors.

- static [Vector3d Subtract](#) ([Vector3d](#) a, [Vector3d](#) b)
Subtract one Vector from another.
- static void [Subtract](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, out [Vector3d](#) result)
Subtract one Vector from another.
- static [Vector3d Multiply](#) ([Vector3d](#) vector, double scale)
Multiplies a vector by a scalar.
- static void [Multiply](#) (ref [Vector3d](#) vector, double scale, out [Vector3d](#) result)
Multiplies a vector by a scalar.
- static [Vector3d Multiply](#) ([Vector3d](#) vector, [Vector3d](#) scale)
Multiplies a vector by the components a vector (scale).
- static void [Multiply](#) (ref [Vector3d](#) vector, ref [Vector3d](#) scale, out [Vector3d](#) result)
Multiplies a vector by the components of a vector (scale).
- static [Vector3d Divide](#) ([Vector3d](#) vector, double scale)
Divides a vector by a scalar.
- static void [Divide](#) (ref [Vector3d](#) vector, double scale, out [Vector3d](#) result)
Divides a vector by a scalar.
- static [Vector3d Divide](#) ([Vector3d](#) vector, [Vector3d](#) scale)
Divides a vector by the components of a vector (scale).
- static void [Divide](#) (ref [Vector3d](#) vector, ref [Vector3d](#) scale, out [Vector3d](#) result)
Divide a vector by the components of a vector (scale).
- static [Vector3d ComponentMin](#) ([Vector3d](#) a, [Vector3d](#) b)
Calculate the component-wise minimum of two vectors.
- static void [ComponentMin](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, out [Vector3d](#) result)
Calculate the component-wise minimum of two vectors.
- static [Vector3d ComponentMax](#) ([Vector3d](#) a, [Vector3d](#) b)
Calculate the component-wise maximum of two vectors.
- static void [ComponentMax](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, out [Vector3d](#) result)

Calculate the component-wise maximum of two vectors.

- static [Vector3d Min](#) ([Vector3d](#) left, [Vector3d](#) right)
Returns the [Vector3d](#) with the minimum magnitude.
- static [Vector3d Max](#) ([Vector3d](#) left, [Vector3d](#) right)
Returns the [Vector3d](#) with the minimum magnitude.
- static [Vector3d Clamp](#) ([Vector3d](#) vec, [Vector3d](#) min, [Vector3d](#) max)
Clamp a vector to the given minimum and maximum vectors.
- static void [Clamp](#) (ref [Vector3d](#) vec, ref [Vector3d](#) min, ref [Vector3d](#) max, out [Vector3d](#) result)
Clamp a vector to the given minimum and maximum vectors.
- static [Vector3d Normalize](#) ([Vector3d](#) vec)
Scale a vector to unit length.
- static void [Normalize](#) (ref [Vector3d](#) vec, out [Vector3d](#) result)
Scale a vector to unit length.
- static [Vector3d NormalizeFast](#) ([Vector3d](#) vec)
Scale a vector to approximately unit length.
- static void [NormalizeFast](#) (ref [Vector3d](#) vec, out [Vector3d](#) result)
Scale a vector to approximately unit length.
- static double [Dot](#) ([Vector3d](#) left, [Vector3d](#) right)
Calculate the dot (scalar) product of two vectors.
- static void [Dot](#) (ref [Vector3d](#) left, ref [Vector3d](#) right, out double result)
Calculate the dot (scalar) product of two vectors.
- static [Vector3d Cross](#) ([Vector3d](#) left, [Vector3d](#) right)
Calculate the cross (vector) product of two vectors.
- static void [Cross](#) (ref [Vector3d](#) left, ref [Vector3d](#) right, out [Vector3d](#) result)
Calculate the cross (vector) product of two vectors.
- static [Vector3d Lerp](#) ([Vector3d](#) a, [Vector3d](#) b, double blend)
Returns a new Vector that is the linear blend of the 2 given Vectors.

- static void [Lerp](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, double blend, out [Vector3d](#) result)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static [Vector3d](#) [BaryCentric](#) ([Vector3d](#) a, [Vector3d](#) b, [Vector3d](#) c, double u, double v)
Interpolate 3 Vectors using Barycentric coordinates.
- static void [BaryCentric](#) (ref [Vector3d](#) a, ref [Vector3d](#) b, ref [Vector3d](#) c, double u, double v, out [Vector3d](#) result)
Interpolate 3 Vectors using Barycentric coordinates.
- static [Vector3d](#) [TransformVector](#) ([Vector3d](#) vec, [Matrix4d](#) mat)
Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.
- static void [TransformVector](#) (ref [Vector3d](#) vec, ref [Matrix4d](#) mat, out [Vector3d](#) result)
Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.
- static [Vector3d](#) [TransformNormal](#) ([Vector3d](#) norm, [Matrix4d](#) mat)
Transform a Normal by the given Matrix.
- static void [TransformNormal](#) (ref [Vector3d](#) norm, ref [Matrix4d](#) mat, out [Vector3d](#) result)
Transform a Normal by the given Matrix.
- static [Vector3d](#) [TransformNormalInverse](#) ([Vector3d](#) norm, [Matrix4d](#) invMat)
Transform a Normal by the (transpose of the) given Matrix.
- static void [TransformNormalInverse](#) (ref [Vector3d](#) norm, ref [Matrix4d](#) invMat, out [Vector3d](#) result)
Transform a Normal by the (transpose of the) given Matrix.
- static [Vector3d](#) [TransformPosition](#) ([Vector3d](#) pos, [Matrix4d](#) mat)
Transform a Position by the given Matrix.
- static void [TransformPosition](#) (ref [Vector3d](#) pos, ref [Matrix4d](#) mat, out [Vector3d](#) result)
Transform a Position by the given Matrix.
- static [Vector3d](#) [Transform](#) ([Vector3d](#) vec, [Matrix4d](#) mat)

Transform a Vector by the given Matrix.

- static void **Transform** (ref **Vector3d** vec, ref **Matrix4d** mat, out **Vector3d** result)
Transform a Vector by the given Matrix.
- static **Vector3d** **Transform** (**Vector3d** vec, **Quaterniond** quat)
Transforms a vector by a quaternion rotation.
- static void **Transform** (ref **Vector3d** vec, ref **Quaterniond** quat, out **Vector3d** result)
Transforms a vector by a quaternion rotation.
- static **Vector3d** **TransformPerspective** (**Vector3d** vec, **Matrix4d** mat)
*Transform a **Vector3d** by the given Matrix, and project the resulting **Vector4** back to a **Vector3**.*
- static void **TransformPerspective** (ref **Vector3d** vec, ref **Matrix4d** mat, out **Vector3d** result)
*Transform a **Vector3d** by the given Matrix, and project the resulting **Vector4d** back to a **Vector3d**.*
- static double **CalculateAngle** (**Vector3d** first, **Vector3d** second)
Calculates the angle (in radians) between two vectors.
- static void **CalculateAngle** (ref **Vector3d** first, ref **Vector3d** second, out double result)
Calculates the angle (in radians) between two vectors.
- static **Vector3d** **operator+** (**Vector3d** left, **Vector3d** right)
Adds two instances.
- static **Vector3d** **operator-** (**Vector3d** left, **Vector3d** right)
Subtracts two instances.
- static **Vector3d** **operator-** (**Vector3d** vec)
Negates an instance.
- static **Vector3d** **operator*** (**Vector3d** vec, double scale)
Multiplies an instance by a scalar.
- static **Vector3d** **operator*** (double scale, **Vector3d** vec)
Multiplies an instance by a scalar.

- static [Vector3d operator/](#) ([Vector3d](#) vec, double scale)
Divides an instance by a scalar.
- static bool [operator==](#) ([Vector3d](#) left, [Vector3d](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([Vector3d](#) left, [Vector3d](#) right)
Compares two instances for inequality.
- static [operator Vector3d](#) ([Vector3](#) v3)
Converts [OpenTK.Vector3](#) to [OpenTK.Vector3d](#).
- static [operator Vector3](#) ([Vector3d](#) v3d)
Converts [OpenTK.Vector3d](#) to [OpenTK.Vector3](#).

Public Attributes

- double [X](#)
The X component of the [Vector3](#).
- double [Y](#)
The Y component of the [Vector3](#).
- double [Z](#)
The Z component of the [Vector3](#).

Static Public Attributes

- static readonly [Vector3d UnitX](#) = new [Vector3d](#)(1, 0, 0)
Defines a unit-length [Vector3d](#) that points towards the X-axis.
- static readonly [Vector3d UnitY](#) = new [Vector3d](#)(0, 1, 0)
Defines a unit-length [Vector3d](#) that points towards the Y-axis.
- static readonly [Vector3d UnitZ](#) = new [Vector3d](#)(0, 0, 1)
/ Defines a unit-length [Vector3d](#) that points towards the Z-axis.
- static readonly [Vector3d Zero](#) = new [Vector3d](#)(0, 0, 0)
Defines a zero-length [Vector3](#).

- static readonly [Vector3d One](#) = new [Vector3d](#)(1, 1, 1)
Defines an instance with all components set to 1.
- static readonly int [SizeInBytes](#) = Marshal.SizeOf(new [Vector3d](#)())
Defines the size of the [Vector3d](#) struct in bytes.

Properties

- double [Length](#) [get]
Gets the length (magnitude) of the vector.
- double [LengthFast](#) [get]
Gets an approximation of the vector length (magnitude).
- double [LengthSquared](#) [get]
Gets the square of the vector length (magnitude).
- [Vector2d Xy](#) [get, set]
Gets or sets an [OpenTK.Vector2d](#) with the X and Y components of this instance.

5.78.1 Detailed Description

Represents a 3D vector using three double-precision floating-point numbers.

5.78.2 Constructor & Destructor Documentation

5.78.2.1 OpenTK.Vector3d.Vector3d (double x, double y, double z)

Constructs a new [Vector3](#).

Parameters

- x** The x component of the [Vector3](#).
- y** The y component of the [Vector3](#).
- z** The z component of the [Vector3](#).

5.78.2.2 OpenTK.Vector3d.Vector3d (Vector2d *v*)

Constructs a new instance from the given [Vector2d](#).

Parameters

- v* The [Vector2d](#) to copy components from.

5.78.2.3 OpenTK.Vector3d.Vector3d (Vector3d *v*)

Constructs a new instance from the given [Vector3d](#).

Parameters

- v* The [Vector3d](#) to copy components from.

5.78.2.4 OpenTK.Vector3d.Vector3d (Vector4d *v*)

Constructs a new instance from the given [Vector4d](#).

Parameters

- v* The [Vector4d](#) to copy components from.

5.78.3 Member Function Documentation**5.78.3.1 void OpenTK.Vector3d.Add (Vector3d *right*)**

Add the Vector passed as parameter to this instance.

Parameters

- right* Right operand. This parameter is only read from.

5.78.3.2 void OpenTK.Vector3d.Add (ref Vector3d *right*)

Add the Vector passed as parameter to this instance.

Parameters

- right* Right operand. This parameter is only read from.

5.78.3.3 static Vector3d OpenTK.Vector3d.Add (Vector3d *a*, Vector3d *b*) [static]

Adds two vectors.

Parameters

- a* Left operand.
- b* Right operand.

Returns

Result of operation.

5.78.3.4 static void OpenTK.Vector3d.Add (ref Vector3d *a*, ref Vector3d *b*, out Vector3d *result*) [static]

Adds two vectors.

Parameters

- a* Left operand.
- b* Right operand.
- result* Result of operation.

5.78.3.5 static Vector3d OpenTK.Vector3d.BaryCentric (Vector3d *a*, Vector3d *b*, Vector3d *c*, double *u*, double *v*) [static]

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector
- b* Second input Vector
- c* Third input Vector
- u* First Barycentric Coordinate
- v* Second Barycentric Coordinate

Returns

a when $u=v=0$, *b* when $u=1,v=0$, *c* when $u=0,v=1$, and a linear combination of *a*,*b*,*c* otherwise

5.78.3.6 `static void OpenTK.Vector3d.BaryCentric (ref Vector3d a, ref Vector3d b, ref Vector3d c, double u, double v, out Vector3d result) [static]`

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector.
- b* Second input Vector.
- c* Third input Vector.
- u* First Barycentric Coordinate.
- v* Second Barycentric Coordinate.
- result* Output Vector. *a* when *u*=*v*=0, *b* when *u*=1,*v*=0, *c* when *u*=0,*v*=1, and a linear combination of *a*,*b*,*c* otherwise

5.78.3.7 `static double OpenTK.Vector3d.CalculateAngle (Vector3d first, Vector3d second) [static]`

Calculates the angle (in radians) between two vectors.

Parameters

- first* The first vector.
- second* The second vector.

Returns

Angle (in radians) between the vectors.

Note that the returned angle is never bigger than the constant Pi.

5.78.3.8 `static void OpenTK.Vector3d.CalculateAngle (ref Vector3d first, ref Vector3d second, out double result) [static]`

Calculates the angle (in radians) between two vectors.

Parameters

- first* The first vector.
- second* The second vector.
- result* Angle (in radians) between the vectors.

Note that the returned angle is never bigger than the constant Pi.

5.78.3.9 `static Vector3d OpenTK.Vector3d.Clamp (Vector3d vec, Vector3d min, Vector3d max) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector
min Minimum vector
max Maximum vector

Returns

The clamped vector

5.78.3.10 `static void OpenTK.Vector3d.Clamp (ref Vector3d vec, ref Vector3d min, ref Vector3d max, out Vector3d result) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector
min Minimum vector
max Maximum vector
result The clamped vector

5.78.3.11 `static Vector3d OpenTK.Vector3d.ComponentMax (Vector3d a, Vector3d b) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand
b Second operand

Returns

The component-wise maximum

5.78.3.12 `static void OpenTK.Vector3d.ComponentMax (ref Vector3d a, ref Vector3d b, out Vector3d result) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand
b Second operand
result The component-wise maximum

5.78.3.13 `static Vector3d OpenTK.Vector3d.ComponentMin (Vector3d a, Vector3d b) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

a First operand
b Second operand

Returns

The component-wise minimum

5.78.3.14 `static void OpenTK.Vector3d.ComponentMin (ref Vector3d a, ref Vector3d b, out Vector3d result) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

a First operand
b Second operand
result The component-wise minimum

5.78.3.15 `static Vector3d OpenTK.Vector3d.Cross (Vector3d left, Vector3d right) [static]`

Caclulate the cross (vector) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The cross product of the two inputs

5.78.3.16 `static void OpenTK.Vector3d.Cross (ref Vector3d left, ref Vector3d right, out Vector3d result) [static]`

Caclulate the cross (vector) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The cross product of the two inputs

Parameters

result The cross product of the two inputs

5.78.3.17 `static void OpenTK.Vector3d.Div (ref Vector3d a, double f, out Vector3d result) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the division

5.78.3.18 `static Vector3d OpenTK.Vector3d.Div (Vector3d a, double f) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand

f Scalar operand

Returns

Result of the division

5.78.3.19 void OpenTK.Vector3d.Div (double *f*)

Divide this instance by a scalar.

Parameters

f Scalar operand.

5.78.3.20 static Vector3d OpenTK.Vector3d.Divide (Vector3d *vector*, double *scale*) [static]

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.78.3.21 static void OpenTK.Vector3d.Divide (ref Vector3d *vector*, double *scale*, out Vector3d *result*) [static]

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.78.3.22 `static void OpenTK.Vector3d.Divide (ref Vector3d vector, ref Vector3d scale, out Vector3d result) [static]`

Divide a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.78.3.23 `static Vector3d OpenTK.Vector3d.Divide (Vector3d vector, Vector3d scale) [static]`

Divides a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.78.3.24 `static void OpenTK.Vector3d.Dot (ref Vector3d left, ref Vector3d right, out double result) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand

right Second operand

result The dot product of the two inputs

5.78.3.25 `static double OpenTK.Vector3d.Dot (Vector3d left, Vector3d right) [static]`

Calculate the dot (scalar) product of two vectors.

Parameters

left First operand
right Second operand

Returns

The dot product of the two inputs

5.78.3.26 bool OpenTK.Vector3d.Equals (Vector3d *other*)

Indicates whether the current vector is equal to another vector.

Parameters

other A vector to compare with this vector.

Returns

true if the current vector is equal to the vector parameter; otherwise, false.

5.78.3.27 override bool OpenTK.Vector3d.Equals (object *obj*)

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.78.3.28 override int OpenTK.Vector3d.GetHashCode ()

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.78.3.29 `static Vector3d OpenTK.Vector3d.Lerp (Vector3d a, Vector3d b, double blend) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

Returns

a when blend=0, b when blend=1, and a linear combination otherwise

5.78.3.30 `static void OpenTK.Vector3d.Lerp (ref Vector3d a, ref Vector3d b, double blend, out Vector3d result) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

result a when blend=0, b when blend=1, and a linear combination otherwise

5.78.3.31 `static Vector3d OpenTK.Vector3d.Max (Vector3d left, Vector3d right) [static]`

Returns the [Vector3d](#) with the minimum magnitude.

Parameters

left Left operand

right Right operand

Returns

The minimum [Vector3](#)

5.78.3.32 `static Vector3d OpenTK.Vector3d.Min (Vector3d left, Vector3d right) [static]`

Returns the [Vector3d](#) with the minimum magnitude.

Parameters

left Left operand
right Right operand

Returns

The minimum [Vector3](#)

5.78.3.33 `void OpenTK.Vector3d.Mult (double f)`

Multiply this instance by a scalar.

Parameters

f Scalar operand.

5.78.3.34 `static Vector3d OpenTK.Vector3d.Mult (Vector3d a, double f) [static]`

Multiply a vector and a scalar.

Parameters

a Vector operand
f Scalar operand

Returns

Result of the multiplication

5.78.3.35 `static void OpenTK.Vector3d.Mult (ref Vector3d a, double f, out Vector3d result) [static]`

Multiply a vector and a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the multiplication

5.78.3.36 `static Vector3d OpenTK.Vector3d.Multiply (Vector3d vector,
double scale) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.78.3.37 `static void OpenTK.Vector3d.Multiply (ref Vector3d vector, double
scale, out Vector3d result) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.78.3.38 `static void OpenTK.Vector3d.Multiply (ref Vector3d vector, ref
Vector3d scale, out Vector3d result) [static]`

Multiplies a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.78.3.39 `static Vector3d OpenTK.Vector3d.Multiply (Vector3d vector,
Vector3d scale) [static]`

Multiplies a vector by the components a vector (scale).

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.78.3.40 `static Vector3d OpenTK.Vector3d.Normalize (Vector3d vec)`
`[static]`

Scale a vector to unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.78.3.41 `static void OpenTK.Vector3d.Normalize (ref Vector3d vec, out Vector3d result)` `[static]`

Scale a vector to unit length.

Parameters

vec The input vector
result The normalized vector

5.78.3.42 `void OpenTK.Vector3d.Normalize ()`

Scales the [Vector3d](#) to unit length.

5.78.3.43 `void OpenTK.Vector3d.NormalizeFast ()`

Scales the [Vector3d](#) to approximately unit length.

5.78.3.44 `static void OpenTK.Vector3d.NormalizeFast (ref Vector3d vec, out Vector3d result) [static]`

Scale a vector to approximately unit length.

Parameters

vec The input vector

result The normalized vector

5.78.3.45 `static Vector3d OpenTK.Vector3d.NormalizeFast (Vector3d vec) [static]`

Scale a vector to approximately unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.78.3.46 `static OpenTK.Vector3d.operator Vector3 (Vector3d v3d) [explicit, static]`

Converts [OpenTK.Vector3d](#) to [OpenTK.Vector3](#).

Parameters

v3d The [Vector3d](#) to convert.

Returns

The resulting [Vector3](#).

5.78.3.47 `static OpenTK.Vector3d.operator Vector3d (Vector3 v3) [explicit, static]`

Converts [OpenTK.Vector3](#) to [OpenTK.Vector3d](#).

Parameters

v3 The [Vector3](#) to convert.

Returns

The resulting [Vector3d](#).

5.78.3.48 `static bool OpenTK.Vector3d.operator!= (Vector3d left, Vector3d right) [static]`

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.78.3.49 `static Vector3d OpenTK.Vector3d.operator* (double scale, Vector3d vec) [static]`

Multiplies an instance by a scalar.

Parameters

scale The scalar.

vec The instance.

Returns

The result of the calculation.

5.78.3.50 `static Vector3d OpenTK.Vector3d.operator* (Vector3d vec, double scale) [static]`

Multiplies an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.78.3.51 `static Vector3d OpenTK.Vector3d.operator+ (Vector3d left,
Vector3d right) [static]`

Adds two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.78.3.52 `static Vector3d OpenTK.Vector3d.operator- (Vector3d left,
Vector3d right) [static]`

Subtracts two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.78.3.53 `static Vector3d OpenTK.Vector3d.operator- (Vector3d vec)
[static]`

Negates an instance.

Parameters

vec The instance.

Returns

The result of the calculation.

5.78.3.54 static Vector3d OpenTK.Vector3d.operator/(Vector3d *vec*, double *scale*) [static]

Divides an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.78.3.55 static bool OpenTK.Vector3d.operator==(Vector3d *left*, Vector3d *right*) [static]

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.78.3.56 void OpenTK.Vector3d.Scale (Vector3d *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.78.3.57 void OpenTK.Vector3d.Scale (double *sx*, double *sy*, double *sz*)

Scales the current [Vector3d](#) by the given amounts.

Parameters

sx The scale of the X component.

sy The scale of the Y component.

sz The scale of the Z component.

5.78.3.58 void OpenTK.Vector3d.Scale (ref Vector3d *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.78.3.59 void OpenTK.Vector3d.Sub (ref Vector3d *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

**5.78.3.60 static Vector3d OpenTK.Vector3d.Sub (Vector3d *a*, Vector3d *b*)
[static]**

Subtract one Vector from another.

Parameters

a First operand

b Second operand

Returns

Result of subtraction

5.78.3.61 void OpenTK.Vector3d.Sub (Vector3d *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.78.3.62 `static void OpenTK.Vector3d.Sub (ref Vector3d a, ref Vector3d b,
out Vector3d result) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand
result Result of subtraction

5.78.3.63 `static Vector3d OpenTK.Vector3d.Subtract (Vector3d a, Vector3d
b) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand

Returns

Result of subtraction

5.78.3.64 `static void OpenTK.Vector3d.Subtract (ref Vector3d a, ref Vector3d
b, out Vector3d result) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand
result Result of subtraction

5.78.3.65 `override string OpenTK.Vector3d.ToString ()`

Returns a System.String that represents the current [Vector3](#).

Returns

5.78.3.66 `static void OpenTK.Vector3d.Transform (ref Vector3d vec, ref Quaterniond quat, out Vector3d result) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.

quat The quaternion to rotate the vector by.

result The result of the operation.

5.78.3.67 `static Vector3d OpenTK.Vector3d.Transform (Vector3d vec, Matrix4d mat) [static]`

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform

mat The desired transformation

Returns

The transformed vector

5.78.3.68 `static void OpenTK.Vector3d.Transform (ref Vector3d vec, ref Matrix4d mat, out Vector3d result) [static]`

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform

mat The desired transformation

result The transformed vector

5.78.3.69 `static Vector3d OpenTK.Vector3d.Transform (Vector3d vec, Quaterniond quat) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.
quat The quaternion to rotate the vector by.

Returns

The result of the operation.

5.78.3.70 static Vector3d OpenTK.Vector3d.TransformNormal (Vector3d *norm*, Matrix4d *mat*) [static]

Transform a Normal by the given Matrix.

This calculates the inverse of the given matrix, use TransformNormalInverse if you already have the inverse to avoid this extra calculation

Parameters

norm The normal to transform
mat The desired transformation

Returns

The transformed normal

5.78.3.71 static void OpenTK.Vector3d.TransformNormal (ref Vector3d *norm*, ref Matrix4d *mat*, out Vector3d *result*) [static]

Transform a Normal by the given Matrix.

This calculates the inverse of the given matrix, use TransformNormalInverse if you already have the inverse to avoid this extra calculation

Parameters

norm The normal to transform
mat The desired transformation
result The transformed normal

5.78.3.72 static Vector3d OpenTK.Vector3d.TransformNormalInverse (Vector3d *norm*, Matrix4d *invMat*) [static]

Transform a Normal by the (transpose of the) given Matrix.

This version doesn't calculate the inverse matrix. Use this version if you already have the inverse of the desired transform to hand

Parameters

norm The normal to transform
invMat The inverse of the desired transformation

Returns

The transformed normal

5.78.3.73 `static void OpenTK.Vector3d.TransformNormalInverse (ref Vector3d norm, ref Matrix4d invMat, out Vector3d result) [static]`

Transform a Normal by the (transpose of the) given Matrix.

This version doesn't calculate the inverse matrix. Use this version if you already have the inverse of the desired transform to hand

Parameters

norm The normal to transform
invMat The inverse of the desired transformation
result The transformed normal

5.78.3.74 `static Vector3d OpenTK.Vector3d.TransformPerspective (Vector3d vec, Matrix4d mat) [static]`

Transform a [Vector3d](#) by the given Matrix, and project the resulting [Vector4](#) back to a [Vector3](#).

Parameters

vec The vector to transform
mat The desired transformation

Returns

The transformed vector

5.78.3.75 `static void OpenTK.Vector3d.TransformPerspective (ref Vector3d vec, ref Matrix4d mat, out Vector3d result) [static]`

Transform a [Vector3d](#) by the given Matrix, and project the resulting [Vector4d](#) back to a [Vector3d](#).

Parameters

vec The vector to transform
mat The desired transformation
result The transformed vector

5.78.3.76 `static Vector3d OpenTK.Vector3d.TransformPosition (Vector3d pos, Matrix4d mat) [static]`

Transform a Position by the given Matrix.

Parameters

pos The position to transform
mat The desired transformation

Returns

The transformed position

5.78.3.77 `static void OpenTK.Vector3d.TransformPosition (ref Vector3d pos, ref Matrix4d mat, out Vector3d result) [static]`

Transform a Position by the given Matrix.

Parameters

pos The position to transform
mat The desired transformation
result The transformed position

5.78.3.78 `static Vector3d OpenTK.Vector3d.TransformVector (Vector3d vec, Matrix4d mat) [static]`

Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.

Parameters

vec The vector to transform
mat The desired transformation

Returns

The transformed vector

5.78.3.79 `static void OpenTK.Vector3d.TransformVector (ref Vector3d vec,
ref Matrix4d mat, out Vector3d result) [static]`

Transform a direction vector by the given Matrix Assumes the matrix has a bottom row of (0,0,0,1), that is the translation part is ignored.

Parameters

vec The vector to transform
mat The desired transformation
result The transformed vector

5.78.4 Member Data Documentation

5.78.4.1 `readonly Vector3d OpenTK.Vector3d.One = new Vector3d(1, 1, 1)
[static]`

Defines an instance with all components set to 1.

5.78.4.2 `readonly int OpenTK.Vector3d.SizeInBytes = Marshal.SizeOf(new
Vector3d()) [static]`

Defines the size of the [Vector3d](#) struct in bytes.

5.78.4.3 `readonly Vector3d OpenTK.Vector3d.UnitX = new Vector3d(1, 0, 0)
[static]`

Defines a unit-length [Vector3d](#) that points towards the X-axis.

5.78.4.4 `readonly Vector3d OpenTK.Vector3d.UnitY = new Vector3d(0, 1, 0)
[static]`

Defines a unit-length [Vector3d](#) that points towards the Y-axis.

5.78.4.5 `readonly Vector3d OpenTK.Vector3d.UnitZ = new Vector3d(0, 0, 1)`
`[static]`

/ Defines a unit-length [Vector3d](#) that points towards the Z-axis.

5.78.4.6 `double OpenTK.Vector3d.X`

The X component of the [Vector3](#).

5.78.4.7 `double OpenTK.Vector3d.Y`

The Y component of the [Vector3](#).

5.78.4.8 `double OpenTK.Vector3d.Z`

The Z component of the [Vector3](#).

5.78.4.9 `readonly Vector3d OpenTK.Vector3d.Zero = new Vector3d(0, 0, 0)`
`[static]`

Defines a zero-length [Vector3](#).

5.78.5 Property Documentation

5.78.5.1 `double OpenTK.Vector3d.Length` `[get]`

Gets the length (magnitude) of the vector.

[LengthFast](#)

See also

[LengthSquared](#)

5.78.5.2 `double OpenTK.Vector3d.LengthFast` `[get]`

Gets an approximation of the vector length (magnitude).

This property uses an approximation of the square root function to calculate vector magnitude, with an upper error bound of 0.001.

[Length](#)

See also

[LengthSquared](#)

5.78.5.3 double OpenTK.Vector3d.LengthSquared [get]

Gets the square of the vector length (magnitude).

This property avoids the costly square root operation required by the Length property. This makes it more suitable for comparisons.

[Length](#)

See also

[LengthFast](#)

5.78.5.4 Vector2d OpenTK.Vector3d.Xy [get, set]

Gets or sets an [OpenTK.Vector2d](#) with the X and Y components of this instance.

5.79 OpenTK.Vector3h Struct Reference

3-component Vector of the [Half](#) type. Occupies 6 Byte total.

Public Member Functions

- [Vector3h](#) ([Half](#) x, [Half](#) y, [Half](#) z)
The new Half3 instance will avoid conversion and copy directly from the [Half](#) parameters.
- [Vector3h](#) (Single x, Single y, Single z)
The new Half3 instance will convert the 3 parameters into 16-bit half-precision floating-point.
- [Vector3h](#) (Single x, Single y, Single z, bool throwOnError)
The new Half3 instance will convert the 3 parameters into 16-bit half-precision floating-point.
- [Vector3h](#) ([Vector3](#) v)
The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point.

- [Vector3h](#) ([Vector3](#) v, bool throwOnError)
The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point.
- [Vector3h](#) (ref [Vector3](#) v)
The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point. This is the fastest constructor.
- [Vector3h](#) (ref [Vector3](#) v, bool throwOnError)
The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point.
- [Vector3h](#) ([Vector3d](#) v)
The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point.
- [Vector3h](#) ([Vector3d](#) v, bool throwOnError)
The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point.
- [Vector3h](#) (ref [Vector3d](#) v)
The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point. This is the faster constructor.
- [Vector3h](#) (ref [Vector3d](#) v, bool throwOnError)
The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point.
- [Vector3 ToVector3](#) ()
Returns this Half3 instance's contents as [Vector3](#).
- [Vector3d ToVector3d](#) ()
Returns this Half3 instance's contents as [Vector3d](#).
- [Vector3h](#) (SerializationInfo info, StreamingContext context)
Constructor used by [ISerializable](#) to deserialize the object.
- void [GetObjectData](#) (SerializationInfo info, StreamingContext context)
Used by [ISerialize](#) to serialize the object.
- void [FromBinaryStream](#) (BinaryReader bin)
Updates the X,Y and Z components of this instance by reading from a [Stream](#).

- void [ToBinaryStream](#) (BinaryWriter bin)
Writes the X,Y and Z components of this instance into a Stream.
- bool [Equals](#) ([Vector3h](#) other)
Returns a value indicating whether this instance is equal to a specified OpenTK.Half3 vector.
- override string [ToString](#) ()
Returns a string that contains this Half3's numbers in human-legible form.

Static Public Member Functions

- static [operator Vector3h](#) ([Vector3](#) v3f)
Converts [OpenTK.Vector3](#) to [OpenTK.Half3](#).
- static [operator Vector3h](#) ([Vector3d](#) v3d)
Converts [OpenTK.Vector3d](#) to [OpenTK.Half3](#).
- static [operator Vector3](#) ([Vector3h](#) h3)
Converts [OpenTK.Half3](#) to [OpenTK.Vector3](#).
- static [operator Vector3d](#) ([Vector3h](#) h3)
Converts [OpenTK.Half3](#) to [OpenTK.Vector3d](#).
- static byte[] [GetBytes](#) ([Vector3h](#) h)
Returns the Half3 as an array of bytes.
- static [Vector3h FromBytes](#) (byte[] value, int startIndex)
Converts an array of bytes into Half3.

Public Attributes

- [Half X](#)
The X component of the Half3.
- [Half Y](#)
The Y component of the Half3.
- [Half Z](#)

The Z component of the Half3.

Static Public Attributes

- static readonly int [SizeInBytes](#) = 6

The size in bytes for an instance of the Half3 struct is 6.

Properties

- [Vector2h Xy](#) [get, set]

Gets or sets an [OpenTK.Vector2h](#) with the X and Y components of this instance.

5.79.1 Detailed Description

3-component Vector of the [Half](#) type. Occupies 6 Byte total.

5.79.2 Constructor & Destructor Documentation

5.79.2.1 OpenTK.Vector3h.Vector3h (Half x, Half y, Half z)

The new Half3 instance will avoid conversion and copy directly from the [Half](#) parameters.

Parameters

- x** An [Half](#) instance of a 16-bit half-precision floating-point number.
- y** An [Half](#) instance of a 16-bit half-precision floating-point number.
- z** An [Half](#) instance of a 16-bit half-precision floating-point number.

5.79.2.2 OpenTK.Vector3h.Vector3h (Single x, Single y, Single z)

The new Half3 instance will convert the 3 parameters into 16-bit half-precision floating-point.

Parameters

- x** 32-bit single-precision floating-point number.
- y** 32-bit single-precision floating-point number.
- z** 32-bit single-precision floating-point number.

5.79.2.3 OpenTK.Vector3h.Vector3h (Single x, Single y, Single z, bool throwOnError)

The new Half3 instance will convert the 3 parameters into 16-bit half-precision floating-point.

Parameters

- x** 32-bit single-precision floating-point number.
- y** 32-bit single-precision floating-point number.
- z** 32-bit single-precision floating-point number.
- throwOnError** Enable checks that will throw if the conversion result is not meaningful.

5.79.2.4 OpenTK.Vector3h.Vector3h (Vector3 v)

The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point.

Parameters

- v** [OpenTK.Vector3](#)

5.79.2.5 OpenTK.Vector3h.Vector3h (Vector3 v, bool throwOnError)

The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point.

Parameters

- v** [OpenTK.Vector3](#)
- throwOnError** Enable checks that will throw if the conversion result is not meaningful.

5.79.2.6 OpenTK.Vector3h.Vector3h (ref Vector3 v)

The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point. This is the fastest constructor.

Parameters

- v** [OpenTK.Vector3](#)

5.79.2.7 `OpenTK.Vector3h.Vector3h (ref Vector3 v, bool throwOnError)`

The new Half3 instance will convert the [Vector3](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector3](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.79.2.8 `OpenTK.Vector3h.Vector3h (Vector3d v)`

The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector3d](#)

5.79.2.9 `OpenTK.Vector3h.Vector3h (Vector3d v, bool throwOnError)`

The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector3d](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.79.2.10 `OpenTK.Vector3h.Vector3h (ref Vector3d v)`

The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point. This is the faster constructor.

Parameters

v [OpenTK.Vector3d](#)

5.79.2.11 OpenTK.Vector3h.Vector3h (ref Vector3d v, bool *throwOnError*)

The new Half3 instance will convert the [Vector3d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector3d](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.79.2.12 OpenTK.Vector3h.Vector3h (SerializationInfo *info*, StreamingContext *context*)

Constructor used by ISerializable to deserialize the object.

Parameters

info

context

5.79.3 Member Function Documentation

5.79.3.1 bool OpenTK.Vector3h.Equals (Vector3h *other*)

Returns a value indicating whether this instance is equal to a specified OpenTK.Half3 vector.

Parameters

other OpenTK.Half3 to compare to this instance..

Returns

True, if other is equal to this instance; false otherwise.

5.79.3.2 void OpenTK.Vector3h.FromBinaryStream (BinaryReader *bin*)

Updates the X,Y and Z components of this instance by reading from a Stream.

Parameters

bin A BinaryReader instance associated with an open Stream.

5.79.3.3 static Vector3h OpenTK.Vector3h.FromBytes (byte[] *value*, int *startIndex*) [static]

Converts an array of bytes into Half3.

Parameters

value A Half3 in it's byte[] representation.

startIndex The starting position within value.

Returns

A new Half3 instance.

5.79.3.4 static byte [] OpenTK.Vector3h.GetBytes (Vector3h *h*) [static]

Returns the Half3 as an array of bytes.

Parameters

h The Half3 to convert.

Returns

The input as byte array.

5.79.3.5 void OpenTK.Vector3h.GetObjectData (SerializationInfo *info*, StreamingContext *context*)

Used by ISerialize to serialize the object.

Parameters

info

context

5.79.3.6 static OpenTK.Vector3h.operator Vector3 (Vector3h *h3*) [explicit, static]

Converts OpenTK.Half3 to [OpenTK.Vector3](#).

Parameters

h3 The Half3 to convert.

Returns

The resulting [Vector3](#).

**5.79.3.7 static OpenTK.Vector3h.operator Vector3d (Vector3h *h3*)
[explicit, static]**

Converts OpenTK.Half3 to [OpenTK.Vector3d](#).

Parameters

h3 The Half3 to convert.

Returns

The resulting [Vector3d](#).

**5.79.3.8 static OpenTK.Vector3h.operator Vector3h (Vector3 *v3f*)
[explicit, static]**

Converts [OpenTK.Vector3](#) to OpenTK.Half3.

Parameters

v3f The [Vector3](#) to convert.

Returns

The resulting [Half](#) vector.

**5.79.3.9 static OpenTK.Vector3h.operator Vector3h (Vector3d *v3d*)
[explicit, static]**

Converts [OpenTK.Vector3d](#) to OpenTK.Half3.

Parameters

v3d The [Vector3d](#) to convert.

Returns

The resulting [Half](#) vector.

5.79.3.10 void OpenTK.Vector3h.ToBinaryStream (BinaryWriter *bin*)

Writes the X,Y and Z components of this instance into a Stream.

Parameters

bin A BinaryWriter instance associated with an open Stream.

5.79.3.11 override string OpenTK.Vector3h.ToString ()

Returns a string that contains this Half3's numbers in human-legible form.

5.79.3.12 Vector3 OpenTK.Vector3h.ToVector3 ()

Returns this Half3 instance's contents as [Vector3](#).

Returns

[OpenTK.Vector3](#)

5.79.3.13 Vector3d OpenTK.Vector3h.ToVector3d ()

Returns this Half3 instance's contents as [Vector3d](#).

5.79.4 Member Data Documentation**5.79.4.1 readonly int OpenTK.Vector3h.SizeInBytes = 6 [static]**

The size in bytes for an instance of the Half3 struct is 6.

5.79.4.2 Half OpenTK.Vector3h.X

The X component of the Half3.

5.79.4.3 Half OpenTK.Vector3h.Y

The Y component of the Half3.

5.79.4.4 Half OpenTK.Vector3h.Z

The Z component of the Half3.

5.79.5 Property Documentation

5.79.5.1 Vector2h OpenTK.Vector3h.Xy [get, set]

Gets or sets an [OpenTK.Vector2h](#) with the X and Y components of this instance.

5.80 OpenTK.Vector4 Struct Reference

Represents a 4D vector using four single-precision floating-point numbers.

Public Member Functions

- [Vector4](#) (float x, float y, float z, float w)
Constructs a new [Vector4](#).
- [Vector4](#) ([Vector2](#) v)
Constructs a new [Vector4](#) from the given [Vector2](#).
- [Vector4](#) ([Vector3](#) v)
Constructs a new [Vector4](#) from the given [Vector3](#).
- [Vector4](#) ([Vector3](#) v, float w)
Constructs a new [Vector4](#) from the specified [Vector3](#) and w component.
- [Vector4](#) ([Vector4](#) v)
Constructs a new [Vector4](#) from the given [Vector4](#).
- void [Add](#) ([Vector4](#) right)
Add the Vector passed as parameter to this instance.
- void [Add](#) (ref [Vector4](#) right)
Add the Vector passed as parameter to this instance.
- void [Sub](#) ([Vector4](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Sub](#) (ref [Vector4](#) right)
Subtract the Vector passed as parameter from this instance.
- void [Mult](#) (float f)
Multiply this instance by a scalar.

- void [Div](#) (float f)
Divide this instance by a scalar.
- void [Normalize](#) ()
Scales the [Vector4](#) to unit length.
- void [NormalizeFast](#) ()
Scales the [Vector4](#) to approximately unit length.
- void [Scale](#) (float sx, float sy, float sz, float sw)
Scales the current [Vector4](#) by the given amounts.
- void [Scale](#) ([Vector4](#) scale)
Scales this instance by the given parameter.
- void [Scale](#) (ref [Vector4](#) scale)
Scales this instance by the given parameter.
- override string [ToString](#) ()
Returns a System.String that represents the current [Vector4](#).
- override int [GetHashCode](#) ()
Returns the hashcode for this instance.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Vector4](#) other)
Indicates whether the current vector is equal to another vector.

Static Public Member Functions

- static [Vector4 Sub](#) ([Vector4](#) a, [Vector4](#) b)
Subtract one Vector from another.
- static void [Sub](#) (ref [Vector4](#) a, ref [Vector4](#) b, out [Vector4](#) result)
Subtract one Vector from another.
- static [Vector4 Mult](#) ([Vector4](#) a, float f)
Multiply a vector and a scalar.

- static void **Mult** (ref **Vector4** a, float f, out **Vector4** result)
Multiply a vector and a scalar.
- static **Vector4 Div** (**Vector4** a, float f)
Divide a vector by a scalar.
- static void **Div** (ref **Vector4** a, float f, out **Vector4** result)
Divide a vector by a scalar.
- static **Vector4 Add** (**Vector4** a, **Vector4** b)
Adds two vectors.
- static void **Add** (ref **Vector4** a, ref **Vector4** b, out **Vector4** result)
Adds two vectors.
- static **Vector4 Subtract** (**Vector4** a, **Vector4** b)
Subtract one Vector from another.
- static void **Subtract** (ref **Vector4** a, ref **Vector4** b, out **Vector4** result)
Subtract one Vector from another.
- static **Vector4 Multiply** (**Vector4** vector, float scale)
Multiplies a vector by a scalar.
- static void **Multiply** (ref **Vector4** vector, float scale, out **Vector4** result)
Multiplies a vector by a scalar.
- static **Vector4 Multiply** (**Vector4** vector, **Vector4** scale)
Multiplies a vector by the components a vector (scale).
- static void **Multiply** (ref **Vector4** vector, ref **Vector4** scale, out **Vector4** result)
Multiplies a vector by the components of a vector (scale).
- static **Vector4 Divide** (**Vector4** vector, float scale)
Divides a vector by a scalar.
- static void **Divide** (ref **Vector4** vector, float scale, out **Vector4** result)
Divides a vector by a scalar.
- static **Vector4 Divide** (**Vector4** vector, **Vector4** scale)
Divides a vector by the components of a vector (scale).

- static void [Divide](#) (ref [Vector4](#) vector, ref [Vector4](#) scale, out [Vector4](#) result)
Divide a vector by the components of a vector (scale).
- static [Vector4](#) [Min](#) ([Vector4](#) a, [Vector4](#) b)
Calculate the component-wise minimum of two vectors.
- static void [Min](#) (ref [Vector4](#) a, ref [Vector4](#) b, out [Vector4](#) result)
Calculate the component-wise minimum of two vectors.
- static [Vector4](#) [Max](#) ([Vector4](#) a, [Vector4](#) b)
Calculate the component-wise maximum of two vectors.
- static void [Max](#) (ref [Vector4](#) a, ref [Vector4](#) b, out [Vector4](#) result)
Calculate the component-wise maximum of two vectors.
- static [Vector4](#) [Clamp](#) ([Vector4](#) vec, [Vector4](#) min, [Vector4](#) max)
Clamp a vector to the given minimum and maximum vectors.
- static void [Clamp](#) (ref [Vector4](#) vec, ref [Vector4](#) min, ref [Vector4](#) max, out [Vector4](#) result)
Clamp a vector to the given minimum and maximum vectors.
- static [Vector4](#) [Normalize](#) ([Vector4](#) vec)
Scale a vector to unit length.
- static void [Normalize](#) (ref [Vector4](#) vec, out [Vector4](#) result)
Scale a vector to unit length.
- static [Vector4](#) [NormalizeFast](#) ([Vector4](#) vec)
Scale a vector to approximately unit length.
- static void [NormalizeFast](#) (ref [Vector4](#) vec, out [Vector4](#) result)
Scale a vector to approximately unit length.
- static float [Dot](#) ([Vector4](#) left, [Vector4](#) right)
Calculate the dot product of two vectors.
- static void [Dot](#) (ref [Vector4](#) left, ref [Vector4](#) right, out float result)
Calculate the dot product of two vectors.
- static [Vector4](#) [Lerp](#) ([Vector4](#) a, [Vector4](#) b, float blend)

Returns a new Vector that is the linear blend of the 2 given Vectors.

- static void **Lerp** (ref **Vector4** a, ref **Vector4** b, float blend, out **Vector4** result)
Returns a new Vector that is the linear blend of the 2 given Vectors.
- static **Vector4** **BaryCentric** (**Vector4** a, **Vector4** b, **Vector4** c, float u, float v)
Interpolate 3 Vectors using Barycentric coordinates.
- static void **BaryCentric** (ref **Vector4** a, ref **Vector4** b, ref **Vector4** c, float u, float v, out **Vector4** result)
Interpolate 3 Vectors using Barycentric coordinates.
- static **Vector4** **Transform** (**Vector4** vec, **Matrix4** mat)
Transform a Vector by the given Matrix.
- static void **Transform** (ref **Vector4** vec, ref **Matrix4** mat, out **Vector4** result)
Transform a Vector by the given Matrix.
- static **Vector4** **Transform** (**Vector4** vec, **Quaternion** quat)
Transforms a vector by a quaternion rotation.
- static void **Transform** (ref **Vector4** vec, ref **Quaternion** quat, out **Vector4** result)
Transforms a vector by a quaternion rotation.
- static **Vector4** **operator+** (**Vector4** left, **Vector4** right)
Adds two instances.
- static **Vector4** **operator-** (**Vector4** left, **Vector4** right)
Subtracts two instances.
- static **Vector4** **operator-** (**Vector4** vec)
Negates an instance.
- static **Vector4** **operator*** (**Vector4** vec, float scale)
Multiplies an instance by a scalar.
- static **Vector4** **operator*** (float scale, **Vector4** vec)
Multiplies an instance by a scalar.
- static **Vector4** **operator/** (**Vector4** vec, float scale)
Divides an instance by a scalar.
- static bool **operator==** (**Vector4** left, **Vector4** right)

Compares two instances for equality.

- static bool `operator!=` (`Vector4` left, `Vector4` right)
Compares two instances for inequality.
- unsafe static `operator float *` (`Vector4` v)
Returns a pointer to the first element of the specified instance.
- static `operator IntPtr` (`Vector4` v)
Returns a pointer to the first element of the specified instance.

Public Attributes

- float `X`
The X component of the `Vector4`.
- float `Y`
The Y component of the `Vector4`.
- float `Z`
The Z component of the `Vector4`.
- float `W`
The W component of the `Vector4`.

Static Public Attributes

- static `Vector4 UnitX` = new `Vector4`(1, 0, 0, 0)
Defines a unit-length `Vector4` that points towards the X-axis.
- static `Vector4 UnitY` = new `Vector4`(0, 1, 0, 0)
Defines a unit-length `Vector4` that points towards the Y-axis.
- static `Vector4 UnitZ` = new `Vector4`(0, 0, 1, 0)
Defines a unit-length `Vector4` that points towards the Z-axis.
- static `Vector4 UnitW` = new `Vector4`(0, 0, 0, 1)
Defines a unit-length `Vector4` that points towards the W-axis.
- static `Vector4 Zero` = new `Vector4`(0, 0, 0, 0)

Defines a zero-length [Vector4](#).

- static readonly [Vector4 One](#) = new [Vector4](#)(1, 1, 1, 1)
Defines an instance with all components set to 1.
- static readonly int [SizeInBytes](#) = Marshal.SizeOf(new [Vector4](#)())
Defines the size of the [Vector4](#) struct in bytes.

Properties

- float [Length](#) [get]
Gets the length (magnitude) of the vector.
- float [LengthFast](#) [get]
Gets an approximation of the vector length (magnitude).
- float [LengthSquared](#) [get]
Gets the square of the vector length (magnitude).
- [Vector2 Xy](#) [get, set]
Gets or sets an [OpenTK.Vector2](#) with the X and Y components of this instance.
- [Vector3 Xyz](#) [get, set]
Gets or sets an [OpenTK.Vector3](#) with the X, Y and Z components of this instance.

5.80.1 Detailed Description

Represents a 4D vector using four single-precision floating-point numbers. The [Vector4](#) structure is suitable for interoperation with unmanaged code requiring four consecutive floats.

5.80.2 Constructor & Destructor Documentation

5.80.2.1 OpenTK.Vector4.Vector4 (float x, float y, float z, float w)

Constructs a new [Vector4](#).

Parameters

- x** The x component of the [Vector4](#).

y The y component of the [Vector4](#).

z The z component of the [Vector4](#).

w The w component of the [Vector4](#).

5.80.2.2 `OpenTK.Vector4.Vector4 (Vector2 v)`

Constructs a new [Vector4](#) from the given [Vector2](#).

Parameters

v The [Vector2](#) to copy components from.

5.80.2.3 `OpenTK.Vector4.Vector4 (Vector3 v)`

Constructs a new [Vector4](#) from the given [Vector3](#).

Parameters

v The [Vector3](#) to copy components from.

5.80.2.4 `OpenTK.Vector4.Vector4 (Vector3 v, float w)`

Constructs a new [Vector4](#) from the specified [Vector3](#) and w component.

Parameters

v The [Vector3](#) to copy components from.

w The w component of the new [Vector4](#).

5.80.2.5 `OpenTK.Vector4.Vector4 (Vector4 v)`

Constructs a new [Vector4](#) from the given [Vector4](#).

Parameters

v The [Vector4](#) to copy components from.

5.80.3 Member Function Documentation

5.80.3.1 void OpenTK.Vector4.Add (Vector4 *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

5.80.3.2 void OpenTK.Vector4.Add (ref Vector4 *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

5.80.3.3 static Vector4 OpenTK.Vector4.Add (Vector4 *a*, Vector4 *b*) [static]

Adds two vectors.

Parameters

a Left operand.

b Right operand.

Returns

Result of operation.

5.80.3.4 static void OpenTK.Vector4.Add (ref Vector4 *a*, ref Vector4 *b*, out Vector4 *result*) [static]

Adds two vectors.

Parameters

a Left operand.

b Right operand.

result Result of operation.

5.80.3.5 static Vector4 OpenTK.Vector4.BaryCentric (Vector4 *a*, Vector4 *b*, Vector4 *c*, float *u*, float *v*) [static]

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector
- b* Second input Vector
- c* Third input Vector
- u* First Barycentric Coordinate
- v* Second Barycentric Coordinate

Returns

a when $u=v=0$, *b* when $u=1,v=0$, *c* when $u=0,v=1$, and a linear combination of *a*,*b*,*c* otherwise

5.80.3.6 static void OpenTK.Vector4.BaryCentric (ref Vector4 *a*, ref Vector4 *b*, ref Vector4 *c*, float *u*, float *v*, out Vector4 *result*) [static]

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

- a* First input Vector.
- b* Second input Vector.
- c* Third input Vector.
- u* First Barycentric Coordinate.
- v* Second Barycentric Coordinate.
- result* Output Vector. *a* when $u=v=0$, *b* when $u=1,v=0$, *c* when $u=0,v=1$, and a linear combination of *a*,*b*,*c* otherwise

5.80.3.7 static Vector4 OpenTK.Vector4.Clamp (Vector4 *vec*, Vector4 *min*, Vector4 *max*) [static]

Clamp a vector to the given minimum and maximum vectors.

Parameters

- vec* [Input](#) vector

min Minimum vector
max Maximum vector

Returns

The clamped vector

5.80.3.8 `static void OpenTK.Vector4.Clamp (ref Vector4 vec, ref Vector4 min, ref Vector4 max, out Vector4 result) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector
min Minimum vector
max Maximum vector
result The clamped vector

5.80.3.9 `static void OpenTK.Vector4.Div (ref Vector4 a, float f, out Vector4 result) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the division

5.80.3.10 `static Vector4 OpenTK.Vector4.Div (Vector4 a, float f) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand

Returns

Result of the division

5.80.3.11 void OpenTK.Vector4.Div (float *f*)

Divide this instance by a scalar.

Parameters

f Scalar operand.

**5.80.3.12 static Vector4 OpenTK.Vector4.Divide (Vector4 *vector*, float *scale*)
[static]**

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

**5.80.3.13 static void OpenTK.Vector4.Divide (ref Vector4 *vector*, float *scale*,
out Vector4 *result*) [static]**

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

**5.80.3.14 static Vector4 OpenTK.Vector4.Divide (Vector4 *vector*, Vector4
scale) [static]**

Divides a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.80.3.15 `static void OpenTK.Vector4.Divide (ref Vector4 vector, ref Vector4 scale, out Vector4 result) [static]`

Divide a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.80.3.16 `static void OpenTK.Vector4.Dot (ref Vector4 left, ref Vector4 right, out float result) [static]`

Calculate the dot product of two vectors.

Parameters

left First operand

right Second operand

result The dot product of the two inputs

5.80.3.17 `static float OpenTK.Vector4.Dot (Vector4 left, Vector4 right) [static]`

Calculate the dot product of two vectors.

Parameters

left First operand

right Second operand

Returns

The dot product of the two inputs

5.80.3.18 `override bool OpenTK.Vector4.Equals (object obj)`

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.80.3.19 bool OpenTK.Vector4.Equals (Vector4 *other*)

Indicates whether the current vector is equal to another vector.

Parameters

other A vector to compare with this vector.

Returns

true if the current vector is equal to the vector parameter; otherwise, false.

5.80.3.20 override int OpenTK.Vector4.GetHashCode ()

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.80.3.21 static Vector4 OpenTK.Vector4.Lerp (Vector4 *a*, Vector4 *b*, float *blend*) [static]

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

Returns

a when blend=0, b when blend=1, and a linear combination otherwise

5.80.3.22 `static void OpenTK.Vector4.Lerp (ref Vector4 a, ref Vector4 b, float blend, out Vector4 result) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

result a when blend=0, b when blend=1, and a linear combination otherwise

5.80.3.23 `static Vector4 OpenTK.Vector4.Max (Vector4 a, Vector4 b) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

Returns

The component-wise maximum

5.80.3.24 `static void OpenTK.Vector4.Max (ref Vector4 a, ref Vector4 b, out Vector4 result) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

result The component-wise maximum

5.80.3.25 `static void OpenTK.Vector4.Min (ref Vector4 a, ref Vector4 b, out Vector4 result) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

a First operand
b Second operand
result The component-wise minimum

5.80.3.26 `static Vector4 OpenTK.Vector4.Min (Vector4 a, Vector4 b)`
`[static]`

Calculate the component-wise minimum of two vectors.

Parameters

a First operand
b Second operand

Returns

The component-wise minimum

5.80.3.27 `void OpenTK.Vector4.Mult (float f)`

Multiply this instance by a scalar.

Parameters

f Scalar operand.

5.80.3.28 `static void OpenTK.Vector4.Mult (ref Vector4 a, float f, out Vector4 result)`
`[static]`

Multiply a vector and a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the multiplication

5.80.3.29 `static Vector4 OpenTK.Vector4.Mult (Vector4 a, float f)`
`[static]`

Multiply a vector and a scalar.

Parameters

a Vector operand

f Scalar operand

Returns

Result of the multiplication

5.80.3.30 `static Vector4 OpenTK.Vector4.Multiply (Vector4 vector, float scale`
`) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.80.3.31 `static void OpenTK.Vector4.Multiply (ref Vector4 vector, float`
`scale, out Vector4 result) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.80.3.32 `static void OpenTK.Vector4.Multiply (ref Vector4 vector, ref Vector4 scale, out Vector4 result) [static]`

Multiplies a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.80.3.33 `static Vector4 OpenTK.Vector4.Multiply (Vector4 vector, Vector4 scale) [static]`

Multiplies a vector by the components a vector (scale).

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.80.3.34 `static Vector4 OpenTK.Vector4.Normalize (Vector4 vec) [static]`

Scale a vector to unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.80.3.35 `static void OpenTK.Vector4.Normalize (ref Vector4 vec, out Vector4 result) [static]`

Scale a vector to unit length.

Parameters

vec The input vector

result The normalized vector

5.80.3.36 void OpenTK.Vector4.Normalize ()

Scales the [Vector4](#) to unit length.

5.80.3.37 void OpenTK.Vector4.NormalizeFast ()

Scales the [Vector4](#) to approximately unit length.

**5.80.3.38 static Vector4 OpenTK.Vector4.NormalizeFast (Vector4 *vec*)
[static]**

Scale a vector to approximately unit length.

Parameters

vec The input vector

Returns

The normalized vector

**5.80.3.39 static void OpenTK.Vector4.NormalizeFast (ref Vector4 *vec*, out
Vector4 *result*) [static]**

Scale a vector to approximately unit length.

Parameters

vec The input vector

result The normalized vector

**5.80.3.40 unsafe static OpenTK.Vector4.operator float * (Vector4 *v*)
[explicit, static]**

Returns a pointer to the first element of the specified instance.

Parameters

v The instance.

Returns

A pointer to the first element of *v*.

5.80.3.41 `static OpenTK.Vector4.operator IntPtr (Vector4 v) [explicit, static]`

Returns a pointer to the first element of the specified instance.

Parameters

v The instance.

Returns

A pointer to the first element of *v*.

5.80.3.42 `static bool OpenTK.Vector4.operator!= (Vector4 left, Vector4 right) [static]`

Compares two instances for inequality.

Parameters

left The first instance.

right The second instance.

Returns

True, if *left* does not equal *right*; false otherwise.

5.80.3.43 `static Vector4 OpenTK.Vector4.operator* (Vector4 vec, float scale) [static]`

Multiplies an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.80.3.44 `static Vector4 OpenTK.Vector4.operator* (float scale, Vector4 vec) [static]`

Multiplies an instance by a scalar.

Parameters

scale The scalar.

vec The instance.

Returns

The result of the calculation.

5.80.3.45 `static Vector4 OpenTK.Vector4.operator+ (Vector4 left, Vector4 right) [static]`

Adds two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.80.3.46 `static Vector4 OpenTK.Vector4.operator- (Vector4 vec) [static]`

Negates an instance.

Parameters

vec The instance.

Returns

The result of the calculation.

5.80.3.47 static Vector4 OpenTK.Vector4.operator- (Vector4 *left*, Vector4 *right*) [static]

Subtracts two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.80.3.48 static Vector4 OpenTK.Vector4.operator/ (Vector4 *vec*, float *scale*) [static]

Divides an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.80.3.49 static bool OpenTK.Vector4.operator== (Vector4 *left*, Vector4 *right*) [static]

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.80.3.50 void OpenTK.Vector4.Scale (ref Vector4 *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.80.3.51 void OpenTK.Vector4.Scale (Vector4 *scale*)

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.80.3.52 void OpenTK.Vector4.Scale (float *sx*, float *sy*, float *sz*, float *sw*)

Scales the current [Vector4](#) by the given amounts.

Parameters

sx The scale of the X component.

sy The scale of the Y component.

sz The scale of the Z component.

sw The scale of the Z component.

5.80.3.53 void OpenTK.Vector4.Sub (ref Vector4 *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.80.3.54 void OpenTK.Vector4.Sub (Vector4 *right*)

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.80.3.55 `static void OpenTK.Vector4.Sub (ref Vector4 a, ref Vector4 b, out Vector4 result) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand
result Result of subtraction

5.80.3.56 `static Vector4 OpenTK.Vector4.Sub (Vector4 a, Vector4 b) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand

Returns

Result of subtraction

5.80.3.57 `static Vector4 OpenTK.Vector4.Subtract (Vector4 a, Vector4 b) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand

Returns

Result of subtraction

5.80.3.58 `static void OpenTK.Vector4.Subtract (ref Vector4 a, ref Vector4 b,
out Vector4 result) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand
result Result of subtraction

5.80.3.59 `override string OpenTK.Vector4.ToString ()`

Returns a System.String that represents the current [Vector4](#).

Returns

5.80.3.60 `static void OpenTK.Vector4.Transform (ref Vector4 vec, ref
Matrix4 mat, out Vector4 result) [static]`

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform
mat The desired transformation
result The transformed vector

5.80.3.61 `static void OpenTK.Vector4.Transform (ref Vector4 vec, ref
Quaternion quat, out Vector4 result) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.
quat The quaternion to rotate the vector by.
result The result of the operation.

5.80.3.62 static Vector4 OpenTK.Vector4.Transform (Vector4 *vec*, Quaternion *quat*) [static]

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.

quat The quaternion to rotate the vector by.

Returns

The result of the operation.

5.80.3.63 static Vector4 OpenTK.Vector4.Transform (Vector4 *vec*, Matrix4 *mat*) [static]

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform

mat The desired transformation

Returns

The transformed vector

5.80.4 Member Data Documentation

5.80.4.1 readonly Vector4 OpenTK.Vector4.One = new Vector4(1, 1, 1, 1) [static]

Defines an instance with all components set to 1.

5.80.4.2 readonly int OpenTK.Vector4.SizeInBytes = Marshal.SizeOf(new Vector4()) [static]

Defines the size of the [Vector4](#) struct in bytes.

5.80.4.3 Vector4 OpenTK.Vector4.UnitW = new Vector4(0, 0, 0, 1) [static]

Defines a unit-length [Vector4](#) that points towards the W-axis.

5.80.4.4 Vector4 OpenTK.Vector4.UnitX = new Vector4(1, 0, 0, 0) [static]

Defines a unit-length [Vector4](#) that points towards the X-axis.

5.80.4.5 Vector4 OpenTK.Vector4.UnitY = new Vector4(0, 1, 0, 0) [static]

Defines a unit-length [Vector4](#) that points towards the Y-axis.

5.80.4.6 Vector4 OpenTK.Vector4.UnitZ = new Vector4(0, 0, 1, 0) [static]

Defines a unit-length [Vector4](#) that points towards the Z-axis.

5.80.4.7 float OpenTK.Vector4.W

The W component of the [Vector4](#).

5.80.4.8 float OpenTK.Vector4.X

The X component of the [Vector4](#).

5.80.4.9 float OpenTK.Vector4.Y

The Y component of the [Vector4](#).

5.80.4.10 float OpenTK.Vector4.Z

The Z component of the [Vector4](#).

5.80.4.11 Vector4 OpenTK.Vector4.Zero = new Vector4(0, 0, 0, 0) [static]

Defines a zero-length [Vector4](#).

5.80.5 Property Documentation**5.80.5.1 float OpenTK.Vector4.Length [get]**

Gets the length (magnitude) of the vector.

[LengthFast](#)

See also

[LengthSquared](#)

5.80.5.2 float `OpenTK.Vector4.LengthFast` [`get`]

Gets an approximation of the vector length (magnitude).

This property uses an approximation of the square root function to calculate vector magnitude, with an upper error bound of 0.001.

[Length](#)

See also

[LengthSquared](#)

5.80.5.3 float `OpenTK.Vector4.LengthSquared` [`get`]

Gets the square of the vector length (magnitude).

This property avoids the costly square root operation required by the `Length` property. This makes it more suitable for comparisons.

[Length](#)

See also

[LengthFast](#)

5.80.5.4 Vector2 `OpenTK.Vector4.Xy` [`get`, `set`]

Gets or sets an [OpenTK.Vector2](#) with the X and Y components of this instance.

5.80.5.5 Vector3 `OpenTK.Vector4.Xyz` [`get`, `set`]

Gets or sets an [OpenTK.Vector3](#) with the X, Y and Z components of this instance.

5.81 `OpenTK.Vector4d` Struct Reference

Represents a 4D vector using four double-precision floating-point numbers.

Public Member Functions

- **Vector4d** (double x, double y, double z, double w)
Constructs a new [Vector4d](#).
- **Vector4d** ([Vector2d](#) v)
Constructs a new [Vector4d](#) from the given [Vector2d](#).
- **Vector4d** ([Vector3d](#) v)
Constructs a new [Vector4d](#) from the given [Vector3d](#).
- **Vector4d** ([Vector3d](#) v, double w)
Constructs a new [Vector4d](#) from the specified [Vector3d](#) and w component.
- **Vector4d** ([Vector4d](#) v)
Constructs a new [Vector4d](#) from the given [Vector4d](#).
- void **Add** ([Vector4d](#) right)
Add the Vector passed as parameter to this instance.
- void **Add** (ref [Vector4d](#) right)
Add the Vector passed as parameter to this instance.
- void **Sub** ([Vector4d](#) right)
Subtract the Vector passed as parameter from this instance.
- void **Sub** (ref [Vector4d](#) right)
Subtract the Vector passed as parameter from this instance.
- void **Mult** (double f)
Multiply this instance by a scalar.
- void **Div** (double f)
Divide this instance by a scalar.
- void **Normalize** ()
Scales the [Vector4d](#) to unit length.
- void **NormalizeFast** ()
Scales the [Vector4d](#) to approximately unit length.
- void **Scale** (double sx, double sy, double sz, double sw)
Scales the current [Vector4d](#) by the given amounts.

- void [Scale](#) ([Vector4d](#) scale)
Scales this instance by the given parameter.
- void [Scale](#) (ref [Vector4d](#) scale)
Scales this instance by the given parameter.
- override string [ToString](#) ()
Returns a System.String that represents the current [Vector4d](#).
- override int [GetHashCode](#) ()
Returns the hashcode for this instance.
- override bool [Equals](#) (object obj)
Indicates whether this instance and a specified object are equal.
- bool [Equals](#) ([Vector4d](#) other)
Indicates whether the current vector is equal to another vector.

Static Public Member Functions

- static [Vector4d Sub](#) ([Vector4d](#) a, [Vector4d](#) b)
Subtract one Vector from another.
- static void [Sub](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, out [Vector4d](#) result)
Subtract one Vector from another.
- static [Vector4d Mult](#) ([Vector4d](#) a, double f)
Multiply a vector and a scalar.
- static void [Mult](#) (ref [Vector4d](#) a, double f, out [Vector4d](#) result)
Multiply a vector and a scalar.
- static [Vector4d Div](#) ([Vector4d](#) a, double f)
Divide a vector by a scalar.
- static void [Div](#) (ref [Vector4d](#) a, double f, out [Vector4d](#) result)
Divide a vector by a scalar.
- static [Vector4d Add](#) ([Vector4d](#) a, [Vector4d](#) b)
Adds two vectors.

- static void [Add](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, out [Vector4d](#) result)
Adds two vectors.
- static [Vector4d Subtract](#) ([Vector4d](#) a, [Vector4d](#) b)
Subtract one Vector from another.
- static void [Subtract](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, out [Vector4d](#) result)
Subtract one Vector from another.
- static [Vector4d Multiply](#) ([Vector4d](#) vector, double scale)
Multiplies a vector by a scalar.
- static void [Multiply](#) (ref [Vector4d](#) vector, double scale, out [Vector4d](#) result)
Multiplies a vector by a scalar.
- static [Vector4d Multiply](#) ([Vector4d](#) vector, [Vector4d](#) scale)
Multiplies a vector by the components a vector (scale).
- static void [Multiply](#) (ref [Vector4d](#) vector, ref [Vector4d](#) scale, out [Vector4d](#) result)
Multiplies a vector by the components of a vector (scale).
- static [Vector4d Divide](#) ([Vector4d](#) vector, double scale)
Divides a vector by a scalar.
- static void [Divide](#) (ref [Vector4d](#) vector, double scale, out [Vector4d](#) result)
Divides a vector by a scalar.
- static [Vector4d Divide](#) ([Vector4d](#) vector, [Vector4d](#) scale)
Divides a vector by the components of a vector (scale).
- static void [Divide](#) (ref [Vector4d](#) vector, ref [Vector4d](#) scale, out [Vector4d](#) result)
Divide a vector by the components of a vector (scale).
- static [Vector4d Min](#) ([Vector4d](#) a, [Vector4d](#) b)
Calculate the component-wise minimum of two vectors.
- static void [Min](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, out [Vector4d](#) result)
Calculate the component-wise minimum of two vectors.
- static [Vector4d Max](#) ([Vector4d](#) a, [Vector4d](#) b)

Calculate the component-wise maximum of two vectors.

- static void [Max](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, out [Vector4d](#) result)

Calculate the component-wise maximum of two vectors.

- static [Vector4d](#) [Clamp](#) ([Vector4d](#) vec, [Vector4d](#) min, [Vector4d](#) max)

Clamp a vector to the given minimum and maximum vectors.

- static void [Clamp](#) (ref [Vector4d](#) vec, ref [Vector4d](#) min, ref [Vector4d](#) max, out [Vector4d](#) result)

Clamp a vector to the given minimum and maximum vectors.

- static [Vector4d](#) [Normalize](#) ([Vector4d](#) vec)

Scale a vector to unit length.

- static void [Normalize](#) (ref [Vector4d](#) vec, out [Vector4d](#) result)

Scale a vector to unit length.

- static [Vector4d](#) [NormalizeFast](#) ([Vector4d](#) vec)

Scale a vector to approximately unit length.

- static void [NormalizeFast](#) (ref [Vector4d](#) vec, out [Vector4d](#) result)

Scale a vector to approximately unit length.

- static double [Dot](#) ([Vector4d](#) left, [Vector4d](#) right)

Calculate the dot product of two vectors.

- static void [Dot](#) (ref [Vector4d](#) left, ref [Vector4d](#) right, out double result)

Calculate the dot product of two vectors.

- static [Vector4d](#) [Lerp](#) ([Vector4d](#) a, [Vector4d](#) b, double blend)

Returns a new Vector that is the linear blend of the 2 given Vectors.

- static void [Lerp](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, double blend, out [Vector4d](#) result)

Returns a new Vector that is the linear blend of the 2 given Vectors.

- static [Vector4d](#) [BaryCentric](#) ([Vector4d](#) a, [Vector4d](#) b, [Vector4d](#) c, double u, double v)

Interpolate 3 Vectors using Barycentric coordinates.

- static void [BaryCentric](#) (ref [Vector4d](#) a, ref [Vector4d](#) b, ref [Vector4d](#) c, double u, double v, out [Vector4d](#) result)

Interpolate 3 Vectors using Barycentric coordinates.

- static [Vector4d Transform](#) ([Vector4d](#) vec, [Matrix4d](#) mat)
Transform a Vector by the given Matrix.
- static void [Transform](#) (ref [Vector4d](#) vec, ref [Matrix4d](#) mat, out [Vector4d](#) result)
Transform a Vector by the given Matrix.
- static [Vector4d Transform](#) ([Vector4d](#) vec, [Quaterniond](#) quat)
Transforms a vector by a quaternion rotation.
- static void [Transform](#) (ref [Vector4d](#) vec, ref [Quaterniond](#) quat, out [Vector4d](#) result)
Transforms a vector by a quaternion rotation.
- static [Vector4d operator+](#) ([Vector4d](#) left, [Vector4d](#) right)
Adds two instances.
- static [Vector4d operator-](#) ([Vector4d](#) left, [Vector4d](#) right)
Subtracts two instances.
- static [Vector4d operator-](#) ([Vector4d](#) vec)
Negates an instance.
- static [Vector4d operator*](#) ([Vector4d](#) vec, double scale)
Multiplies an instance by a scalar.
- static [Vector4d operator*](#) (double scale, [Vector4d](#) vec)
Multiplies an instance by a scalar.
- static [Vector4d operator/](#) ([Vector4d](#) vec, double scale)
Divides an instance by a scalar.
- static bool [operator==](#) ([Vector4d](#) left, [Vector4d](#) right)
Compares two instances for equality.
- static bool [operator!=](#) ([Vector4d](#) left, [Vector4d](#) right)
Compares two instances for inequality.
- unsafe static [operator double *](#) ([Vector4d](#) v)
Returns a pointer to the first element of the specified instance.
- static [operator IntPtr](#) ([Vector4d](#) v)

Returns a pointer to the first element of the specified instance.

- static [operator Vector4d](#) ([Vector4](#) v4)
Converts [OpenTK.Vector4](#) to [OpenTK.Vector4d](#).
- static [operator Vector4](#) ([Vector4d](#) v4d)
Converts [OpenTK.Vector4d](#) to [OpenTK.Vector4](#).

Public Attributes

- double [X](#)
The X component of the [Vector4d](#).
- double [Y](#)
The Y component of the [Vector4d](#).
- double [Z](#)
The Z component of the [Vector4d](#).
- double [W](#)
The W component of the [Vector4d](#).

Static Public Attributes

- static [Vector4d UnitX](#) = new [Vector4d](#)(1, 0, 0, 0)
Defines a unit-length [Vector4d](#) that points towards the X-axis.
- static [Vector4d UnitY](#) = new [Vector4d](#)(0, 1, 0, 0)
Defines a unit-length [Vector4d](#) that points towards the Y-axis.
- static [Vector4d UnitZ](#) = new [Vector4d](#)(0, 0, 1, 0)
Defines a unit-length [Vector4d](#) that points towards the Z-axis.
- static [Vector4d UnitW](#) = new [Vector4d](#)(0, 0, 0, 1)
Defines a unit-length [Vector4d](#) that points towards the W-axis.
- static [Vector4d Zero](#) = new [Vector4d](#)(0, 0, 0, 0)
Defines a zero-length [Vector4d](#).
- static readonly [Vector4d One](#) = new [Vector4d](#)(1, 1, 1, 1)

Defines an instance with all components set to 1.

- static readonly int [SizeInBytes](#) = Marshal.SizeOf(new [Vector4d](#)())

Defines the size of the [Vector4d](#) struct in bytes.

Properties

- double [Length](#) [get]
Gets the length (magnitude) of the vector.
- double [LengthFast](#) [get]
Gets an approximation of the vector length (magnitude).
- double [LengthSquared](#) [get]
Gets the square of the vector length (magnitude).
- [Vector2d Xy](#) [get, set]
Gets or sets an [OpenTK.Vector2d](#) with the X and Y components of this instance.
- [Vector3d Xyz](#) [get, set]
Gets or sets an [OpenTK.Vector3d](#) with the X, Y and Z components of this instance.

5.81.1 Detailed Description

Represents a 4D vector using four double-precision floating-point numbers.

5.81.2 Constructor & Destructor Documentation

5.81.2.1 [OpenTK.Vector4d.Vector4d](#) (double *x*, double *y*, double *z*, double *w*)

Constructs a new [Vector4d](#).

Parameters

- x* The x component of the [Vector4d](#).
- y* The y component of the [Vector4d](#).
- z* The z component of the [Vector4d](#).
- w* The w component of the [Vector4d](#).

5.81.2.2 OpenTK.Vector4d.Vector4d (Vector2d *v*)

Constructs a new [Vector4d](#) from the given [Vector2d](#).

Parameters

- v* The [Vector2d](#) to copy components from.

5.81.2.3 OpenTK.Vector4d.Vector4d (Vector3d *v*)

Constructs a new [Vector4d](#) from the given [Vector3d](#).

Parameters

- v* The [Vector3d](#) to copy components from.

5.81.2.4 OpenTK.Vector4d.Vector4d (Vector3d *v*, double *w*)

Constructs a new [Vector4d](#) from the specified [Vector3d](#) and *w* component.

Parameters

- v* The [Vector3d](#) to copy components from.
- w* The *w* component of the new [Vector4](#).

5.81.2.5 OpenTK.Vector4d.Vector4d (Vector4d *v*)

Constructs a new [Vector4d](#) from the given [Vector4d](#).

Parameters

- v* The [Vector4d](#) to copy components from.

5.81.3 Member Function Documentation**5.81.3.1 void OpenTK.Vector4d.Add (Vector4d *right*)**

Add the Vector passed as parameter to this instance.

Parameters

- right* Right operand. This parameter is only read from.

5.81.3.2 void OpenTK.Vector4d.Add (ref Vector4d *right*)

Add the Vector passed as parameter to this instance.

Parameters

right Right operand. This parameter is only read from.

5.81.3.3 static Vector4d OpenTK.Vector4d.Add (Vector4d *a*, Vector4d *b*) [static]

Adds two vectors.

Parameters

a Left operand.

b Right operand.

Returns

Result of operation.

5.81.3.4 static void OpenTK.Vector4d.Add (ref Vector4d *a*, ref Vector4d *b*, out Vector4d *result*) [static]

Adds two vectors.

Parameters

a Left operand.

b Right operand.

result Result of operation.

5.81.3.5 static Vector4d OpenTK.Vector4d.BaryCentric (Vector4d *a*, Vector4d *b*, Vector4d *c*, double *u*, double *v*) [static]

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

a First input Vector

b Second input Vector

c Third input Vector
u First Barycentric Coordinate
v Second Barycentric Coordinate

Returns

a when $u=v=0$, b when $u=1,v=0$, c when $u=0,v=1$, and a linear combination of a,b,c otherwise

5.81.3.6 `static void OpenTK.Vector4d.BaryCentric (ref Vector4d a, ref Vector4d b, ref Vector4d c, double u, double v, out Vector4d result) [static]`

Interpolate 3 Vectors using Barycentric coordinates.

Parameters

a First input Vector.
b Second input Vector.
c Third input Vector.
u First Barycentric Coordinate.
v Second Barycentric Coordinate.
result Output Vector. a when $u=v=0$, b when $u=1,v=0$, c when $u=0,v=1$, and a linear combination of a,b,c otherwise

5.81.3.7 `static Vector4d OpenTK.Vector4d.Clamp (Vector4d vec, Vector4d min, Vector4d max) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec [Input](#) vector
min Minimum vector
max Maximum vector

Returns

The clamped vector

5.81.3.8 `static void OpenTK.Vector4d.Clamp (ref Vector4d vec, ref Vector4d min, ref Vector4d max, out Vector4d result) [static]`

Clamp a vector to the given minimum and maximum vectors.

Parameters

vec Input vector
min Minimum vector
max Maximum vector
result The clamped vector

5.81.3.9 `static void OpenTK.Vector4d.Div (ref Vector4d a, double f, out Vector4d result) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand
result Result of the division

5.81.3.10 `static Vector4d OpenTK.Vector4d.Div (Vector4d a, double f) [static]`

Divide a vector by a scalar.

Parameters

a Vector operand
f Scalar operand

Returns

Result of the division

5.81.3.11 `void OpenTK.Vector4d.Div (double f)`

Divide this instance by a scalar.

Parameters

f Scalar operand.

5.81.3.12 static Vector4d OpenTK.Vector4d.Divide (Vector4d *vector*, double *scale*) [static]

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.81.3.13 static void OpenTK.Vector4d.Divide (ref Vector4d *vector*, double *scale*, out Vector4d *result*) [static]

Divides a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.81.3.14 static Vector4d OpenTK.Vector4d.Divide (Vector4d *vector*, Vector4d *scale*) [static]

Divides a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.81.3.15 `static void OpenTK.Vector4d.Divide (ref Vector4d vector, ref Vector4d scale, out Vector4d result) [static]`

Divide a vector by the components of a vector (scale).

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.81.3.16 `static void OpenTK.Vector4d.Dot (ref Vector4d left, ref Vector4d right, out double result) [static]`

Calculate the dot product of two vectors.

Parameters

left First operand

right Second operand

result The dot product of the two inputs

5.81.3.17 `static double OpenTK.Vector4d.Dot (Vector4d left, Vector4d right) [static]`

Calculate the dot product of two vectors.

Parameters

left First operand

right Second operand

Returns

The dot product of the two inputs

5.81.3.18 `override bool OpenTK.Vector4d.Equals (object obj)`

Indicates whether this instance and a specified object are equal.

Parameters

obj The object to compare to.

Returns

True if the instances are equal; false otherwise.

5.81.3.19 bool OpenTK.Vector4d.Equals (Vector4d *other*)

Indicates whether the current vector is equal to another vector.

Parameters

other A vector to compare with this vector.

Returns

true if the current vector is equal to the vector parameter; otherwise, false.

5.81.3.20 override int OpenTK.Vector4d.GetHashCode ()

Returns the hashcode for this instance.

Returns

A System.Int32 containing the unique hashcode for this instance.

5.81.3.21 static Vector4d OpenTK.Vector4d.Lerp (Vector4d *a*, Vector4d *b*, double *blend*) [static]

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

Returns

a when blend=0, b when blend=1, and a linear combination otherwise

5.81.3.22 `static void OpenTK.Vector4d.Lerp (ref Vector4d a, ref Vector4d b, double blend, out Vector4d result) [static]`

Returns a new Vector that is the linear blend of the 2 given Vectors.

Parameters

a First input vector

b Second input vector

blend The blend factor. a when blend=0, b when blend=1.

result a when blend=0, b when blend=1, and a linear combination otherwise

5.81.3.23 `static void OpenTK.Vector4d.Max (ref Vector4d a, ref Vector4d b, out Vector4d result) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

result The component-wise maximum

5.81.3.24 `static Vector4d OpenTK.Vector4d.Max (Vector4d a, Vector4d b) [static]`

Calculate the component-wise maximum of two vectors.

Parameters

a First operand

b Second operand

Returns

The component-wise maximum

5.81.3.25 `static void OpenTK.Vector4d.Min (ref Vector4d a, ref Vector4d b, out Vector4d result) [static]`

Calculate the component-wise minimum of two vectors.

Parameters

- a* First operand
- b* Second operand
- result* The component-wise minimum

5.81.3.26 `static Vector4d OpenTK.Vector4d.Min (Vector4d a, Vector4d b)`
`[static]`

Calculate the component-wise minimum of two vectors.

Parameters

- a* First operand
- b* Second operand

Returns

The component-wise minimum

5.81.3.27 `void OpenTK.Vector4d.Mult (double f)`

Multiply this instance by a scalar.

Parameters

- f* Scalar operand.

5.81.3.28 `static Vector4d OpenTK.Vector4d.Mult (Vector4d a, double f)`
`[static]`

Multiply a vector and a scalar.

Parameters

- a* Vector operand
- f* Scalar operand

Returns

Result of the multiplication

5.81.3.29 `static void OpenTK.Vector4d.Mult (ref Vector4d a, double f, out Vector4d result) [static]`

Multiply a vector and a scalar.

Parameters

a Vector operand

f Scalar operand

result Result of the multiplication

5.81.3.30 `static Vector4d OpenTK.Vector4d.Multiply (Vector4d vector, double scale) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

Returns

Result of the operation.

5.81.3.31 `static void OpenTK.Vector4d.Multiply (ref Vector4d vector, double scale, out Vector4d result) [static]`

Multiplies a vector by a scalar.

Parameters

vector Left operand.

scale Right operand.

result Result of the operation.

5.81.3.32 `static void OpenTK.Vector4d.Multiply (ref Vector4d vector, ref Vector4d scale, out Vector4d result) [static]`

Multiplies a vector by the components of a vector (scale).

Parameters

vector Left operand.
scale Right operand.
result Result of the operation.

5.81.3.33 `static Vector4d OpenTK.Vector4d.Multiply (Vector4d vector,
Vector4d scale) [static]`

Multiplies a vector by the components a vector (scale).

Parameters

vector Left operand.
scale Right operand.

Returns

Result of the operation.

5.81.3.34 `static Vector4d OpenTK.Vector4d.Normalize (Vector4d vec)
[static]`

Scale a vector to unit length.

Parameters

vec The input vector

Returns

The normalized vector

5.81.3.35 `static void OpenTK.Vector4d.Normalize (ref Vector4d vec, out
Vector4d result) [static]`

Scale a vector to unit length.

Parameters

vec The input vector
result The normalized vector

5.81.3.36 void OpenTK.Vector4d.Normalize ()

Scales the [Vector4d](#) to unit length.

5.81.3.37 void OpenTK.Vector4d.NormalizeFast ()

Scales the [Vector4d](#) to approximately unit length.

**5.81.3.38 static Vector4d OpenTK.Vector4d.NormalizeFast (Vector4d *vec*)
[static]**

Scale a vector to approximately unit length.

Parameters

vec The input vector

Returns

The normalized vector

**5.81.3.39 static void OpenTK.Vector4d.NormalizeFast (ref Vector4d *vec*, out
Vector4d *result*) [static]**

Scale a vector to approximately unit length.

Parameters

vec The input vector

result The normalized vector

**5.81.3.40 unsafe static OpenTK.Vector4d.operator double * (Vector4d *v*)
[explicit, static]**

Returns a pointer to the first element of the specified instance.

Parameters

v The instance.

Returns

A pointer to the first element of *v*.

**5.81.3.41 static OpenTK.Vector4d.operator IntPtr (Vector4d *v*)
[explicit, static]**

Returns a pointer to the first element of the specified instance.

Parameters

v The instance.

Returns

A pointer to the first element of *v*.

**5.81.3.42 static OpenTK.Vector4d.operator Vector4 (Vector4d *v4d*)
[explicit, static]**

Converts [OpenTK.Vector4d](#) to [OpenTK.Vector4](#).

Parameters

v4d The [Vector4d](#) to convert.

Returns

The resulting [Vector4](#).

**5.81.3.43 static OpenTK.Vector4d.operator Vector4d (Vector4 *v4*)
[explicit, static]**

Converts [OpenTK.Vector4](#) to [OpenTK.Vector4d](#).

Parameters

v4 The [Vector4](#) to convert.

Returns

The resulting [Vector4d](#).

**5.81.3.44 static bool OpenTK.Vector4d.operator!= (Vector4d *left*, Vector4d
right) [static]**

Compares two instances for inequality.

Parameters

left The first instance.
right The second instance.

Returns

True, if left does not equal right; false otherwise.

5.81.3.45 `static Vector4d OpenTK.Vector4d.operator* (Vector4d vec, double scale) [static]`

Multiplies an instance by a scalar.

Parameters

vec The instance.
scale The scalar.

Returns

The result of the calculation.

5.81.3.46 `static Vector4d OpenTK.Vector4d.operator* (double scale, Vector4d vec) [static]`

Multiplies an instance by a scalar.

Parameters

scale The scalar.
vec The instance.

Returns

The result of the calculation.

5.81.3.47 `static Vector4d OpenTK.Vector4d.operator+ (Vector4d left, Vector4d right) [static]`

Adds two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.81.3.48 `static Vector4d OpenTK.Vector4d.operator- (Vector4d left, Vector4d right) [static]`

Subtracts two instances.

Parameters

left The first instance.

right The second instance.

Returns

The result of the calculation.

5.81.3.49 `static Vector4d OpenTK.Vector4d.operator- (Vector4d vec) [static]`

Negates an instance.

Parameters

vec The instance.

Returns

The result of the calculation.

5.81.3.50 `static Vector4d OpenTK.Vector4d.operator/ (Vector4d vec, double scale) [static]`

Divides an instance by a scalar.

Parameters

vec The instance.

scale The scalar.

Returns

The result of the calculation.

5.81.3.51 `static bool OpenTK.Vector4d.operator==(Vector4d left, Vector4d right) [static]`

Compares two instances for equality.

Parameters

left The first instance.

right The second instance.

Returns

True, if left equals right; false otherwise.

5.81.3.52 `void OpenTK.Vector4d.Scale (ref Vector4d scale)`

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.81.3.53 `void OpenTK.Vector4d.Scale (double sx, double sy, double sz, double sw)`

Scales the current [Vector4d](#) by the given amounts.

Parameters

sx The scale of the X component.

sy The scale of the Y component.

sz The scale of the Z component.

sw The scale of the W component.

5.81.3.54 `void OpenTK.Vector4d.Scale (Vector4d scale)`

Scales this instance by the given parameter.

Parameters

scale The scaling of the individual components.

5.81.3.55 `static void OpenTK.Vector4d.Sub (ref Vector4d a, ref Vector4d b,
out Vector4d result) [static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand
result Result of subtraction

5.81.3.56 `static Vector4d OpenTK.Vector4d.Sub (Vector4d a, Vector4d b)
[static]`

Subtract one Vector from another.

Parameters

a First operand
b Second operand

Returns

Result of subtraction

5.81.3.57 `void OpenTK.Vector4d.Sub (ref Vector4d right)`

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.81.3.58 `void OpenTK.Vector4d.Sub (Vector4d right)`

Subtract the Vector passed as parameter from this instance.

Parameters

right Right operand. This parameter is only read from.

5.81.3.59 `static Vector4d OpenTK.Vector4d.Subtract (Vector4d a, Vector4d b) [static]`

Subtract one Vector from another.

Parameters

- a* First operand
- b* Second operand

Returns

Result of subtraction

5.81.3.60 `static void OpenTK.Vector4d.Subtract (ref Vector4d a, ref Vector4d b, out Vector4d result) [static]`

Subtract one Vector from another.

Parameters

- a* First operand
- b* Second operand
- result* Result of subtraction

5.81.3.61 `override string OpenTK.Vector4d.ToString ()`

Returns a System.String that represents the current [Vector4d](#).

Returns

5.81.3.62 `static void OpenTK.Vector4d.Transform (ref Vector4d vec, ref Matrix4d mat, out Vector4d result) [static]`

Transform a Vector by the given Matrix.

Parameters

- vec* The vector to transform
- mat* The desired transformation
- result* The transformed vector

5.81.3.63 `static void OpenTK.Vector4d.Transform (ref Vector4d vec, ref Quaterniond quat, out Vector4d result) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.

quat The quaternion to rotate the vector by.

result The result of the operation.

5.81.3.64 `static Vector4d OpenTK.Vector4d.Transform (Vector4d vec, Matrix4d mat) [static]`

Transform a Vector by the given Matrix.

Parameters

vec The vector to transform

mat The desired transformation

Returns

The transformed vector

5.81.3.65 `static Vector4d OpenTK.Vector4d.Transform (Vector4d vec, Quaterniond quat) [static]`

Transforms a vector by a quaternion rotation.

Parameters

vec The vector to transform.

quat The quaternion to rotate the vector by.

Returns

The result of the operation.

5.81.4 Member Data Documentation

5.81.4.1 `readonly Vector4d OpenTK.Vector4d.One = new Vector4d(1, 1, 1, 1) [static]`

Defines an instance with all components set to 1.

5.81.4.2 `readonly int OpenTK.Vector4d.SizeInBytes = Marshal.SizeOf(new Vector4d()) [static]`

Defines the size of the [Vector4d](#) struct in bytes.

5.81.4.3 `Vector4d OpenTK.Vector4d.UnitW = new Vector4d(0, 0, 0, 1) [static]`

Defines a unit-length [Vector4d](#) that points towards the W-axis.

5.81.4.4 `Vector4d OpenTK.Vector4d.UnitX = new Vector4d(1, 0, 0, 0) [static]`

Defines a unit-length [Vector4d](#) that points towards the X-axis.

5.81.4.5 `Vector4d OpenTK.Vector4d.UnitY = new Vector4d(0, 1, 0, 0) [static]`

Defines a unit-length [Vector4d](#) that points towards the Y-axis.

5.81.4.6 `Vector4d OpenTK.Vector4d.UnitZ = new Vector4d(0, 0, 1, 0) [static]`

Defines a unit-length [Vector4d](#) that points towards the Z-axis.

5.81.4.7 `double OpenTK.Vector4d.W`

The W component of the [Vector4d](#).

5.81.4.8 `double OpenTK.Vector4d.X`

The X component of the [Vector4d](#).

5.81.4.9 `double OpenTK.Vector4d.Y`

The Y component of the [Vector4d](#).

5.81.4.10 `double OpenTK.Vector4d.Z`

The Z component of the [Vector4d](#).

5.81.4.11 **Vector4d** `OpenTK.Vector4d.Zero = new Vector4d(0, 0, 0, 0)` `[static]`

Defines a zero-length [Vector4d](#).

5.81.5 Property Documentation

5.81.5.1 **double** `OpenTK.Vector4d.Length` `[get]`

Gets the length (magnitude) of the vector.

[LengthFast](#)

See also

[LengthSquared](#)

5.81.5.2 **double** `OpenTK.Vector4d.LengthFast` `[get]`

Gets an approximation of the vector length (magnitude).

This property uses an approximation of the square root function to calculate vector magnitude, with an upper error bound of 0.001.

[Length](#)

See also

[LengthSquared](#)

5.81.5.3 **double** `OpenTK.Vector4d.LengthSquared` `[get]`

Gets the square of the vector length (magnitude).

This property avoids the costly square root operation required by the `Length` property. This makes it more suitable for comparisons.

[Length](#)

5.81.5.4 **Vector2d** `OpenTK.Vector4d.Xy` `[get, set]`

Gets or sets an [OpenTK.Vector2d](#) with the X and Y components of this instance.

5.81.5.5 **Vector3d** `OpenTK.Vector4d.Xyz` `[get, set]`

Gets or sets an [OpenTK.Vector3d](#) with the X, Y and Z components of this instance.

5.82 OpenTK.Vector4h Struct Reference

4-component Vector of the [Half](#) type. Occupies 8 Byte total.

Public Member Functions

- [Vector4h](#) ([Half](#) x, [Half](#) y, [Half](#) z, [Half](#) w)
The new Half4 instance will avoid conversion and copy directly from the [Half](#) parameters.
- [Vector4h](#) ([Single](#) x, [Single](#) y, [Single](#) z, [Single](#) w)
The new Half4 instance will convert the 4 parameters into 16-bit half-precision floating-point.
- [Vector4h](#) ([Single](#) x, [Single](#) y, [Single](#) z, [Single](#) w, bool throwOnError)
The new Half4 instance will convert the 4 parameters into 16-bit half-precision floating-point.
- [Vector4h](#) ([Vector4](#) v)
The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point.
- [Vector4h](#) ([Vector4](#) v, bool throwOnError)
The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point.
- [Vector4h](#) (ref [Vector4](#) v)
The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point. This is the fastest constructor.
- [Vector4h](#) (ref [Vector4](#) v, bool throwOnError)
The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point.
- [Vector4h](#) ([Vector4d](#) v)
The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point.
- [Vector4h](#) ([Vector4d](#) v, bool throwOnError)
The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point.
- [Vector4h](#) (ref [Vector4d](#) v)

The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point. This is the faster constructor.

- [Vector4h](#) (ref [Vector4d](#) v, bool throwOnError)
The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point.
- [Vector4 ToVector4](#) ()
Returns this Half4 instance's contents as [Vector4](#).
- [Vector4d ToVector4d](#) ()
Returns this Half4 instance's contents as [Vector4d](#).
- [Vector4h](#) (SerializationInfo info, StreamingContext context)
Constructor used by ISerializable to deserialize the object.
- void [GetObjectData](#) (SerializationInfo info, StreamingContext context)
Used by ISerialize to serialize the object.
- void [FromBinaryStream](#) (BinaryReader bin)
Updates the X,Y,Z and W components of this instance by reading from a Stream.
- void [ToBinaryStream](#) (BinaryWriter bin)
Writes the X,Y,Z and W components of this instance into a Stream.
- bool [Equals](#) ([Vector4h](#) other)
Returns a value indicating whether this instance is equal to a specified OpenTK.Half4 vector.
- override string [ToString](#) ()
Returns a string that contains this Half4's numbers in human-legible form.

Static Public Member Functions

- static [operator Vector4h](#) ([Vector4](#) v4f)
Converts [OpenTK.Vector4](#) to [OpenTK.Half4](#).
- static [operator Vector4h](#) ([Vector4d](#) v4d)
Converts [OpenTK.Vector4d](#) to [OpenTK.Half4](#).
- static [operator Vector4](#) ([Vector4h](#) h4)
Converts [OpenTK.Half4](#) to [OpenTK.Vector4](#).

- static [operator Vector4d](#) ([Vector4h](#) h4)
Converts `OpenTK.Half4` to `OpenTK.Vector4d`.
- static `byte[]` [GetBytes](#) ([Vector4h](#) h)
Returns the `Half4` as an array of bytes.
- static [Vector4h FromBytes](#) (`byte[]` value, `int` startIndex)
Converts an array of bytes into `Half4`.

Public Attributes

- [Half X](#)
The X component of the `Half4`.
- [Half Y](#)
The Y component of the `Half4`.
- [Half Z](#)
The Z component of the `Half4`.
- [Half W](#)
The W component of the `Half4`.

Static Public Attributes

- static readonly `int` [SizeInBytes](#) = 8
The size in bytes for an instance of the `Half4` struct is 8.

Properties

- [Vector2h Xy](#) [`get`, `set`]
Gets or sets an `OpenTK.Vector2h` with the X and Y components of this instance.
- [Vector3h Xyz](#) [`get`, `set`]
Gets or sets an `OpenTK.Vector3h` with the X, Y and Z components of this instance.

5.82.1 Detailed Description

4-component Vector of the [Half](#) type. Occupies 8 Byte total.

5.82.2 Constructor & Destructor Documentation

5.82.2.1 `OpenTK.Vector4h.Vector4h (Half x, Half y, Half z, Half w)`

The new Half4 instance will avoid conversion and copy directly from the [Half](#) parameters.

Parameters

- x* An [Half](#) instance of a 16-bit half-precision floating-point number.
- y* An [Half](#) instance of a 16-bit half-precision floating-point number.
- z* An [Half](#) instance of a 16-bit half-precision floating-point number.
- w* An [Half](#) instance of a 16-bit half-precision floating-point number.

5.82.2.2 `OpenTK.Vector4h.Vector4h (Single x, Single y, Single z, Single w)`

The new Half4 instance will convert the 4 parameters into 16-bit half-precision floating-point.

Parameters

- x* 32-bit single-precision floating-point number.
- y* 32-bit single-precision floating-point number.
- z* 32-bit single-precision floating-point number.
- w* 32-bit single-precision floating-point number.

5.82.2.3 `OpenTK.Vector4h.Vector4h (Single x, Single y, Single z, Single w, bool throwOnError)`

The new Half4 instance will convert the 4 parameters into 16-bit half-precision floating-point.

Parameters

- x* 32-bit single-precision floating-point number.
- y* 32-bit single-precision floating-point number.
- z* 32-bit single-precision floating-point number.

w 32-bit single-precision floating-point number.

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.82.2.4 OpenTK.Vector4h.Vector4h (Vector4 *v*)

The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector4](#)

5.82.2.5 OpenTK.Vector4h.Vector4h (Vector4 *v*, bool *throwOnError*)

The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector4](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.82.2.6 OpenTK.Vector4h.Vector4h (ref Vector4 *v*)

The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point. This is the fastest constructor.

Parameters

v [OpenTK.Vector4](#)

5.82.2.7 OpenTK.Vector4h.Vector4h (ref Vector4 *v*, bool *throwOnError*)

The new Half4 instance will convert the [Vector4](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector4](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.82.2.8 OpenTK.Vector4h.Vector4h (Vector4d v)

The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector4d](#)

5.82.2.9 OpenTK.Vector4h.Vector4h (Vector4d v, bool throwOnError)

The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector4d](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.82.2.10 OpenTK.Vector4h.Vector4h (ref Vector4d v)

The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point. This is the faster constructor.

Parameters

v [OpenTK.Vector4d](#)

5.82.2.11 OpenTK.Vector4h.Vector4h (ref Vector4d v, bool throwOnError)

The new Half4 instance will convert the [Vector4d](#) into 16-bit half-precision floating-point.

Parameters

v [OpenTK.Vector4d](#)

throwOnError Enable checks that will throw if the conversion result is not meaningful.

5.82.2.12 OpenTK.Vector4h.Vector4h (SerializationInfo *info*, StreamingContext *context*)

Constructor used by ISerializable to deserialize the object.

Parameters

info
context

5.82.3 Member Function Documentation

5.82.3.1 bool OpenTK.Vector4h.Equals (Vector4h *other*)

Returns a value indicating whether this instance is equal to a specified OpenTK.Half4 vector.

Parameters

other OpenTK.Half4 to compare to this instance..

Returns

True, if *other* is equal to this instance; false otherwise.

5.82.3.2 void OpenTK.Vector4h.FromBinaryStream (BinaryReader *bin*)

Updates the X,Y,Z and W components of this instance by reading from a Stream.

Parameters

bin A BinaryReader instance associated with an open Stream.

5.82.3.3 static Vector4h OpenTK.Vector4h.FromBytes (byte[] *value*, int *startIndex*) [static]

Converts an array of bytes into Half4.

Parameters

value A Half4 in it's byte[] representation.
startIndex The starting position within *value*.

Returns

A new Half4 instance.

5.82.3.4 static byte [] OpenTK.Vector4h.GetBytes (Vector4h *h*) [static]

Returns the Half4 as an array of bytes.

Parameters

h The Half4 to convert.

Returns

The input as byte array.

5.82.3.5 void OpenTK.Vector4h.GetObjectData (SerializationInfo *info*, StreamingContext *context*)

Used by ISerialize to serialize the object.

Parameters

info

context

5.82.3.6 static OpenTK.Vector4h.operator Vector4 (Vector4h *h4*) [explicit, static]

Converts OpenTK.Half4 to [OpenTK.Vector4](#).

Parameters

h4 The Half4 to convert.

Returns

The resulting [Vector4](#).

5.82.3.7 static OpenTK.Vector4h.operator Vector4d (Vector4h *h4*) [explicit, static]

Converts OpenTK.Half4 to [OpenTK.Vector4d](#).

Parameters

h4 The Half4 to convert.

Returns

The resulting [Vector4d](#).

5.82.3.8 static OpenTK.Vector4h.operator Vector4h (Vector4 *v4f*) [explicit, static]

Converts [OpenTK.Vector4](#) to OpenTK.Half4.

Parameters

v4f The [Vector4](#) to convert.

Returns

The resulting [Half](#) vector.

5.82.3.9 static OpenTK.Vector4h.operator Vector4h (Vector4d *v4d*) [explicit, static]

Converts [OpenTK.Vector4d](#) to OpenTK.Half4.

Parameters

v4d The [Vector4d](#) to convert.

Returns

The resulting [Half](#) vector.

5.82.3.10 void OpenTK.Vector4h.ToBinaryStream (BinaryWriter *bin*)

Writes the X,Y,Z and W components of this instance into a Stream.

Parameters

bin A BinaryWriter instance associated with an open Stream.

5.82.3.11 override string OpenTK.Vector4h.ToString ()

Returns a string that contains this Half4's numbers in human-legible form.

5.82.3.12 Vector4 OpenTK.Vector4h.ToVector4 ()

Returns this Half4 instance's contents as [Vector4](#).

Returns

[OpenTK.Vector4](#)

5.82.3.13 Vector4d OpenTK.Vector4h.ToVector4d ()

Returns this Half4 instance's contents as [Vector4d](#).

5.82.4 Member Data Documentation

5.82.4.1 readonly int OpenTK.Vector4h.SizeInBytes = 8 [static]

The size in bytes for an instance of the Half4 struct is 8.

5.82.4.2 Half OpenTK.Vector4h.W

The W component of the Half4.

5.82.4.3 Half OpenTK.Vector4h.X

The X component of the Half4.

5.82.4.4 Half OpenTK.Vector4h.Y

The Y component of the Half4.

5.82.4.5 Half OpenTK.Vector4h.Z

The Z component of the Half4.

5.82.5 Property Documentation

5.82.5.1 Vector2h OpenTK.Vector4h.Xy [get, set]

Gets or sets an [OpenTK.Vector2h](#) with the X and Y components of this instance.

5.82.5.2 Vector3h OpenTK.Vector4h.Xyz [get, set]

Gets or sets an [OpenTK.Vector3h](#) with the X, Y and Z components of this instance.

Index

- A
 - OpenTK::Graphics::Color4, [408](#)
 - OpenTK::Input, [252](#)
- Accessible
 - OpenTK::Audio::OpenAL::XRamExtension, [341](#)
- Accum
 - OpenTK::Graphics::OpenGL::GL, [953](#)
- AccumulatorFormat
 - OpenTK::Graphics::GraphicsMode, [798](#)
- ActiveTexture
 - OpenTK::Graphics::ES10::GL, [442](#)
 - OpenTK::Graphics::ES11::GL, [527](#)
 - OpenTK::Graphics::ES20::GL, [656](#)
 - OpenTK::Graphics::OpenGL::GL, [953](#)
- Add
 - OpenTK::Quaternion, [1608](#)
 - OpenTK::Quaterniond, [1623](#)
 - OpenTK::Vector2, [1641](#)
 - OpenTK::Vector2d, [1665](#), [1666](#)
 - OpenTK::Vector3, [1701](#), [1702](#)
 - OpenTK::Vector3d, [1734](#), [1735](#)
 - OpenTK::Vector4, [1777](#)
 - OpenTK::Vector4d, [1804](#), [1805](#)
- AddOverlays
 - OpenTK::Platform::Windows, [279](#)
- ALBufferState
 - OpenTK::Audio::OpenAL, [20](#)
- ALCapability
 - OpenTK::Audio::OpenAL, [20](#)
- AlcContextAttributes
 - OpenTK::Audio::OpenAL, [20](#)
- AlcError
 - OpenTK::Audio::OpenAL, [21](#)
- AlcGetInteger
 - OpenTK::Audio::OpenAL, [21](#)
- AlcGetString
 - OpenTK::Audio::OpenAL, [22](#)
- AlcGetStringList
 - OpenTK::Audio::OpenAL, [23](#)
- ALDistanceModel
 - OpenTK::Audio::OpenAL, [23](#)
- ALError
 - OpenTK::Audio::OpenAL, [23](#)
- ALFormat
 - OpenTK::Audio::OpenAL, [24](#)
- ALGetBufferi
 - OpenTK::Audio::OpenAL, [25](#)
- ALGetFloat
 - OpenTK::Audio::OpenAL, [26](#)
- ALGetInteger
 - OpenTK::Audio::OpenAL, [26](#)
- ALGetSourcei
 - OpenTK::Audio::OpenAL, [26](#)
- ALGetString
 - OpenTK::Audio::OpenAL, [27](#)
- AliceBlue
 - OpenTK::Graphics::Color4, [408](#)
- AllAttributes
 - OpenTK::Audio::OpenAL, [22](#)
- AllDevicesSpecifier
 - OpenTK::Audio::OpenAL, [22](#), [23](#)
- ALLEVENTS
 - OpenTK::Platform::Windows, [276](#)
- ALLINPUT
 - OpenTK::Platform::Windows, [276](#)
- ALLlistener3f
 - OpenTK::Audio::OpenAL, [27](#)
- ALLlistenerf
 - OpenTK::Audio::OpenAL, [27](#)
- ALLlistenerfv

- OpenTK::Audio::OpenAL, 28
- ALLPOSTMESSAGE
 - OpenTK::Platform::Windows, 276
- Alpha
 - OpenTK::Graphics::ColorFormat, 427
- AlphaFunc
 - OpenTK::Graphics::ES10::GL, 442
 - OpenTK::Graphics::ES11::GL, 528
 - OpenTK::Graphics::OpenGL::GL, 954
- ALSource3f
 - OpenTK::Audio::OpenAL, 28
- ALSource3i
 - OpenTK::Audio::OpenAL, 28
- ALSourceb
 - OpenTK::Audio::OpenAL, 28
- ALSourcef
 - OpenTK::Audio::OpenAL, 29
- ALSourcei
 - OpenTK::Audio::OpenAL, 30
- ALSourceState
 - OpenTK::Audio::OpenAL, 31
- ALSourceType
 - OpenTK::Audio::OpenAL, 31
- AltLeft
 - OpenTK::Input, 249
- AltRight
 - OpenTK::Input, 249
- AntiqueWhite
 - OpenTK::Graphics::Color4, 408
- APPKEYS
 - OpenTK::Platform::Windows, 277
- Aqua
 - OpenTK::Graphics::Color4, 409
- Aquamarine
 - OpenTK::Graphics::Color4, 409
- AreTexturesResident
 - OpenTK::Graphics::OpenGL::GL, 954–956
- ArrayElement
 - OpenTK::Graphics::OpenGL::GL, 956
- AspectRatio
 - OpenTK::GLControl, 392
- Assert
 - OpenTK::Graphics::GraphicsContext, 785
- AttachShader
 - OpenTK::Graphics::ES20::GL, 657
 - OpenTK::Graphics::OpenGL::GL, 956, 957
- Attr_Specified
 - OpenTK::Platform::Windows, 279
- Attributes
 - OpenTK::Platform::Windows, 278
- AttributesSize
 - OpenTK::Audio::OpenAL, 22
- AudioCapture
 - OpenTK::Audio::AudioCapture, 297
- AudioContext
 - OpenTK::Audio::AudioContext, 302–304
- AudioContextException
 - OpenTK::Audio::AudioContextException, 309
- AudioDeviceException
 - OpenTK::Audio::AudioDeviceException, 310
- AudioException
 - OpenTK::Audio::AudioException, 311
- AudioValueException
 - OpenTK::Audio::AudioValueException, 312
- AutoGeneratedAttribute
 - OpenTK::AutoGeneratedAttribute, 344
- Automatic
 - OpenTK::Audio::OpenAL::XRamExtension, 341
- Autowah
 - OpenTK::Audio::OpenAL, 40
- AutowahAttackTime
 - OpenTK::Audio::OpenAL, 36
- AutowahPeakGain
 - OpenTK::Audio::OpenAL, 36
- AutowahReleaseTime
 - OpenTK::Audio::OpenAL, 36
- AutowahResonance
 - OpenTK::Audio::OpenAL, 36
- AuxiliaryEffectSlot

- OpenTK::Audio::OpenAL::EffectsExtension, 319, 320
- AvailableDevices
 - OpenTK::Audio::AudioCapture, 298
 - OpenTK::Audio::AudioContext, 307
- AvailableDisplays
 - OpenTK::DisplayDevice, 370
- AvailableResolutions
 - OpenTK::DisplayDevice, 370
- AvailableSamples
 - OpenTK::Audio::AudioCapture, 298
- Axis
 - OpenTK::Input::JoystickDevice, 1510
 - OpenTK::Input::JoystickMoveEventArgs, 1512
- Axis0
 - OpenTK::Input, 248
- Axis1
 - OpenTK::Input, 248
- Axis2
 - OpenTK::Input, 248
- Axis3
 - OpenTK::Input, 248
- Axis4
 - OpenTK::Input, 248
- Axis5
 - OpenTK::Input, 248
- Axis6
 - OpenTK::Input, 248
- Axis7
 - OpenTK::Input, 248
- Axis8
 - OpenTK::Input, 248
- Axis9
 - OpenTK::Input, 248
- Azure
 - OpenTK::Graphics::Color4, 409
- B
 - OpenTK::Graphics::Color4, 408
 - OpenTK::Input, 252
- Back
 - OpenTK::Input, 251
- BackSlash
 - OpenTK::Input, 253
- BackSpace
 - OpenTK::Input, 251
- BandpassGain
 - OpenTK::Audio::OpenAL, 41
- BandpassGainLF
 - OpenTK::Audio::OpenAL, 41
- BaryCentric
 - OpenTK::Vector2, 1641, 1642
 - OpenTK::Vector2d, 1666, 1667
 - OpenTK::Vector3, 1702, 1703
 - OpenTK::Vector3d, 1735
 - OpenTK::Vector4, 1777, 1778
 - OpenTK::Vector4d, 1805, 1806
- Begin
 - OpenTK::Graphics::OpenGL::GL, 957
- BeginQuery
 - OpenTK::Graphics::OpenGL::GL, 957
- Beige
 - OpenTK::Graphics::Color4, 409
- BezierCurve
 - OpenTK::BezierCurve, 346, 347
- BezierCurveCubic
 - OpenTK::BezierCurveCubic, 351
- BezierCurveQuadric
 - OpenTK::BezierCurveQuadric, 353, 354
- BindAttribLocation
 - OpenTK::Graphics::ES20::GL, 657
 - OpenTK::Graphics::OpenGL::GL, 958
- BindBuffer
 - OpenTK::Graphics::ES11::GL, 528
 - OpenTK::Graphics::ES20::GL, 658
 - OpenTK::Graphics::OpenGL::GL, 958, 959
- BindEffect
 - OpenTK::Audio::OpenAL::EffectsExtension, 320
- BindEffectToAuxiliarySlot
 - OpenTK::Audio::OpenAL::EffectsExtension, 321
- BindFilterToSource
 - OpenTK::Audio::OpenAL::EffectsExtension, 321

- BindingsBase
 - OpenTK::BindingsBase, [356](#)
- BindSourceToAuxiliarySlot
 - OpenTK::Audio::OpenAL::EffectsExtension, [965–967](#)
 - [322](#)
- BindTexture
 - OpenTK::Graphics::ES10::GL, [443](#)
 - OpenTK::Graphics::ES11::GL, [529](#)
 - OpenTK::Graphics::ES20::GL, [658](#), [659](#)
 - OpenTK::Graphics::OpenGL::GL, [959](#)
- Bisque
 - OpenTK::Graphics::Color4, [409](#)
- Bitmap
 - OpenTK::Graphics::OpenGL::GL, [960](#)
- Bits
 - OpenTK::Audio::OpenAL, [26](#)
- BitsPerPixel
 - OpenTK::DisplayDevice, [370](#)
 - OpenTK::DisplayResolution, [373](#)
 - OpenTK::Graphics::ColorFormat, [427](#)
- Black
 - OpenTK::Graphics::Color4, [409](#)
- BlanchedAlmond
 - OpenTK::Graphics::Color4, [409](#)
- BlendColor
 - OpenTK::Graphics::ES20::GL, [659](#)
 - OpenTK::Graphics::OpenGL::GL, [961](#)
- BlendEquation
 - OpenTK::Graphics::ES20::GL, [659](#)
 - OpenTK::Graphics::OpenGL::GL, [961](#), [962](#)
- BlendEquationSeparate
 - OpenTK::Graphics::ES20::GL, [660](#)
 - OpenTK::Graphics::OpenGL::GL, [962](#), [963](#)
- BlendFunc
 - OpenTK::Graphics::ES10::GL, [443](#)
 - OpenTK::Graphics::ES11::GL, [529](#)
 - OpenTK::Graphics::ES20::GL, [660](#)
 - OpenTK::Graphics::OpenGL::GL, [963](#), [964](#)
- BlendFuncSeparate
 - OpenTK::Graphics::ES20::GL, [661](#)
 - OpenTK::Graphics::OpenGL::GL, [965–967](#)
- Blue
 - OpenTK::Graphics::Color4, [409](#)
 - OpenTK::Graphics::ColorFormat, [427](#)
- BlueViolet
 - OpenTK::Graphics::Color4, [409](#)
- Bottom
 - OpenTK::Box2, [360](#)
- Bounds
 - OpenTK::DisplayDevice, [370](#)
 - OpenTK::DisplayResolution, [373](#)
 - OpenTK::INativeWindow, [1496](#)
 - OpenTK::NativeWindow, [1595](#)
- Box2
 - OpenTK::Box2, [359](#)
- BracketLeft
 - OpenTK::Input, [253](#)
- BracketRight
 - OpenTK::Input, [253](#)
- Brown
 - OpenTK::Graphics::Color4, [409](#)
- Buffer
 - OpenTK::Audio::OpenAL, [30](#)
- BufferData
 - OpenTK::Graphics::ES11::GL, [530](#)
 - OpenTK::Graphics::ES20::GL, [661](#)
 - OpenTK::Graphics::OpenGL::GL, [967](#)
- BufferData< T2 >
 - OpenTK::Graphics::ES11::GL, [530–532](#)
 - OpenTK::Graphics::ES20::GL, [662–664](#)
 - OpenTK::Graphics::OpenGL::GL, [968–970](#)
- Buffers
 - OpenTK::Graphics::GraphicsMode, [798](#)
- BuffersProcessed
 - OpenTK::Audio::OpenAL, [27](#)
- BuffersQueued
 - OpenTK::Audio::OpenAL, [26](#)

- BufferSubData
 - OpenTK::Graphics::ES11::GL, [532](#)
 - OpenTK::Graphics::ES20::GL, [664](#)
 - OpenTK::Graphics::OpenGL::GL, [970](#)
- BufferSubData< T3 >
 - OpenTK::Graphics::ES11::GL, [533](#), [534](#)
 - OpenTK::Graphics::ES20::GL, [664](#)–[666](#)
 - OpenTK::Graphics::OpenGL::GL, [970](#)–[972](#)
- BurlyWood
 - OpenTK::Graphics::Color4, [410](#)
- Button
 - OpenTK::Input::JoystickButtonEventArgs, [1508](#)
 - OpenTK::Input::JoystickDevice, [1510](#)
 - OpenTK::Input::MouseButtonEventArgs, [1520](#)
- Button0
 - OpenTK::Input, [248](#)
- Button1
 - OpenTK::Input, [248](#), [253](#)
- Button10
 - OpenTK::Input, [248](#)
- Button11
 - OpenTK::Input, [249](#)
- Button12
 - OpenTK::Input, [249](#)
- Button13
 - OpenTK::Input, [249](#)
- Button14
 - OpenTK::Input, [249](#)
- Button15
 - OpenTK::Input, [249](#)
- Button2
 - OpenTK::Input, [248](#), [253](#)
- Button3
 - OpenTK::Input, [248](#), [253](#)
- Button4
 - OpenTK::Input, [248](#), [253](#)
- Button5
 - OpenTK::Input, [248](#), [253](#)
- Button6
 - OpenTK::Input, [248](#), [253](#)
- Button7
 - OpenTK::Input, [248](#), [253](#)
- Button8
 - OpenTK::Input, [248](#), [253](#)
- Button9
 - OpenTK::Input, [248](#), [253](#)
- ButtonDown
 - OpenTK::Input::JoystickDevice, [1509](#)
 - OpenTK::Input::MouseDevice, [1524](#)
- ButtonUp
 - OpenTK::Input::JoystickDevice, [1509](#)
 - OpenTK::Input::MouseDevice, [1524](#)
- ByteOffset
 - OpenTK::Audio::OpenAL, [26](#), [30](#)
- C
 - OpenTK::Input, [252](#)
- CadetBlue
 - OpenTK::Graphics::Color4, [410](#)
- CalculateAngle
 - OpenTK::Vector3, [1703](#)
 - OpenTK::Vector3d, [1736](#)
- CalculateLength
 - OpenTK::BezierCurve, [347](#), [348](#)
 - OpenTK::BezierCurveCubic, [351](#)
 - OpenTK::BezierCurveQuadric, [354](#)
- CalculatePoint
 - OpenTK::BezierCurve, [348](#), [349](#)
 - OpenTK::BezierCurveCubic, [351](#)
 - OpenTK::BezierCurveQuadric, [354](#)
- CallList
 - OpenTK::Graphics::OpenGL::GL, [972](#), [973](#)
- CallLists
 - OpenTK::Graphics::OpenGL::GL, [973](#)
- CallLists< T2 >
 - OpenTK::Graphics::OpenGL::GL, [973](#)–[975](#)
- CapsLock
 - OpenTK::Input, [251](#)
- CaptureDeviceSpecifier
 - OpenTK::Audio::OpenAL, [22](#), [23](#)

- CAPTUREMOUSE
 - OpenTK::Platform::Windows, [277](#)
- Category
 - OpenTK::AutoGeneratedAttribute, [344](#)
- ChangeResolution
 - OpenTK::DisplayDevice, [368](#)
- Channels
 - OpenTK::Audio::OpenAL, [26](#)
- Chartreuse
 - OpenTK::Graphics::Color4, [410](#)
- CheckErrors
 - OpenTK::Audio::AudioCapture, [297](#)
 - OpenTK::Audio::AudioContext, [305](#)
- Chocolate
 - OpenTK::Graphics::Color4, [410](#)
- Chorus
 - OpenTK::Audio::OpenAL, [40](#)
- ChorusDelay
 - OpenTK::Audio::OpenAL, [34](#)
- ChorusDepth
 - OpenTK::Audio::OpenAL, [34](#)
- ChorusFeedback
 - OpenTK::Audio::OpenAL, [34](#)
- ChorusPhase
 - OpenTK::Audio::OpenAL, [38](#)
- ChorusRate
 - OpenTK::Audio::OpenAL, [34](#)
- ChorusWaveform
 - OpenTK::Audio::OpenAL, [38](#)
- Clamp
 - OpenTK::Vector2, [1642](#), [1643](#)
 - OpenTK::Vector2d, [1667](#)
 - OpenTK::Vector3, [1704](#)
 - OpenTK::Vector3d, [1736](#), [1737](#)
 - OpenTK::Vector4, [1778](#), [1779](#)
 - OpenTK::Vector4d, [1806](#)
- Clear
 - OpenTK::Graphics::ES10::GL, [444](#)
 - OpenTK::Graphics::ES11::GL, [535](#)
 - OpenTK::Graphics::ES20::GL, [666](#)
 - OpenTK::Graphics::OpenGL::GL, [975](#)
 - OpenTK::Input, [251](#)
- ClearAccum
 - OpenTK::Graphics::OpenGL::GL, [975](#)
- ClearColor
 - OpenTK::Graphics::ES10::GL, [444](#)
 - OpenTK::Graphics::ES11::GL, [535](#)
 - OpenTK::Graphics::ES20::GL, [667](#)
 - OpenTK::Graphics::OpenGL::GL, [976](#)
- ClearDepth
 - OpenTK::Graphics::ES10::GL, [444](#)
 - OpenTK::Graphics::ES11::GL, [535](#)
 - OpenTK::Graphics::ES20::GL, [667](#)
 - OpenTK::Graphics::OpenGL::GL, [976](#)
- ClearIndex
 - OpenTK::Graphics::OpenGL::GL, [976](#)
- ClearStencil
 - OpenTK::Graphics::ES10::GL, [445](#)
 - OpenTK::Graphics::ES11::GL, [536](#)
 - OpenTK::Graphics::ES20::GL, [667](#)
 - OpenTK::Graphics::OpenGL::GL, [976](#)
- ClientActiveTexture
 - OpenTK::Graphics::ES10::GL, [445](#)
 - OpenTK::Graphics::ES11::GL, [536](#)
 - OpenTK::Graphics::OpenGL::GL, [977](#)
- ClientRectangle
 - OpenTK::INativeWindow, [1496](#)
 - OpenTK::NativeWindow, [1595](#)
- ClientSize
 - OpenTK::INativeWindow, [1496](#)
 - OpenTK::NativeWindow, [1595](#)
- ClipPlane
 - OpenTK::Graphics::ES11::GL, [536](#), [537](#)
 - OpenTK::Graphics::OpenGL::GL, [977](#), [978](#)
- Close
 - OpenTK::INativeWindow, [1495](#)
 - OpenTK::NativeWindow, [1591](#)
- Closed
 - OpenTK::INativeWindow, [1498](#)
 - OpenTK::NativeWindow, [1598](#)
- Closing

- OpenTK::INativeWindow, [1498](#)
- OpenTK::NativeWindow, [1598](#)
- Color3
 - OpenTK::Graphics::OpenGL::GL, [978–986](#)
- Color4
 - OpenTK::Graphics::Color4, [405](#)
 - OpenTK::Graphics::ES10::GL, [445](#)
 - OpenTK::Graphics::ES11::GL, [537](#)
 - OpenTK::Graphics::OpenGL::GL, [986–994](#)
- ColorFormat
 - OpenTK::Graphics::ColorFormat, [425](#)
 - OpenTK::Graphics::GraphicsMode, [798](#)
- ColorMask
 - OpenTK::Graphics::ES10::GL, [445](#)
 - OpenTK::Graphics::ES11::GL, [538](#)
 - OpenTK::Graphics::ES20::GL, [667](#)
 - OpenTK::Graphics::OpenGL::GL, [994](#)
- ColorMaterial
 - OpenTK::Graphics::OpenGL::GL, [995](#)
- ColorPointer
 - OpenTK::Graphics::ES10::GL, [446](#)
 - OpenTK::Graphics::ES11::GL, [538](#)
 - OpenTK::Graphics::OpenGL::GL, [995](#)
- ColorPointer< T3 >
 - OpenTK::Graphics::ES10::GL, [446–448](#)
 - OpenTK::Graphics::ES11::GL, [538–540](#)
 - OpenTK::Graphics::OpenGL::GL, [996, 997](#)
- ColorSubTable
 - OpenTK::Graphics::OpenGL::GL, [998](#)
- ColorSubTable< T5 >
 - OpenTK::Graphics::OpenGL::GL, [998–1001](#)
- ColorTable
 - OpenTK::Graphics::OpenGL::GL, [1002](#)
- ColorTable< T5 >
 - OpenTK::Graphics::OpenGL::GL, [1003–1006](#)
- ColorTableParameter
 - OpenTK::Graphics::OpenGL::GL, [1007–1009](#)
- Column0
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- Column1
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- Column2
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- Column3
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- Comma
 - OpenTK::Input, [253](#)
- CompareTo
 - OpenTK::ContextHandle, [364](#)
 - OpenTK::Half, [1485](#)
- CompileShader
 - OpenTK::Graphics::ES20::GL, [668](#)
 - OpenTK::Graphics::OpenGL::GL, [1009, 1010](#)
- ComponentMax
 - OpenTK::Vector2, [1643](#)
 - OpenTK::Vector3, [1704, 1705](#)
 - OpenTK::Vector3d, [1737](#)
- ComponentMin
 - OpenTK::Vector2, [1643, 1644](#)
 - OpenTK::Vector3, [1705](#)
 - OpenTK::Vector3d, [1738](#)
- CompressedTexImage1D
 - OpenTK::Graphics::OpenGL::GL, [1010](#)
- CompressedTexImage1D< T6 >
 - OpenTK::Graphics::OpenGL::GL, [1010–1013](#)
- CompressedTexImage2D
 - OpenTK::Graphics::ES10::GL, [448](#)
 - OpenTK::Graphics::ES11::GL, [540](#)
 - OpenTK::Graphics::ES20::GL, [668](#)

- OpenTK::Graphics::OpenGL::GL, [1013](#)
- CompressedTexImage2D< T7 >
 - OpenTK::Graphics::ES10::GL, [449–452](#)
 - OpenTK::Graphics::ES11::GL, [541–544](#)
 - OpenTK::Graphics::ES20::GL, [669–672](#)
 - OpenTK::Graphics::OpenGL::GL, [1014–1017](#)
- CompressedTexImage3D
 - OpenTK::Graphics::OpenGL::GL, [1018](#)
- CompressedTexImage3D< T8 >
 - OpenTK::Graphics::OpenGL::GL, [1019–1021](#)
- CompressedTexSubImage1D
 - OpenTK::Graphics::OpenGL::GL, [1022](#)
- CompressedTexSubImage1D< T6 >
 - OpenTK::Graphics::OpenGL::GL, [1023–1025](#)
- CompressedTexSubImage2D
 - OpenTK::Graphics::ES10::GL, [453](#)
 - OpenTK::Graphics::ES11::GL, [545](#)
 - OpenTK::Graphics::ES20::GL, [673](#)
 - OpenTK::Graphics::OpenGL::GL, [1025](#)
- CompressedTexSubImage2D< T8 >
 - OpenTK::Graphics::ES10::GL, [453–455](#)
 - OpenTK::Graphics::ES11::GL, [546–548](#)
 - OpenTK::Graphics::ES20::GL, [673–675](#)
 - OpenTK::Graphics::OpenGL::GL, [1026–1028](#)
- CompressedTexSubImage3D
 - OpenTK::Graphics::OpenGL::GL, [1029](#)
- CompressedTexSubImage3D< T10 >
 - OpenTK::Graphics::OpenGL::GL, [1029–1032](#)
- Compressor
 - OpenTK::Audio::OpenAL, [40](#)
- CompressorOnoff
 - OpenTK::Audio::OpenAL, [39](#)
- ConeInnerAngle
 - OpenTK::Audio::OpenAL, [30](#)
- ConeOuterAngle
 - OpenTK::Audio::OpenAL, [30](#)
- Conjugate
 - OpenTK::Quaternion, [1608](#), [1609](#)
 - OpenTK::Quaterniond, [1623](#), [1624](#)
- Context
 - OpenTK::GameWindow, [385](#)
 - OpenTK::GLControl, [392](#)
 - OpenTK::Graphics::IGraphicsContextInternal, [804](#)
- ContextExistsException
 - OpenTK::ContextExistsException, [361](#)
- ContextHandle
 - OpenTK::ContextHandle, [363](#)
- ControlLeft
 - OpenTK::Input, [249](#)
- ControlPoint
 - OpenTK::BezierCurveQuadric, [355](#)
- ControlRight
 - OpenTK::Input, [249](#)
- ConvolutionFilter1D
 - OpenTK::Graphics::OpenGL::GL, [1032](#)
- ConvolutionFilter1D< T5 >
 - OpenTK::Graphics::OpenGL::GL, [1033–1036](#)
- ConvolutionFilter2D
 - OpenTK::Graphics::OpenGL::GL, [1037](#)
- ConvolutionFilter2D< T6 >
 - OpenTK::Graphics::OpenGL::GL, [1038–1041](#)
- ConvolutionParameter
 - OpenTK::Graphics::OpenGL::GL, [1042–1044](#)
- CopyColorSubTable
 - OpenTK::Graphics::OpenGL::GL, [1045](#)
- CopyColorTable
 - OpenTK::Graphics::OpenGL::GL, [1045](#)

- CopyConvolutionFilter1D
 - OpenTK::Graphics::OpenGL::GL, 1046
- CopyConvolutionFilter2D
 - OpenTK::Graphics::OpenGL::GL, 1046
- CopyPixels
 - OpenTK::Graphics::OpenGL::GL, 1047
- CopyTexImage1D
 - OpenTK::Graphics::OpenGL::GL, 1047
- CopyTexImage2D
 - OpenTK::Graphics::ES10::GL, 456
 - OpenTK::Graphics::ES11::GL, 548
 - OpenTK::Graphics::ES20::GL, 676
 - OpenTK::Graphics::OpenGL::GL, 1048
- CopyTexSubImage1D
 - OpenTK::Graphics::OpenGL::GL, 1049
- CopyTexSubImage2D
 - OpenTK::Graphics::ES10::GL, 457
 - OpenTK::Graphics::ES11::GL, 549
 - OpenTK::Graphics::ES20::GL, 677
 - OpenTK::Graphics::OpenGL::GL, 1050
- CopyTexSubImage3D
 - OpenTK::Graphics::OpenGL::GL, 1050
- Coral
 - OpenTK::Graphics::Color4, 410
- CoreClass
 - OpenTK::BindingsBase, 357
- CoreFunctionMap
 - OpenTK::BindingsBase, 357
- CornflowerBlue
 - OpenTK::Graphics::Color4, 410
- Cornsilk
 - OpenTK::Graphics::Color4, 410
- Count
 - OpenTK::Input::JoystickAxisCollection, 1506
 - OpenTK::Input::JoystickButtonCollection, 1507
- CreateDummyContext
 - OpenTK::Graphics::GraphicsContext, 785, 786
- CreateFromAxisAngle
 - OpenTK::Matrix4, 1540
 - OpenTK::Matrix4d, 1565, 1566
- CreateOrthographic
 - OpenTK::Matrix4, 1540, 1541
 - OpenTK::Matrix4d, 1566
- CreateOrthographicOffCenter
 - OpenTK::Matrix4, 1541
 - OpenTK::Matrix4d, 1567
- CreatePerspectiveFieldOfView
 - OpenTK::Matrix4, 1542
 - OpenTK::Matrix4d, 1567, 1568
- CreatePerspectiveOffCenter
 - OpenTK::Matrix4, 1543, 1544
 - OpenTK::Matrix4d, 1569
- CreateProgram
 - OpenTK::Graphics::ES20::GL, 678
 - OpenTK::Graphics::OpenGL::GL, 1051
- CreateRotationX
 - OpenTK::Matrix4, 1544
 - OpenTK::Matrix4d, 1570
- CreateRotationY
 - OpenTK::Matrix4, 1545
 - OpenTK::Matrix4d, 1570, 1571
- CreateRotationZ
 - OpenTK::Matrix4, 1545
 - OpenTK::Matrix4d, 1571
- CreateShader
 - OpenTK::Graphics::ES20::GL, 678
 - OpenTK::Graphics::OpenGL::GL, 1051
- CreateTranslation
 - OpenTK::Matrix4, 1546, 1547
 - OpenTK::Matrix4d, 1571, 1572
- Crimson
 - OpenTK::Graphics::Color4, 410
- Cross
 - OpenTK::Vector3, 1706
 - OpenTK::Vector3d, 1738, 1739
- CullFace
 - OpenTK::Graphics::ES10::GL, 458
 - OpenTK::Graphics::ES11::GL, 550
 - OpenTK::Graphics::ES20::GL, 678

- OpenTK::Graphics::OpenGL::GL, 1051
- CurrentContext
 - OpenTK::Audio::AudioContext, 307
 - OpenTK::Graphics::GraphicsContext, 787
- CurrentDevice
 - OpenTK::Audio::AudioCapture, 299
 - OpenTK::Audio::AudioContext, 308
- CurrentError
 - OpenTK::Audio::AudioCapture, 299
 - OpenTK::Audio::AudioContext, 308
- Cyan
 - OpenTK::Graphics::Color4, 410
- D
 - OpenTK::Input, 252
- DarkBlue
 - OpenTK::Graphics::Color4, 410
- DarkCyan
 - OpenTK::Graphics::Color4, 411
- DarkGoldenrod
 - OpenTK::Graphics::Color4, 411
- DarkGray
 - OpenTK::Graphics::Color4, 411
- DarkGreen
 - OpenTK::Graphics::Color4, 411
- DarkKhaki
 - OpenTK::Graphics::Color4, 411
- DarkMagenta
 - OpenTK::Graphics::Color4, 411
- DarkOliveGreen
 - OpenTK::Graphics::Color4, 411
- DarkOrange
 - OpenTK::Graphics::Color4, 411
- DarkOrchid
 - OpenTK::Graphics::Color4, 411
- DarkRed
 - OpenTK::Graphics::Color4, 412
- DarkSalmon
 - OpenTK::Graphics::Color4, 412
- DarkSeaGreen
 - OpenTK::Graphics::Color4, 412
- DarkSlateBlue
 - OpenTK::Graphics::Color4, 412
- DarkSlateGray
 - OpenTK::Graphics::Color4, 412
- DarkTurquoise
 - OpenTK::Graphics::Color4, 412
- DarkViolet
 - OpenTK::Graphics::Color4, 412
- Debug
 - OpenTK::Graphics, 44
- DeepPink
 - OpenTK::Graphics::Color4, 412
- DeepSkyBlue
 - OpenTK::Graphics::Color4, 412
- Default
 - OpenTK::DisplayDevice, 370
 - OpenTK::Graphics, 44
 - OpenTK::Graphics::GraphicsMode, 798
- DefaultAllDevicesSpecifier
 - OpenTK::Audio::OpenAL, 22
- DefaultDevice
 - OpenTK::Audio::AudioCapture, 299
 - OpenTK::Audio::AudioContext, 308
- DefaultDeviceSpecifier
 - OpenTK::Audio::OpenAL, 22
- DelegatesClass
 - OpenTK::BindingsBase, 357
- Delete
 - OpenTK::Input, 251
- DeleteAuxiliaryEffectSlot
 - OpenTK::Audio::OpenAL::EffectsExtension, 322, 323
- DeleteAuxiliaryEffectSlots
 - OpenTK::Audio::OpenAL::EffectsExtension, 323, 324
- DeleteBuffers
 - OpenTK::Graphics::ES11::GL, 550, 551
 - OpenTK::Graphics::ES20::GL, 678, 679
 - OpenTK::Graphics::OpenGL::GL, 1052, 1053
- DeleteEffect
 - OpenTK::Audio::OpenAL::EffectsExtension, 324
- DeleteEffects
 - OpenTK::Audio::OpenAL::EffectsExtension, 324, 325

- DeleteFilter
 - OpenTK::Audio::OpenAL::EffectsExtensionOpenTK::Graphics::ES10::GL, [460](#)
 - [325](#), [326](#)
- DeleteFilters
 - OpenTK::Audio::OpenAL::EffectsExtensionOpenTK::Graphics::ES11::GL, [553](#)
 - OpenTK::Graphics::ES20::GL, [682](#)
 - OpenTK::Graphics::OpenGL::GL, [1058](#)
- DeleteLists
 - OpenTK::Graphics::OpenGL::GL, [1053](#)
- DeleteProgram
 - OpenTK::Graphics::ES20::GL, [680](#)
 - OpenTK::Graphics::OpenGL::GL, [1053](#), [1054](#)
- DeleteQueries
 - OpenTK::Graphics::OpenGL::GL, [1054](#), [1055](#)
- DeleteShader
 - OpenTK::Graphics::ES20::GL, [680](#)
 - OpenTK::Graphics::OpenGL::GL, [1055](#), [1056](#)
- DeleteTextures
 - OpenTK::Graphics::ES10::GL, [458](#), [459](#)
 - OpenTK::Graphics::ES11::GL, [552](#), [553](#)
 - OpenTK::Graphics::ES20::GL, [681](#), [682](#)
 - OpenTK::Graphics::OpenGL::GL, [1056](#), [1057](#)
- Delta
 - OpenTK::Input::JoystickMoveEventArgs, [1512](#)
 - OpenTK::Input::MouseWheelEventArgs, [1531](#)
- DeltaPrecise
 - OpenTK::Input::MouseWheelEventArgs, [1531](#)
- Depth
 - OpenTK::Graphics::GraphicsMode, [798](#)
- DepthFunc
 - OpenTK::Graphics::ES10::GL, [459](#)
 - OpenTK::Graphics::ES11::GL, [553](#)
 - OpenTK::Graphics::ES20::GL, [682](#)
 - OpenTK::Graphics::OpenGL::GL, [1057](#)
- DepthMask
 - OpenTK::Graphics::ES10::GL, [460](#)
 - OpenTK::Graphics::ES11::GL, [553](#)
 - OpenTK::Graphics::ES20::GL, [682](#)
 - OpenTK::Graphics::OpenGL::GL, [1058](#)
- DepthRange
 - OpenTK::Graphics::ES10::GL, [460](#)
 - OpenTK::Graphics::ES11::GL, [554](#)
 - OpenTK::Graphics::ES20::GL, [682](#)
 - OpenTK::Graphics::OpenGL::GL, [1058](#)
- Description
 - OpenTK::Input::IInputDevice, [1502](#)
 - OpenTK::Input::JoystickDevice, [1510](#)
 - OpenTK::Input::KeyboardDevice, [1515](#)
 - OpenTK::Input::MouseDevice, [1523](#)
- DetachShader
 - OpenTK::Graphics::ES20::GL, [683](#)
 - OpenTK::Graphics::OpenGL::GL, [1058](#)
- Determinant
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- DeviceID
 - OpenTK::Input::KeyboardDevice, [1515](#)
 - OpenTK::Input::MouseDevice, [1523](#)
- DeviceType
 - OpenTK::Input::IInputDevice, [1502](#)
 - OpenTK::Input::JoystickDevice, [1510](#)
 - OpenTK::Input::KeyboardDevice, [1515](#)
 - OpenTK::Input::MouseDevice, [1523](#)
- DimGray
 - OpenTK::Graphics::Color4, [412](#)
- DirectRendering
 - OpenTK::Graphics::GraphicsContext, [787](#)
- DisplayName
 - OpenTK::Platform::Windows, [278](#)
- Dispose
 - OpenTK::Audio::AudioCapture, [297](#)

- OpenTK::Audio::AudioContext, 305
- OpenTK::GameWindow, 382
- OpenTK::GLControl, 391
- OpenTK::Graphics::GraphicsContext, 786
- OpenTK::NativeWindow, 1591
- Disposed
 - OpenTK::INativeWindow, 1499
 - OpenTK::NativeWindow, 1598
- DistanceModel
 - OpenTK::Audio::OpenAL, 26
- Distortion
 - OpenTK::Audio::OpenAL, 40
- DistortionEdge
 - OpenTK::Audio::OpenAL, 34
- DistortionEQBandwidth
 - OpenTK::Audio::OpenAL, 34
- DistortionEQCenter
 - OpenTK::Audio::OpenAL, 34
- DistortionGain
 - OpenTK::Audio::OpenAL, 34
- DistortionLowpassCutoff
 - OpenTK::Audio::OpenAL, 34
- Div
 - OpenTK::Vector2, 1644, 1645
 - OpenTK::Vector2d, 1668
 - OpenTK::Vector3, 1706, 1707
 - OpenTK::Vector3d, 1739, 1740
 - OpenTK::Vector4, 1779
 - OpenTK::Vector4d, 1807
- Divide
 - OpenTK::Vector2, 1645, 1646
 - OpenTK::Vector2d, 1668, 1669
 - OpenTK::Vector3, 1707, 1708
 - OpenTK::Vector3d, 1740, 1741
 - OpenTK::Vector4, 1780
 - OpenTK::Vector4d, 1807, 1808
- DodgerBlue
 - OpenTK::Graphics::Color4, 413
- DopplerFactor
 - OpenTK::Audio::OpenAL, 26
- DopplerVelocity
 - OpenTK::Audio::OpenAL, 26
- Dot
 - OpenTK::Vector2, 1646
 - OpenTK::Vector2d, 1670
 - OpenTK::Vector3, 1708, 1709
 - OpenTK::Vector3d, 1741
 - OpenTK::Vector4, 1781
 - OpenTK::Vector4d, 1809
- Down
 - OpenTK::Input, 250
- DrawArrays
 - OpenTK::Graphics::ES10::GL, 460
 - OpenTK::Graphics::ES11::GL, 554
 - OpenTK::Graphics::ES20::GL, 683
 - OpenTK::Graphics::OpenGL::GL, 1059
- DrawBuffer
 - OpenTK::Graphics::OpenGL::GL, 1059
- DrawBuffers
 - OpenTK::Graphics::OpenGL::GL, 1059, 1060
- DrawElements
 - OpenTK::Graphics::ES10::GL, 461
 - OpenTK::Graphics::ES11::GL, 554
 - OpenTK::Graphics::ES20::GL, 684
 - OpenTK::Graphics::OpenGL::GL, 1060
- DrawElements< T3 >
 - OpenTK::Graphics::ES10::GL, 461–463
 - OpenTK::Graphics::ES11::GL, 555, 556
 - OpenTK::Graphics::ES20::GL, 684, 685
 - OpenTK::Graphics::OpenGL::GL, 1061, 1062
- DRAWFRAME
 - OpenTK::Platform::Windows, 278
- DrawPixels
 - OpenTK::Graphics::OpenGL::GL, 1063
- DrawPixels< T4 >
 - OpenTK::Graphics::OpenGL::GL, 1063–1066
- DrawRangeElements
 - OpenTK::Graphics::OpenGL::GL, 1066, 1067
- DrawRangeElements< T5 >

- OpenTK::Graphics::OpenGL::GL,
1067–1072
- E
 - OpenTK::Input, 252
- EaxReverbAirAbsorptionGainHF
 - OpenTK::Audio::OpenAL, 38
- EaxReverbDecayHFRatio
 - OpenTK::Audio::OpenAL, 37
- EaxReverbDecayLFRatio
 - OpenTK::Audio::OpenAL, 37
- EaxReverbDecayTime
 - OpenTK::Audio::OpenAL, 37
- EaxReverbDensity
 - OpenTK::Audio::OpenAL, 36
- EaxReverbDiffusion
 - OpenTK::Audio::OpenAL, 36
- EaxReverbEchoDepth
 - OpenTK::Audio::OpenAL, 37
- EaxReverbEchoTime
 - OpenTK::Audio::OpenAL, 37
- EaxReverbGain
 - OpenTK::Audio::OpenAL, 36
- EaxReverbGainHF
 - OpenTK::Audio::OpenAL, 37
- EaxReverbGainLF
 - OpenTK::Audio::OpenAL, 37
- EaxReverbHFReference
 - OpenTK::Audio::OpenAL, 38
- EaxReverbLateReverbDelay
 - OpenTK::Audio::OpenAL, 37
- EaxReverbLateReverbGain
 - OpenTK::Audio::OpenAL, 37
- EaxReverbLateReverbPan
 - OpenTK::Audio::OpenAL, 32
- EaxReverbLFReference
 - OpenTK::Audio::OpenAL, 38
- EaxReverbModulationDepth
 - OpenTK::Audio::OpenAL, 38
- EaxReverbModulationTime
 - OpenTK::Audio::OpenAL, 37
- EaxReverbReflectionsDelay
 - OpenTK::Audio::OpenAL, 37
- EaxReverbReflectionsGain
 - OpenTK::Audio::OpenAL, 37
- EaxReverbReflectionsPan
 - OpenTK::Audio::OpenAL, 32
- OpenTK::Audio::OpenAL, 32
- Echo
 - OpenTK::Audio::OpenAL, 40
- EchoDamping
 - OpenTK::Audio::OpenAL, 35
- EchoDelay
 - OpenTK::Audio::OpenAL, 34
- EchoFeedback
 - OpenTK::Audio::OpenAL, 35
- EchoLRDelay
 - OpenTK::Audio::OpenAL, 34
- EchoSpread
 - OpenTK::Audio::OpenAL, 35
- EdgeFlag
 - OpenTK::Graphics::OpenGL::GL,
1072
- EdgeFlagPointer
 - OpenTK::Graphics::OpenGL::GL,
1073
- EdgeFlagPointer< T1 >
 - OpenTK::Graphics::OpenGL::GL,
1073, 1074
- Effect
 - OpenTK::Audio::OpenAL::EffectsExtension,
327, 328
- EffectsExtension
 - OpenTK::Audio::OpenAL::EffectsExtension,
319
- EffectslotAuxiliarySendAuto
 - OpenTK::Audio::OpenAL, 32
- EffectslotEffect
 - OpenTK::Audio::OpenAL, 32
- EffectslotGain
 - OpenTK::Audio::OpenAL, 31
- EffectType
 - OpenTK::Audio::OpenAL, 40
- EfxAirAbsorptionFactor
 - OpenTK::Audio::OpenAL, 30
- EfxAuxiliaryf
 - OpenTK::Audio::OpenAL, 31
- EfxAuxiliaryi
 - OpenTK::Audio::OpenAL, 31
- EfxAuxiliarySendFilter
 - OpenTK::Audio::OpenAL, 28
- EfxAuxiliarySendFilterGainAuto
 - OpenTK::Audio::OpenAL, 29

- EfxDirectFilterGainHighFrequencyAuto
 - OpenTK::Audio::OpenAL, [29](#)
- EfxEffect3f
 - OpenTK::Audio::OpenAL, [32](#)
- EfxEffectf
 - OpenTK::Audio::OpenAL, [32](#)
- EfxEffecti
 - OpenTK::Audio::OpenAL, [38](#)
- EfxEffectType
 - OpenTK::Audio::OpenAL, [40](#)
- EfxFilterf
 - OpenTK::Audio::OpenAL, [41](#)
- EfxFilteri
 - OpenTK::Audio::OpenAL, [41](#)
- EfxFilterType
 - OpenTK::Audio::OpenAL, [41](#)
- EfxFormantFilterSettings
 - OpenTK::Audio::OpenAL, [41](#)
- EfxMajorVersion
 - OpenTK::Audio::OpenAL, [22](#)
- EfxMaxAuxiliarySends
 - OpenTK::Audio::OpenAL, [21](#), [22](#)
- EfxMinorVersion
 - OpenTK::Audio::OpenAL, [22](#)
- EfxRoomRolloffFactor
 - OpenTK::Audio::OpenAL, [30](#)
- Embedded
 - OpenTK::Graphics, [44](#)
- Enable
 - OpenTK::Graphics::ES10::GL, [463](#)
 - OpenTK::Graphics::ES11::GL, [557](#)
 - OpenTK::Graphics::ES20::GL, [686](#)
 - OpenTK::Graphics::OpenGL::GL, [1074](#), [1075](#)
- EnableClientState
 - OpenTK::Graphics::ES10::GL, [463](#)
 - OpenTK::Graphics::ES11::GL, [557](#)
 - OpenTK::Graphics::OpenGL::GL, [1075](#)
- EnableVertexAttribArray
 - OpenTK::Graphics::ES20::GL, [686](#)
 - OpenTK::Graphics::OpenGL::GL, [1075](#), [1076](#)
- End
 - OpenTK::Input, [251](#)
- EndAnchor
 - OpenTK::BezierCurveCubic, [352](#)
 - OpenTK::BezierCurveQuadric, [355](#)
- EnsureUndisposed
 - OpenTK::NativeWindow, [1591](#)
- Enter
 - OpenTK::Input, [250](#)
- EntryPoint
 - OpenTK::AutoGeneratedAttribute, [344](#)
- Epsilon
 - OpenTK::Half, [1490](#)
- Equalizer
 - OpenTK::Audio::OpenAL, [41](#)
- EqualizerHighCutoff
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerHighGain
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerLowCutoff
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerLowGain
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerMid1Center
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerMid1Gain
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerMid1Width
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerMid2Center
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerMid2Gain
 - OpenTK::Audio::OpenAL, [36](#)
- EqualizerMid2Width
 - OpenTK::Audio::OpenAL, [36](#)
- Equals
 - OpenTK::Audio::AudioContext, [305](#)
 - OpenTK::ContextHandle, [364](#)
 - OpenTK::DisplayResolution, [372](#)
 - OpenTK::Graphics::Color4, [406](#)
 - OpenTK::Graphics::ColorFormat, [425](#)
 - OpenTK::Graphics::GraphicsMode, [797](#)
 - OpenTK::Half, [1485](#)
 - OpenTK::Input::KeyboardState, [1518](#)
 - OpenTK::Input::MouseState, [1529](#)

- OpenTK::Matrix4, [1547](#)
- OpenTK::Matrix4d, [1573](#)
- OpenTK::Quaternion, [1609](#)
- OpenTK::Quaterniond, [1624](#)
- OpenTK::Vector2, [1647](#)
- OpenTK::Vector2d, [1670](#), [1671](#)
- OpenTK::Vector2h, [1690](#)
- OpenTK::Vector3, [1709](#)
- OpenTK::Vector3d, [1742](#)
- OpenTK::Vector3h, [1765](#)
- OpenTK::Vector4, [1781](#), [1782](#)
- OpenTK::Vector4d, [1809](#), [1810](#)
- OpenTK::Vector4h, [1831](#)
- ErrorChecking
 - OpenTK::Graphics::GraphicsContext, [F11](#)
[788](#)
 - OpenTK::Graphics::IGraphicsContext, [F12](#)
[802](#)
- Escape
 - OpenTK::Input, [250](#)
- EvalCoord1
 - OpenTK::Graphics::OpenGL::GL, [1076](#), [1077](#)
- EvalCoord2
 - OpenTK::Graphics::OpenGL::GL, [1077–1079](#)
- EvalMesh1
 - OpenTK::Graphics::OpenGL::GL, [1080](#)
- EvalMesh2
 - OpenTK::Graphics::OpenGL::GL, [1080](#)
- EvalPoint1
 - OpenTK::Graphics::OpenGL::GL, [1080](#)
- EvalPoint2
 - OpenTK::Graphics::OpenGL::GL, [1081](#)
- EXCLUDE
 - OpenTK::Platform::Windows, [276](#)
- ExeType
 - OpenTK::Platform::Windows, [278](#)
- EXINPUTSINK
 - OpenTK::Platform::Windows, [277](#)
- Exists
 - OpenTK::INativeWindow, [1496](#)
- OpenTK::NativeWindow, [1596](#)
- Exit
 - OpenTK::GameWindow, [382](#)
- ExponentDistance
 - OpenTK::Audio::OpenAL, [23](#)
- Extensions
 - OpenTK::Audio::OpenAL, [22](#), [27](#)
- F
 - OpenTK::Input, [252](#)
- F1
 - OpenTK::Input, [249](#)
- F10
 - OpenTK::Input, [250](#)
- F11
 - OpenTK::Input, [250](#)
- F12
 - OpenTK::Input, [250](#)
- F13
 - OpenTK::Input, [250](#)
- F14
 - OpenTK::Input, [250](#)
- F15
 - OpenTK::Input, [250](#)
- F16
 - OpenTK::Input, [250](#)
- F17
 - OpenTK::Input, [250](#)
- F18
 - OpenTK::Input, [250](#)
- F19
 - OpenTK::Input, [250](#)
- F2
 - OpenTK::Input, [249](#)
- F20
 - OpenTK::Input, [250](#)
- F21
 - OpenTK::Input, [250](#)
- F22
 - OpenTK::Input, [250](#)
- F23
 - OpenTK::Input, [250](#)
- F24
 - OpenTK::Input, [250](#)
- F25
 - OpenTK::Input, [250](#)

- F26
 - OpenTK::Input, [250](#)
- F27
 - OpenTK::Input, [250](#)
- F28
 - OpenTK::Input, [250](#)
- F29
 - OpenTK::Input, [250](#)
- F3
 - OpenTK::Input, [249](#)
- F30
 - OpenTK::Input, [250](#)
- F31
 - OpenTK::Input, [250](#)
- F32
 - OpenTK::Input, [250](#)
- F33
 - OpenTK::Input, [250](#)
- F34
 - OpenTK::Input, [250](#)
- F35
 - OpenTK::Input, [250](#)
- F4
 - OpenTK::Input, [249](#)
- F5
 - OpenTK::Input, [249](#)
- F6
 - OpenTK::Input, [249](#)
- F7
 - OpenTK::Input, [250](#)
- F8
 - OpenTK::Input, [250](#)
- F9
 - OpenTK::Input, [250](#)
- FeedbackBuffer
 - OpenTK::Graphics::OpenGL::GL, [1081](#)
- Filter
 - OpenTK::Audio::OpenAL::EffectsExtensionOpenTK::Graphics::OpenGL::GL, [328](#), [329](#)
- FilterType
 - OpenTK::Audio::OpenAL, [41](#)
- Finish
 - OpenTK::Graphics::ES10::GL, [464](#)
 - OpenTK::Graphics::ES11::GL, [557](#)
 - OpenTK::Graphics::ES20::GL, [687](#)
 - OpenTK::Graphics::OpenGL::GL, [1082](#)
- Firebrick
 - OpenTK::Graphics::Color4, [413](#)
- FirstControlPoint
 - OpenTK::BezierCurveCubic, [352](#)
- Flanger
 - OpenTK::Audio::OpenAL, [40](#)
- FlangerDelay
 - OpenTK::Audio::OpenAL, [35](#)
- FlangerDepth
 - OpenTK::Audio::OpenAL, [35](#)
- FlangerFeedback
 - OpenTK::Audio::OpenAL, [35](#)
- FlangerPhase
 - OpenTK::Audio::OpenAL, [38](#)
- FlangerRate
 - OpenTK::Audio::OpenAL, [35](#)
- FlangerWaveform
 - OpenTK::Audio::OpenAL, [38](#)
- FloralWhite
 - OpenTK::Graphics::Color4, [413](#)
- Flush
 - OpenTK::Graphics::ES10::GL, [464](#)
 - OpenTK::Graphics::ES11::GL, [558](#)
 - OpenTK::Graphics::ES20::GL, [687](#)
 - OpenTK::Graphics::OpenGL::GL, [1082](#)
- Focused
 - OpenTK::INativeWindow, [1496](#)
 - OpenTK::NativeWindow, [1596](#)
- FocusedChanged
 - OpenTK::INativeWindow, [1499](#)
 - OpenTK::NativeWindow, [1598](#)
- Fog
 - OpenTK::Graphics::ES10::GL, [464](#), [465](#)
 - OpenTK::Graphics::ES11::GL, [558](#)
 - OpenTK::Graphics::OpenGL::GL, [1082–1084](#)
- FogCoord
 - OpenTK::Graphics::OpenGL::GL, [1084](#)
- FogCoordPointer
 - OpenTK::Graphics::OpenGL::GL, [1085](#)

- FogCoordPointer< T2 >
 - OpenTK::Graphics::OpenGL::GL, [1085](#), [1086](#)
- FORCEMINIMIZE
 - OpenTK::Platform::Windows, [280](#)
- ForestGreen
 - OpenTK::Graphics::Color4, [413](#)
- ForwardCompatible
 - OpenTK::Graphics, [44](#)
- Four
 - OpenTK::Audio::AudioContext, [302](#)
- FRAMECHANGED
 - OpenTK::Platform::Windows, [278](#)
- FrameEventArgs
 - OpenTK::FrameEventArgs, [375](#)
- Frequency
 - OpenTK::Audio::OpenAL, [21](#), [26](#)
- FrequencyShifter
 - OpenTK::Audio::OpenAL, [40](#)
- FrequencyShifterFrequency
 - OpenTK::Audio::OpenAL, [35](#)
- FrequencyShifterLeftDirection
 - OpenTK::Audio::OpenAL, [38](#)
- FrequencyShifterRightDirection
 - OpenTK::Audio::OpenAL, [39](#)
- FromAxisAngle
 - OpenTK::Quaternion, [1610](#)
 - OpenTK::Quaterniond, [1624](#)
- FromBinaryStream
 - OpenTK::Half, [1486](#)
 - OpenTK::Vector2h, [1690](#)
 - OpenTK::Vector3h, [1765](#)
 - OpenTK::Vector4h, [1831](#)
- FromBytes
 - OpenTK::Half, [1486](#)
 - OpenTK::Vector2h, [1690](#)
 - OpenTK::Vector3h, [1765](#)
 - OpenTK::Vector4h, [1831](#)
- FromTLRB
 - OpenTK::Box2, [360](#)
- FrontFace
 - OpenTK::Graphics::ES10::GL, [465](#)
 - OpenTK::Graphics::ES11::GL, [559](#)
 - OpenTK::Graphics::ES20::GL, [687](#)
 - OpenTK::Graphics::OpenGL::GL, [1087](#)
- Frustum
 - OpenTK::Graphics::ES10::GL, [465](#)
 - OpenTK::Graphics::ES11::GL, [559](#)
 - OpenTK::Graphics::OpenGL::GL, [1087](#)
 - OpenTK::Matrix4, [1547](#)
 - OpenTK::Matrix4d, [1573](#)
- Fuchsia
 - OpenTK::Graphics::Color4, [413](#)
- G
 - OpenTK::Graphics::Color4, [408](#)
 - OpenTK::Input, [252](#)
- Gain
 - OpenTK::Audio::OpenAL, [27](#), [30](#)
- Gainsboro
 - OpenTK::Graphics::Color4, [413](#)
- GameWindow
 - OpenTK::GameWindow, [380–382](#)
- GdiCharset
 - OpenTK::Platform::Windows, [275](#)
- GenAuxiliaryEffectSlot
 - OpenTK::Audio::OpenAL::EffectsExtension, [329](#), [330](#)
- GenAuxiliaryEffectSlots
 - OpenTK::Audio::OpenAL::EffectsExtension, [330](#), [331](#)
- GenBuffers
 - OpenTK::Graphics::ES11::GL, [559](#), [560](#)
 - OpenTK::Graphics::ES20::GL, [687](#), [688](#)
 - OpenTK::Graphics::OpenGL::GL, [1087](#), [1088](#)
- GenEffect
 - OpenTK::Audio::OpenAL::EffectsExtension, [331](#), [332](#)
- GenEffects
 - OpenTK::Audio::OpenAL::EffectsExtension, [332](#), [333](#)
- GenFilter
 - OpenTK::Audio::OpenAL::EffectsExtension, [333](#)
- GenFilters
 - OpenTK::Audio::OpenAL::EffectsExtension, [333](#), [334](#)

- GenLists
 - OpenTK::Graphics::OpenGL::GL, [1089](#)
- GenQueries
 - OpenTK::Graphics::OpenGL::GL, [1089](#), [1090](#)
- GenTextures
 - OpenTK::Graphics::ES10::GL, [465](#)–[467](#)
 - OpenTK::Graphics::ES11::GL, [560](#)–[562](#)
 - OpenTK::Graphics::ES20::GL, [688](#)–[690](#)
 - OpenTK::Graphics::OpenGL::GL, [1090](#)–[1092](#)
- GetActiveAttrib
 - OpenTK::Graphics::ES20::GL, [690](#)–[692](#)
 - OpenTK::Graphics::OpenGL::GL, [1092](#), [1093](#)
- GetActiveUniform
 - OpenTK::Graphics::ES20::GL, [693](#)–[695](#)
 - OpenTK::Graphics::OpenGL::GL, [1094](#)–[1096](#)
- GetAddress
 - OpenTK::BindingsBase, [357](#)
 - OpenTK::Graphics::GraphicsBindingsBase, [781](#)
 - OpenTK::Graphics::IGraphicsContextInternal, [804](#)
- GetAttachedShaders
 - OpenTK::Graphics::ES20::GL, [696](#)–[698](#)
 - OpenTK::Graphics::OpenGL::GL, [1096](#)–[1098](#)
- GetAttribLocation
 - OpenTK::Graphics::ES20::GL, [698](#)
 - OpenTK::Graphics::OpenGL::GL, [1098](#)
- GetAuxiliaryEffectSlot
 - OpenTK::Audio::OpenAL::EffectsExtension, [334](#), [335](#)
- GetBufferMode
 - OpenTK::Audio::OpenAL::XRamExtension, [342](#)
- GetBufferParameter
 - OpenTK::Graphics::ES11::GL, [562](#), [563](#)
 - OpenTK::Graphics::ES20::GL, [698](#), [699](#)
 - OpenTK::Graphics::OpenGL::GL, [1099](#), [1100](#)
- GetBufferPointer
 - OpenTK::Graphics::OpenGL::GL, [1100](#)
- GetBufferPointer< T2 >
 - OpenTK::Graphics::OpenGL::GL, [1100](#)–[1102](#)
- GetBufferSubData
 - OpenTK::Graphics::OpenGL::GL, [1102](#)
- GetBufferSubData< T3 >
 - OpenTK::Graphics::OpenGL::GL, [1102](#)–[1104](#)
- GetBytes
 - OpenTK::Half, [1486](#)
 - OpenTK::Vector2h, [1690](#)
 - OpenTK::Vector3h, [1766](#)
 - OpenTK::Vector4h, [1831](#)
- GetClipPlane
 - OpenTK::Graphics::ES11::GL, [563](#), [564](#)
 - OpenTK::Graphics::OpenGL::GL, [1104](#), [1105](#)
- GetColorTable
 - OpenTK::Graphics::OpenGL::GL, [1105](#)
- GetColorTable< T3 >
 - OpenTK::Graphics::OpenGL::GL, [1106](#)–[1108](#)
- GetColorTableParameter
 - OpenTK::Graphics::OpenGL::GL, [1109](#)–[1112](#)
- GetCompressedTexImage
 - OpenTK::Graphics::OpenGL::GL, [1112](#)
- GetCompressedTexImage< T2 >
 - OpenTK::Graphics::OpenGL::GL, [1113](#), [1114](#)
- GetConvolutionFilter

- OpenTK::Graphics::OpenGL::GL,
1115
- GetConvolutionFilter< T3 >
 - OpenTK::Graphics::OpenGL::GL,
1115–1118
- GetConvolutionParameter
 - OpenTK::Graphics::OpenGL::GL,
1118–1121
- GetEffect
 - OpenTK::Audio::OpenAL::EffectsExtension,
335–337
- GetError
 - OpenTK::Graphics::ES10::GL, 467
 - OpenTK::Graphics::ES11::GL, 564
 - OpenTK::Graphics::ES20::GL, 700
 - OpenTK::Graphics::OpenGL::GL,
1121
- GetFilter
 - OpenTK::Audio::OpenAL::EffectsExtension,
337, 338
- GetHashCode
 - OpenTK::Audio::AudioContext, 305
 - OpenTK::ContextHandle, 364
 - OpenTK::DisplayResolution, 372
 - OpenTK::Graphics::Color4, 406
 - OpenTK::Graphics::ColorFormat,
426
 - OpenTK::Graphics::GraphicsMode,
797
 - OpenTK::Input::KeyboardDevice,
1514
 - OpenTK::Input::MouseDevice, 1523
 - OpenTK::Matrix4, 1548
 - OpenTK::Matrix4d, 1574
 - OpenTK::Quaternion, 1610
 - OpenTK::Quaterniond, 1625
 - OpenTK::Vector2, 1647
 - OpenTK::Vector2d, 1671
 - OpenTK::Vector3, 1709
 - OpenTK::Vector3d, 1742
 - OpenTK::Vector4, 1782
 - OpenTK::Vector4d, 1810
- GetHistogram
 - OpenTK::Graphics::OpenGL::GL,
1121
- GetHistogram< T4 >
 - OpenTK::Graphics::OpenGL::GL,
1122–1124
- GetHistogramParameter
 - OpenTK::Graphics::OpenGL::GL,
1125–1127
- GetLight
 - OpenTK::Graphics::ES11::GL, 564,
565
 - OpenTK::Graphics::OpenGL::GL,
1127–1130
- GetMap
 - OpenTK::Graphics::OpenGL::GL,
1130–1134
- GetMaterial
 - OpenTK::Graphics::ES11::GL, 566
 - OpenTK::Graphics::OpenGL::GL,
1135–1137
- GetMinmax
 - OpenTK::Graphics::OpenGL::GL,
1137
- GetMinmax< T4 >
 - OpenTK::Graphics::OpenGL::GL,
1138–1140
- GetMinmaxParameter
 - OpenTK::Graphics::OpenGL::GL,
1141–1143
- GetObjectData
 - OpenTK::Half, 1486
 - OpenTK::Vector2h, 1691
 - OpenTK::Vector3h, 1766
 - OpenTK::Vector4h, 1832
- GetPixelMap
 - OpenTK::Graphics::OpenGL::GL,
1143–1148
- GetPointer
 - OpenTK::Graphics::ES11::GL, 567
 - OpenTK::Graphics::OpenGL::GL,
1148
- GetPointer< T1 >
 - OpenTK::Graphics::ES11::GL, 567,
568
 - OpenTK::Graphics::OpenGL::GL,
1148–1150
- GetPolygonStipple
 - OpenTK::Graphics::OpenGL::GL,
1150, 1151

- GetProgram
 - OpenTK::Graphics::ES20::GL, [700–702](#)
 - OpenTK::Graphics::OpenGL::GL, [1151–1153](#)
- GetProgramInfoLog
 - OpenTK::Graphics::ES20::GL, [702–704](#)
 - OpenTK::Graphics::OpenGL::GL, [1153–1155](#)
- GetQuery
 - OpenTK::Graphics::OpenGL::GL, [1155, 1156](#)
- GetQueryObject
 - OpenTK::Graphics::OpenGL::GL, [1156–1158](#)
- GetRamFree
 - OpenTK::Audio::OpenAL::XRamExtension, [343](#)
- GetRamSize
 - OpenTK::Audio::OpenAL::XRamExtension, [343](#)
- GetSeparableFilter
 - OpenTK::Graphics::OpenGL::GL, [1159](#)
- GetSeparableFilter< T3, T4, T5 >
 - OpenTK::Graphics::OpenGL::GL, [1159–1162](#)
- GetSeparableFilter< T4, T5 >
 - OpenTK::Graphics::OpenGL::GL, [1163–1165](#)
- GetSeparableFilter< T5 >
 - OpenTK::Graphics::OpenGL::GL, [1166–1168](#)
- GetShader
 - OpenTK::Graphics::ES20::GL, [704–706](#)
 - OpenTK::Graphics::OpenGL::GL, [1169–1171](#)
- GetShaderInfoLog
 - OpenTK::Graphics::ES20::GL, [706–708](#)
 - OpenTK::Graphics::OpenGL::GL, [1171, 1172](#)
- GetShaderSource
 - OpenTK::Graphics::ES20::GL, [708–710](#)
 - OpenTK::Graphics::OpenGL::GL, [1172, 1173](#)
- GetString
 - OpenTK::Graphics::ES10::GL, [467](#)
 - OpenTK::Graphics::ES11::GL, [569](#)
 - OpenTK::Graphics::ES20::GL, [710](#)
 - OpenTK::Graphics::OpenGL::GL, [1174](#)
- GetTexEnv
 - OpenTK::Graphics::ES11::GL, [569–572](#)
 - OpenTK::Graphics::OpenGL::GL, [1175–1177](#)
- GetTexGen
 - OpenTK::Graphics::OpenGL::GL, [1178–1180](#)
- GetTexImage
 - OpenTK::Graphics::OpenGL::GL, [1181](#)
- GetTexImage< T4 >
 - OpenTK::Graphics::OpenGL::GL, [1182–1184](#)
- GetTexLevelParameter
 - OpenTK::Graphics::OpenGL::GL, [1185–1189](#)
- GetTexParameter
 - OpenTK::Graphics::ES11::GL, [572–575](#)
 - OpenTK::Graphics::ES20::GL, [711–713](#)
 - OpenTK::Graphics::OpenGL::GL, [1189–1192](#)
- GetUniform
 - OpenTK::Graphics::ES20::GL, [714–717](#)
 - OpenTK::Graphics::OpenGL::GL, [1192–1196](#)
- GetUniformLocation
 - OpenTK::Graphics::ES20::GL, [717, 718](#)
 - OpenTK::Graphics::OpenGL::GL, [1197](#)
- GetVertexAttrib

- OpenTK::Graphics::ES20::GL, [718–722](#)
- OpenTK::Graphics::OpenGL::GL, [1197–1204](#)
- GetVertexAttribPointer
 - OpenTK::Graphics::ES20::GL, [723](#)
 - OpenTK::Graphics::OpenGL::GL, [1204, 1205](#)
- GetVertexAttribPointer< T2 >
 - OpenTK::Graphics::ES20::GL, [723–726](#)
 - OpenTK::Graphics::OpenGL::GL, [1205–1208](#)
- GhostWhite
 - OpenTK::Graphics::Color4, [413](#)
- GLControl
 - OpenTK::GLControl, [390](#)
- Gold
 - OpenTK::Graphics::Color4, [413](#)
- Goldenrod
 - OpenTK::Graphics::Color4, [413](#)
- GrabScreenshot
 - OpenTK::GLControl, [391](#)
- GraphicsContext
 - OpenTK::Graphics::GraphicsContext, [784, 785](#)
- GraphicsContextException
 - OpenTK::Graphics::GraphicsContextException, [789](#)
- GraphicsContextFlags
 - OpenTK::Graphics, [44](#)
- GraphicsContextMissingException
 - OpenTK::Graphics::GraphicsContextMissingException, [790](#)
- GraphicsErrorException
 - OpenTK::Graphics::GraphicsErrorException, [792](#)
- GraphicsException
 - OpenTK::GraphicsException, [1480](#)
- GraphicsMode
 - OpenTK::GLControl, [393](#)
 - OpenTK::Graphics::GraphicsContext, [788](#)
 - OpenTK::Graphics::GraphicsMode, [795, 796](#)
- OpenTK::Graphics::IGraphicsContext, [802](#)
- GraphicsModeException
 - OpenTK::Graphics::GraphicsModeException, [800](#)
- Gray
 - OpenTK::Graphics::Color4, [413](#)
- Green
 - OpenTK::Graphics::Color4, [414](#)
 - OpenTK::Graphics::ColorFormat, [427](#)
- GreenYellow
 - OpenTK::Graphics::Color4, [414](#)
- GWL
 - OpenTK::Platform::Windows, [275](#)
- H
 - OpenTK::Input, [252](#)
- Half
 - OpenTK::Half, [1484, 1485](#)
- Handle
 - OpenTK::ContextHandle, [366](#)
- Hardware
 - OpenTK::Audio::OpenAL::XRamExtension, [341](#)
- Height
 - OpenTK::Box2, [361](#)
 - OpenTK::DisplayDevice, [370](#)
 - OpenTK::DisplayResolution, [373](#)
 - OpenTK::INativeWindow, [1496](#)
 - OpenTK::NativeWindow, [1596](#)
- Hid
 - OpenTK::Input, [248](#)
 - OpenTK::Platform::Windows, [279](#)
- HIDEWINDOW
 - OpenTK::Platform::Windows, [278](#)
- Highpass
 - OpenTK::Audio::OpenAL, [41](#)
- HighpassGain
 - OpenTK::Audio::OpenAL, [41](#)
- HighpassGainLF
 - OpenTK::Audio::OpenAL, [41](#)
- Hint
 - OpenTK::Graphics::ES10::GL, [467](#)
 - OpenTK::Graphics::ES11::GL, [575](#)

- OpenTK::Graphics::ES20::GL, [727](#)
- OpenTK::Graphics::OpenGL::GL, [1209](#)
- Histogram
 - OpenTK::Graphics::OpenGL::GL, [1209](#)
- Home
 - OpenTK::Input, [251](#)
- Honeydew
 - OpenTK::Graphics::Color4, [414](#)
- HOTKEY
 - OpenTK::Platform::Windows, [276](#)
- HotPink
 - OpenTK::Graphics::Color4, [414](#)
- I
 - OpenTK::Input, [252](#)
- Icon
 - OpenTK::INativeWindow, [1497](#)
 - OpenTK::NativeWindow, [1596](#)
 - OpenTK::Platform::Windows, [278](#)
- IconChanged
 - OpenTK::INativeWindow, [1499](#)
 - OpenTK::NativeWindow, [1598](#)
- IconLocation
 - OpenTK::Platform::Windows, [278](#)
- Identity
 - OpenTK::Matrix4, [1555](#)
 - OpenTK::Matrix4d, [1581](#)
 - OpenTK::Quaternion, [1617](#)
 - OpenTK::Quaterniond, [1632](#)
- IllegalCommand
 - OpenTK::Audio::OpenAL, [24](#)
- IllegalEnum
 - OpenTK::Audio::OpenAL, [24](#)
- Implementation
 - OpenTK::Graphics::IGraphicsContext, [804](#)
- Index
 - OpenTK::Graphics::GraphicsMode, [799](#)
 - OpenTK::Graphics::OpenGL::GL, [1210](#), [1211](#)
- IndexMask
 - OpenTK::Graphics::OpenGL::GL, [1212](#)
- IndexPointer
 - OpenTK::Graphics::OpenGL::GL, [1212](#)
- IndexPointer< T2 >
 - OpenTK::Graphics::OpenGL::GL, [1212–1214](#)
- IndianRed
 - OpenTK::Graphics::Color4, [414](#)
- Indigo
 - OpenTK::Graphics::Color4, [414](#)
- Init
 - OpenTK::Toolkit, [1633](#)
- Initial
 - OpenTK::Audio::OpenAL, [31](#)
- InitNames
 - OpenTK::Graphics::OpenGL::GL, [1214](#)
- INPUT
 - OpenTK::Platform::Windows, [276](#)
- INPUT_LEGACY
 - OpenTK::Platform::Windows, [276](#)
- InputDeviceType
 - OpenTK::Input, [247](#)
- InputDriver
 - OpenTK::INativeWindow, [1497](#)
 - OpenTK::NativeWindow, [1596](#)
- INPUTSINK
 - OpenTK::Platform::Windows, [277](#)
- Insert
 - OpenTK::Input, [251](#)
- InterleavedArrays
 - OpenTK::Graphics::OpenGL::GL, [1214](#)
- InterleavedArrays< T2 >
 - OpenTK::Graphics::OpenGL::GL, [1215](#), [1216](#)
- Invalid
 - OpenTK::Audio::OpenAL, [20](#)
- InvalidContext
 - OpenTK::Audio::OpenAL, [21](#)
- InvalidDevice
 - OpenTK::Audio::OpenAL, [21](#)
- InvalidEnum
 - OpenTK::Audio::OpenAL, [21](#), [24](#)
- InvalidName
 - OpenTK::Audio::OpenAL, [24](#)

- InvalidOperation
 - OpenTK::Audio::OpenAL, [24](#)
- InvalidValue
 - OpenTK::Audio::OpenAL, [21](#), [24](#)
- InverseDistance
 - OpenTK::Audio::OpenAL, [23](#)
- InverseDistanceClamped
 - OpenTK::Audio::OpenAL, [23](#)
- Invert
 - OpenTK::Matrix4, [1548](#)
 - OpenTK::Matrix4d, [1574](#)
 - OpenTK::Quaternion, [1610](#)
 - OpenTK::Quaterniond, [1625](#)
- IsAuxiliaryEffectSlot
 - OpenTK::Audio::OpenAL::EffectsExtension, [338](#)
- IsBuffer
 - OpenTK::Graphics::ES11::GL, [576](#)
 - OpenTK::Graphics::ES20::GL, [727](#)
 - OpenTK::Graphics::OpenGL::GL, [1216](#), [1217](#)
- IsCurrent
 - OpenTK::Graphics::GraphicsContext, [788](#)
 - OpenTK::Graphics::IGraphicsContext, [802](#)
- IsDisposed
 - OpenTK::Graphics::GraphicsContext, [788](#)
 - OpenTK::Graphics::IGraphicsContext, [802](#)
 - OpenTK::NativeWindow, [1596](#)
- IsEffect
 - OpenTK::Audio::OpenAL::EffectsExtension, [339](#)
- IsEnabled
 - OpenTK::Graphics::ES11::GL, [576](#)
 - OpenTK::Graphics::ES20::GL, [727](#)
 - OpenTK::Graphics::OpenGL::GL, [1217](#)
- IsExiting
 - OpenTK::GameWindow, [385](#)
- IsFilter
 - OpenTK::Audio::OpenAL::EffectsExtension, [339](#)
- IsIdle
 - OpenTK::GLControl, [393](#)
- IsIndexed
 - OpenTK::Graphics::ColorFormat, [427](#)
- IsInitialized
 - OpenTK::Audio::OpenAL::EffectsExtension, [340](#)
 - OpenTK::Audio::OpenAL::XRamExtension, [343](#)
- IsKeyDown
 - OpenTK::Input::KeyboardState, [1518](#)
- IsKeyUp
 - OpenTK::Input::KeyboardState, [1518](#)
- IsList
 - OpenTK::Graphics::OpenGL::GL, [1218](#)
- IsNaN
 - OpenTK::Half, [1491](#)
- IsNegativeInfinity
 - OpenTK::Half, [1491](#)
- IsPositiveInfinity
 - OpenTK::Half, [1491](#)
- IsPressed
 - OpenTK::Input::MouseButtonEventArgs, [1520](#)
- IsPrimary
 - OpenTK::DisplayDevice, [370](#)
- IsProcessing
 - OpenTK::Audio::AudioContext, [308](#)
- IsProgram
 - OpenTK::Graphics::ES20::GL, [728](#)
 - OpenTK::Graphics::OpenGL::GL, [1218](#)
- IsQuery
 - OpenTK::Graphics::OpenGL::GL, [1218](#), [1219](#)
- IsRunning
 - OpenTK::Audio::AudioCapture, [299](#)
- IsShader
 - OpenTK::Graphics::ES20::GL, [728](#)
 - OpenTK::Graphics::OpenGL::GL, [1218](#)
- IsSynchronized
 - OpenTK::Audio::AudioContext, [308](#)

- IsTexture
 - OpenTK::Graphics::ES11::GL, [576](#)
 - OpenTK::Graphics::ES20::GL, [728](#), [729](#)
 - OpenTK::Graphics::OpenGL::GL, [1219](#)
- IsZero
 - OpenTK::Half, [1491](#)
- Ivory
 - OpenTK::Graphics::Color4, [414](#)
- J
 - OpenTK::Input, [252](#)
- JoystickAxis
 - OpenTK::Input, [248](#)
- JoystickButton
 - OpenTK::Input, [248](#)
- JoystickMoveEventArgs
 - OpenTK::Input::JoystickMoveEventArgs, [1512](#)
- Joysticks
 - OpenTK::GameWindow, [385](#)
 - OpenTK::Input::IJoystickDriver, [1503](#)
- K
 - OpenTK::Input, [252](#)
- KEY
 - OpenTK::Platform::Windows, [276](#)
- Key
 - OpenTK::Input, [249](#)
 - OpenTK::Input::KeyboardKeyEventArgs, [1517](#)
- Keyboard
 - OpenTK::GameWindow, [385](#)
 - OpenTK::Input, [247](#)
 - OpenTK::Input::IKeyboardDriver, [1504](#)
- KeyboardKeyEventArgs
 - OpenTK::Input::KeyboardKeyEventArgs, [1517](#)
- KeyChar
 - OpenTK::KeyPressEventArgs, [1532](#)
- KeyDown
 - OpenTK::Input::KeyboardDevice, [1516](#)
- Keypad0
 - OpenTK::Input, [251](#)
- Keypad1
 - OpenTK::Input, [251](#)
- Keypad2
 - OpenTK::Input, [251](#)
- Keypad3
 - OpenTK::Input, [251](#)
- Keypad4
 - OpenTK::Input, [251](#)
- Keypad5
 - OpenTK::Input, [251](#)
- Keypad6
 - OpenTK::Input, [251](#)
- Keypad7
 - OpenTK::Input, [251](#)
- Keypad8
 - OpenTK::Input, [251](#)
- Keypad9
 - OpenTK::Input, [251](#)
- KeypadAdd
 - OpenTK::Input, [251](#)
- KeypadDecimal
 - OpenTK::Input, [251](#)
- KeypadDivide
 - OpenTK::Input, [251](#)
- KeypadEnter
 - OpenTK::Input, [251](#)
- KeypadMinus
 - OpenTK::Input, [251](#)
- KeypadMultiply
 - OpenTK::Input, [251](#)
- KeypadPlus
 - OpenTK::Input, [251](#)
- KeypadSubtract
 - OpenTK::Input, [251](#)
- KeyPress
 - OpenTK::INativeWindow, [1499](#)
 - OpenTK::NativeWindow, [1599](#)
- KeyPressEventArgs
 - OpenTK::KeyPressEventArgs, [1532](#)
- KeyRepeat
 - OpenTK::Input::KeyboardDevice, [1515](#)
- KeyUp

- OpenTK::Input::KeyboardDevice, [1516](#)
- Khaki
 - OpenTK::Graphics::Color4, [414](#)
- L
 - OpenTK::Input, [252](#)
- LAlt
 - OpenTK::Input, [249](#)
- LargeIcon
 - OpenTK::Platform::Windows, [279](#)
- LastButton
 - OpenTK::Input, [253](#)
- LastKey
 - OpenTK::Input, [253](#)
- Lavender
 - OpenTK::Graphics::Color4, [414](#)
- LavenderBlush
 - OpenTK::Graphics::Color4, [414](#)
- LawnGreen
 - OpenTK::Graphics::Color4, [415](#)
- LBracket
 - OpenTK::Input, [253](#)
- LControl
 - OpenTK::Input, [249](#)
- Left
 - OpenTK::Box2, [360](#)
 - OpenTK::Input, [250](#), [253](#)
- LemonChiffon
 - OpenTK::Graphics::Color4, [415](#)
- Length
 - OpenTK::Quaternion, [1617](#)
 - OpenTK::Quaterniond, [1632](#)
 - OpenTK::Vector2, [1658](#)
 - OpenTK::Vector2d, [1683](#)
 - OpenTK::Vector3, [1725](#)
 - OpenTK::Vector3d, [1758](#)
 - OpenTK::Vector4, [1795](#)
 - OpenTK::Vector4d, [1824](#)
- LengthFast
 - OpenTK::Vector2, [1658](#)
 - OpenTK::Vector3, [1725](#)
 - OpenTK::Vector3d, [1758](#)
 - OpenTK::Vector4, [1796](#)
 - OpenTK::Vector4d, [1824](#)
- LengthSquared
- OpenTK::Quaternion, [1617](#)
- OpenTK::Quaterniond, [1632](#)
- OpenTK::Vector2, [1659](#)
- OpenTK::Vector2d, [1683](#)
- OpenTK::Vector3, [1725](#)
- OpenTK::Vector3d, [1759](#)
- OpenTK::Vector4, [1796](#)
- OpenTK::Vector4d, [1824](#)
- Lerp
 - OpenTK::Vector2, [1647](#), [1648](#)
 - OpenTK::Vector2d, [1671](#)
 - OpenTK::Vector3, [1710](#)
 - OpenTK::Vector3d, [1742](#), [1743](#)
 - OpenTK::Vector4, [1782](#)
 - OpenTK::Vector4d, [1810](#)
- Light
 - OpenTK::Graphics::ES10::GL, [468](#), [469](#)
 - OpenTK::Graphics::ES11::GL, [577](#), [578](#)
 - OpenTK::Graphics::OpenGL::GL, [1220–1222](#)
- LightBlue
 - OpenTK::Graphics::Color4, [415](#)
- LightCoral
 - OpenTK::Graphics::Color4, [415](#)
- LightCyan
 - OpenTK::Graphics::Color4, [415](#)
- LightGoldenrodYellow
 - OpenTK::Graphics::Color4, [415](#)
- LightGray
 - OpenTK::Graphics::Color4, [415](#)
- LightGreen
 - OpenTK::Graphics::Color4, [415](#)
- LightModel
 - OpenTK::Graphics::ES10::GL, [469](#), [470](#)
 - OpenTK::Graphics::ES11::GL, [578](#), [579](#)
 - OpenTK::Graphics::OpenGL::GL, [1222–1224](#)
- LightPink
 - OpenTK::Graphics::Color4, [415](#)
- LightSalmon
 - OpenTK::Graphics::Color4, [416](#)
- LightSeaGreen

- OpenTK::Graphics::Color4, [416](#)
- LightSkyBlue
 - OpenTK::Graphics::Color4, [416](#)
- LightSlateGray
 - OpenTK::Graphics::Color4, [416](#)
- LightSteelBlue
 - OpenTK::Graphics::Color4, [416](#)
- LightYellow
 - OpenTK::Graphics::Color4, [416](#)
- Lime
 - OpenTK::Graphics::Color4, [416](#)
- LimeGreen
 - OpenTK::Graphics::Color4, [416](#)
- LinearDistance
 - OpenTK::Audio::OpenAL, [23](#)
- LinearDistanceClamped
 - OpenTK::Audio::OpenAL, [23](#)
- Linen
 - OpenTK::Graphics::Color4, [416](#)
- LineStipple
 - OpenTK::Graphics::OpenGL::GL, [1224](#), [1225](#)
- LineWidth
 - OpenTK::Graphics::ES10::GL, [470](#)
 - OpenTK::Graphics::ES11::GL, [579](#)
 - OpenTK::Graphics::ES20::GL, [729](#)
 - OpenTK::Graphics::OpenGL::GL, [1225](#)
- LinkOverlay
 - OpenTK::Platform::Windows, [278](#)
- LinkProgram
 - OpenTK::Graphics::ES20::GL, [729](#)
 - OpenTK::Graphics::OpenGL::GL, [1225](#), [1226](#)
- ListBase
 - OpenTK::Graphics::OpenGL::GL, [1226](#)
- Load
 - OpenTK::GameWindow, [388](#)
 - OpenTK::Platform::IGameWindow, [1602](#)
- LoadAll
 - OpenTK::Graphics::GraphicsContext, [786](#)
 - OpenTK::Graphics::IGraphicsContext, [801](#)
- OpenTK::Graphics::IGraphicsContextInternal, [804](#)
- OpenTK::Graphics::OpenGL::GL, [1226](#)
- LoadIdentity
 - OpenTK::Graphics::ES10::GL, [470](#)
 - OpenTK::Graphics::ES11::GL, [579](#)
 - OpenTK::Graphics::OpenGL::GL, [1226](#)
- LoadMatrix
 - OpenTK::Graphics::ES10::GL, [470](#), [471](#)
 - OpenTK::Graphics::ES11::GL, [579](#), [580](#)
 - OpenTK::Graphics::OpenGL::GL, [1226–1228](#)
- LoadName
 - OpenTK::Graphics::OpenGL::GL, [1228](#)
- LoadTransposeMatrix
 - OpenTK::Graphics::OpenGL::GL, [1228–1230](#)
- Location
 - OpenTK::INativeWindow, [1497](#)
 - OpenTK::NativeWindow, [1596](#)
- LogicOp
 - OpenTK::Graphics::ES10::GL, [471](#)
 - OpenTK::Graphics::ES11::GL, [580](#)
 - OpenTK::Graphics::OpenGL::GL, [1230](#)
- LookAt
 - OpenTK::Matrix4, [1549](#)
 - OpenTK::Matrix4d, [1574](#), [1575](#)
- Looping
 - OpenTK::Audio::OpenAL, [29](#)
- Lowpass
 - OpenTK::Audio::OpenAL, [41](#)
- LowpassGain
 - OpenTK::Audio::OpenAL, [41](#)
- LowpassGainHF
 - OpenTK::Audio::OpenAL, [41](#)
- LShift
 - OpenTK::Input, [249](#)
- LWin
 - OpenTK::Input, [249](#)

- M
 - OpenTK::Input, [252](#)
- M11
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- M12
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1583](#)
- M13
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1583](#)
- M14
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M21
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M22
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M23
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M24
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M31
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M32
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M33
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1583](#)
- M34
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1584](#)
- M41
 - OpenTK::Matrix4, [1557](#)
 - OpenTK::Matrix4d, [1584](#)
- M42
 - OpenTK::Matrix4, [1558](#)
 - OpenTK::Matrix4d, [1584](#)
- M43
 - OpenTK::Matrix4, [1558](#)
 - OpenTK::Matrix4d, [1584](#)
- M44
 - OpenTK::Matrix4, [1558](#)
 - OpenTK::Matrix4d, [1584](#)
- Magenta
 - OpenTK::Graphics::Color4, [416](#)
- Major
 - OpenTK::Graphics::GraphicsContextVersion, [791](#)
- MajorVersion
 - OpenTK::Audio::OpenAL, [21](#)
- MakeCurrent
 - OpenTK::Audio::AudioContext, [306](#)
 - OpenTK::GameWindow, [383](#)
 - OpenTK::GLControl, [391](#)
 - OpenTK::Graphics::GraphicsContext, [786](#)
 - OpenTK::Graphics::IGraphicsContext, [801](#)
 - OpenTK::Platform::IGameWindow, [1601](#)
- Map1
 - OpenTK::Graphics::OpenGL::GL, [1230–1233](#)
- Map2
 - OpenTK::Graphics::OpenGL::GL, [1234–1238](#)
- MapBuffer
 - OpenTK::Graphics::OpenGL::GL, [1239](#)
- MapGrid1
 - OpenTK::Graphics::OpenGL::GL, [1240](#)
- MapGrid2
 - OpenTK::Graphics::OpenGL::GL, [1241](#)
- MapVirtualKeyType
 - OpenTK::Platform::Windows, [275](#)
- Maroon
 - OpenTK::Graphics::Color4, [417](#)
- Material
 - OpenTK::Graphics::ES10::GL, [471](#), [472](#)
 - OpenTK::Graphics::ES11::GL, [580](#), [581](#)

- OpenTK::Graphics::OpenGL::GL, 1241–1243
- Matrix4
 - OpenTK::Matrix4, 1539
- Matrix4d
 - OpenTK::Matrix4d, 1564, 1565
- MatrixMode
 - OpenTK::Graphics::ES10::GL, 472
 - OpenTK::Graphics::ES11::GL, 581
 - OpenTK::Graphics::OpenGL::GL, 1243
- Max
 - OpenTK::Vector2, 1648
 - OpenTK::Vector2d, 1672
 - OpenTK::Vector3, 1710
 - OpenTK::Vector3d, 1743
 - OpenTK::Vector4, 1783
 - OpenTK::Vector4d, 1811
- MaxAuxiliarySends
 - OpenTK::Audio::AudioContext, 302
- MaxDistance
 - OpenTK::Audio::OpenAL, 29
- MaxGain
 - OpenTK::Audio::OpenAL, 30
- MaxValue
 - OpenTK::Half, 1490
- MediumAquamarine
 - OpenTK::Graphics::Color4, 417
- MediumBlue
 - OpenTK::Graphics::Color4, 417
- MediumOrchid
 - OpenTK::Graphics::Color4, 417
- MediumPurple
 - OpenTK::Graphics::Color4, 417
- MediumSeaGreen
 - OpenTK::Graphics::Color4, 417
- MediumSlateBlue
 - OpenTK::Graphics::Color4, 417
- MediumSpringGreen
 - OpenTK::Graphics::Color4, 417
- MediumTurquoise
 - OpenTK::Graphics::Color4, 417
- MediumVioletRed
 - OpenTK::Graphics::Color4, 418
- Menu
 - OpenTK::Input, 249
- Message
 - OpenTK::ContextExistsException, 362
- Middle
 - OpenTK::Input, 253
- MidnightBlue
 - OpenTK::Graphics::Color4, 418
- Min
 - OpenTK::Vector2, 1648
 - OpenTK::Vector2d, 1672, 1673
 - OpenTK::Vector3, 1711
 - OpenTK::Vector3d, 1743
 - OpenTK::Vector4, 1783, 1784
 - OpenTK::Vector4d, 1811, 1812
- MinGain
 - OpenTK::Audio::OpenAL, 30
- MINIMIZE
 - OpenTK::Platform::Windows, 279
- Minmax
 - OpenTK::Graphics::OpenGL::GL, 1244
- MinNormalizedValue
 - OpenTK::Half, 1490
- Minor
 - OpenTK::Graphics::GraphicsContextVersion, 791
- MinorVersion
 - OpenTK::Audio::OpenAL, 22
- MintCream
 - OpenTK::Graphics::Color4, 418
- Minus
 - OpenTK::Input, 253
- MinValue
 - OpenTK::Half, 1491
- MistyRose
 - OpenTK::Graphics::Color4, 418
- Moccasin
 - OpenTK::Graphics::Color4, 418
- Mono16
 - OpenTK::Audio::OpenAL, 24
- Mono8
 - OpenTK::Audio::OpenAL, 24
- MonoALawExt
 - OpenTK::Audio::OpenAL, 24
- MonoDoubleExt
 - OpenTK::Audio::OpenAL, 24

- MonoFloat32Ext
 - OpenTK::Audio::OpenAL, [24](#)
- MonoIma4Ext
 - OpenTK::Audio::OpenAL, [24](#)
- MonoMuLawExt
 - OpenTK::Audio::OpenAL, [24](#)
- MonoSources
 - OpenTK::Audio::OpenAL, [21](#)
- MOUSE
 - OpenTK::Platform::Windows, [276](#)
- Mouse
 - OpenTK::GameWindow, [385](#)
 - OpenTK::Input, [247](#)
 - OpenTK::Input::IMouseDriver, [1505](#)
- MOUSE_ATTRIBUTES_CHANGED
 - OpenTK::Platform::Windows, [277](#)
- MOUSE_MOVE_ABSOLUTE
 - OpenTK::Platform::Windows, [277](#)
- MOUSE_MOVE_RELATIVE
 - OpenTK::Platform::Windows, [277](#)
- MOUSE_VIRTUAL_DESKTOP
 - OpenTK::Platform::Windows, [277](#)
- MOUSEBUTTON
 - OpenTK::Platform::Windows, [276](#)
- MouseButton
 - OpenTK::Input, [253](#)
- MouseButtonEventArgs
 - OpenTK::Input::MouseButtonEventArgs, [1520](#)
- MouseEnter
 - OpenTK::INativeWindow, [1499](#)
 - OpenTK::NativeWindow, [1599](#)
- MouseEventArgs
 - OpenTK::Input::MouseEventArgs, [1526](#)
- MouseKeys
 - OpenTK::Platform::Windows, [275](#)
- MouseLeave
 - OpenTK::INativeWindow, [1499](#)
 - OpenTK::NativeWindow, [1599](#)
- MOUSEMOVE
 - OpenTK::Platform::Windows, [276](#)
- MouseMoveEventArgs
 - OpenTK::Input::MouseMoveEventArgs, [1528](#)
- MouseWheelEventArgs
 - OpenTK::Input::MouseWheelEventArgs, [1530](#), [1531](#)
- Move
 - OpenTK::INativeWindow, [1499](#)
 - OpenTK::Input::JoystickDevice, [1510](#)
 - OpenTK::Input::MouseDevice, [1524](#)
 - OpenTK::NativeWindow, [1599](#)
- Mp3Ext
 - OpenTK::Audio::OpenAL, [24](#)
- Mult
 - OpenTK::Matrix4, [1549](#), [1550](#)
 - OpenTK::Matrix4d, [1575](#), [1576](#)
 - OpenTK::Quaternion, [1611](#)
 - OpenTK::Quaterniond, [1625](#), [1626](#)
 - OpenTK::Vector2, [1649](#)
 - OpenTK::Vector2d, [1673](#)
 - OpenTK::Vector3, [1711](#)
 - OpenTK::Vector3d, [1744](#)
 - OpenTK::Vector4, [1784](#)
 - OpenTK::Vector4d, [1812](#)
- Multi51Chn16Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi51Chn32Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi51Chn8Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi61Chn16Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi61Chn32Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi61Chn8Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi71Chn16Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi71Chn32Ext
 - OpenTK::Audio::OpenAL, [25](#)
- Multi71Chn8Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiDrawArrays
 - OpenTK::Graphics::OpenGL::GL, [1244](#), [1245](#)
- MultiDrawElements
 - OpenTK::Graphics::OpenGL::GL, [1245](#), [1246](#)

- MultiDrawElements< T3 >
 - OpenTK::Graphics::OpenGL::GL, [1247–1253](#)
- Multiply
 - OpenTK::Quaternion, [1611](#), [1612](#)
 - OpenTK::Quaterniond, [1626](#), [1627](#)
 - OpenTK::Vector2, [1649](#), [1650](#)
 - OpenTK::Vector2d, [1674](#), [1675](#)
 - OpenTK::Vector3, [1712](#), [1713](#)
 - OpenTK::Vector3d, [1744](#), [1745](#)
 - OpenTK::Vector4, [1785](#), [1786](#)
 - OpenTK::Vector4d, [1813](#), [1814](#)
- MultiQuad16Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiQuad32Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiQuad8Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiRear16Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiRear32Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiRear8Ext
 - OpenTK::Audio::OpenAL, [25](#)
- MultiTexCoord1
 - OpenTK::Graphics::OpenGL::GL, [1254–1256](#)
- MultiTexCoord2
 - OpenTK::Graphics::OpenGL::GL, [1257–1262](#)
- MultiTexCoord3
 - OpenTK::Graphics::OpenGL::GL, [1263–1268](#)
- MultiTexCoord4
 - OpenTK::Graphics::ES10::GL, [472](#)
 - OpenTK::Graphics::ES11::GL, [581](#)
 - OpenTK::Graphics::OpenGL::GL, [1269–1274](#)
- MultiMatrix
 - OpenTK::Graphics::ES10::GL, [473](#)
 - OpenTK::Graphics::ES11::GL, [582](#)
 - OpenTK::Graphics::OpenGL::GL, [1275](#), [1276](#)
- MultiTransposeMatrix
 - OpenTK::Graphics::OpenGL::GL, [1276](#), [1277](#)
- N
 - OpenTK::Input, [252](#)
- NativeWindow
 - OpenTK::NativeWindow, [1589](#), [1590](#)
- NavajoWhite
 - OpenTK::Graphics::Color4, [418](#)
- Navy
 - OpenTK::Graphics::Color4, [418](#)
- NCXBUTTONDBLCLK
 - OpenTK::Platform::Windows, [280](#)
- NCXBUTTONDOWN
 - OpenTK::Platform::Windows, [280](#)
- NCXBUTTONUP
 - OpenTK::Platform::Windows, [280](#)
- NewList
 - OpenTK::Graphics::OpenGL::GL, [1278](#)
- NOACTIVATE
 - OpenTK::Platform::Windows, [278](#)
- NOCOPYBITS
 - OpenTK::Platform::Windows, [278](#)
- NoError
 - OpenTK::Audio::OpenAL, [21](#), [24](#)
- NOHOTKEYS
 - OpenTK::Platform::Windows, [277](#)
- NOLEGACY
 - OpenTK::Platform::Windows, [277](#)
- NOMOVE
 - OpenTK::Platform::Windows, [277](#)
- None
 - OpenTK::Audio::OpenAL, [23](#)
- NOOWNERZORDER
 - OpenTK::Platform::Windows, [278](#)
- NOREDRA
 - OpenTK::Platform::Windows, [278](#)
- NOREPOSITION
 - OpenTK::Platform::Windows, [278](#)
- Normal3
 - OpenTK::Graphics::ES10::GL, [473](#)
 - OpenTK::Graphics::ES11::GL, [582](#)
 - OpenTK::Graphics::OpenGL::GL, [1278–1284](#)
- Normalize
 - OpenTK::Quaternion, [1612](#), [1613](#)
 - OpenTK::Quaterniond, [1627](#), [1628](#)

- OpenTK::Vector2, [1651](#)
- OpenTK::Vector2d, [1675](#), [1676](#)
- OpenTK::Vector3, [1713](#)
- OpenTK::Vector3d, [1746](#)
- OpenTK::Vector4, [1786](#), [1787](#)
- OpenTK::Vector4d, [1814](#)
- NormalizeFast
 - OpenTK::Vector2, [1651](#), [1652](#)
 - OpenTK::Vector2d, [1676](#)
 - OpenTK::Vector3, [1714](#)
 - OpenTK::Vector3d, [1746](#), [1747](#)
 - OpenTK::Vector4, [1787](#)
 - OpenTK::Vector4d, [1815](#)
- NormalPointer
 - OpenTK::Graphics::ES10::GL, [474](#)
 - OpenTK::Graphics::ES11::GL, [583](#)
 - OpenTK::Graphics::OpenGL::GL, [1284](#)
- NormalPointer< T2 >
 - OpenTK::Graphics::ES10::GL, [474](#), [475](#)
 - OpenTK::Graphics::ES11::GL, [583](#), [584](#)
 - OpenTK::Graphics::OpenGL::GL, [1285](#), [1286](#)
- NOSENDCHANGING
 - OpenTK::Platform::Windows, [278](#)
- NOSIZE
 - OpenTK::Platform::Windows, [277](#)
- NOZORDER
 - OpenTK::Platform::Windows, [278](#)
- Null
 - OpenTK::Audio::OpenAL, [40](#), [41](#)
- Number0
 - OpenTK::Input, [252](#)
- Number1
 - OpenTK::Input, [252](#)
- Number2
 - OpenTK::Input, [252](#)
- Number3
 - OpenTK::Input, [252](#)
- Number4
 - OpenTK::Input, [252](#)
- Number5
 - OpenTK::Input, [252](#)
- Number6
 - OpenTK::Input, [252](#)
- Number7
 - OpenTK::Input, [252](#)
- Number8
 - OpenTK::Input, [252](#)
- Number9
 - OpenTK::Input, [253](#)
- NumberOfButtons
 - OpenTK::Input::MouseDevice, [1523](#)
- NumberOfFunctionKeys
 - OpenTK::Input::KeyboardDevice, [1515](#)
- NumberOfKeys
 - OpenTK::Input::KeyboardDevice, [1515](#)
- NumberOfLeds
 - OpenTK::Input::KeyboardDevice, [1515](#)
- NumberOfWheels
 - OpenTK::Input::MouseDevice, [1523](#)
- NumLock
 - OpenTK::Input, [251](#)
- O
 - OpenTK::Input, [252](#)
- OldLace
 - OpenTK::Graphics::Color4, [418](#)
- Olive
 - OpenTK::Graphics::Color4, [418](#)
- OliveDrab
 - OpenTK::Graphics::Color4, [419](#)
- OnClosed
 - OpenTK::NativeWindow, [1591](#)
- OnClosing
 - OpenTK::GameWindow, [383](#)
 - OpenTK::NativeWindow, [1591](#)
- OnDisposed
 - OpenTK::NativeWindow, [1591](#)
- One
 - OpenTK::Audio::AudioContext, [302](#)
 - OpenTK::Vector2, [1657](#)
 - OpenTK::Vector2d, [1683](#)
 - OpenTK::Vector3, [1723](#)
 - OpenTK::Vector3d, [1757](#)
 - OpenTK::Vector4, [1794](#)
 - OpenTK::Vector4d, [1822](#)

- OnFocusedChanged
 - OpenTK::NativeWindow, [1592](#)
- OnHandleCreated
 - OpenTK::GLControl, [391](#)
- OnHandleDestroyed
 - OpenTK::GLControl, [391](#)
- OnIconChanged
 - OpenTK::NativeWindow, [1592](#)
- OnKeyPress
 - OpenTK::NativeWindow, [1592](#)
- OnLoad
 - OpenTK::GameWindow, [383](#)
- OnMouseEnter
 - OpenTK::NativeWindow, [1592](#)
- OnMouseLeave
 - OpenTK::NativeWindow, [1593](#)
- OnMove
 - OpenTK::NativeWindow, [1593](#)
- OnPaint
 - OpenTK::GLControl, [392](#)
- OnParentChanged
 - OpenTK::GLControl, [392](#)
- OnRenderFrame
 - OpenTK::GameWindow, [383](#)
- OnResize
 - OpenTK::GLControl, [392](#)
 - OpenTK::NativeWindow, [1593](#)
- OnTitleChanged
 - OpenTK::NativeWindow, [1593](#)
- OnUnload
 - OpenTK::GameWindow, [383](#)
- OnUpdateFrame
 - OpenTK::GameWindow, [384](#)
- OnVisibleChanged
 - OpenTK::NativeWindow, [1593](#)
- OnWindowBorderChanged
 - OpenTK::NativeWindow, [1594](#)
- OnWindowInfoChanged
 - OpenTK::GameWindow, [384](#)
- OnWindowStateChanged
 - OpenTK::NativeWindow, [1594](#)
- OpenIcon
 - OpenTK::Platform::Windows, [279](#)
- OpenTK.Audio, [13](#)
- OpenTK.Audio.OpenAL, [14](#)
- OpenTK.Compute, [43](#)
- OpenTK.Graphics, [43](#)
- OpenTK.Graphics.ES10, [45](#)
- OpenTK.Graphics.ES11, [52](#)
- OpenTK.Graphics.ES20, [68](#)
- OpenTK.Graphics.OpenGL, [85](#)
- OpenTK.Input, [244](#)
- OpenTK.Platform, [254](#)
- OpenTK.Platform.Dummy, [254](#)
- OpenTK.Platform.Egl, [254](#)
- OpenTK.Platform.MacOS, [254](#)
- OpenTK.Platform.MacOS.Carbon, [255](#)
- OpenTK.Platform.Windows, [258](#)
- OpenTK.Platform.X11, [280](#)
- OpenTK.Properties, [294](#)
- OpenTK::Audio::AudioCapture, [295](#)
 - AudioCapture, [297](#)
 - AvailableDevices, [298](#)
 - AvailableSamples, [298](#)
 - CheckErrors, [297](#)
 - CurrentDevice, [299](#)
 - CurrentError, [299](#)
 - DefaultDevice, [299](#)
 - Dispose, [297](#)
 - IsRunning, [299](#)
 - ReadSamples, [297](#)
 - ReadSamples< TBuffer >, [298](#)
 - SampleFormat, [299](#)
 - SampleFrequency, [299](#)
 - Start, [298](#)
 - Stop, [298](#)
- OpenTK::Audio::AudioContext, [299](#)
 - AudioContext, [302–304](#)
 - AvailableDevices, [307](#)
 - CheckErrors, [305](#)
 - CurrentContext, [307](#)
 - CurrentDevice, [308](#)
 - CurrentError, [308](#)
 - DefaultDevice, [308](#)
 - Dispose, [305](#)
 - Equals, [305](#)
 - Four, [302](#)
 - GetHashCode, [305](#)
 - IsProcessing, [308](#)
 - IsSynchronized, [308](#)
 - MakeCurrent, [306](#)
 - MaxAuxiliarySends, [302](#)

- One, [302](#)
- Process, [306](#)
- SupportsExtension, [306](#)
- Suspend, [307](#)
- Three, [302](#)
- ToString, [307](#)
- Two, [302](#)
- UseDriverDefault, [302](#)
- OpenTK::Audio::AudioContextException, [308](#)
 - AudioContextException, [309](#)
- OpenTK::Audio::AudioDeviceException, [309](#)
 - AudioDeviceException, [310](#)
- OpenTK::Audio::AudioException, [310](#)
 - AudioException, [311](#)
- OpenTK::Audio::AudioValueException, [311](#)
 - AudioValueException, [312](#)
- OpenTK::Audio::OpenAL
 - ALBufferState, [20](#)
 - ALCapability, [20](#)
 - AlcContextAttributes, [20](#)
 - AlcError, [21](#)
 - AlcGetInteger, [21](#)
 - AlcGetString, [22](#)
 - AlcGetStringList, [23](#)
 - ALDistanceModel, [23](#)
 - ALError, [23](#)
 - ALFormat, [24](#)
 - ALGetBufferi, [25](#)
 - ALGetFloat, [26](#)
 - ALGetInteger, [26](#)
 - ALGetSourceci, [26](#)
 - ALGetString, [27](#)
 - AllAttributes, [22](#)
 - AllDevicesSpecifier, [22, 23](#)
 - ALLlistener3f, [27](#)
 - ALLlistenerf, [27](#)
 - ALLlistenerfv, [28](#)
 - ALSource3f, [28](#)
 - ALSource3i, [28](#)
 - ALSourceb, [28](#)
 - ALSourcef, [29](#)
 - ALSourceci, [30](#)
 - ALSourceState, [31](#)
 - ALSourceType, [31](#)
 - AttributesSize, [22](#)
 - Autowah, [40](#)
 - AutowahAttackTime, [36](#)
 - AutowahPeakGain, [36](#)
 - AutowahReleaseTime, [36](#)
 - AutowahResonance, [36](#)
 - BandpassGain, [41](#)
 - BandpassGainLF, [41](#)
 - Bits, [26](#)
 - Buffer, [30](#)
 - BuffersProcessed, [27](#)
 - BuffersQueued, [26](#)
 - ByteOffset, [26, 30](#)
 - CaptureDeviceSpecifier, [22, 23](#)
 - Channels, [26](#)
 - Chorus, [40](#)
 - ChorusDelay, [34](#)
 - ChorusDepth, [34](#)
 - ChorusFeedback, [34](#)
 - ChorusPhase, [38](#)
 - ChorusRate, [34](#)
 - ChorusWaveform, [38](#)
 - Compressor, [40](#)
 - CompressorOnoff, [39](#)
 - ConeInnerAngle, [30](#)
 - ConeOuterAngle, [30](#)
 - DefaultAllDevicesSpecifier, [22](#)
 - DefaultDeviceSpecifier, [22](#)
 - DistanceModel, [26](#)
 - Distortion, [40](#)
 - DistortionEdge, [34](#)
 - DistortionEQBandwidth, [34](#)
 - DistortionEQCenter, [34](#)
 - DistortionGain, [34](#)
 - DistortionLowpassCutoff, [34](#)
 - DopplerFactor, [26](#)
 - DopplerVelocity, [26](#)
 - EaxReverbAirAbsorptionGainHF, [38](#)
 - EaxReverbDecayHFRatio, [37](#)
 - EaxReverbDecayLFRatio, [37](#)
 - EaxReverbDecayTime, [37](#)
 - EaxReverbDensity, [36](#)
 - EaxReverbDiffusion, [36](#)
 - EaxReverbEchoDepth, [37](#)

- EaxReverbEchoTime, 37
- EaxReverbGain, 36
- EaxReverbGainHF, 37
- EaxReverbGainLF, 37
- EaxReverbHFReference, 38
- EaxReverbLateReverbDelay, 37
- EaxReverbLateReverbGain, 37
- EaxReverbLateReverbPan, 32
- EaxReverbLFReference, 38
- EaxReverbModulationDepth, 38
- EaxReverbModulationTime, 37
- EaxReverbReflectionsDelay, 37
- EaxReverbReflectionsGain, 37
- EaxReverbReflectionsPan, 32
- Echo, 40
- EchoDamping, 35
- EchoDelay, 34
- EchoFeedback, 35
- EchoLRDelay, 34
- EchoSpread, 35
- EffectslotAuxiliarySendAuto, 32
- EffectslotEffect, 32
- EffectslotGain, 31
- EffectType, 40
- EfxAirAbsorptionFactor, 30
- EfxAuxiliaryf, 31
- EfxAuxiliaryi, 31
- EfxAuxiliarySendFilter, 28
- EfxAuxiliarySendFilterGainAuto, 29
- EfxDirectFilterGainHighFrequencyAuto, 29
- EfxEffect3f, 32
- EfxEffectf, 32
- EfxEffecti, 38
- EfxEffectType, 40
- EfxFilterf, 41
- EfxFilteri, 41
- EfxFilterType, 41
- EfxFormantFilterSettings, 41
- EfxMajorVersion, 22
- EfxMaxAuxiliarySends, 21, 22
- EfxMinorVersion, 22
- EfxRoomRolloffFactor, 30
- Equalizer, 41
- EqualizerHighCutoff, 36
- EqualizerHighGain, 36
- EqualizerLowCutoff, 36
- EqualizerLowGain, 36
- EqualizerMid1Center, 36
- EqualizerMid1Gain, 36
- EqualizerMid1Width, 36
- EqualizerMid2Center, 36
- EqualizerMid2Gain, 36
- EqualizerMid2Width, 36
- ExponentDistance, 23
- Extensions, 22, 27
- FilterType, 41
- Flanger, 40
- FlangerDelay, 35
- FlangerDepth, 35
- FlangerFeedback, 35
- FlangerPhase, 38
- FlangerRate, 35
- FlangerWaveform, 38
- Frequency, 21, 26
- FrequencyShifter, 40
- FrequencyShifterFrequency, 35
- FrequencyShifterLeftDirection, 38
- FrequencyShifterRightDirection, 39
- Gain, 27, 30
- Highpass, 41
- HighpassGain, 41
- HighpassGainLF, 41
- IllegalCommand, 24
- IllegalEnum, 24
- Initial, 31
- Invalid, 20
- InvalidContext, 21
- InvalidDevice, 21
- InvalidEnum, 21, 24
- InvalidName, 24
- InvalidOperation, 24
- InvalidValue, 21, 24
- InverseDistance, 23
- InverseDistanceClamped, 23
- LinearDistance, 23
- LinearDistanceClamped, 23
- Looping, 29
- Lowpass, 41
- LowpassGain, 41
- LowpassGainHF, 41

- MajorVersion, [21](#)
- MaxDistance, [29](#)
- MaxGain, [30](#)
- MinGain, [30](#)
- MinorVersion, [22](#)
- Mono16, [24](#)
- Mono8, [24](#)
- MonoALawExt, [24](#)
- MonoDoubleExt, [24](#)
- MonoFloat32Ext, [24](#)
- MonoIma4Ext, [24](#)
- MonoMuLawExt, [24](#)
- MonoSources, [21](#)
- Mp3Ext, [24](#)
- Multi51Chn16Ext, [25](#)
- Multi51Chn32Ext, [25](#)
- Multi51Chn8Ext, [25](#)
- Multi61Chn16Ext, [25](#)
- Multi61Chn32Ext, [25](#)
- Multi61Chn8Ext, [25](#)
- Multi71Chn16Ext, [25](#)
- Multi71Chn32Ext, [25](#)
- Multi71Chn8Ext, [25](#)
- MultiQuad16Ext, [25](#)
- MultiQuad32Ext, [25](#)
- MultiQuad8Ext, [25](#)
- MultiRear16Ext, [25](#)
- MultiRear32Ext, [25](#)
- MultiRear8Ext, [25](#)
- NoError, [21](#), [24](#)
- None, [23](#)
- Null, [40](#), [41](#)
- Orientation, [28](#)
- Paused, [31](#)
- Pending, [20](#)
- Pitch, [29](#)
- PitchShifter, [40](#)
- PitchShifterCoarseTune, [39](#)
- PitchShifterFineTune, [39](#)
- Playing, [31](#)
- Position, [27](#), [28](#)
- ReferenceDistance, [29](#)
- Refresh, [21](#)
- Renderer, [27](#)
- Reverb, [40](#)
- ReverbAirAbsorptionGainHF, [33](#)
- ReverbDecayHFLimit, [39](#)
- ReverbDecayHFRatio, [33](#)
- ReverbDecayTime, [33](#)
- ReverbDensity, [32](#)
- ReverbDiffusion, [32](#)
- ReverbGain, [33](#)
- ReverbGainHF, [33](#)
- ReverbLateReverbDelay, [33](#)
- ReverbLateReverbGain, [33](#)
- ReverbReflectionsDelay, [33](#)
- ReverbReflectionsGain, [33](#)
- ReverbRoomRolloffFactor, [34](#)
- RingModulator, [40](#)
- RingModulatorFrequency, [35](#)
- RingModulatorHighpassCutoff, [35](#)
- RingModulatorWaveform, [39](#)
- RolloffFactor, [29](#)
- SampleOffset, [26](#), [30](#)
- SecOffset, [30](#)
- Size, [26](#)
- SourceRelative, [29](#)
- SourceState, [26](#)
- SourceType, [30](#)
- Static, [31](#)
- Stereo8, [24](#)
- StereoALawExt, [24](#)
- StereoDoubleExt, [25](#)
- StereoFloat32Ext, [24](#)
- StereoIma4Ext, [24](#)
- StereoMuLawExt, [24](#)
- Streaming, [31](#)
- Sync, [21](#)
- Unused, [20](#)
- Velocity, [28](#)
- Vendor, [27](#)
- Version, [27](#)
- VocalMorpher, [40](#)
- VocalMorpherPhonemeA, [39](#), [42](#)
- VocalMorpherPhonemeAA, [42](#)
- VocalMorpherPhonemeACoarse-Tuning, [39](#)
- VocalMorpherPhonemeAE, [42](#)
- VocalMorpherPhonemeAH, [42](#)
- VocalMorpherPhonemeAO, [42](#)
- VocalMorpherPhonemeB, [39](#), [42](#)

- VocalMorpherPhonemeBCoarse-Tuning, 39
- VocalMorpherPhonemeD, 42
- VocalMorpherPhonemeE, 42
- VocalMorpherPhonemeEH, 42
- VocalMorpherPhonemeER, 42
- VocalMorpherPhonemeF, 42
- VocalMorpherPhonemeG, 42
- VocalMorpherPhonemeI, 42
- VocalMorpherPhonemeIH, 42
- VocalMorpherPhonemeIY, 42
- VocalMorpherPhonemeJ, 42
- VocalMorpherPhonemeK, 42
- VocalMorpherPhonemeL, 42
- VocalMorpherPhonemeM, 42
- VocalMorpherPhonemeN, 42
- VocalMorpherPhonemeO, 42
- VocalMorpherPhonemeP, 42
- VocalMorpherPhonemeR, 42
- VocalMorpherPhonemeS, 42
- VocalMorpherPhonemeT, 42
- VocalMorpherPhonemeU, 42
- VocalMorpherPhonemeUH, 42
- VocalMorpherPhonemeUW, 42
- VocalMorpherPhonemeV, 42
- VocalMorpherPhonemeZ, 42
- VocalMorpherRate, 35
- VocalMorpherWaveform, 39
- VorbisExt, 24
- OpenTK::Audio::OpenAL::EffectsExtension, 312
 - AuxiliaryEffectSlot, 319, 320
 - BindEffect, 320
 - BindEffectToAuxiliarySlot, 321
 - BindFilterToSource, 321
 - BindSourceToAuxiliarySlot, 322
 - DeleteAuxiliaryEffectSlot, 322, 323
 - DeleteAuxiliaryEffectSlots, 323, 324
 - DeleteEffect, 324
 - DeleteEffects, 324, 325
 - DeleteFilter, 325, 326
 - DeleteFilters, 326, 327
 - Effect, 327, 328
 - EffectsExtension, 319
 - Filter, 328, 329
 - GenAuxiliaryEffectSlot, 329, 330
 - GenAuxiliaryEffectSlots, 330, 331
 - GenEffect, 331, 332
 - GenEffects, 332, 333
 - GenFilter, 333
 - GenFilters, 333, 334
 - GetAuxiliaryEffectSlot, 334, 335
 - GetEffect, 335–337
 - GetFilter, 337, 338
 - IsAuxiliaryEffectSlot, 338
 - IsEffect, 339
 - IsFilter, 339
 - IsInitialized, 340
- OpenTK::Audio::OpenAL::XRamExtension, 340
 - Accessible, 341
 - Automatic, 341
 - GetBufferMode, 342
 - GetRamFree, 343
 - GetRamSize, 343
 - Hardware, 341
 - IsInitialized, 343
 - SetBufferMode, 342, 343
 - XRamExtension, 342
 - XRamStorage, 341
- OpenTK::AutoGeneratedAttribute, 344
 - AutoGeneratedAttribute, 344
 - Category, 344
 - EntryPoint, 344
 - Version, 345
- OpenTK::BezierCurve, 345
 - BezierCurve, 346, 347
 - CalculateLength, 347, 348
 - CalculatePoint, 348, 349
 - Parallel, 349
 - Points, 349
- OpenTK::BezierCurveCubic, 350
 - BezierCurveCubic, 351
 - CalculateLength, 351
 - CalculatePoint, 351
 - EndAnchor, 352
 - FirstControlPoint, 352
 - Parallel, 352
 - SecondControlPoint, 352
 - StartAnchor, 352
- OpenTK::BezierCurveQuadric, 352

- BezierCurveQuadric, [353](#), [354](#)
- CalculateLength, [354](#)
- CalculatePoint, [354](#)
- ControlPoint, [355](#)
- EndAnchor, [355](#)
- Parallel, [355](#)
- StartAnchor, [355](#)
- OpenTK::BindingsBase, [355](#)
 - BindingsBase, [356](#)
 - CoreClass, [357](#)
 - CoreFunctionMap, [357](#)
 - DelegatesClass, [357](#)
 - GetAddress, [357](#)
 - RebuildExtensionList, [357](#)
 - SyncRoot, [357](#)
- OpenTK::Box2, [358](#)
 - Bottom, [360](#)
 - Box2, [359](#)
 - FromTLRB, [360](#)
 - Height, [361](#)
 - Left, [360](#)
 - Right, [360](#)
 - Top, [360](#)
 - ToString, [360](#)
 - Width, [361](#)
- OpenTK::ContextExistsException, [361](#)
 - ContextExistsException, [361](#)
 - Message, [362](#)
- OpenTK::ContextHandle, [362](#)
 - CompareTo, [364](#)
 - ContextHandle, [363](#)
 - Equals, [364](#)
 - GetHashCode, [364](#)
 - Handle, [366](#)
 - operator ContextHandle, [364](#)
 - operator IntPtr, [365](#)
 - operator==, [365](#)
 - ToString, [366](#)
 - Zero, [366](#)
- OpenTK::DisplayDevice, [366](#)
 - AvailableDisplays, [370](#)
 - AvailableResolutions, [370](#)
 - BitsPerPixel, [370](#)
 - Bounds, [370](#)
 - ChangeResolution, [368](#)
 - Default, [370](#)
 - Height, [370](#)
 - IsPrimary, [370](#)
 - RefreshRate, [370](#)
 - RestoreResolution, [368](#)
 - SelectResolution, [369](#)
 - ToString, [369](#)
 - Width, [370](#)
- OpenTK::DisplayResolution, [371](#)
 - BitsPerPixel, [373](#)
 - Bounds, [373](#)
 - Equals, [372](#)
 - GetHashCode, [372](#)
 - Height, [373](#)
 - operator==, [373](#)
 - RefreshRate, [373](#)
 - ToString, [373](#)
 - Width, [374](#)
- OpenTK::FrameEventArgs, [374](#)
 - FrameEventArgs, [375](#)
 - Time, [375](#)
- OpenTK::GameWindow, [375](#)
 - Context, [385](#)
 - Dispose, [382](#)
 - Exit, [382](#)
 - GameWindow, [380–382](#)
 - IsExiting, [385](#)
 - Joysticks, [385](#)
 - Keyboard, [385](#)
 - Load, [388](#)
 - MakeCurrent, [383](#)
 - Mouse, [385](#)
 - OnClosing, [383](#)
 - OnLoad, [383](#)
 - OnRenderFrame, [383](#)
 - OnUnload, [383](#)
 - OnUpdateFrame, [384](#)
 - OnWindowInfoChanged, [384](#)
 - RenderFrame, [388](#)
 - RenderFrequency, [386](#)
 - RenderPeriod, [386](#)
 - RenderTime, [386](#)
 - Run, [384](#), [385](#)
 - SwapBuffers, [385](#)
 - TargetRenderFrequency, [386](#)
 - TargetRenderPeriod, [386](#)
 - TargetUpdateFrequency, [386](#)

- TargetUpdatePeriod, 387
- Unload, 388
- UpdateFrame, 388
- UpdateFrequency, 387
- UpdatePeriod, 387
- UpdateTime, 387
- VSync, 387
- WindowState, 387
- OpenTK::GLControl, 388
 - AspectRatio, 392
 - Context, 392
 - Dispose, 391
 - GLControl, 390
 - GrabScreenshot, 391
 - GraphicsMode, 393
 - IsIdle, 393
 - MakeCurrent, 391
 - OnHandleCreated, 391
 - OnHandleDestroyed, 391
 - OnPaint, 392
 - OnParentChanged, 392
 - OnResize, 392
 - SwapBuffers, 392
 - VSync, 393
 - WindowInfo, 393
- OpenTK::Graphics
 - Debug, 44
 - Default, 44
 - Embedded, 44
 - ForwardCompatible, 44
 - GraphicsContextFlags, 44
- OpenTK::Graphics::Color4, 393
 - A, 408
 - AliceBlue, 408
 - AntiqueWhite, 408
 - Aqua, 409
 - Aquamarine, 409
 - Azure, 409
 - B, 408
 - Beige, 409
 - Bisque, 409
 - Black, 409
 - BlanchedAlmond, 409
 - Blue, 409
 - BlueViolet, 409
 - Brown, 409
 - BurlyWood, 410
 - CadetBlue, 410
 - Chartreuse, 410
 - Chocolate, 410
 - Color4, 405
 - Coral, 410
 - CornflowerBlue, 410
 - Cornsilk, 410
 - Crimson, 410
 - Cyan, 410
 - DarkBlue, 410
 - DarkCyan, 411
 - DarkGoldenrod, 411
 - DarkGray, 411
 - DarkGreen, 411
 - DarkKhaki, 411
 - DarkMagenta, 411
 - DarkOliveGreen, 411
 - DarkOrange, 411
 - DarkOrchid, 411
 - DarkRed, 412
 - DarkSalmon, 412
 - DarkSeaGreen, 412
 - DarkSlateBlue, 412
 - DarkSlateGray, 412
 - DarkTurquoise, 412
 - DarkViolet, 412
 - DeepPink, 412
 - DeepSkyBlue, 412
 - DimGray, 412
 - DodgerBlue, 413
 - Equals, 406
 - Firebrick, 413
 - FloralWhite, 413
 - ForestGreen, 413
 - Fuchsia, 413
 - G, 408
 - Gainsboro, 413
 - GetHashCode, 406
 - GhostWhite, 413
 - Gold, 413
 - Goldenrod, 413
 - Gray, 413
 - Green, 414
 - GreenYellow, 414
 - Honeydew, 414

- HotPink, [414](#)
- IndianRed, [414](#)
- Indigo, [414](#)
- Ivory, [414](#)
- Khaki, [414](#)
- Lavender, [414](#)
- LavenderBlush, [414](#)
- LawnGreen, [415](#)
- LemonChiffon, [415](#)
- LightBlue, [415](#)
- LightCoral, [415](#)
- LightCyan, [415](#)
- LightGoldenrodYellow, [415](#)
- LightGray, [415](#)
- LightGreen, [415](#)
- LightPink, [415](#)
- LightSalmon, [416](#)
- LightSeaGreen, [416](#)
- LightSkyBlue, [416](#)
- LightSlateGray, [416](#)
- LightSteelBlue, [416](#)
- LightYellow, [416](#)
- Lime, [416](#)
- LimeGreen, [416](#)
- Linen, [416](#)
- Magenta, [416](#)
- Maroon, [417](#)
- MediumAquamarine, [417](#)
- MediumBlue, [417](#)
- MediumOrchid, [417](#)
- MediumPurple, [417](#)
- MediumSeaGreen, [417](#)
- MediumSlateBlue, [417](#)
- MediumSpringGreen, [417](#)
- MediumTurquoise, [417](#)
- MediumVioletRed, [418](#)
- MidnightBlue, [418](#)
- MintCream, [418](#)
- MistyRose, [418](#)
- Moccasin, [418](#)
- NavajoWhite, [418](#)
- Navy, [418](#)
- OldLace, [418](#)
- Olive, [418](#)
- OliveDrab, [419](#)
- operator Color4, [406](#)
- operator System.Drawing.Color, [406](#)
- operator==, [407](#)
- Orange, [419](#)
- OrangeRed, [419](#)
- Orchid, [419](#)
- PaleGoldenrod, [419](#)
- PaleGreen, [419](#)
- PaleTurquoise, [419](#)
- PaleVioletRed, [419](#)
- PapayaWhip, [419](#)
- PeachPuff, [420](#)
- Peru, [420](#)
- Pink, [420](#)
- Plum, [420](#)
- PowderBlue, [420](#)
- Purple, [420](#)
- R, [408](#)
- Red, [420](#)
- RosyBrown, [420](#)
- RoyalBlue, [420](#)
- SaddleBrown, [420](#)
- Salmon, [421](#)
- SandyBrown, [421](#)
- SeaGreen, [421](#)
- SeaShell, [421](#)
- Sienna, [421](#)
- Silver, [421](#)
- SkyBlue, [421](#)
- SlateBlue, [421](#)
- SlateGray, [421](#)
- Snow, [421](#)
- SpringGreen, [422](#)
- SteelBlue, [422](#)
- Tan, [422](#)
- Teal, [422](#)
- Thistle, [422](#)
- ToArgb, [407](#)
- Tomato, [422](#)
- ToString, [408](#)
- Transparent, [422](#)
- Turquoise, [422](#)
- Violet, [422](#)
- Wheat, [422](#)
- White, [423](#)
- WhiteSmoke, [423](#)
- Yellow, [423](#)

- YellowGreen, [423](#)
- OpenTK::Graphics::ColorFormat, [423](#)
 - Alpha, [427](#)
 - BitsPerPixel, [427](#)
 - Blue, [427](#)
 - ColorFormat, [425](#)
 - Equals, [425](#)
 - GetHashCode, [426](#)
 - Green, [427](#)
 - IsIndexed, [427](#)
 - operator ColorFormat, [426](#)
 - operator==, [426](#)
 - Red, [427](#)
 - ToString, [427](#)
- OpenTK::Graphics::ES10::GL, [428](#)
 - ActiveTexture, [442](#)
 - AlphaFunc, [442](#)
 - BindTexture, [443](#)
 - BlendFunc, [443](#)
 - Clear, [444](#)
 - ClearColor, [444](#)
 - ClearDepth, [444](#)
 - ClearStencil, [445](#)
 - ClientActiveTexture, [445](#)
 - Color4, [445](#)
 - ColorMask, [445](#)
 - ColorPointer, [446](#)
 - ColorPointer< T3 >, [446–448](#)
 - CompressedTexImage2D, [448](#)
 - CompressedTexImage2D< T7 >, [449–452](#)
 - CompressedTexSubImage2D, [453](#)
 - CompressedTexSubImage2D< T8 >, [453–455](#)
 - CopyTexImage2D, [456](#)
 - CopyTexSubImage2D, [457](#)
 - CullFace, [458](#)
 - DeleteTextures, [458, 459](#)
 - DepthFunc, [459](#)
 - DepthMask, [460](#)
 - DepthRange, [460](#)
 - DrawArrays, [460](#)
 - DrawElements, [461](#)
 - DrawElements< T3 >, [461–463](#)
 - Enable, [463](#)
 - EnableClientState, [463](#)
 - Finish, [464](#)
 - Flush, [464](#)
 - Fog, [464, 465](#)
 - FrontFace, [465](#)
 - Frustum, [465](#)
 - GenTextures, [465–467](#)
 - GetError, [467](#)
 - GetString, [467](#)
 - Hint, [467](#)
 - Light, [468, 469](#)
 - LightModel, [469, 470](#)
 - LineWidth, [470](#)
 - LoadIdentity, [470](#)
 - LoadMatrix, [470, 471](#)
 - LogicOp, [471](#)
 - Material, [471, 472](#)
 - MatrixMode, [472](#)
 - MultiTexCoord4, [472](#)
 - MultMatrix, [473](#)
 - Normal3, [473](#)
 - NormalPointer, [474](#)
 - NormalPointer< T2 >, [474, 475](#)
 - Ortho, [476](#)
 - PixelStore, [476](#)
 - PointSize, [477](#)
 - PolygonOffset, [477](#)
 - PushMatrix, [477](#)
 - ReadPixels, [477](#)
 - ReadPixels< T6 >, [478–480](#)
 - Rotate, [481](#)
 - SampleCoverage, [481](#)
 - Scale, [482](#)
 - Scissor, [482](#)
 - ShadeModel, [482](#)
 - StencilFunc, [482, 483](#)
 - StencilMask, [483](#)
 - StencilOp, [484](#)
 - SyncRoot, [505](#)
 - TexCoordPointer, [484](#)
 - TexCoordPointer< T3 >, [485, 486](#)
 - TexEnv, [487, 488](#)
 - TexImage2D, [489](#)
 - TexImage2D< T8 >, [490, 492, 493, 495](#)
 - TexParameter, [496](#)
 - TexSubImage2D, [497](#)

- TexSubImage2D< T8 >, [498–500](#)
- Translate, [501](#)
- VertexPointer, [502](#)
- VertexPointer< T3 >, [502–504](#)
- Viewport, [504](#)
- OpenTK::Graphics::ES11::GL, [505](#)
 - ActiveTexture, [527](#)
 - AlphaFunc, [528](#)
 - BindBuffer, [528](#)
 - BindTexture, [529](#)
 - BlendFunc, [529](#)
 - BufferData, [530](#)
 - BufferData< T2 >, [530–532](#)
 - BufferSubData, [532](#)
 - BufferSubData< T3 >, [533, 534](#)
 - Clear, [535](#)
 - ClearColor, [535](#)
 - ClearDepth, [535](#)
 - ClearStencil, [536](#)
 - ClientActiveTexture, [536](#)
 - ClipPlane, [536, 537](#)
 - Color4, [537](#)
 - ColorMask, [538](#)
 - ColorPointer, [538](#)
 - ColorPointer< T3 >, [538–540](#)
 - CompressedTexImage2D, [540](#)
 - CompressedTexImage2D< T7 >, [541–544](#)
 - CompressedTexSubImage2D, [545](#)
 - CompressedTexSubImage2D< T8 >, [546–548](#)
 - CopyTexImage2D, [548](#)
 - CopyTexSubImage2D, [549](#)
 - CullFace, [550](#)
 - DeleteBuffers, [550, 551](#)
 - DeleteTextures, [552, 553](#)
 - DepthFunc, [553](#)
 - DepthMask, [553](#)
 - DepthRange, [554](#)
 - DrawArrays, [554](#)
 - DrawElements, [554](#)
 - DrawElements< T3 >, [555, 556](#)
 - Enable, [557](#)
 - EnableClientState, [557](#)
 - Finish, [557](#)
 - Flush, [558](#)
 - Fog, [558](#)
 - FrontFace, [559](#)
 - Frustum, [559](#)
 - GenBuffers, [559, 560](#)
 - GenTextures, [560–562](#)
 - GetBufferParameter, [562, 563](#)
 - GetClipPlane, [563, 564](#)
 - GetError, [564](#)
 - GetLight, [564, 565](#)
 - GetMaterial, [566](#)
 - GetPointer, [567](#)
 - GetPointer< T1 >, [567, 568](#)
 - GetString, [569](#)
 - GetTexEnv, [569–572](#)
 - GetTexParameter, [572–575](#)
 - Hint, [575](#)
 - IsBuffer, [576](#)
 - IsEnabled, [576](#)
 - IsTexture, [576](#)
 - Light, [577, 578](#)
 - LightModel, [578, 579](#)
 - LineWidth, [579](#)
 - LoadIdentity, [579](#)
 - LoadMatrix, [579, 580](#)
 - LogicOp, [580](#)
 - Material, [580, 581](#)
 - MatrixMode, [581](#)
 - MultiTexCoord4, [581](#)
 - MultMatrix, [582](#)
 - Normal3, [582](#)
 - NormalPointer, [583](#)
 - NormalPointer< T2 >, [583, 584](#)
 - Ortho, [585](#)
 - PixelStore, [585](#)
 - PointParameter, [586](#)
 - PointSize, [587](#)
 - PolygonOffset, [587](#)
 - PushMatrix, [587](#)
 - ReadPixels, [587](#)
 - ReadPixels< T6 >, [588–590](#)
 - Rotate, [591](#)
 - SampleCoverage, [591](#)
 - Scale, [592](#)
 - Scissor, [592](#)
 - ShadeModel, [592](#)
 - StencilFunc, [592, 593](#)

- StencilMask, [593](#)
- StencilOp, [594](#)
- SyncRoot, [619](#)
- TexCoordPointer, [594](#)
- TexCoordPointer< T3 >, [594–596](#)
- TexEnv, [596–600](#)
- TexImage2D, [600](#)
- TexImage2D< T8 >, [602](#), [603](#), [605](#), [606](#)
- TexParameter, [608–610](#)
- TexSubImage2D, [611](#)
- TexSubImage2D< T8 >, [612–615](#)
- Translate, [616](#)
- VertexPointer, [616](#)
- VertexPointer< T3 >, [616–618](#)
- Viewport, [618](#)
- OpenTK::Graphics::ES20::GL, [619](#)
 - ActiveTexture, [656](#)
 - AttachShader, [657](#)
 - BindAttribLocation, [657](#)
 - BindBuffer, [658](#)
 - BindTexture, [658](#), [659](#)
 - BlendColor, [659](#)
 - BlendEquation, [659](#)
 - BlendEquationSeparate, [660](#)
 - BlendFunc, [660](#)
 - BlendFuncSeparate, [661](#)
 - BufferData, [661](#)
 - BufferData< T2 >, [662–664](#)
 - BufferSubData, [664](#)
 - BufferSubData< T3 >, [664–666](#)
 - Clear, [666](#)
 - ClearColor, [667](#)
 - ClearDepth, [667](#)
 - ClearStencil, [667](#)
 - ColorMask, [667](#)
 - CompileShader, [668](#)
 - CompressedTexImage2D, [668](#)
 - CompressedTexImage2D< T7 >, [669–672](#)
 - CompressedTexSubImage2D, [673](#)
 - CompressedTexSubImage2D< T8 >, [673–675](#)
 - CopyTexImage2D, [676](#)
 - CopyTexSubImage2D, [677](#)
 - CreateProgram, [678](#)
 - CreateShader, [678](#)
 - CullFace, [678](#)
 - DeleteBuffers, [678](#), [679](#)
 - DeleteProgram, [680](#)
 - DeleteShader, [680](#)
 - DeleteTextures, [681](#), [682](#)
 - DepthFunc, [682](#)
 - DepthMask, [682](#)
 - DepthRange, [682](#)
 - DetachShader, [683](#)
 - DrawArrays, [683](#)
 - DrawElements, [684](#)
 - DrawElements< T3 >, [684](#), [685](#)
 - Enable, [686](#)
 - EnableVertexAttribArray, [686](#)
 - Finish, [687](#)
 - Flush, [687](#)
 - FrontFace, [687](#)
 - GenBuffers, [687](#), [688](#)
 - GenTextures, [688–690](#)
 - GetActiveAttrib, [690–692](#)
 - GetActiveUniform, [693–695](#)
 - GetAttachedShaders, [696–698](#)
 - GetAttribLocation, [698](#)
 - GetBufferParameter, [698](#), [699](#)
 - GetError, [700](#)
 - GetProgram, [700–702](#)
 - GetProgramInfoLog, [702–704](#)
 - GetShader, [704–706](#)
 - GetShaderInfoLog, [706–708](#)
 - GetShaderSource, [708–710](#)
 - GetString, [710](#)
 - GetTexParameter, [711–713](#)
 - GetUniform, [714–717](#)
 - GetUniformLocation, [717](#), [718](#)
 - GetVertexAttrib, [718–722](#)
 - GetVertexAttribPointer, [723](#)
 - GetVertexAttribPointer< T2 >, [723–726](#)
 - Hint, [727](#)
 - IsBuffer, [727](#)
 - IsEnabled, [727](#)
 - IsProgram, [728](#)
 - IsShader, [728](#)
 - IsTexture, [728](#), [729](#)
 - LineWidth, [729](#)

- LinkProgram, [729](#)
- PixelStore, [729](#)
- PolygonOffset, [730](#)
- ReadPixels, [730](#)
- ReadPixels< T6 >, [731–733](#)
- SampleCoverage, [734](#)
- Scissor, [734](#)
- ShaderSource, [735](#), [736](#)
- StencilFunc, [737](#)
- StencilFuncSeparate, [738](#)
- StencilMask, [739](#)
- StencilMaskSeparate, [739](#)
- StencilOp, [740](#)
- StencilOpSeparate, [740](#)
- SyncRoot, [780](#)
- TexImage2D, [741](#)
- TexImage2D< T8 >, [742](#), [744](#), [746](#), [747](#)
- TexParameter, [749–751](#)
- TexSubImage2D, [752](#)
- TexSubImage2D< T8 >, [753–755](#)
- Uniform1, [756–758](#)
- Uniform2, [758–760](#)
- Uniform3, [760–762](#)
- Uniform4, [762–764](#)
- UseProgram, [764](#)
- ValidateProgram, [765](#)
- VertexAttrib1, [765](#), [766](#)
- VertexAttrib2, [767](#), [768](#)
- VertexAttrib3, [769](#), [770](#)
- VertexAttrib4, [771](#), [772](#)
- VertexAttribPointer, [773](#)
- VertexAttribPointer< T5 >, [774–779](#)
- Viewport, [780](#)
- OpenTK::Graphics::GraphicsBindingsBase, [780](#)
 - GetAddress, [781](#)
- OpenTK::Graphics::GraphicsContext, [781](#)
 - Assert, [785](#)
 - CreateDummyContext, [785](#), [786](#)
 - CurrentContext, [787](#)
 - DirectRendering, [787](#)
 - Dispose, [786](#)
 - ErrorChecking, [788](#)
 - GraphicsContext, [784](#), [785](#)
 - GraphicsMode, [788](#)
 - IsCurrent, [788](#)
 - IsDisposed, [788](#)
 - LoadAll, [786](#)
 - MakeCurrent, [786](#)
 - ShareContexts, [788](#)
 - SwapBuffers, [787](#)
 - Update, [787](#)
 - VSync, [788](#)
- OpenTK::Graphics::GraphicsContextException, [789](#)
 - GraphicsContextException, [789](#)
- OpenTK::Graphics::GraphicsContextMissingException, [790](#)
 - GraphicsContextMissingException, [790](#)
- OpenTK::Graphics::GraphicsContextVersion, [791](#)
 - Major, [791](#)
 - Minor, [791](#)
 - Renderer, [791](#)
 - Vendor, [791](#)
- OpenTK::Graphics::GraphicsErrorException, [792](#)
 - GraphicsErrorException, [792](#)
- OpenTK::Graphics::GraphicsMode, [793](#)
 - AccumulatorFormat, [798](#)
 - Buffers, [798](#)
 - ColorFormat, [798](#)
 - Default, [798](#)
 - Depth, [798](#)
 - Equals, [797](#)
 - GetHashCode, [797](#)
 - GraphicsMode, [795](#), [796](#)
 - Index, [799](#)
 - Samples, [799](#)
 - Stencil, [799](#)
 - Stereo, [799](#)
 - ToString, [798](#)
- OpenTK::Graphics::GraphicsModeException, [799](#)
 - GraphicsModeException, [800](#)
- OpenTK::Graphics::IGraphicsContext, [800](#)
 - ErrorChecking, [802](#)

- GraphicsMode, [802](#)
- IsCurrent, [802](#)
- IsDisposed, [802](#)
- LoadAll, [801](#)
- MakeCurrent, [801](#)
- SwapBuffers, [802](#)
- Update, [802](#)
- VSync, [803](#)
- OpenTK::Graphics::IGraphicsContextInternal, [803](#)
 - Context, [804](#)
 - GetAddress, [804](#)
 - Implementation, [804](#)
 - LoadAll, [804](#)
- OpenTK::Graphics::OpenGL::GL, [805](#)
 - Accum, [953](#)
 - ActiveTexture, [953](#)
 - AlphaFunc, [954](#)
 - AreTexturesResident, [954–956](#)
 - ArrayElement, [956](#)
 - AttachShader, [956, 957](#)
 - Begin, [957](#)
 - BeginQuery, [957](#)
 - BindAttribLocation, [958](#)
 - BindBuffer, [958, 959](#)
 - BindTexture, [959](#)
 - Bitmap, [960](#)
 - BlendColor, [961](#)
 - BlendEquation, [961, 962](#)
 - BlendEquationSeparate, [962, 963](#)
 - BlendFunc, [963, 964](#)
 - BlendFuncSeparate, [965–967](#)
 - BufferData, [967](#)
 - BufferData< T2 >, [968–970](#)
 - BufferSubData, [970](#)
 - BufferSubData< T3 >, [970–972](#)
 - CallList, [972, 973](#)
 - CallLists, [973](#)
 - CallLists< T2 >, [973–975](#)
 - Clear, [975](#)
 - ClearAccum, [975](#)
 - ClearColor, [976](#)
 - ClearDepth, [976](#)
 - ClearIndex, [976](#)
 - ClearStencil, [976](#)
 - ClientActiveTexture, [977](#)
 - ClipPlane, [977, 978](#)
 - Color3, [978–986](#)
 - Color4, [986–994](#)
 - ColorMask, [994](#)
 - ColorMaterial, [995](#)
 - ColorPointer, [995](#)
 - ColorPointer< T3 >, [996, 997](#)
 - ColorSubTable, [998](#)
 - ColorSubTable< T5 >, [998–1001](#)
 - ColorTable, [1002](#)
 - ColorTable< T5 >, [1003–1006](#)
 - ColorTableParameter, [1007–1009](#)
 - CompileShader, [1009, 1010](#)
 - CompressedTexImage1D, [1010](#)
 - CompressedTexImage1D< T6 >, [1010–1013](#)
 - CompressedTexImage2D, [1013](#)
 - CompressedTexImage2D< T7 >, [1014–1017](#)
 - CompressedTexImage3D, [1018](#)
 - CompressedTexImage3D< T8 >, [1019–1021](#)
 - CompressedTexSubImage1D, [1022](#)
 - CompressedTexSubImage1D< T6 >, [1023–1025](#)
 - CompressedTexSubImage2D, [1025](#)
 - CompressedTexSubImage2D< T8 >, [1026–1028](#)
 - CompressedTexSubImage3D, [1029](#)
 - CompressedTexSubImage3D< T10 >, [1029–1032](#)
 - ConvolutionFilter1D, [1032](#)
 - ConvolutionFilter1D< T5 >, [1033–1036](#)
 - ConvolutionFilter2D, [1037](#)
 - ConvolutionFilter2D< T6 >, [1038–1041](#)
 - ConvolutionParameter, [1042–1044](#)
 - CopyColorSubTable, [1045](#)
 - CopyColorTable, [1045](#)
 - CopyConvolutionFilter1D, [1046](#)
 - CopyConvolutionFilter2D, [1046](#)
 - CopyPixels, [1047](#)
 - CopyTexImage1D, [1047](#)
 - CopyTexImage2D, [1048](#)
 - CopyTexSubImage1D, [1049](#)

- CopyTexSubImage2D, [1050](#)
- CopyTexSubImage3D, [1050](#)
- CreateProgram, [1051](#)
- CreateShader, [1051](#)
- CullFace, [1051](#)
- DeleteBuffers, [1052](#), [1053](#)
- DeleteLists, [1053](#)
- DeleteProgram, [1053](#), [1054](#)
- DeleteQueries, [1054](#), [1055](#)
- DeleteShader, [1055](#), [1056](#)
- DeleteTextures, [1056](#), [1057](#)
- DepthFunc, [1057](#)
- DepthMask, [1058](#)
- DepthRange, [1058](#)
- DetachShader, [1058](#)
- DrawArrays, [1059](#)
- DrawBuffer, [1059](#)
- DrawBuffers, [1059](#), [1060](#)
- DrawElements, [1060](#)
- DrawElements< T3 >, [1061](#), [1062](#)
- DrawPixels, [1063](#)
- DrawPixels< T4 >, [1063](#)–[1066](#)
- DrawRangeElements, [1066](#), [1067](#)
- DrawRangeElements< T5 >, [1067](#)–[1072](#)
- EdgeFlag, [1072](#)
- EdgeFlagPointer, [1073](#)
- EdgeFlagPointer< T1 >, [1073](#), [1074](#)
- Enable, [1074](#), [1075](#)
- EnableClientState, [1075](#)
- EnableVertexAttribArray, [1075](#), [1076](#)
- EvalCoord1, [1076](#), [1077](#)
- EvalCoord2, [1077](#)–[1079](#)
- EvalMesh1, [1080](#)
- EvalMesh2, [1080](#)
- EvalPoint1, [1080](#)
- EvalPoint2, [1081](#)
- FeedbackBuffer, [1081](#)
- Finish, [1082](#)
- Flush, [1082](#)
- Fog, [1082](#)–[1084](#)
- FogCoord, [1084](#)
- FogCoordPointer, [1085](#)
- FogCoordPointer< T2 >, [1085](#), [1086](#)
- FrontFace, [1087](#)
- Frustum, [1087](#)
- GenBuffers, [1087](#), [1088](#)
- GenLists, [1089](#)
- GenQueries, [1089](#), [1090](#)
- GenTextures, [1090](#)–[1092](#)
- GetActiveAttrib, [1092](#), [1093](#)
- GetActiveUniform, [1094](#)–[1096](#)
- GetAttachedShaders, [1096](#)–[1098](#)
- GetAttribLocation, [1098](#)
- GetBufferParameter, [1099](#), [1100](#)
- GetBufferPointer, [1100](#)
- GetBufferPointer< T2 >, [1100](#)–[1102](#)
- GetBufferSubData, [1102](#)
- GetBufferSubData< T3 >, [1102](#)–[1104](#)
- GetClipPlane, [1104](#), [1105](#)
- GetColorTable, [1105](#)
- GetColorTable< T3 >, [1106](#)–[1108](#)
- GetColorTableParameter, [1109](#)–[1112](#)
- GetCompressedTexImage, [1112](#)
- GetCompressedTexImage< T2 >, [1113](#), [1114](#)
- GetConvolutionFilter, [1115](#)
- GetConvolutionFilter< T3 >, [1115](#)–[1118](#)
- GetConvolutionParameter, [1118](#)–[1121](#)
- GetError, [1121](#)
- GetHistogram, [1121](#)
- GetHistogram< T4 >, [1122](#)–[1124](#)
- GetHistogramParameter, [1125](#)–[1127](#)
- GetLight, [1127](#)–[1130](#)
- GetMap, [1130](#)–[1134](#)
- GetMaterial, [1135](#)–[1137](#)
- GetMinmax, [1137](#)
- GetMinmax< T4 >, [1138](#)–[1140](#)
- GetMinmaxParameter, [1141](#)–[1143](#)
- GetPixelMap, [1143](#)–[1148](#)
- GetPointer, [1148](#)
- GetPointer< T1 >, [1148](#)–[1150](#)
- GetPolygonStipple, [1150](#), [1151](#)
- GetProgram, [1151](#)–[1153](#)
- GetProgramInfoLog, [1153](#)–[1155](#)

- GetQuery, [1155](#), [1156](#)
- GetQueryObject, [1156–1158](#)
- GetSeparableFilter, [1159](#)
- GetSeparableFilter< T3, T4, T5 >, [1159–1162](#)
- GetSeparableFilter< T4, T5 >, [1163–1165](#)
- GetSeparableFilter< T5 >, [1166–1168](#)
- GetShader, [1169–1171](#)
- GetShaderInfoLog, [1171](#), [1172](#)
- GetShaderSource, [1172](#), [1173](#)
- GetString, [1174](#)
- GetTexEnv, [1175–1177](#)
- GetTexGen, [1178–1180](#)
- GetTexImage, [1181](#)
- GetTexImage< T4 >, [1182–1184](#)
- GetTexLevelParameter, [1185–1189](#)
- GetTexParameter, [1189–1192](#)
- GetUniform, [1192–1196](#)
- GetUniformLocation, [1197](#)
- GetVertexAttrib, [1197–1204](#)
- GetVertexAttribPointer, [1204](#), [1205](#)
- GetVertexAttribPointer< T2 >, [1205–1208](#)
- Hint, [1209](#)
- Histogram, [1209](#)
- Index, [1210](#), [1211](#)
- IndexMask, [1212](#)
- IndexPointer, [1212](#)
- IndexPointer< T2 >, [1212–1214](#)
- InitNames, [1214](#)
- InterleavedArrays, [1214](#)
- InterleavedArrays< T2 >, [1215](#), [1216](#)
- IsBuffer, [1216](#), [1217](#)
- IsEnabled, [1217](#)
- IsList, [1218](#)
- IsProgram, [1218](#)
- IsQuery, [1218](#), [1219](#)
- IsShader, [1219](#)
- IsTexture, [1219](#)
- Light, [1220–1222](#)
- LightModel, [1222–1224](#)
- LineStipple, [1224](#), [1225](#)
- LineWidth, [1225](#)
- LinkProgram, [1225](#), [1226](#)
- ListBase, [1226](#)
- LoadAll, [1226](#)
- LoadIdentity, [1226](#)
- LoadMatrix, [1226–1228](#)
- LoadName, [1228](#)
- LoadTransposeMatrix, [1228–1230](#)
- LogicOp, [1230](#)
- Map1, [1230–1233](#)
- Map2, [1234–1238](#)
- MapBuffer, [1239](#)
- MapGrid1, [1240](#)
- MapGrid2, [1241](#)
- Material, [1241–1243](#)
- MatrixMode, [1243](#)
- Minmax, [1244](#)
- MultiDrawArrays, [1244](#), [1245](#)
- MultiDrawElements, [1245](#), [1246](#)
- MultiDrawElements< T3 >, [1247–1253](#)
- MultiTexCoord1, [1254–1256](#)
- MultiTexCoord2, [1257–1262](#)
- MultiTexCoord3, [1263–1268](#)
- MultiTexCoord4, [1269–1274](#)
- MultMatrix, [1275](#), [1276](#)
- MultTransposeMatrix, [1276](#), [1277](#)
- NewList, [1278](#)
- Normal3, [1278–1284](#)
- NormalPointer, [1284](#)
- NormalPointer< T2 >, [1285](#), [1286](#)
- Ortho, [1286](#)
- PassThrough, [1287](#)
- PixelMap, [1287–1292](#)
- PixelStore, [1293](#)
- PixelTransfer, [1294](#)
- PixelZoom, [1295](#)
- PointParameter, [1295–1297](#)
- PointSize, [1297](#)
- PolygonMode, [1297](#)
- PolygonOffset, [1298](#)
- PolygonStipple, [1298](#), [1299](#)
- PrioritizeTextures, [1299](#), [1300](#)
- PushAttrib, [1301](#)
- PushClientAttrib, [1301](#)
- PushMatrix, [1301](#)
- PushName, [1301](#), [1302](#)

- RasterPos2, [1302–1305](#)
- RasterPos3, [1305–1308](#)
- RasterPos4, [1308–1311](#)
- ReadBuffer, [1311](#)
- ReadPixels, [1312](#)
- ReadPixels< T6 >, [1312–1315](#)
- Rect, [1316–1319](#)
- RenderMode, [1319](#)
- ResetHistogram, [1319](#)
- ResetMinmax, [1320](#)
- Rotate, [1320](#)
- SampleCoverage, [1320](#)
- Scale, [1321](#)
- Scissor, [1321](#)
- SecondaryColor3, [1321–1328](#)
- SecondaryColorPointer, [1328](#)
- SecondaryColorPointer< T3 >, [1328–1330](#)
- SelectBuffer, [1331](#), [1332](#)
- SeparableFilter2D, [1332](#)
- SeparableFilter2D< T6, T7 >, [1333–1335](#), [1337](#)
- SeparableFilter2D< T7 >, [1338–1341](#)
- ShadeModel, [1343](#)
- ShaderSource, [1343](#), [1344](#)
- StencilFunc, [1344](#), [1345](#)
- StencilFuncSeparate, [1345](#), [1346](#)
- StencilMask, [1346](#)
- StencilMaskSeparate, [1347](#)
- StencilOp, [1347](#)
- StencilOpSeparate, [1348](#)
- SyncRoot, [1479](#)
- TexCoord1, [1348–1350](#)
- TexCoord2, [1350–1354](#)
- TexCoord3, [1354–1358](#)
- TexCoord4, [1358–1362](#)
- TexCoordPointer, [1362](#)
- TexCoordPointer< T3 >, [1363](#), [1364](#)
- TexEnv, [1365–1368](#)
- TexGen, [1369–1372](#)
- TexImage1D, [1372](#)
- TexImage1D< T7 >, [1374](#), [1375](#), [1377](#), [1378](#)
- TexImage2D, [1379](#)
- TexImage2D< T8 >, [1381](#), [1382](#), [1384](#), [1386](#)
- TexImage3D, [1387](#)
- TexImage3D< T9 >, [1389](#), [1390](#), [1392](#), [1393](#)
- TexParameter, [1395–1397](#)
- TexSubImage1D, [1398](#)
- TexSubImage1D< T6 >, [1398–1401](#)
- TexSubImage2D, [1402](#)
- TexSubImage2D< T8 >, [1403–1406](#)
- TexSubImage3D, [1406](#)
- TexSubImage3D< T10 >, [1407–1410](#)
- Translate, [1411](#), [1412](#)
- Uniform1, [1412–1415](#)
- Uniform2, [1415–1417](#)
- Uniform3, [1418–1420](#)
- Uniform4, [1421–1423](#)
- UseProgram, [1424](#)
- ValidateProgram, [1424](#)
- Vertex2, [1425–1428](#)
- Vertex3, [1429–1432](#)
- Vertex4, [1433–1436](#)
- VertexAttrib1, [1437–1439](#)
- VertexAttrib2, [1440–1445](#)
- VertexAttrib3, [1446–1451](#)
- VertexAttrib4, [1452–1463](#)
- VertexAttribPointer, [1463](#)
- VertexAttribPointer< T5 >, [1464–1469](#)
- VertexPointer, [1470](#)
- VertexPointer< T3 >, [1470–1472](#)
- Viewport, [1472](#)
- WindowPos2, [1473–1476](#)
- WindowPos3, [1476–1479](#)
- OpenTK::GraphicsException, [1479](#)
- GraphicsException, [1480](#)
- OpenTK::Half, [1480](#)
- CompareTo, [1485](#)
- Epsilon, [1490](#)
- Equals, [1485](#)
- FromBinaryStream, [1486](#)
- FromBytes, [1486](#)
- GetBytes, [1486](#)
- GetObjectData, [1486](#)
- Half, [1484](#), [1485](#)

- IsNaN, [1491](#)
- IsNegativeInfinity, [1491](#)
- IsPositiveInfinity, [1491](#)
- IsZero, [1491](#)
- MaxValue, [1490](#)
- MinNormalizedValue, [1490](#)
- MinValue, [1491](#)
- operator double, [1487](#)
- operator float, [1487](#)
- operator Half, [1487](#)
- Parse, [1488](#)
- SizeInBytes, [1491](#)
- ToBinaryStream, [1488](#)
- ToSingle, [1489](#)
- ToString, [1489](#)
- TryParse, [1489](#), [1490](#)
- OpenTK::INativeWindow, [1491](#)
 - Bounds, [1496](#)
 - ClientRectangle, [1496](#)
 - ClientSize, [1496](#)
 - Close, [1495](#)
 - Closed, [1498](#)
 - Closing, [1498](#)
 - Disposed, [1499](#)
 - Exists, [1496](#)
 - Focused, [1496](#)
 - FocusedChanged, [1499](#)
 - Height, [1496](#)
 - Icon, [1497](#)
 - IconChanged, [1499](#)
 - InputDriver, [1497](#)
 - KeyPress, [1499](#)
 - Location, [1497](#)
 - MouseEnter, [1499](#)
 - MouseLeave, [1499](#)
 - Move, [1499](#)
 - PointToClient, [1495](#)
 - PointToScreen, [1495](#)
 - ProcessEvents, [1495](#)
 - Resize, [1500](#)
 - Size, [1497](#)
 - Title, [1497](#)
 - TitleChanged, [1500](#)
 - Visible, [1497](#)
 - VisibleChanged, [1500](#)
 - Width, [1497](#)
 - WindowBorder, [1498](#)
 - WindowBorderChanged, [1500](#)
 - WindowInfo, [1498](#)
 - WindowState, [1498](#)
 - WindowStateChanged, [1500](#)
 - X, [1498](#)
 - Y, [1498](#)
- OpenTK::Input
 - A, [252](#)
 - AltLeft, [249](#)
 - AltRight, [249](#)
 - Axis0, [248](#)
 - Axis1, [248](#)
 - Axis2, [248](#)
 - Axis3, [248](#)
 - Axis4, [248](#)
 - Axis5, [248](#)
 - Axis6, [248](#)
 - Axis7, [248](#)
 - Axis8, [248](#)
 - Axis9, [248](#)
 - B, [252](#)
 - Back, [251](#)
 - BackSlash, [253](#)
 - BackSpace, [251](#)
 - BracketLeft, [253](#)
 - BracketRight, [253](#)
 - Button0, [248](#)
 - Button1, [248](#), [253](#)
 - Button10, [248](#)
 - Button11, [249](#)
 - Button12, [249](#)
 - Button13, [249](#)
 - Button14, [249](#)
 - Button15, [249](#)
 - Button2, [248](#), [253](#)
 - Button3, [248](#), [253](#)
 - Button4, [248](#), [253](#)
 - Button5, [248](#), [253](#)
 - Button6, [248](#), [253](#)
 - Button7, [248](#), [253](#)
 - Button8, [248](#), [253](#)
 - Button9, [248](#), [253](#)
 - C, [252](#)
 - CapsLock, [251](#)
 - Clear, [251](#)

Comma, [253](#)
ControlLeft, [249](#)
ControlRight, [249](#)
D, [252](#)
Delete, [251](#)
Down, [250](#)
E, [252](#)
End, [251](#)
Enter, [250](#)
Escape, [250](#)
F, [252](#)
F1, [249](#)
F10, [250](#)
F11, [250](#)
F12, [250](#)
F13, [250](#)
F14, [250](#)
F15, [250](#)
F16, [250](#)
F17, [250](#)
F18, [250](#)
F19, [250](#)
F2, [249](#)
F20, [250](#)
F21, [250](#)
F22, [250](#)
F23, [250](#)
F24, [250](#)
F25, [250](#)
F26, [250](#)
F27, [250](#)
F28, [250](#)
F29, [250](#)
F3, [249](#)
F30, [250](#)
F31, [250](#)
F32, [250](#)
F33, [250](#)
F34, [250](#)
F35, [250](#)
F4, [249](#)
F5, [249](#)
F6, [249](#)
F7, [250](#)
F8, [250](#)
F9, [250](#)
G, [252](#)
H, [252](#)
Hid, [248](#)
Home, [251](#)
I, [252](#)
InputDeviceType, [247](#)
Insert, [251](#)
J, [252](#)
JoystickAxis, [248](#)
JoystickButton, [248](#)
K, [252](#)
Key, [249](#)
Keyboard, [247](#)
Keypad0, [251](#)
Keypad1, [251](#)
Keypad2, [251](#)
Keypad3, [251](#)
Keypad4, [251](#)
Keypad5, [251](#)
Keypad6, [251](#)
Keypad7, [251](#)
Keypad8, [251](#)
Keypad9, [251](#)
KeypadAdd, [251](#)
KeypadDecimal, [251](#)
KeypadDivide, [251](#)
KeypadEnter, [251](#)
KeypadMinus, [251](#)
KeypadMultiply, [251](#)
KeypadPlus, [251](#)
KeypadSubtract, [251](#)
L, [252](#)
LAlt, [249](#)
LastButton, [253](#)
LastKey, [253](#)
LBracket, [253](#)
LControl, [249](#)
Left, [250](#), [253](#)
LShift, [249](#)
LWin, [249](#)
M, [252](#)
Menu, [249](#)
Middle, [253](#)
Minus, [253](#)
Mouse, [247](#)
MouseButton, [253](#)

- N, [252](#)
- Number0, [252](#)
- Number1, [252](#)
- Number2, [252](#)
- Number3, [252](#)
- Number4, [252](#)
- Number5, [252](#)
- Number6, [252](#)
- Number7, [252](#)
- Number8, [252](#)
- Number9, [253](#)
- NumLock, [251](#)
- O, [252](#)
- P, [252](#)
- PageDown, [251](#)
- PageUp, [251](#)
- Pause, [251](#)
- Period, [253](#)
- Plus, [253](#)
- PrintScreen, [251](#)
- Q, [252](#)
- Quote, [253](#)
- R, [252](#)
- RAlt, [249](#)
- RBracket, [253](#)
- RControl, [249](#)
- Right, [250](#), [253](#)
- RShift, [249](#)
- RWin, [249](#)
- S, [252](#)
- ScrollLock, [251](#)
- Semicolon, [253](#)
- ShiftLeft, [249](#)
- ShiftRight, [249](#)
- Slash, [253](#)
- Sleep, [251](#)
- Space, [251](#)
- T, [252](#)
- Tab, [251](#)
- Tilde, [253](#)
- U, [252](#)
- Unknown, [249](#)
- Up, [250](#)
- V, [252](#)
- W, [252](#)
- WinLeft, [249](#)
- WinRight, [249](#)
- X, [252](#)
- Y, [252](#)
- Z, [252](#)
- OpenTK::Input::GamePad, [1500](#)
- OpenTK::Input::GamePadState, [1501](#)
- OpenTK::Input::InputDevice, [1501](#)
 - Description, [1502](#)
 - DeviceType, [1502](#)
- OpenTK::Input::InputDriver, [1502](#)
 - Poll, [1503](#)
- OpenTK::Input::IJoystickDriver, [1503](#)
 - Joysticks, [1503](#)
- OpenTK::Input::IKeyboardDriver, [1504](#)
 - Keyboard, [1504](#)
- OpenTK::Input::IMouseDriver, [1504](#)
 - Mouse, [1505](#)
- OpenTK::Input::JoystickAxisCollection, [1505](#)
 - Count, [1506](#)
 - this, [1506](#)
- OpenTK::Input::JoystickButtonCollection, [1506](#)
 - Count, [1507](#)
 - this, [1507](#)
- OpenTK::Input::JoystickButtonEventArgs, [1507](#)
 - Button, [1508](#)
 - Pressed, [1508](#)
- OpenTK::Input::JoystickDevice, [1508](#)
 - Axis, [1510](#)
 - Button, [1510](#)
 - ButtonDown, [1509](#)
 - ButtonUp, [1509](#)
 - Description, [1510](#)
 - DeviceType, [1510](#)
 - Move, [1510](#)
- OpenTK::Input::JoystickEventArgs, [1511](#)
- OpenTK::Input::JoystickMoveEventArgs, [1511](#)
 - Axis, [1512](#)
 - Delta, [1512](#)
 - JoystickMoveEventArgs, [1512](#)
 - Value, [1512](#)
- OpenTK::Input::KeyboardDevice, [1513](#)
 - Description, [1515](#)

- DeviceID, [1515](#)
- DeviceType, [1515](#)
- GetHashCode, [1514](#)
- KeyDown, [1516](#)
- KeyRepeat, [1515](#)
- KeyUp, [1516](#)
- NumberOfFunctionKeys, [1515](#)
- NumberOfKeys, [1515](#)
- NumberOfLeds, [1515](#)
- this, [1516](#)
- ToString, [1514](#)
- OpenTK::Input::KeyboardKeyEventArgs, [1516](#)
 - Key, [1517](#)
 - KeyboardKeyEventArgs, [1517](#)
- OpenTK::Input::KeyboardState, [1517](#)
 - Equals, [1518](#)
 - IsKeyDown, [1518](#)
 - IsKeyUp, [1518](#)
- OpenTK::Input::MouseButtonEventArgs, [1519](#)
 - Button, [1520](#)
 - IsPressed, [1520](#)
 - MouseButtonEventArgs, [1520](#)
- OpenTK::Input::MouseDevice, [1521](#)
 - ButtonDown, [1524](#)
 - ButtonUp, [1524](#)
 - Description, [1523](#)
 - DeviceID, [1523](#)
 - DeviceType, [1523](#)
 - GetHashCode, [1523](#)
 - Move, [1524](#)
 - NumberOfButtons, [1523](#)
 - NumberOfWheels, [1523](#)
 - this, [1523](#)
 - ToString, [1523](#)
 - Wheel, [1524](#)
 - WheelChanged, [1525](#)
 - WheelPrecise, [1524](#)
 - X, [1524](#)
 - Y, [1524](#)
- OpenTK::Input::MouseEventArgs, [1525](#)
 - MouseEventArgs, [1526](#)
 - Position, [1526](#)
 - X, [1526](#)
 - Y, [1527](#)
- OpenTK::Input::MouseMoveEventArgs, [1527](#)
 - MouseEventArgs, [1528](#)
 - XDelta, [1528](#)
 - YDelta, [1528](#)
- OpenTK::Input::MouseState, [1529](#)
 - Equals, [1529](#)
- OpenTK::Input::MouseWheelEventArgs, [1529](#)
 - Delta, [1531](#)
 - DeltaPrecise, [1531](#)
 - MouseWheelEventArgs, [1530](#), [1531](#)
 - Value, [1531](#)
 - ValuePrecise, [1531](#)
- OpenTK::KeyPressEventArgs, [1532](#)
 - KeyChar, [1532](#)
 - KeyPressEventArgs, [1532](#)
- OpenTK::Matrix4, [1533](#)
 - Column0, [1556](#)
 - Column1, [1556](#)
 - Column2, [1556](#)
 - Column3, [1556](#)
 - CreateFromAxisAngle, [1540](#)
 - CreateOrthographic, [1540](#), [1541](#)
 - CreateOrthographicOffCenter, [1541](#)
 - CreatePerspectiveFieldOfView, [1542](#)
 - CreatePerspectiveOffCenter, [1543](#), [1544](#)
 - CreateRotationX, [1544](#)
 - CreateRotationY, [1545](#)
 - CreateRotationZ, [1545](#)
 - CreateTranslation, [1546](#), [1547](#)
 - Determinant, [1556](#)
 - Equals, [1547](#)
 - Frustum, [1547](#)
 - GetHashCode, [1548](#)
 - Identity, [1555](#)
 - Invert, [1548](#)
 - LookAt, [1549](#)
 - M11, [1556](#)
 - M12, [1556](#)
 - M13, [1556](#)
 - M14, [1557](#)
 - M21, [1557](#)
 - M22, [1557](#)

- M23, [1557](#)
- M24, [1557](#)
- M31, [1557](#)
- M32, [1557](#)
- M33, [1557](#)
- M34, [1557](#)
- M41, [1557](#)
- M42, [1558](#)
- M43, [1558](#)
- M44, [1558](#)
- Matrix4, [1539](#)
- Mult, [1549](#), [1550](#)
- operator*, [1550](#)
- operator==, [1551](#)
- Perspective, [1551](#)
- Rotate, [1551](#), [1552](#)
- RotateX, [1552](#)
- RotateY, [1552](#)
- RotateZ, [1553](#)
- Row0, [1555](#)
- Row1, [1555](#)
- Row2, [1555](#)
- Row3, [1556](#)
- Scale, [1553](#)
- ToString, [1554](#)
- Translation, [1554](#)
- Transpose, [1554](#), [1555](#)
- OpenTK::Matrix4d, [1558](#)
 - Column0, [1582](#)
 - Column1, [1582](#)
 - Column2, [1582](#)
 - Column3, [1582](#)
 - CreateFromAxisAngle, [1565](#), [1566](#)
 - CreateOrthographic, [1566](#)
 - CreateOrthographicOffCenter, [1567](#)
 - CreatePerspectiveFieldOfView, [1567](#), [1568](#)
 - CreatePerspectiveOffCenter, [1569](#)
 - CreateRotationX, [1570](#)
 - CreateRotationY, [1570](#), [1571](#)
 - CreateRotationZ, [1571](#)
 - CreateTranslation, [1571](#), [1572](#)
 - Determinant, [1582](#)
 - Equals, [1573](#)
 - Frustum, [1573](#)
 - GetHashCode, [1574](#)
 - Identity, [1581](#)
 - Invert, [1574](#)
 - LookAt, [1574](#), [1575](#)
 - M11, [1582](#)
 - M12, [1583](#)
 - M13, [1583](#)
 - M14, [1583](#)
 - M21, [1583](#)
 - M22, [1583](#)
 - M23, [1583](#)
 - M24, [1583](#)
 - M31, [1583](#)
 - M32, [1583](#)
 - M33, [1583](#)
 - M34, [1584](#)
 - M41, [1584](#)
 - M42, [1584](#)
 - M43, [1584](#)
 - M44, [1584](#)
 - Matrix4d, [1564](#), [1565](#)
 - Mult, [1575](#), [1576](#)
 - operator*, [1576](#)
 - operator==, [1576](#)
 - Perspective, [1577](#)
 - Rotate, [1577](#)
 - RotateX, [1578](#)
 - RotateY, [1578](#)
 - RotateZ, [1578](#)
 - Row0, [1581](#)
 - Row1, [1582](#)
 - Row2, [1582](#)
 - Row3, [1582](#)
 - Scale, [1579](#)
 - ToString, [1580](#)
 - Translation, [1580](#)
 - Transpose, [1581](#)
- OpenTK::NativeWindow, [1584](#)
 - Bounds, [1595](#)
 - ClientRectangle, [1595](#)
 - ClientSize, [1595](#)
 - Close, [1591](#)
 - Closed, [1598](#)
 - Closing, [1598](#)
 - Dispose, [1591](#)
 - Disposed, [1598](#)
 - EnsureUndisposed, [1591](#)

- Exists, [1596](#)
- Focused, [1596](#)
- FocusedChanged, [1598](#)
- Height, [1596](#)
- Icon, [1596](#)
- IconChanged, [1598](#)
- InputDriver, [1596](#)
- IsDisposed, [1596](#)
- KeyPress, [1599](#)
- Location, [1596](#)
- MouseEnter, [1599](#)
- MouseLeave, [1599](#)
- Move, [1599](#)
- NativeWindow, [1589](#), [1590](#)
- OnClosed, [1591](#)
- OnClosing, [1591](#)
- OnDisposed, [1591](#)
- OnFocusedChanged, [1592](#)
- OnIconChanged, [1592](#)
- OnKeyPress, [1592](#)
- OnMouseEnter, [1592](#)
- OnMouseLeave, [1593](#)
- OnMove, [1593](#)
- OnResize, [1593](#)
- OnTitleChanged, [1593](#)
- OnVisibleChanged, [1593](#)
- OnWindowBorderChanged, [1594](#)
- OnWindowStateChanged, [1594](#)
- PointToClient, [1594](#)
- PointToScreen, [1594](#)
- ProcessEvents, [1595](#)
- Resize, [1599](#)
- Size, [1597](#)
- Title, [1597](#)
- TitleChanged, [1599](#)
- Visible, [1597](#)
- VisibleChanged, [1599](#)
- Width, [1597](#)
- WindowBorder, [1597](#)
- WindowBorderChanged, [1600](#)
- WindowInfo, [1597](#)
- WindowState, [1597](#)
- WindowStateChanged, [1600](#)
- X, [1598](#)
- Y, [1598](#)
- OpenTK::Platform::IGameWindow, [1600](#)
- Load, [1602](#)
- MakeCurrent, [1601](#)
- RenderFrame, [1602](#)
- Run, [1601](#)
- SwapBuffers, [1602](#)
- Unload, [1602](#)
- UpdateFrame, [1602](#)
- OpenTK::Platform::IWindowInfo, [1603](#)
- OpenTK::Platform::Windows
 - AddOverlays, [279](#)
 - ALLEVENTS, [276](#)
 - ALLINPUT, [276](#)
 - ALLPOSTMESSAGE, [276](#)
 - APPKEYS, [277](#)
 - Attr_Specified, [279](#)
 - Attributes, [278](#)
 - CAPTUREMOUSE, [277](#)
 - DisplayName, [278](#)
 - DRAWFRAME, [278](#)
 - EXCLUDE, [276](#)
 - ExeType, [278](#)
 - EXINPUTSINK, [277](#)
 - FORCEMINIMIZE, [280](#)
 - FRAMECHANGED, [278](#)
 - GdiCharset, [275](#)
 - GWL, [275](#)
 - HIDE, [279](#)
 - HIDEWINDOW, [278](#)
 - HOTKEY, [276](#)
 - Icon, [278](#)
 - IconLocation, [278](#)
 - INPUT, [276](#)
 - INPUT_LEGACY, [276](#)
 - INPUTSINK, [277](#)
 - KEY, [276](#)
 - LargeIcon, [279](#)
 - LinkOverlay, [278](#)
 - MapVirtualKeyType, [275](#)
 - MINIMIZE, [279](#)
 - MOUSE, [276](#)
 - MOUSE_ATTRIBUTES_-
CHANGED, [277](#)
 - MOUSE_MOVE_ABSOLUTE, [277](#)
 - MOUSE_MOVE_RELATIVE, [277](#)
 - MOUSE_VIRTUAL_DESKTOP,
[277](#)

- MOUSEBUTTON, [276](#)
- MouseKeys, [275](#)
- MOUSEMOVE, [276](#)
- NCXBUTTONDBLCLK, [280](#)
- NCXBUTTONDOWN, [280](#)
- NCXBUTTONUP, [280](#)
- NOACTIVATE, [278](#)
- NOCOPYBITS, [278](#)
- NOHOTKEYS, [277](#)
- NOLEGACY, [277](#)
- NOMOVE, [277](#)
- NOOWNERZORDER, [278](#)
- NOREDRA, [278](#)
- NOREPOSITION, [278](#)
- NOSENDCHANGING, [278](#)
- NOSIZE, [277](#)
- NOZORDER, [278](#)
- OpenIcon, [279](#)
- OverlayIndex, [279](#)
- PAGEONLY, [276](#)
- PAINT, [276](#)
- PIDL, [279](#)
- POSTMESSAGE, [276](#)
- QueueStatusFlags, [275](#)
- RAWINPUT, [276](#)
- RawInputDeviceFlags, [276](#)
- RawMouseFlags, [277](#)
- REMOVE, [276](#)
- RESTORE, [279](#)
- ScanCodeToVirtualKey, [275](#)
- ScanCodeToVirtualKeyExtended, [275](#)
- Selected, [279](#)
- SENDMESSAGE, [276](#)
- SetWindowPosFlags, [277](#)
- ShellIconSize, [279](#)
- ShGetFileIconFlags, [278](#)
- SHOW, [279](#)
- SHOWDEFAULT, [279](#)
- SHOWMAXIMIZED, [279](#)
- SHOWMINIMIZED, [279](#)
- SHOWMINNOACTIVE, [279](#)
- SHOWNA, [279](#)
- SHOWNOACTIVATE, [279](#)
- SHOWNORMAL, [279](#)
- SHOWWINDOW, [278](#)
- ShowWindowCommand, [279](#)
- ShowWindowMessageIdentifiers, [280](#)
- SmallIcon, [279](#)
- SysIconIndex, [278](#)
- TIMER, [276](#)
- TypeName, [278](#)
- UseFileAttributes, [279](#)
- VirtualKeyToCharacter, [275](#)
- VirtualKeyToScanCode, [275](#)
- WindowMessage, [280](#)
- XBUTTONDBLCLK, [280](#)
- XBUTTONDOWN, [280](#)
- XBUTTONUP, [280](#)
- OpenTK::Platform::X11
 - XKey, [294](#)
- OpenTK::PlatformException, [1603](#)
 - PlatformException, [1603](#)
- OpenTK::Properties::Resources, [1603](#)
- OpenTK::Quaternion, [1604](#)
 - Add, [1608](#)
 - Conjugate, [1608](#), [1609](#)
 - Equals, [1609](#)
 - FromAxisAngle, [1610](#)
 - GetHashCode, [1610](#)
 - Identity, [1617](#)
 - Invert, [1610](#)
 - Length, [1617](#)
 - LengthSquared, [1617](#)
 - Mult, [1611](#)
 - Multiply, [1611](#), [1612](#)
 - Normalize, [1612](#), [1613](#)
 - operator*, [1613](#), [1614](#)
 - operator+, [1614](#)
 - operator-, [1615](#)
 - operator==, [1615](#)
 - Quaternion, [1607](#), [1608](#)
 - Slerp, [1615](#)
 - Sub, [1616](#)
 - ToAxisAngle, [1616](#), [1617](#)
 - ToString, [1617](#)
 - W, [1618](#)
 - X, [1618](#)
 - XYZ, [1618](#)
 - Xyz, [1618](#)
 - Y, [1618](#)

- Z, [1618](#)
- OpenTK::Quaterniond, [1618](#)
 - Add, [1623](#)
 - Conjugate, [1623](#), [1624](#)
 - Equals, [1624](#)
 - FromAxisAngle, [1624](#)
 - GetHashCode, [1625](#)
 - Identity, [1632](#)
 - Invert, [1625](#)
 - Length, [1632](#)
 - LengthSquared, [1632](#)
 - Mult, [1625](#), [1626](#)
 - Multiply, [1626](#), [1627](#)
 - Normalize, [1627](#), [1628](#)
 - operator*, [1628](#), [1629](#)
 - operator+, [1629](#)
 - operator-, [1629](#)
 - operator==, [1630](#)
 - Quaterniond, [1622](#)
 - Slerp, [1630](#)
 - Sub, [1630](#), [1631](#)
 - ToAxisAngle, [1631](#)
 - ToString, [1631](#)
 - W, [1632](#)
 - X, [1632](#)
 - XYZ, [1632](#)
 - Xyz, [1632](#)
 - Y, [1633](#)
 - Z, [1633](#)
- OpenTK::Toolkit, [1633](#)
 - Init, [1633](#)
- OpenTK::Vector2, [1634](#)
 - Add, [1641](#)
 - BaryCentric, [1641](#), [1642](#)
 - Clamp, [1642](#), [1643](#)
 - ComponentMax, [1643](#)
 - ComponentMin, [1643](#), [1644](#)
 - Div, [1644](#), [1645](#)
 - Divide, [1645](#), [1646](#)
 - Dot, [1646](#)
 - Equals, [1647](#)
 - GetHashCode, [1647](#)
 - Length, [1658](#)
 - LengthFast, [1658](#)
 - LengthSquared, [1659](#)
 - Lerp, [1647](#), [1648](#)
 - Max, [1648](#)
 - Min, [1648](#)
 - Mult, [1649](#)
 - Multiply, [1649](#), [1650](#)
 - Normalize, [1651](#)
 - NormalizeFast, [1651](#), [1652](#)
 - One, [1657](#)
 - operator*, [1652](#), [1653](#)
 - operator+, [1653](#)
 - operator-, [1653](#), [1654](#)
 - operator/, [1654](#)
 - operator==, [1654](#)
 - PerpendicularLeft, [1659](#)
 - PerpendicularRight, [1659](#)
 - Scale, [1654](#), [1655](#)
 - SizeInBytes, [1657](#)
 - Sub, [1655](#), [1656](#)
 - Subtract, [1656](#)
 - ToString, [1656](#)
 - Transform, [1657](#)
 - UnitX, [1657](#)
 - UnitY, [1658](#)
 - Vector2, [1640](#)
 - X, [1658](#)
 - Y, [1658](#)
 - Zero, [1658](#)
- OpenTK::Vector2d, [1659](#)
 - Add, [1665](#), [1666](#)
 - BaryCentric, [1666](#), [1667](#)
 - Clamp, [1667](#)
 - Div, [1668](#)
 - Divide, [1668](#), [1669](#)
 - Dot, [1670](#)
 - Equals, [1670](#), [1671](#)
 - GetHashCode, [1671](#)
 - Length, [1683](#)
 - LengthSquared, [1683](#)
 - Lerp, [1671](#)
 - Max, [1672](#)
 - Min, [1672](#), [1673](#)
 - Mult, [1673](#)
 - Multiply, [1674](#), [1675](#)
 - Normalize, [1675](#), [1676](#)
 - NormalizeFast, [1676](#)
 - One, [1683](#)
 - operator Vector2, [1676](#)

- operator Vector2d, 1676
- operator*, 1677
- operator+, 1678
- operator-, 1678
- operator/, 1679
- operator==, 1679
- PerpendicularLeft, 1684
- PerpendicularRight, 1684
- Scale, 1679, 1680
- SizeInBytes, 1683
- Sub, 1680, 1681
- Subtract, 1681
- ToString, 1682
- Transform, 1682
- UnitX, 1683
- UnitY, 1683
- Vector2d, 1665
- X, 1683
- Y, 1683
- Zero, 1683
- OpenTK::Vector2h, 1684
 - Equals, 1690
 - FromBinaryStream, 1690
 - FromBytes, 1690
 - GetBytes, 1690
 - GetObjectData, 1691
 - operator Vector2, 1691
 - operator Vector2d, 1691
 - operator Vector2h, 1691, 1692
 - SizeInBytes, 1693
 - ToBinaryStream, 1692
 - ToString, 1692
 - ToVector2, 1692
 - ToVector2d, 1692
 - Vector2h, 1687–1689
 - X, 1693
 - Y, 1693
- OpenTK::Vector3, 1693
 - Add, 1701, 1702
 - BaryCentric, 1702, 1703
 - CalculateAngle, 1703
 - Clamp, 1704
 - ComponentMax, 1704, 1705
 - ComponentMin, 1705
 - Cross, 1706
 - Div, 1706, 1707
 - Divide, 1707, 1708
 - Dot, 1708, 1709
 - Equals, 1709
 - GetHashCode, 1709
 - Length, 1725
 - LengthFast, 1725
 - LengthSquared, 1725
 - Lerp, 1710
 - Max, 1710
 - Min, 1711
 - Mult, 1711
 - Multiply, 1712, 1713
 - Normalize, 1713
 - NormalizeFast, 1714
 - One, 1723
 - operator*, 1714, 1715
 - operator+, 1715
 - operator-, 1715, 1716
 - operator/, 1716
 - operator==, 1716
 - Scale, 1717
 - SizeInBytes, 1723
 - Sub, 1717, 1718
 - Subtract, 1718, 1719
 - ToString, 1719
 - Transform, 1719, 1720
 - TransformNormal, 1720, 1721
 - TransformNormalInverse, 1721
 - TransformPerspective, 1722
 - TransformPosition, 1722
 - TransformVector, 1723
 - UnitX, 1724
 - UnitY, 1724
 - UnitZ, 1724
 - Vector3, 1701
 - X, 1724
 - Xy, 1725
 - Y, 1724
 - Z, 1724
 - Zero, 1724
- OpenTK::Vector3d, 1725
 - Add, 1734, 1735
 - BaryCentric, 1735
 - CalculateAngle, 1736
 - Clamp, 1736, 1737
 - ComponentMax, 1737

- ComponentMin, [1738](#)
- Cross, [1738](#), [1739](#)
- Div, [1739](#), [1740](#)
- Divide, [1740](#), [1741](#)
- Dot, [1741](#)
- Equals, [1742](#)
- GetHashCode, [1742](#)
- Length, [1758](#)
- LengthFast, [1758](#)
- LengthSquared, [1759](#)
- Lerp, [1742](#), [1743](#)
- Max, [1743](#)
- Min, [1743](#)
- Mult, [1744](#)
- Multiply, [1744](#), [1745](#)
- Normalize, [1746](#)
- NormalizeFast, [1746](#), [1747](#)
- One, [1757](#)
- operator Vector3, [1747](#)
- operator Vector3d, [1747](#)
- operator*, [1748](#)
- operator+, [1748](#)
- operator-, [1749](#)
- operator/, [1749](#)
- operator==, [1750](#)
- Scale, [1750](#)
- SizeInBytes, [1757](#)
- Sub, [1751](#)
- Subtract, [1752](#)
- ToString, [1752](#)
- Transform, [1752](#), [1753](#)
- TransformNormal, [1754](#)
- TransformNormalInverse, [1754](#), [1755](#)
- TransformPerspective, [1755](#)
- TransformPosition, [1756](#)
- TransformVector, [1756](#), [1757](#)
- UnitX, [1757](#)
- UnitY, [1757](#)
- UnitZ, [1757](#)
- Vector3d, [1733](#), [1734](#)
- X, [1758](#)
- Xy, [1759](#)
- Y, [1758](#)
- Z, [1758](#)
- Zero, [1758](#)
- OpenTK::Vector3h, [1759](#)
 - Equals, [1765](#)
 - FromBinaryStream, [1765](#)
 - FromBytes, [1765](#)
 - GetBytes, [1766](#)
 - GetObjectData, [1766](#)
 - operator Vector3, [1766](#)
 - operator Vector3d, [1767](#)
 - operator Vector3h, [1767](#)
 - SizeInBytes, [1768](#)
 - ToBinaryStream, [1767](#)
 - ToString, [1768](#)
 - ToVector3, [1768](#)
 - ToVector3d, [1768](#)
 - Vector3h, [1762](#)–[1765](#)
 - X, [1768](#)
 - Xy, [1769](#)
 - Y, [1768](#)
 - Z, [1768](#)
- OpenTK::Vector4, [1769](#)
 - Add, [1777](#)
 - BaryCentric, [1777](#), [1778](#)
 - Clamp, [1778](#), [1779](#)
 - Div, [1779](#)
 - Divide, [1780](#)
 - Dot, [1781](#)
 - Equals, [1781](#), [1782](#)
 - GetHashCode, [1782](#)
 - Length, [1795](#)
 - LengthFast, [1796](#)
 - LengthSquared, [1796](#)
 - Lerp, [1782](#)
 - Max, [1783](#)
 - Min, [1783](#), [1784](#)
 - Mult, [1784](#)
 - Multiply, [1785](#), [1786](#)
 - Normalize, [1786](#), [1787](#)
 - NormalizeFast, [1787](#)
 - One, [1794](#)
 - operator float *, [1787](#)
 - operator IntPtr, [1788](#)
 - operator*, [1788](#)
 - operator+, [1789](#)
 - operator-, [1789](#)
 - operator/, [1790](#)
 - operator==, [1790](#)

- Scale, [1790](#), [1791](#)
- SizeInBytes, [1794](#)
- Sub, [1791](#), [1792](#)
- Subtract, [1792](#)
- ToString, [1793](#)
- Transform, [1793](#), [1794](#)
- UnitW, [1794](#)
- UnitX, [1794](#)
- UnitY, [1795](#)
- UnitZ, [1795](#)
- Vector4, [1775](#), [1776](#)
- W, [1795](#)
- X, [1795](#)
- Xy, [1796](#)
- Xyz, [1796](#)
- Y, [1795](#)
- Z, [1795](#)
- Zero, [1795](#)
- OpenTK::Vector4d, [1796](#)
 - Add, [1804](#), [1805](#)
 - BaryCentric, [1805](#), [1806](#)
 - Clamp, [1806](#)
 - Div, [1807](#)
 - Divide, [1807](#), [1808](#)
 - Dot, [1809](#)
 - Equals, [1809](#), [1810](#)
 - GetHashCode, [1810](#)
 - Length, [1824](#)
 - LengthFast, [1824](#)
 - LengthSquared, [1824](#)
 - Lerp, [1810](#)
 - Max, [1811](#)
 - Min, [1811](#), [1812](#)
 - Mult, [1812](#)
 - Multiply, [1813](#), [1814](#)
 - Normalize, [1814](#)
 - NormalizeFast, [1815](#)
 - One, [1822](#)
 - operator double *, [1815](#)
 - operator IntPtr, [1815](#)
 - operator Vector4, [1816](#)
 - operator Vector4d, [1816](#)
 - operator*, [1817](#)
 - operator+, [1817](#)
 - operator-, [1818](#)
 - operator/, [1818](#)
 - operator==, [1818](#)
 - Scale, [1819](#)
 - SizeInBytes, [1822](#)
 - Sub, [1819](#), [1820](#)
 - Subtract, [1820](#), [1821](#)
 - ToString, [1821](#)
 - Transform, [1821](#), [1822](#)
 - UnitW, [1823](#)
 - UnitX, [1823](#)
 - UnitY, [1823](#)
 - UnitZ, [1823](#)
 - Vector4d, [1803](#), [1804](#)
 - W, [1823](#)
 - X, [1823](#)
 - Xy, [1824](#)
 - Xyz, [1824](#)
 - Y, [1823](#)
 - Z, [1823](#)
 - Zero, [1823](#)
- OpenTK::Vector4h, [1825](#)
 - Equals, [1831](#)
 - FromBinaryStream, [1831](#)
 - FromBytes, [1831](#)
 - GetBytes, [1831](#)
 - GetObjectData, [1832](#)
 - operator Vector4, [1832](#)
 - operator Vector4d, [1832](#)
 - operator Vector4h, [1832](#), [1833](#)
 - SizeInBytes, [1834](#)
 - ToBinaryStream, [1833](#)
 - ToString, [1833](#)
 - ToVector4, [1833](#)
 - ToVector4d, [1833](#)
 - Vector4h, [1828–1830](#)
 - W, [1834](#)
 - X, [1834](#)
 - Xy, [1834](#)
 - Xyz, [1834](#)
 - Y, [1834](#)
 - Z, [1834](#)
- operator Color4
 - OpenTK::Graphics::Color4, [406](#)
- operator ColorFormat
 - OpenTK::Graphics::ColorFormat, [426](#)
- operator ContextHandle

- OpenTK::ContextHandle, 364
- operator double
 - OpenTK::Half, 1487
- operator double *
 - OpenTK::Vector4d, 1815
- operator float
 - OpenTK::Half, 1487
- operator float *
 - OpenTK::Vector4, 1787
- operator Half
 - OpenTK::Half, 1487
- operator IntPtr
 - OpenTK::ContextHandle, 365
 - OpenTK::Vector4, 1788
 - OpenTK::Vector4d, 1815
- operator System.Drawing.Color
 - OpenTK::Graphics::Color4, 406
- operator Vector2
 - OpenTK::Vector2d, 1676
 - OpenTK::Vector2h, 1691
- operator Vector2d
 - OpenTK::Vector2d, 1676
 - OpenTK::Vector2h, 1691
- operator Vector2h
 - OpenTK::Vector2h, 1691, 1692
- operator Vector3
 - OpenTK::Vector3d, 1747
 - OpenTK::Vector3h, 1766
- operator Vector3d
 - OpenTK::Vector3d, 1747
 - OpenTK::Vector3h, 1767
- operator Vector3h
 - OpenTK::Vector3h, 1767
- operator Vector4
 - OpenTK::Vector4d, 1816
 - OpenTK::Vector4h, 1832
- operator Vector4d
 - OpenTK::Vector4d, 1816
 - OpenTK::Vector4h, 1832
- operator Vector4h
 - OpenTK::Vector4h, 1832, 1833
- operator *
 - OpenTK::Matrix4, 1550
 - OpenTK::Matrix4d, 1576
 - OpenTK::Quaternion, 1613, 1614
 - OpenTK::Quaterniond, 1628, 1629
 - OpenTK::Vector2, 1652, 1653
 - OpenTK::Vector2d, 1677
 - OpenTK::Vector3, 1714, 1715
 - OpenTK::Vector3d, 1748
 - OpenTK::Vector4, 1788
 - OpenTK::Vector4d, 1817
- operator +
 - OpenTK::Quaternion, 1614
 - OpenTK::Quaterniond, 1629
 - OpenTK::Vector2, 1653
 - OpenTK::Vector2d, 1678
 - OpenTK::Vector3, 1715
 - OpenTK::Vector3d, 1748
 - OpenTK::Vector4, 1789
 - OpenTK::Vector4d, 1817
- operator -
 - OpenTK::Quaternion, 1615
 - OpenTK::Quaterniond, 1629
 - OpenTK::Vector2, 1653, 1654
 - OpenTK::Vector2d, 1678
 - OpenTK::Vector3, 1715, 1716
 - OpenTK::Vector3d, 1749
 - OpenTK::Vector4, 1789
 - OpenTK::Vector4d, 1818
- operator /
 - OpenTK::Vector2, 1654
 - OpenTK::Vector2d, 1679
 - OpenTK::Vector3, 1716
 - OpenTK::Vector3d, 1749
 - OpenTK::Vector4, 1790
 - OpenTK::Vector4d, 1818
- operator ==
 - OpenTK::ContextHandle, 365
 - OpenTK::DisplayResolution, 373
 - OpenTK::Graphics::Color4, 407
 - OpenTK::Graphics::ColorFormat, 426
 - OpenTK::Matrix4, 1551
 - OpenTK::Matrix4d, 1576
 - OpenTK::Quaternion, 1615
 - OpenTK::Quaterniond, 1630
 - OpenTK::Vector2, 1654
 - OpenTK::Vector2d, 1679
 - OpenTK::Vector3, 1716
 - OpenTK::Vector3d, 1750
 - OpenTK::Vector4, 1790

- OpenTK::Vector4d, [1818](#)
- Orange
 - OpenTK::Graphics::Color4, [419](#)
- OrangeRed
 - OpenTK::Graphics::Color4, [419](#)
- Orchid
 - OpenTK::Graphics::Color4, [419](#)
- Orientation
 - OpenTK::Audio::OpenAL, [28](#)
- Ortho
 - OpenTK::Graphics::ES10::GL, [476](#)
 - OpenTK::Graphics::ES11::GL, [585](#)
 - OpenTK::Graphics::OpenGL::GL, [1286](#)
- OverlayIndex
 - OpenTK::Platform::Windows, [279](#)
- P
 - OpenTK::Input, [252](#)
- PageDown
 - OpenTK::Input, [251](#)
- PAGEONLY
 - OpenTK::Platform::Windows, [276](#)
- PageUp
 - OpenTK::Input, [251](#)
- PAINT
 - OpenTK::Platform::Windows, [276](#)
- PaleGoldenrod
 - OpenTK::Graphics::Color4, [419](#)
- PaleGreen
 - OpenTK::Graphics::Color4, [419](#)
- PaleTurquoise
 - OpenTK::Graphics::Color4, [419](#)
- PaleVioletRed
 - OpenTK::Graphics::Color4, [419](#)
- PapayaWhip
 - OpenTK::Graphics::Color4, [419](#)
- Parallel
 - OpenTK::BezierCurve, [349](#)
 - OpenTK::BezierCurveCubic, [352](#)
 - OpenTK::BezierCurveQuadric, [355](#)
- Parse
 - OpenTK::Half, [1488](#)
- PassThrough
 - OpenTK::Graphics::OpenGL::GL, [1287](#)
- Pause
 - OpenTK::Input, [251](#)
- Paused
 - OpenTK::Audio::OpenAL, [31](#)
- PeachPuff
 - OpenTK::Graphics::Color4, [420](#)
- Pending
 - OpenTK::Audio::OpenAL, [20](#)
- Period
 - OpenTK::Input, [253](#)
- PerpendicularLeft
 - OpenTK::Vector2, [1659](#)
 - OpenTK::Vector2d, [1684](#)
- PerpendicularRight
 - OpenTK::Vector2, [1659](#)
 - OpenTK::Vector2d, [1684](#)
- Perspective
 - OpenTK::Matrix4, [1551](#)
 - OpenTK::Matrix4d, [1577](#)
- Peru
 - OpenTK::Graphics::Color4, [420](#)
- PIDL
 - OpenTK::Platform::Windows, [279](#)
- Pink
 - OpenTK::Graphics::Color4, [420](#)
- Pitch
 - OpenTK::Audio::OpenAL, [29](#)
- PitchShifter
 - OpenTK::Audio::OpenAL, [40](#)
- PitchShifterCoarseTune
 - OpenTK::Audio::OpenAL, [39](#)
- PitchShifterFineTune
 - OpenTK::Audio::OpenAL, [39](#)
- PixelMap
 - OpenTK::Graphics::OpenGL::GL, [1287–1292](#)
- PixelStore
 - OpenTK::Graphics::ES10::GL, [476](#)
 - OpenTK::Graphics::ES11::GL, [585](#)
 - OpenTK::Graphics::ES20::GL, [729](#)
 - OpenTK::Graphics::OpenGL::GL, [1293](#)
- PixelTransfer
 - OpenTK::Graphics::OpenGL::GL, [1294](#)
- PixelZoom

- OpenTK::Graphics::OpenGL::GL, [1295](#)
- PlatformException
 - OpenTK::PlatformException, [1603](#)
- Playing
 - OpenTK::Audio::OpenAL, [31](#)
- Plum
 - OpenTK::Graphics::Color4, [420](#)
- Plus
 - OpenTK::Input, [253](#)
- PointParameter
 - OpenTK::Graphics::ES11::GL, [586](#)
 - OpenTK::Graphics::OpenGL::GL, [1295–1297](#)
- Points
 - OpenTK::BezierCurve, [349](#)
- PointSize
 - OpenTK::Graphics::ES10::GL, [477](#)
 - OpenTK::Graphics::ES11::GL, [587](#)
 - OpenTK::Graphics::OpenGL::GL, [1297](#)
- PointToClient
 - OpenTK::INativeWindow, [1495](#)
 - OpenTK::NativeWindow, [1594](#)
- PointToScreen
 - OpenTK::INativeWindow, [1495](#)
 - OpenTK::NativeWindow, [1594](#)
- Poll
 - OpenTK::Input::IInputDriver, [1503](#)
- PolygonMode
 - OpenTK::Graphics::OpenGL::GL, [1297](#)
- PolygonOffset
 - OpenTK::Graphics::ES10::GL, [477](#)
 - OpenTK::Graphics::ES11::GL, [587](#)
 - OpenTK::Graphics::ES20::GL, [730](#)
 - OpenTK::Graphics::OpenGL::GL, [1298](#)
- PolygonStipple
 - OpenTK::Graphics::OpenGL::GL, [1298, 1299](#)
- Position
 - OpenTK::Audio::OpenAL, [27, 28](#)
 - OpenTK::Input::MouseEventArgs, [1526](#)
- POSTMESSAGE
 - OpenTK::Platform::Windows, [276](#)
- PowderBlue
 - OpenTK::Graphics::Color4, [420](#)
- Pressed
 - OpenTK::Input::JoystickButtonEventArgs, [1508](#)
- PrintScreen
 - OpenTK::Input, [251](#)
- PrioritizeTextures
 - OpenTK::Graphics::OpenGL::GL, [1299, 1300](#)
- Process
 - OpenTK::Audio::AudioContext, [306](#)
- ProcessEvent
 - OpenTK::INativeWindow, [1495](#)
 - OpenTK::NativeWindow, [1595](#)
- Purple
 - OpenTK::Graphics::Color4, [420](#)
- PushAttrib
 - OpenTK::Graphics::OpenGL::GL, [1301](#)
- PushClientAttrib
 - OpenTK::Graphics::OpenGL::GL, [1301](#)
- PushMatrix
 - OpenTK::Graphics::ES10::GL, [477](#)
 - OpenTK::Graphics::ES11::GL, [587](#)
 - OpenTK::Graphics::OpenGL::GL, [1301](#)
- PushName
 - OpenTK::Graphics::OpenGL::GL, [1301, 1302](#)
- Q
 - OpenTK::Input, [252](#)
- Quaternion
 - OpenTK::Quaternion, [1607, 1608](#)
- Quaterniond
 - OpenTK::Quaterniond, [1622](#)
- QueueStatusFlags
 - OpenTK::Platform::Windows, [275](#)
- Quote
 - OpenTK::Input, [253](#)
- R
 - OpenTK::Graphics::Color4, [408](#)

- OpenTK::Input, [252](#)
- RAlt
 - OpenTK::Input, [249](#)
- RasterPos2
 - OpenTK::Graphics::OpenGL::GL, [1302–1305](#)
- RasterPos3
 - OpenTK::Graphics::OpenGL::GL, [1305–1308](#)
- RasterPos4
 - OpenTK::Graphics::OpenGL::GL, [1308–1311](#)
- RAWINPUT
 - OpenTK::Platform::Windows, [276](#)
- RawInputDeviceFlags
 - OpenTK::Platform::Windows, [276](#)
- RawMouseFlags
 - OpenTK::Platform::Windows, [277](#)
- RBracket
 - OpenTK::Input, [253](#)
- RControl
 - OpenTK::Input, [249](#)
- ReadBuffer
 - OpenTK::Graphics::OpenGL::GL, [1311](#)
- ReadPixels
 - OpenTK::Graphics::ES10::GL, [477](#)
 - OpenTK::Graphics::ES11::GL, [587](#)
 - OpenTK::Graphics::ES20::GL, [730](#)
 - OpenTK::Graphics::OpenGL::GL, [1312](#)
- ReadPixels< T6 >
 - OpenTK::Graphics::ES10::GL, [478–480](#)
 - OpenTK::Graphics::ES11::GL, [588–590](#)
 - OpenTK::Graphics::ES20::GL, [731–733](#)
 - OpenTK::Graphics::OpenGL::GL, [1312–1315](#)
- ReadSamples
 - OpenTK::Audio::AudioCapture, [297](#)
- ReadSamples< TBuffer >
 - OpenTK::Audio::AudioCapture, [298](#)
- RebuildExtensionList
 - OpenTK::BindingsBase, [357](#)
- Rect
 - OpenTK::Graphics::OpenGL::GL, [1316–1319](#)
- Red
 - OpenTK::Graphics::Color4, [420](#)
 - OpenTK::Graphics::ColorFormat, [427](#)
- ReferenceDistance
 - OpenTK::Audio::OpenAL, [29](#)
- Refresh
 - OpenTK::Audio::OpenAL, [21](#)
- RefreshRate
 - OpenTK::DisplayDevice, [370](#)
 - OpenTK::DisplayResolution, [373](#)
- REMOVE
 - OpenTK::Platform::Windows, [276](#)
- Renderer
 - OpenTK::Audio::OpenAL, [27](#)
 - OpenTK::Graphics::GraphicsContextVersion, [791](#)
- RenderFrame
 - OpenTK::GameWindow, [388](#)
 - OpenTK::Platform::IGameWindow, [1602](#)
- RenderFrequency
 - OpenTK::GameWindow, [386](#)
- RenderMode
 - OpenTK::Graphics::OpenGL::GL, [1319](#)
- RenderPeriod
 - OpenTK::GameWindow, [386](#)
- RenderTime
 - OpenTK::GameWindow, [386](#)
- ResetHistogram
 - OpenTK::Graphics::OpenGL::GL, [1319](#)
- ResetMinmax
 - OpenTK::Graphics::OpenGL::GL, [1320](#)
- Resize
 - OpenTK::INativeWindow, [1500](#)
 - OpenTK::NativeWindow, [1599](#)
- RESTORE
 - OpenTK::Platform::Windows, [279](#)
- RestoreResolution
 - OpenTK::DisplayDevice, [368](#)

- Reverb
 - OpenTK::Audio::OpenAL, [40](#)
- ReverbAirAbsorptionGainHF
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbDecayHFLimit
 - OpenTK::Audio::OpenAL, [39](#)
- ReverbDecayHFRatio
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbDecayTime
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbDensity
 - OpenTK::Audio::OpenAL, [32](#)
- ReverbDiffusion
 - OpenTK::Audio::OpenAL, [32](#)
- ReverbGain
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbGainHF
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbLateReverbDelay
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbLateReverbGain
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbReflectionsDelay
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbReflectionsGain
 - OpenTK::Audio::OpenAL, [33](#)
- ReverbRoomRolloffFactor
 - OpenTK::Audio::OpenAL, [34](#)
- Right
 - OpenTK::Box2, [360](#)
 - OpenTK::Input, [250](#), [253](#)
- RingModulator
 - OpenTK::Audio::OpenAL, [40](#)
- RingModulatorFrequency
 - OpenTK::Audio::OpenAL, [35](#)
- RingModulatorHighpassCutoff
 - OpenTK::Audio::OpenAL, [35](#)
- RingModulatorWaveform
 - OpenTK::Audio::OpenAL, [39](#)
- RolloffFactor
 - OpenTK::Audio::OpenAL, [29](#)
- RosyBrown
 - OpenTK::Graphics::Color4, [420](#)
- Rotate
 - OpenTK::Graphics::ES10::GL, [481](#)
 - OpenTK::Graphics::ES11::GL, [591](#)
 - OpenTK::Graphics::OpenGL::GL, [1320](#)
 - OpenTK::Matrix4, [1551](#), [1552](#)
 - OpenTK::Matrix4d, [1577](#)
- RotateX
 - OpenTK::Matrix4, [1552](#)
 - OpenTK::Matrix4d, [1578](#)
- RotateY
 - OpenTK::Matrix4, [1552](#)
 - OpenTK::Matrix4d, [1578](#)
- RotateZ
 - OpenTK::Matrix4, [1553](#)
 - OpenTK::Matrix4d, [1578](#)
- Row0
 - OpenTK::Matrix4, [1555](#)
 - OpenTK::Matrix4d, [1581](#)
- Row1
 - OpenTK::Matrix4, [1555](#)
 - OpenTK::Matrix4d, [1582](#)
- Row2
 - OpenTK::Matrix4, [1555](#)
 - OpenTK::Matrix4d, [1582](#)
- Row3
 - OpenTK::Matrix4, [1556](#)
 - OpenTK::Matrix4d, [1582](#)
- RoyalBlue
 - OpenTK::Graphics::Color4, [420](#)
- RShift
 - OpenTK::Input, [249](#)
- Run
 - OpenTK::GameWindow, [384](#), [385](#)
 - OpenTK::Platform::IGameWindow, [1601](#)
- RWin
 - OpenTK::Input, [249](#)
- S
 - OpenTK::Input, [252](#)
- SaddleBrown
 - OpenTK::Graphics::Color4, [420](#)
- Salmon
 - OpenTK::Graphics::Color4, [421](#)
- SampleCoverage
 - OpenTK::Graphics::ES10::GL, [481](#)
 - OpenTK::Graphics::ES11::GL, [591](#)
 - OpenTK::Graphics::ES20::GL, [734](#)

- OpenTK::Graphics::OpenGL::GL, [1320](#)
- SampleFormat
 - OpenTK::Audio::AudioCapture, [299](#)
- SampleFrequency
 - OpenTK::Audio::AudioCapture, [299](#)
- SampleOffset
 - OpenTK::Audio::OpenAL, [26](#), [30](#)
- Samples
 - OpenTK::Graphics::GraphicsMode, [799](#)
- SandyBrown
 - OpenTK::Graphics::Color4, [421](#)
- Scale
 - OpenTK::Graphics::ES10::GL, [482](#)
 - OpenTK::Graphics::ES11::GL, [592](#)
 - OpenTK::Graphics::OpenGL::GL, [1321](#)
 - OpenTK::Matrix4, [1553](#)
 - OpenTK::Matrix4d, [1579](#)
 - OpenTK::Vector2, [1654](#), [1655](#)
 - OpenTK::Vector2d, [1679](#), [1680](#)
 - OpenTK::Vector3, [1717](#)
 - OpenTK::Vector3d, [1750](#)
 - OpenTK::Vector4, [1790](#), [1791](#)
 - OpenTK::Vector4d, [1819](#)
- ScanCodeToVirtualKey
 - OpenTK::Platform::Windows, [275](#)
- ScanCodeToVirtualKeyExtended
 - OpenTK::Platform::Windows, [275](#)
- Scissor
 - OpenTK::Graphics::ES10::GL, [482](#)
 - OpenTK::Graphics::ES11::GL, [592](#)
 - OpenTK::Graphics::ES20::GL, [734](#)
 - OpenTK::Graphics::OpenGL::GL, [1321](#)
- ScrollLock
 - OpenTK::Input, [251](#)
- SeaGreen
 - OpenTK::Graphics::Color4, [421](#)
- SeaShell
 - OpenTK::Graphics::Color4, [421](#)
- SecOffset
 - OpenTK::Audio::OpenAL, [30](#)
- SecondaryColor3
 - OpenTK::Graphics::OpenGL::GL, [1321–1328](#)
- SecondaryColorPointer
 - OpenTK::Graphics::OpenGL::GL, [1328](#)
- SecondaryColorPointer< T3 >
 - OpenTK::Graphics::OpenGL::GL, [1328–1330](#)
- SecondControlPoint
 - OpenTK::BezierCurveCubic, [352](#)
- SelectBuffer
 - OpenTK::Graphics::OpenGL::GL, [1331](#), [1332](#)
- Selected
 - OpenTK::Platform::Windows, [279](#)
- SelectResolution
 - OpenTK::DisplayDevice, [369](#)
- Semicolon
 - OpenTK::Input, [253](#)
- SENDMESSAGE
 - OpenTK::Platform::Windows, [276](#)
- SeparableFilter2D
 - OpenTK::Graphics::OpenGL::GL, [1332](#)
- SeparableFilter2D< T6, T7 >
 - OpenTK::Graphics::OpenGL::GL, [1333–1335](#), [1337](#)
- SeparableFilter2D< T7 >
 - OpenTK::Graphics::OpenGL::GL, [1338–1341](#)
- SetBufferMode
 - OpenTK::Audio::OpenAL::XRamExtension, [342](#), [343](#)
- SetWindowPosFlags
 - OpenTK::Platform::Windows, [277](#)
- ShadeModel
 - OpenTK::Graphics::ES10::GL, [482](#)
 - OpenTK::Graphics::ES11::GL, [592](#)
 - OpenTK::Graphics::OpenGL::GL, [1343](#)
- ShaderSource
 - OpenTK::Graphics::ES20::GL, [735](#), [736](#)
 - OpenTK::Graphics::OpenGL::GL, [1343](#), [1344](#)
- ShareContexts

- OpenTK::Graphics::GraphicsContext,
788
- ShellIconSize
 - OpenTK::Platform::Windows, 279
- ShGetFileIconFlags
 - OpenTK::Platform::Windows, 278
- ShiftLeft
 - OpenTK::Input, 249
- ShiftRight
 - OpenTK::Input, 249
- SHOW
 - OpenTK::Platform::Windows, 279
- SHOWDEFAULT
 - OpenTK::Platform::Windows, 279
- SHOWMAXIMIZED
 - OpenTK::Platform::Windows, 279
- SHOWMINIMIZED
 - OpenTK::Platform::Windows, 279
- SHOWMINNOACTIVE
 - OpenTK::Platform::Windows, 279
- SHOWNA
 - OpenTK::Platform::Windows, 279
- SHOWNOACTIVATE
 - OpenTK::Platform::Windows, 279
- SHOWNORMAL
 - OpenTK::Platform::Windows, 279
- SHOWWINDOW
 - OpenTK::Platform::Windows, 278
- ShowWindowCommand
 - OpenTK::Platform::Windows, 279
- ShowWindowMessageIdentifiers
 - OpenTK::Platform::Windows, 280
- Sienna
 - OpenTK::Graphics::Color4, 421
- Silver
 - OpenTK::Graphics::Color4, 421
- Size
 - OpenTK::Audio::OpenAL, 26
 - OpenTK::INativeWindow, 1497
 - OpenTK::NativeWindow, 1597
- SizeInBytes
 - OpenTK::Half, 1491
 - OpenTK::Vector2, 1657
 - OpenTK::Vector2d, 1683
 - OpenTK::Vector2h, 1693
 - OpenTK::Vector3, 1723
 - OpenTK::Vector3d, 1757
 - OpenTK::Vector3h, 1768
 - OpenTK::Vector4, 1794
 - OpenTK::Vector4d, 1822
 - OpenTK::Vector4h, 1834
- SkyBlue
 - OpenTK::Graphics::Color4, 421
- Slash
 - OpenTK::Input, 253
- SlateBlue
 - OpenTK::Graphics::Color4, 421
- SlateGray
 - OpenTK::Graphics::Color4, 421
- Sleep
 - OpenTK::Input, 251
- Slerp
 - OpenTK::Quaternion, 1615
 - OpenTK::Quaterniond, 1630
- SmallIcon
 - OpenTK::Platform::Windows, 279
- Snow
 - OpenTK::Graphics::Color4, 421
- SourceRelative
 - OpenTK::Audio::OpenAL, 29
- SourceState
 - OpenTK::Audio::OpenAL, 26
- SourceType
 - OpenTK::Audio::OpenAL, 30
- Space
 - OpenTK::Input, 251
- SpringGreen
 - OpenTK::Graphics::Color4, 422
- Start
 - OpenTK::Audio::AudioCapture, 298
- StartAnchor
 - OpenTK::BezierCurveCubic, 352
 - OpenTK::BezierCurveQuadric, 355
- Static
 - OpenTK::Audio::OpenAL, 31
- SteelBlue
 - OpenTK::Graphics::Color4, 422
- Stencil
 - OpenTK::Graphics::GraphicsMode,
799
- StencilFunc

- OpenTK::Graphics::ES10::GL, [482](#), [483](#)
- OpenTK::Graphics::ES11::GL, [592](#), [593](#)
- OpenTK::Graphics::ES20::GL, [737](#)
- OpenTK::Graphics::OpenGL::GL, [1344](#), [1345](#)
- StencilFuncSeparate
 - OpenTK::Graphics::ES20::GL, [738](#)
 - OpenTK::Graphics::OpenGL::GL, [1345](#), [1346](#)
- StencilMask
 - OpenTK::Graphics::ES10::GL, [483](#)
 - OpenTK::Graphics::ES11::GL, [593](#)
 - OpenTK::Graphics::ES20::GL, [739](#)
 - OpenTK::Graphics::OpenGL::GL, [1346](#)
- StencilMaskSeparate
 - OpenTK::Graphics::ES20::GL, [739](#)
 - OpenTK::Graphics::OpenGL::GL, [1347](#)
- StencilOp
 - OpenTK::Graphics::ES10::GL, [484](#)
 - OpenTK::Graphics::ES11::GL, [594](#)
 - OpenTK::Graphics::ES20::GL, [740](#)
 - OpenTK::Graphics::OpenGL::GL, [1347](#)
- StencilOpSeparate
 - OpenTK::Graphics::ES20::GL, [740](#)
 - OpenTK::Graphics::OpenGL::GL, [1348](#)
- Stereo
 - OpenTK::Graphics::GraphicsMode, [799](#)
- Stereo8
 - OpenTK::Audio::OpenAL, [24](#)
- StereoALawExt
 - OpenTK::Audio::OpenAL, [24](#)
- StereoDoubleExt
 - OpenTK::Audio::OpenAL, [25](#)
- StereoFloat32Ext
 - OpenTK::Audio::OpenAL, [24](#)
- StereoIma4Ext
 - OpenTK::Audio::OpenAL, [24](#)
- StereoMuLawExt
 - OpenTK::Audio::OpenAL, [24](#)
- Stop
 - OpenTK::Audio::AudioCapture, [298](#)
- Streaming
 - OpenTK::Audio::OpenAL, [31](#)
- Sub
 - OpenTK::Quaternion, [1616](#)
 - OpenTK::Quaterniond, [1630](#), [1631](#)
 - OpenTK::Vector2, [1655](#), [1656](#)
 - OpenTK::Vector2d, [1680](#), [1681](#)
 - OpenTK::Vector3, [1717](#), [1718](#)
 - OpenTK::Vector3d, [1751](#)
 - OpenTK::Vector4, [1791](#), [1792](#)
 - OpenTK::Vector4d, [1819](#), [1820](#)
- Subtract
 - OpenTK::Vector2, [1656](#)
 - OpenTK::Vector2d, [1681](#)
 - OpenTK::Vector3, [1718](#), [1719](#)
 - OpenTK::Vector3d, [1752](#)
 - OpenTK::Vector4, [1792](#)
 - OpenTK::Vector4d, [1820](#), [1821](#)
- SupportsExtension
 - OpenTK::Audio::AudioContext, [306](#)
- Suspend
 - OpenTK::Audio::AudioContext, [307](#)
- SwapBuffers
 - OpenTK::GameWindow, [385](#)
 - OpenTK::GLControl, [392](#)
 - OpenTK::Graphics::GraphicsContext, [787](#)
 - OpenTK::Graphics::IGraphicsContext, [802](#)
 - OpenTK::Platform::IGameWindow, [1602](#)
- Sync
 - OpenTK::Audio::OpenAL, [21](#)
- SyncRoot
 - OpenTK::BindingsBase, [357](#)
 - OpenTK::Graphics::ES10::GL, [505](#)
 - OpenTK::Graphics::ES11::GL, [619](#)
 - OpenTK::Graphics::ES20::GL, [780](#)
 - OpenTK::Graphics::OpenGL::GL, [1479](#)
- SysIconIndex
 - OpenTK::Platform::Windows, [278](#)
- T

- OpenTK::Input, [252](#)
- Tab
 - OpenTK::Input, [251](#)
- Tan
 - OpenTK::Graphics::Color4, [422](#)
- TargetRenderFrequency
 - OpenTK::GameWindow, [386](#)
- TargetRenderPeriod
 - OpenTK::GameWindow, [386](#)
- TargetUpdateFrequency
 - OpenTK::GameWindow, [386](#)
- TargetUpdatePeriod
 - OpenTK::GameWindow, [387](#)
- Teal
 - OpenTK::Graphics::Color4, [422](#)
- TexCoord1
 - OpenTK::Graphics::OpenGL::GL, [1348–1350](#)
- TexCoord2
 - OpenTK::Graphics::OpenGL::GL, [1350–1354](#)
- TexCoord3
 - OpenTK::Graphics::OpenGL::GL, [1354–1358](#)
- TexCoord4
 - OpenTK::Graphics::OpenGL::GL, [1358–1362](#)
- TexCoordPointer
 - OpenTK::Graphics::ES10::GL, [484](#)
 - OpenTK::Graphics::ES11::GL, [594](#)
 - OpenTK::Graphics::OpenGL::GL, [1362](#)
- TexCoordPointer< T3 >
 - OpenTK::Graphics::ES10::GL, [485](#), [486](#)
 - OpenTK::Graphics::ES11::GL, [594–596](#)
 - OpenTK::Graphics::OpenGL::GL, [1363](#), [1364](#)
- TexEnv
 - OpenTK::Graphics::ES10::GL, [487](#), [488](#)
 - OpenTK::Graphics::ES11::GL, [596–600](#)
 - OpenTK::Graphics::OpenGL::GL, [1365–1368](#)
- TexGen
 - OpenTK::Graphics::OpenGL::GL, [1369–1372](#)
- TexImage1D
 - OpenTK::Graphics::OpenGL::GL, [1372](#)
- TexImage1D< T7 >
 - OpenTK::Graphics::OpenGL::GL, [1374](#), [1375](#), [1377](#), [1378](#)
- TexImage2D
 - OpenTK::Graphics::ES10::GL, [489](#)
 - OpenTK::Graphics::ES11::GL, [600](#)
 - OpenTK::Graphics::ES20::GL, [741](#)
 - OpenTK::Graphics::OpenGL::GL, [1379](#)
- TexImage2D< T8 >
 - OpenTK::Graphics::ES10::GL, [490](#), [492](#), [493](#), [495](#)
 - OpenTK::Graphics::ES11::GL, [602](#), [603](#), [605](#), [606](#)
 - OpenTK::Graphics::ES20::GL, [742](#), [744](#), [746](#), [747](#)
 - OpenTK::Graphics::OpenGL::GL, [1381](#), [1382](#), [1384](#), [1386](#)
- TexImage3D
 - OpenTK::Graphics::OpenGL::GL, [1387](#)
- TexImage3D< T9 >
 - OpenTK::Graphics::OpenGL::GL, [1389](#), [1390](#), [1392](#), [1393](#)
- TexParameter
 - OpenTK::Graphics::ES10::GL, [496](#)
 - OpenTK::Graphics::ES11::GL, [608–610](#)
 - OpenTK::Graphics::ES20::GL, [749–751](#)
 - OpenTK::Graphics::OpenGL::GL, [1395–1397](#)
- TexSubImage1D
 - OpenTK::Graphics::OpenGL::GL, [1398](#)
- TexSubImage1D< T6 >
 - OpenTK::Graphics::OpenGL::GL, [1398–1401](#)
- TexSubImage2D
 - OpenTK::Graphics::ES10::GL, [497](#)

- OpenTK::Graphics::ES11::GL, [611](#)
- OpenTK::Graphics::ES20::GL, [752](#)
- OpenTK::Graphics::OpenGL::GL, [1402](#)
- TexSubImage2D< T8 >
 - OpenTK::Graphics::ES10::GL, [498](#)–[500](#)
 - OpenTK::Graphics::ES11::GL, [612](#)–[615](#)
 - OpenTK::Graphics::ES20::GL, [753](#)–[755](#)
 - OpenTK::Graphics::OpenGL::GL, [1403](#)–[1406](#)
- TexSubImage3D
 - OpenTK::Graphics::OpenGL::GL, [1406](#)
- TexSubImage3D< T10 >
 - OpenTK::Graphics::OpenGL::GL, [1407](#)–[1410](#)
- this
 - OpenTK::Input::JoystickAxisCollection, [1506](#)
 - OpenTK::Input::JoystickButtonCollection, [1507](#)
 - OpenTK::Input::KeyboardDevice, [1516](#)
 - OpenTK::Input::MouseDevice, [1523](#)
- Thistle
 - OpenTK::Graphics::Color4, [422](#)
- Three
 - OpenTK::Audio::AudioContext, [302](#)
- Tilde
 - OpenTK::Input, [253](#)
- Time
 - OpenTK::FrameEventArgs, [375](#)
- TIMER
 - OpenTK::Platform::Windows, [276](#)
- Title
 - OpenTK::INativeWindow, [1497](#)
 - OpenTK::NativeWindow, [1597](#)
- TitleChanged
 - OpenTK::INativeWindow, [1500](#)
 - OpenTK::NativeWindow, [1599](#)
- ToArgb
 - OpenTK::Graphics::Color4, [407](#)
- ToAxisAngle
 - OpenTK::Quaternion, [1616](#), [1617](#)
 - OpenTK::Quaterniond, [1631](#)
- ToBinaryStream
 - OpenTK::Half, [1488](#)
 - OpenTK::Vector2h, [1692](#)
 - OpenTK::Vector3h, [1767](#)
 - OpenTK::Vector4h, [1833](#)
- Tomato
 - OpenTK::Graphics::Color4, [422](#)
- Top
 - OpenTK::Box2, [360](#)
- ToSingle
 - OpenTK::Half, [1489](#)
- ToString
 - OpenTK::Audio::AudioContext, [307](#)
 - OpenTK::Box2, [360](#)
 - OpenTK::ContextHandle, [366](#)
 - OpenTK::DisplayDevice, [369](#)
 - OpenTK::DisplayResolution, [373](#)
 - OpenTK::Graphics::Color4, [408](#)
 - OpenTK::Graphics::ColorFormat, [427](#)
 - OpenTK::Graphics::GraphicsMode, [798](#)
 - OpenTK::Half, [1489](#)
 - OpenTK::Input::KeyboardDevice, [1514](#)
 - OpenTK::Input::MouseDevice, [1523](#)
 - OpenTK::Matrix4, [1554](#)
 - OpenTK::Matrix4d, [1580](#)
 - OpenTK::Quaternion, [1617](#)
 - OpenTK::Quaterniond, [1631](#)
 - OpenTK::Vector2, [1656](#)
 - OpenTK::Vector2d, [1682](#)
 - OpenTK::Vector2h, [1692](#)
 - OpenTK::Vector3, [1719](#)
 - OpenTK::Vector3d, [1752](#)
 - OpenTK::Vector3h, [1768](#)
 - OpenTK::Vector4, [1793](#)
 - OpenTK::Vector4d, [1821](#)
 - OpenTK::Vector4h, [1833](#)
- ToVector2
 - OpenTK::Vector2h, [1692](#)
- ToVector2d
 - OpenTK::Vector2h, [1692](#)
- ToVector3

- OpenTK::Vector3h, [1768](#)
- ToVector3d
 - OpenTK::Vector3h, [1768](#)
- ToVector4
 - OpenTK::Vector4h, [1833](#)
- ToVector4d
 - OpenTK::Vector4h, [1833](#)
- Transform
 - OpenTK::Vector2, [1657](#)
 - OpenTK::Vector2d, [1682](#)
 - OpenTK::Vector3, [1719](#), [1720](#)
 - OpenTK::Vector3d, [1752](#), [1753](#)
 - OpenTK::Vector4, [1793](#), [1794](#)
 - OpenTK::Vector4d, [1821](#), [1822](#)
- TransformNormal
 - OpenTK::Vector3, [1720](#), [1721](#)
 - OpenTK::Vector3d, [1754](#)
- TransformNormalInverse
 - OpenTK::Vector3, [1721](#)
 - OpenTK::Vector3d, [1754](#), [1755](#)
- TransformPerspective
 - OpenTK::Vector3, [1722](#)
 - OpenTK::Vector3d, [1755](#)
- TransformPosition
 - OpenTK::Vector3, [1722](#)
 - OpenTK::Vector3d, [1756](#)
- TransformVector
 - OpenTK::Vector3, [1723](#)
 - OpenTK::Vector3d, [1756](#), [1757](#)
- Translate
 - OpenTK::Graphics::ES10::GL, [501](#)
 - OpenTK::Graphics::ES11::GL, [616](#)
 - OpenTK::Graphics::OpenGL::GL, [1411](#), [1412](#)
- Translation
 - OpenTK::Matrix4, [1554](#)
 - OpenTK::Matrix4d, [1580](#)
- Transparent
 - OpenTK::Graphics::Color4, [422](#)
- Transpose
 - OpenTK::Matrix4, [1554](#), [1555](#)
 - OpenTK::Matrix4d, [1581](#)
- TryParse
 - OpenTK::Half, [1489](#), [1490](#)
- Turquoise
 - OpenTK::Graphics::Color4, [422](#)
- Two
 - OpenTK::Audio::AudioContext, [302](#)
- TypeName
 - OpenTK::Platform::Windows, [278](#)
- U
 - OpenTK::Input, [252](#)
- Uniform1
 - OpenTK::Graphics::ES20::GL, [756](#)–[758](#)
 - OpenTK::Graphics::OpenGL::GL, [1412](#)–[1415](#)
- Uniform2
 - OpenTK::Graphics::ES20::GL, [758](#)–[760](#)
 - OpenTK::Graphics::OpenGL::GL, [1415](#)–[1417](#)
- Uniform3
 - OpenTK::Graphics::ES20::GL, [760](#)–[762](#)
 - OpenTK::Graphics::OpenGL::GL, [1418](#)–[1420](#)
- Uniform4
 - OpenTK::Graphics::ES20::GL, [762](#)–[764](#)
 - OpenTK::Graphics::OpenGL::GL, [1421](#)–[1423](#)
- UnitW
 - OpenTK::Vector4, [1794](#)
 - OpenTK::Vector4d, [1823](#)
- UnitX
 - OpenTK::Vector2, [1657](#)
 - OpenTK::Vector2d, [1683](#)
 - OpenTK::Vector3, [1724](#)
 - OpenTK::Vector3d, [1757](#)
 - OpenTK::Vector4, [1794](#)
 - OpenTK::Vector4d, [1823](#)
- UnitY
 - OpenTK::Vector2, [1658](#)
 - OpenTK::Vector2d, [1683](#)
 - OpenTK::Vector3, [1724](#)
 - OpenTK::Vector3d, [1757](#)
 - OpenTK::Vector4, [1795](#)
 - OpenTK::Vector4d, [1823](#)
- UnitZ
 - OpenTK::Vector3, [1724](#)

- OpenTK::Vector3d, [1757](#)
- OpenTK::Vector4, [1795](#)
- OpenTK::Vector4d, [1823](#)
- Unknown
 - OpenTK::Input, [249](#)
- Unload
 - OpenTK::GameWindow, [388](#)
 - OpenTK::Platform::IGameWindow, [1602](#)
- Unused
 - OpenTK::Audio::OpenAL, [20](#)
- Up
 - OpenTK::Input, [250](#)
- Update
 - OpenTK::Graphics::GraphicsContext, [787](#)
 - OpenTK::Graphics::IGraphicsContext, [802](#)
- UpdateFrame
 - OpenTK::GameWindow, [388](#)
 - OpenTK::Platform::IGameWindow, [1602](#)
- UpdateFrequency
 - OpenTK::GameWindow, [387](#)
- UpdatePeriod
 - OpenTK::GameWindow, [387](#)
- UpdateTime
 - OpenTK::GameWindow, [387](#)
- UseDriverDefault
 - OpenTK::Audio::AudioContext, [302](#)
- UseFileAttributes
 - OpenTK::Platform::Windows, [279](#)
- UseProgram
 - OpenTK::Graphics::ES20::GL, [764](#)
 - OpenTK::Graphics::OpenGL::GL, [1424](#)
- V
 - OpenTK::Input, [252](#)
- ValidateProgram
 - OpenTK::Graphics::ES20::GL, [765](#)
 - OpenTK::Graphics::OpenGL::GL, [1424](#)
- Value
 - OpenTK::Input::JoystickMoveEventArgs, [1512](#)
 - OpenTK::Input::MouseWheelEventArgs, [1531](#)
- ValuePrecise
 - OpenTK::Input::MouseWheelEventArgs, [1531](#)
- Vector2
 - OpenTK::Vector2, [1640](#)
- Vector2d
 - OpenTK::Vector2d, [1665](#)
- Vector2h
 - OpenTK::Vector2h, [1687–1689](#)
- Vector3
 - OpenTK::Vector3, [1701](#)
- Vector3d
 - OpenTK::Vector3d, [1733](#), [1734](#)
- Vector3h
 - OpenTK::Vector3h, [1762–1765](#)
- Vector4
 - OpenTK::Vector4, [1775](#), [1776](#)
- Vector4d
 - OpenTK::Vector4d, [1803](#), [1804](#)
- Vector4h
 - OpenTK::Vector4h, [1828–1830](#)
- Velocity
 - OpenTK::Audio::OpenAL, [28](#)
- Vendor
 - OpenTK::Audio::OpenAL, [27](#)
 - OpenTK::Graphics::GraphicsContextVersion, [791](#)
- Version
 - OpenTK::Audio::OpenAL, [27](#)
 - OpenTK::AutoGeneratedAttribute, [345](#)
- Vertex2
 - OpenTK::Graphics::OpenGL::GL, [1425–1428](#)
- Vertex3
 - OpenTK::Graphics::OpenGL::GL, [1429–1432](#)
- Vertex4
 - OpenTK::Graphics::OpenGL::GL, [1433–1436](#)
- VertexAttrib1
 - OpenTK::Graphics::ES20::GL, [765](#), [766](#)

- OpenTK::Graphics::OpenGL::GL, [1437–1439](#)
- VertexAttrib2
 - OpenTK::Graphics::ES20::GL, [767](#), [768](#)
 - OpenTK::Graphics::OpenGL::GL, [1440–1445](#)
- VertexAttrib3
 - OpenTK::Graphics::ES20::GL, [769](#), [770](#)
 - OpenTK::Graphics::OpenGL::GL, [1446–1451](#)
- VertexAttrib4
 - OpenTK::Graphics::ES20::GL, [771](#), [772](#)
 - OpenTK::Graphics::OpenGL::GL, [1452–1463](#)
- VertexAttribPointer
 - OpenTK::Graphics::ES20::GL, [773](#)
 - OpenTK::Graphics::OpenGL::GL, [1463](#)
- VertexAttribPointer< T5 >
 - OpenTK::Graphics::ES20::GL, [774–779](#)
 - OpenTK::Graphics::OpenGL::GL, [1464–1469](#)
- VertexPointer
 - OpenTK::Graphics::ES10::GL, [502](#)
 - OpenTK::Graphics::ES11::GL, [616](#)
 - OpenTK::Graphics::OpenGL::GL, [1470](#)
- VertexPointer< T3 >
 - OpenTK::Graphics::ES10::GL, [502–504](#)
 - OpenTK::Graphics::ES11::GL, [616–618](#)
 - OpenTK::Graphics::OpenGL::GL, [1470–1472](#)
- Viewport
 - OpenTK::Graphics::ES10::GL, [504](#)
 - OpenTK::Graphics::ES11::GL, [618](#)
 - OpenTK::Graphics::ES20::GL, [780](#)
 - OpenTK::Graphics::OpenGL::GL, [1472](#)
- Violet
 - OpenTK::Graphics::Color4, [422](#)
- VirtualKeyToCharacter
 - OpenTK::Platform::Windows, [275](#)
- VirtualKeyToScanCode
 - OpenTK::Platform::Windows, [275](#)
- Visible
 - OpenTK::INativeWindow, [1497](#)
 - OpenTK::NativeWindow, [1597](#)
- VisibleChanged
 - OpenTK::INativeWindow, [1500](#)
 - OpenTK::NativeWindow, [1599](#)
- VocalMorpher
 - OpenTK::Audio::OpenAL, [40](#)
- VocalMorpherPhonemeA
 - OpenTK::Audio::OpenAL, [39](#), [42](#)
- VocalMorpherPhonemeAA
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeACoarseTuning
 - OpenTK::Audio::OpenAL, [39](#)
- VocalMorpherPhonemeAE
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeAH
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeAO
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeB
 - OpenTK::Audio::OpenAL, [39](#), [42](#)
- VocalMorpherPhonemeBCoarseTuning
 - OpenTK::Audio::OpenAL, [39](#)
- VocalMorpherPhonemeD
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeE
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeEH
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeER
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeF
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeG
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeI
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeIH
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeIY
 - OpenTK::Audio::OpenAL, [42](#)

- VocalMorpherPhonemeJ
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeK
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeL
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeM
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeN
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeO
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeP
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeR
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeS
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeT
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeU
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeUH
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeUW
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeV
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherPhonemeZ
 - OpenTK::Audio::OpenAL, [42](#)
- VocalMorpherRate
 - OpenTK::Audio::OpenAL, [35](#)
- VocalMorpherWaveform
 - OpenTK::Audio::OpenAL, [39](#)
- VorbisExt
 - OpenTK::Audio::OpenAL, [24](#)
- VSync
 - OpenTK::GameWindow, [387](#)
 - OpenTK::GLControl, [393](#)
 - OpenTK::Graphics::GraphicsContext, [788](#)
 - OpenTK::Graphics::IGraphicsContext, [803](#)
- W
 - OpenTK::Input, [252](#)
- OpenTK::Quaternion, [1618](#)
- OpenTK::Quaterniond, [1632](#)
- OpenTK::Vector4, [1795](#)
- OpenTK::Vector4d, [1823](#)
- OpenTK::Vector4h, [1834](#)
- Wheat
 - OpenTK::Graphics::Color4, [422](#)
- Wheel
 - OpenTK::Input::MouseDevice, [1524](#)
- WheelChanged
 - OpenTK::Input::MouseDevice, [1525](#)
- WheelPrecise
 - OpenTK::Input::MouseDevice, [1524](#)
- White
 - OpenTK::Graphics::Color4, [423](#)
- WhiteSmoke
 - OpenTK::Graphics::Color4, [423](#)
- Width
 - OpenTK::Box2, [361](#)
 - OpenTK::DisplayDevice, [370](#)
 - OpenTK::DisplayResolution, [374](#)
 - OpenTK::INativeWindow, [1497](#)
 - OpenTK::NativeWindow, [1597](#)
- WindowBorder
 - OpenTK::INativeWindow, [1498](#)
 - OpenTK::NativeWindow, [1597](#)
- WindowBorderChanged
 - OpenTK::INativeWindow, [1500](#)
 - OpenTK::NativeWindow, [1600](#)
- WindowInfo
 - OpenTK::GLControl, [393](#)
 - OpenTK::INativeWindow, [1498](#)
 - OpenTK::NativeWindow, [1597](#)
- WindowMessage
 - OpenTK::Platform::Windows, [280](#)
- WindowPos2
 - OpenTK::Graphics::OpenGL::GL, [1473–1476](#)
- WindowPos3
 - OpenTK::Graphics::OpenGL::GL, [1476–1479](#)
- WindowState
 - OpenTK::GameWindow, [387](#)
 - OpenTK::INativeWindow, [1498](#)
 - OpenTK::NativeWindow, [1597](#)
- WindowStateChanged

- OpenTK::INativeWindow, [1500](#)
- OpenTK::NativeWindow, [1600](#)
- WinLeft
 - OpenTK::Input, [249](#)
- WinRight
 - OpenTK::Input, [249](#)
- X
 - OpenTK::INativeWindow, [1498](#)
 - OpenTK::Input, [252](#)
 - OpenTK::Input::MouseDevice, [1524](#)
 - OpenTK::Input::MouseEventArgs, [1526](#)
 - OpenTK::NativeWindow, [1598](#)
 - OpenTK::Quaternion, [1618](#)
 - OpenTK::Quaterniond, [1632](#)
 - OpenTK::Vector2, [1658](#)
 - OpenTK::Vector2d, [1683](#)
 - OpenTK::Vector2h, [1693](#)
 - OpenTK::Vector3, [1724](#)
 - OpenTK::Vector3d, [1758](#)
 - OpenTK::Vector3h, [1768](#)
 - OpenTK::Vector4, [1795](#)
 - OpenTK::Vector4d, [1823](#)
 - OpenTK::Vector4h, [1834](#)
- XBUTTONDBLCLK
 - OpenTK::Platform::Windows, [280](#)
- XBUTTONDOWN
 - OpenTK::Platform::Windows, [280](#)
- XBUTTONUP
 - OpenTK::Platform::Windows, [280](#)
- XDelta
 - OpenTK::Input::MouseMoveEventArgs, [1528](#)
- XKey
 - OpenTK::Platform::X11, [294](#)
- XRamExtension
 - OpenTK::Audio::OpenAL::XRamExtension, [342](#)
- XRamStorage
 - OpenTK::Audio::OpenAL::XRamExtension, [341](#)
- Xy
 - OpenTK::Vector3, [1725](#)
 - OpenTK::Vector3d, [1759](#)
 - OpenTK::Vector3h, [1769](#)
- XYZ
 - OpenTK::Quaternion, [1618](#)
 - OpenTK::Quaterniond, [1632](#)
- Xyz
 - OpenTK::Quaternion, [1618](#)
 - OpenTK::Quaterniond, [1632](#)
 - OpenTK::Vector4, [1796](#)
 - OpenTK::Vector4d, [1824](#)
 - OpenTK::Vector4h, [1834](#)
- Y
 - OpenTK::INativeWindow, [1498](#)
 - OpenTK::Input, [252](#)
 - OpenTK::Input::MouseDevice, [1524](#)
 - OpenTK::Input::MouseEventArgs, [1527](#)
 - OpenTK::NativeWindow, [1598](#)
 - OpenTK::Quaternion, [1618](#)
 - OpenTK::Quaterniond, [1633](#)
 - OpenTK::Vector2, [1658](#)
 - OpenTK::Vector2d, [1683](#)
 - OpenTK::Vector2h, [1693](#)
 - OpenTK::Vector3, [1724](#)
 - OpenTK::Vector3d, [1758](#)
 - OpenTK::Vector3h, [1768](#)
 - OpenTK::Vector4, [1795](#)
 - OpenTK::Vector4d, [1823](#)
 - OpenTK::Vector4h, [1834](#)
- YDelta
 - OpenTK::Input::MouseMoveEventArgs, [1528](#)
- Yellow
 - OpenTK::Graphics::Color4, [423](#)
- YellowGreen
 - OpenTK::Graphics::Color4, [423](#)
- Z
 - OpenTK::Input, [252](#)
 - OpenTK::Quaternion, [1618](#)
 - OpenTK::Quaterniond, [1633](#)
 - OpenTK::Vector3, [1724](#)
 - OpenTK::Vector3d, [1758](#)
 - OpenTK::Vector3h, [1768](#)

OpenTK::Vector4, [1795](#)

OpenTK::Vector4d, [1823](#)

OpenTK::Vector4h, [1834](#)

Zero

OpenTK::ContextHandle, [366](#)

OpenTK::Vector2, [1658](#)

OpenTK::Vector2d, [1683](#)

OpenTK::Vector3, [1724](#)

OpenTK::Vector3d, [1758](#)

OpenTK::Vector4, [1795](#)

OpenTK::Vector4d, [1823](#)