

1 Afterwarp Framework	1
2 Namespace Index	3
2.1 Namespace List	3
3 Hierarchical Index	5
3.1 Class Hierarchy	5
4 Class Index	9
4.1 Class List	9
5 File Index	15
5.1 File List	15
6 Namespace Documentation	17
6.1 Afterwarp Namespace Reference	17
6.1.1 Enumeration Type Documentation	25
6.1.1.1 AlphaFormatRequest	25
6.1.1.2 AppCursor	26
6.1.1.3 ApplicationEvent	26
6.1.1.4 ApplicationWindowState	27
6.1.1.5 BlendFactor	27
6.1.1.6 BlendingEffect	27
6.1.1.7 BlendOp	28
6.1.1.8 BufferAccessType	28
6.1.1.9 BufferDataType	28
6.1.1.10 CameraCommand	29
6.1.1.11 CanvasContextState	29
6.1.1.12 ClustersCullingMode	29
6.1.1.13 ColorDitheringFormat	30
6.1.1.14 ComparisonFunc	30
6.1.1.15 ComputeTextureAccess	30
6.1.1.16 DepthFogDistance	31
6.1.1.17 DevicePlatform	31
6.1.1.18 DeviceTechnology	31
6.1.1.19 ElementFormat	32
6.1.1.20 FileChooserDialog	32
6.1.1.21 FogFormula	32
6.1.1.22 FontBorder	33
6.1.1.23 FontSlant	33
6.1.1.24 FontStretch	33
6.1.1.25 FontWeight	33
6.1.1.26 Key	35
6.1.1.27 KeyEvent	37

6.1.1.28 LineCaps	37
6.1.1.29 MeshAlign	38
6.1.1.30 ModelTransform	38
6.1.1.31 MouseButton	38
6.1.1.32 MouseEvent	39
6.1.1.33 ObjectModelViewCompare	39
6.1.1.34 PathJoint	39
6.1.1.35 PixelFormat	40
6.1.1.36 PointShape	41
6.1.1.37 PrimitiveTopology	43
6.1.1.38 SceneMeshTexture	43
6.1.1.39 SceneSamplerType	43
6.1.1.40 SceneTextureType	44
6.1.1.41 SelectionHighlightTextureType	44
6.1.1.42 ShaderElement	44
6.1.1.43 ShaderType	45
6.1.1.44 StencilOp	45
6.1.1.45 SuperSampleSDF	45
6.1.1.46 TechniqueLighting	46
6.1.1.47 TechniqueShadows	46
6.1.1.48 TextAlignment	46
6.1.1.49 TextureAddress	46
6.1.1.50 TextureCabinetFilterType	47
6.1.1.51 TextureCabinetPass	47
6.1.1.52 TextureCabinetType	47
6.1.1.53 TextureFidelity	48
6.1.1.54 TextureFilter	48
6.1.1.55 TextureType	48
6.1.1.56 TriangleFace	48
6.1.1.57 WidgetAlignment	49
6.1.1.58 WidgetManagerTextureType	49
6.1.1.59 WidgetPropertyBehavior	49
6.1.1.60 WidgetPropertyType	50
7 Class Documentation	51
7.1 Afterwarp.ActorCamera Class Reference	51
7.1.1 Detailed Description	53
7.1.2 Constructor & Destructor Documentation	53
7.1.2.1 ActorCamera() [1/2]	53
7.1.2.2 ActorCamera() [2/2]	53
7.1.3 Member Function Documentation	53
7.1.3.1 Command()	53

7.1.3.2 GetPosition()	54
7.1.3.3 GetQuaternion()	54
7.1.3.4 GetRotation()	54
7.1.3.5 SetPosition()	54
7.1.3.6 SetQuaternion()	54
7.1.3.7 SetRotation()	54
7.1.3.8 Zoom()	54
7.1.3.9 ZoomOrtho()	55
7.1.4 Property Documentation	55
7.1.4.1 Ceiling	55
7.1.4.2 Constraints	55
7.1.4.3 Distance	55
7.1.4.4 Forward	55
7.1.4.5 Right	55
7.1.4.6 View	56
7.2 Afterwarp.AmbientOcclusionParameters Struct Reference	56
7.2.1 Detailed Description	57
7.2.2 Constructor & Destructor Documentation	57
7.2.2.1 AmbientOcclusionParameters()	57
7.2.3 Member Data Documentation	57
7.2.3.1 Attenuation	57
7.2.3.2 BlurFallOff	57
7.2.3.3 BlurSigma	57
7.2.3.4 Contrast	57
7.2.3.5 DepthBias	58
7.2.3.6 KernelBias	58
7.2.3.7 Radius	58
7.2.3.8 Samples	58
7.2.3.9 Strength	58
7.2.4 Property Documentation	58
7.2.4.1 Default	58
7.3 Afterwarp.API Struct Reference	59
7.4 Afterwarp.Application Class Reference	59
7.4.1 Detailed Description	62
7.4.2 Constructor & Destructor Documentation	62
7.4.2.1 Application()	62
7.4.3 Member Function Documentation	62
7.4.3.1 BoolCallback()	62
7.4.3.2 CaptureMouseInput()	62
7.4.3.3 ConvertPortableKey()	62
7.4.3.4 EventCallback()	63
7.4.3.5 EventHookCallback()	63

7.4.3.6 Execute()	63
7.4.3.7 FileChooserDialog()	63
7.4.3.8 Invalidate()	63
7.4.3.9 KeyboardCallback()	63
7.4.3.10 MouseCallback()	64
7.4.3.11 ReadTextFromClipboard()	64
7.4.3.12 ReleaseMouseInput()	64
7.4.3.13 SetIcons() [1/2]	64
7.4.3.14 SetIcons() [2/2]	64
7.4.3.15 StatusCallback()	64
7.4.3.16 Terminate()	65
7.4.3.17 TranslateVirtualKey()	65
7.4.3.18 WriteTextToClipboard()	65
7.4.4 Property Documentation	65
7.4.4.1 ClientRect	65
7.4.4.2 ClientSize	65
7.4.4.3 Cursor	65
7.4.4.4 ExecutablePath	65
7.4.4.5 IconTitle	66
7.4.4.6 MinimalSize	66
7.4.4.7 MouseInputCaptured	66
7.4.4.8 Title	66
7.4.4.9 WindowHandle	66
7.4.4.10 WindowRect	66
7.4.4.11 WindowScale	66
7.4.4.12 WindowState	66
7.4.5 Event Documentation	67
7.4.5.1 OnCreate	67
7.4.5.2 OnDestroy	67
7.4.5.3 OnHook	67
7.4.5.4 OnIdle	67
7.4.5.5 OnKeyboard	67
7.4.5.6 OnMouse	67
7.4.5.7 OnRender	67
7.4.5.8 OnResize	68
7.4.5.9 OnStatus	68
7.5 Afterwarp.ApplicationConfiguration Struct Reference	68
7.5.1 Detailed Description	69
7.5.2 Constructor & Destructor Documentation	69
7.5.2.1 ApplicationConfiguration()	69
7.5.3 Member Data Documentation	69
7.5.3.1 IconBig	69

7.5.3.2 IconSmall	70
7.5.3.3 Instance	70
7.5.3.4 Position	70
7.5.3.5 Size	70
7.5.3.6 State	70
7.5.3.7 WindowClassNameBytes	70
7.5.4 Property Documentation	70
7.5.4.1 WindowClassName	70
7.6 Afterwarp.Canvas.Attribute Struct Reference	71
7.6.1 Detailed Description	71
7.6.2 Member Data Documentation	71
7.6.2.1 ColorAdjust	71
7.6.2.2 Cubic	71
7.6.2.3 Outline	71
7.6.2.4 Projected	72
7.6.2.5 SDF	72
7.6.2.6 Transform	72
7.7 Afterwarp.Device.Attribute Struct Reference	72
7.7.1 Detailed Description	72
7.7.2 Member Data Documentation	72
7.7.2.1 Debug	72
7.7.2.2 Legacy	73
7.7.2.3 LimitedExtensions	73
7.7.2.4 Software	73
7.8 Afterwarp.ObjectModel.Attribute Struct Reference	73
7.8.1 Detailed Description	73
7.8.2 Member Data Documentation	74
7.8.2.1 DisableRealign	74
7.8.2.2 Hierarchyless	74
7.8.2.3 NonViewable	74
7.8.2.4 Selectable	74
7.8.2.5 Transparent	74
7.8.2.6 Visible	74
7.9 Afterwarp.Scene.Attribute Struct Reference	74
7.9.1 Detailed Description	75
7.9.2 Member Data Documentation	75
7.9.2.1 Coloring	75
7.9.2.2 DepthPrePass	75
7.9.2.3 Glassy	75
7.9.2.4 Instancing	76
7.9.2.5 LinearDepths	76
7.9.2.6 Modeling	76

7.9.2.7 Normals	76
7.9.2.8 ShadowsCubic	76
7.9.2.9 SinglePass	76
7.9.2.10 TexturingCubic	76
7.10 Afterwarp.Texture.Attribute Struct Reference	77
7.10.1 Detailed Description	77
7.10.2 Member Data Documentation	77
7.10.2.1 Compute	77
7.10.2.2 Drawable	77
7.10.2.3 Dynamic	77
7.10.2.4 MipMapping	77
7.10.2.5 PremultipliedAlpha	78
7.10.2.6 Scratch	78
7.11 Afterwarp.TextureCabinet.Attribute Struct Reference	78
7.11.1 Detailed Description	79
7.11.2 Member Data Documentation	79
7.11.2.1 AmbientOcclusion	79
7.11.2.2 AmbientOcclusionDownscale	79
7.11.2.3 Bloom	79
7.11.2.4 BloomCubic	79
7.11.2.5 BloomDownscale	79
7.11.2.6 Glassy	79
7.11.2.7 GlassyFast	80
7.11.2.8 GlassyFog	80
7.11.2.9 GlassyFrosted	80
7.11.2.10 LinearDepths	80
7.12 Afterwarp.AutoDrawOption Struct Reference	80
7.12.1 Detailed Description	81
7.12.2 Member Data Documentation	81
7.12.2.1 MaterialIgnore	81
7.12.2.2 MaterialSkipOpaque	81
7.12.2.3 MaterialSkipTransparent	81
7.12.2.4 MaterialTransparency	81
7.12.2.5 ObjectDisableInstancing	81
7.12.2.6 ObjectHighlighted	81
7.12.2.7 ObjectSkipHavingMaterials	82
7.12.2.8 ObjectSkipNonTransparent	82
7.12.2.9 ObjectSkipNotHavingMaterials	82
7.12.2.10 ObjectSkipTransparent	82
7.12.2.11 PostUnbind	82
7.12.2.12 ResetTextures	82
7.13 Afterwarp.RenderingState.Blend Struct Reference	82

7.13.1 Detailed Description	83
7.13.2 Constructor & Destructor Documentation	83
7.13.2.1 Blend()	83
7.13.3 Member Data Documentation	83
7.13.3.1 Dest	83
7.13.3.2 Op	83
7.13.3.3 Source	84
7.13.4 Property Documentation	84
7.13.4.1 Default	84
7.14 Afterwarp.Buffer Class Reference	84
7.14.1 Detailed Description	86
7.14.2 Constructor & Destructor Documentation	86
7.14.2.1 Buffer()	86
7.14.3 Member Function Documentation	86
7.14.3.1 Copy()	86
7.14.3.2 Retrieve()	86
7.14.3.3 Update()	86
7.14.4 Property Documentation	87
7.14.4.1 AccessType	87
7.14.4.2 DataType	87
7.14.4.3 Device	87
7.14.4.4 Format	87
7.14.4.5 Pitch	87
7.14.4.6 PlatformHandle	87
7.14.4.7 Size	87
7.15 Afterwarp.CameraConstraints Struct Reference	88
7.15.1 Detailed Description	88
7.15.2 Constructor & Destructor Documentation	88
7.15.2.1 CameraConstraints()	88
7.15.3 Member Data Documentation	88
7.15.3.1 PositionMax	88
7.15.3.2 PositionMin	88
7.15.3.3 RotationMax	89
7.15.3.4 RotationMin	89
7.16 Afterwarp.Canvas Class Reference	89
7.16.1 Detailed Description	93
7.16.2 Constructor & Destructor Documentation	94
7.16.2.1 Canvas()	94
7.16.3 Member Function Documentation	94
7.16.3.1 Arc() [1/4]	94
7.16.3.2 Arc() [2/4]	94
7.16.3.3 Arc() [3/4]	94

7.16.3.4 Arc() [4/4]	95
7.16.3.5 Begin()	95
7.16.3.6 Circle()	95
7.16.3.7 Draw()	95
7.16.3.8 Ellipse()	95
7.16.3.9 End()	96
7.16.3.10 FillRect() [1/2]	96
7.16.3.11 FillRect() [2/2]	96
7.16.3.12 FillRoundRect()	96
7.16.3.13 FillRoundRectBottom()	96
7.16.3.14 FillRoundRectTop()	97
7.16.3.15 FillRoundRectTopInverse()	97
7.16.3.16 Flush()	97
7.16.3.17 FrameRect() [1/2]	97
7.16.3.18 FrameRect() [2/2]	97
7.16.3.19 FrameRoundRect()	98
7.16.3.20 Hexagon() [1/2]	98
7.16.3.21 Hexagon() [2/2]	98
7.16.3.22 Highlight()	98
7.16.3.23 Line()	99
7.16.3.24 LineCircle()	99
7.16.3.25 LineEllipse()	99
7.16.3.26 LineHexagon() [1/2]	99
7.16.3.27 LineHexagon() [2/2]	99
7.16.3.28 LineQuad()	100
7.16.3.29 Lines()	100
7.16.3.30 LineTriangle()	100
7.16.3.31 Pixel()	100
7.16.3.32 Pixels()	100
7.16.3.33 Quad() [1/2]	101
7.16.3.34 Quad() [2/2]	101
7.16.3.35 QuadImage()	101
7.16.3.36 QuadRegion()	101
7.16.3.37 RectWithHole()	101
7.16.3.38 Reset()	102
7.16.3.39 ResetSamplerState()	102
7.16.3.40 Ribbon() [1/3]	102
7.16.3.41 Ribbon() [2/3]	102
7.16.3.42 Ribbon() [3/3]	102
7.16.3.43 RoundRect()	103
7.16.3.44 RoundRectRegion()	103
7.16.3.45 Tape() [1/3]	103

7.16.3.46 Tape() [2/3]	103
7.16.3.47 Tape() [3/3]	104
7.16.3.48 TexturedTriangles()	104
7.16.3.49 ThickLine()	104
7.16.3.50 ThickLineCircle()	104
7.16.3.51 ThickLineEllipse()	105
7.16.3.52 ThickLineHexagon() [1/2]	105
7.16.3.53 ThickLineHexagon() [2/2]	105
7.16.3.54 ThickLineQuad()	105
7.16.3.55 ThickLineTriangle()	106
7.16.3.56 Triangle() [1/2]	106
7.16.3.57 Triangle() [2/2]	106
7.16.3.58 TriangleRegion()	106
7.16.3.59 Triangles()	107
7.16.4 Member Data Documentation	107
7.16.4.1 HexagonDelta	107
7.16.5 Property Documentation	107
7.16.5.1 Attributes	107
7.16.5.2 BatchCount	107
7.16.5.3 ClipRect	108
7.16.5.4 ContextState	108
7.16.5.5 Device	108
7.16.5.6 SamplerState	108
7.16.5.7 SignedDistanceField	108
7.16.5.8 Transform	108
7.16.5.9 ViewProjection	108
7.16.5.10 World	109
7.17 Afterwarp.CanvasBuffer Class Reference	109
7.17.1 Detailed Description	110
7.17.2 Constructor & Destructor Documentation	111
7.17.2.1 CanvasBuffer()	111
7.17.3 Member Function Documentation	111
7.17.3.1 Clear()	111
7.17.3.2 ClearAndShrink()	111
7.17.4 Property Documentation	111
7.17.4.1 Colors	111
7.17.4.2 Extents	111
7.17.4.3 IndexCount	111
7.17.4.4 Indices	111
7.17.4.5 VertexCount	112
7.17.4.6 Vertices	112
7.18 Afterwarp.CanvasSamplerState Struct Reference	112

7.18.1 Detailed Description	113
7.18.2 Constructor & Destructor Documentation	113
7.18.2.1 CanvasSamplerState()	113
7.18.3 Member Data Documentation	113
7.18.3.1 AddressU	113
7.18.3.2 AddressV	113
7.18.3.3 BorderColor	113
7.18.3.4 FilterMag	113
7.18.3.5 FilterMin	114
7.18.3.6 FilterMip	114
7.18.4 Property Documentation	114
7.18.4.1 Default	114
7.19 Afterwarp.ColorDithering Class Reference	114
7.19.1 Detailed Description	116
7.19.2 Constructor & Destructor Documentation	116
7.19.2.1 ColorDithering()	116
7.19.3 Member Function Documentation	116
7.19.3.1 Execute() [1/2]	116
7.19.3.2 Execute() [2/2]	116
7.19.4 Property Documentation	116
7.19.4.1 Device	116
7.19.4.2 Format	117
7.20 Afterwarp.ColorPair Struct Reference	117
7.20.1 Detailed Description	117
7.20.2 Constructor & Destructor Documentation	118
7.20.2.1 ColorPair() [1/2]	118
7.20.2.2 ColorPair() [2/2]	118
7.20.3 Member Data Documentation	118
7.20.3.1 First	118
7.20.3.2 Second	118
7.20.4 Property Documentation	118
7.20.4.1 Black	118
7.20.4.2 TranslucentBlack	118
7.20.4.3 TranslucentWhite	119
7.20.4.4 White	119
7.21 Afterwarp.ColorRect Struct Reference	119
7.21.1 Detailed Description	120
7.21.2 Constructor & Destructor Documentation	120
7.21.2.1 ColorRect() [1/2]	120
7.21.2.2 ColorRect() [2/2]	120
7.21.3 Member Function Documentation	120
7.21.3.1 GradientX() [1/2]	120

7.21.3.2 GradientX() [2/2]	121
7.21.3.3 GradientY() [1/2]	121
7.21.3.4 GradientY() [2/2]	121
7.21.4 Member Data Documentation	121
7.21.4.1 BottomLeft	121
7.21.4.2 BottomRight	121
7.21.4.3 TopLeft	121
7.21.4.4 TopRight	121
7.21.5 Property Documentation	122
7.21.5.1 Black	122
7.21.5.2 TranslucentBlack	122
7.21.5.3 TranslucentWhite	122
7.21.5.4 White	122
7.22 Afterwarp.ComputeBindTextureFormat Struct Reference	122
7.22.1 Detailed Description	123
7.22.2 Constructor & Destructor Documentation	123
7.22.2.1 ComputeBindTextureFormat()	123
7.22.3 Member Data Documentation	123
7.22.3.1 Access	123
7.22.3.2 Channel	123
7.22.3.3 Format	124
7.22.3.4 Layer	124
7.22.3.5 MipLevel	124
7.23 Afterwarp.ComputeProgram Class Reference	124
7.23.1 Detailed Description	126
7.23.2 Constructor & Destructor Documentation	126
7.23.2.1 ComputeProgram()	126
7.23.3 Member Function Documentation	126
7.23.3.1 Begin()	126
7.23.3.2 Bind() [1/2]	126
7.23.3.3 Bind() [2/2]	127
7.23.3.4 Commit()	127
7.23.3.5 Dispatch()	127
7.23.3.6 End()	127
7.23.3.7 ResetBindings()	127
7.23.3.8 Unbind() [1/2]	127
7.23.3.9 Unbind() [2/2]	128
7.23.4 Property Documentation	128
7.23.4.1 Device	128
7.24 Afterwarp.CustomObject Class Reference	128
7.24.1 Detailed Description	130
7.24.2 Member Function Documentation	130

7.24.2.1 Dispose() [1/2]	130
7.24.2.2 Dispose() [2/2]	131
7.24.2.3 Equals() [1/2]	131
7.24.2.4 Equals() [2/2]	131
7.24.2.5 GetHashCode()	131
7.24.3 Member Data Documentation	131
7.24.3.1 _handle	131
7.24.4 Property Documentation	131
7.24.4.1 Handle	131
7.25 Afterwarp.Device Class Reference	132
7.25.1 Detailed Description	133
7.25.2 Constructor & Destructor Documentation	134
7.25.2.1 Device()	134
7.25.3 Member Function Documentation	134
7.25.3.1 Clear()	134
7.25.3.2 MemoryBarrier()	134
7.25.3.3 ResetCache()	134
7.25.4 Property Documentation	134
7.25.4.1 Attributes	134
7.25.4.2 Behavior	134
7.25.4.3 LegacyBits	135
7.25.4.4 Platform	135
7.25.4.5 RenderingState	135
7.25.4.6 Scissor	135
7.25.4.7 TechFeatureVersion	135
7.25.4.8 Technology	135
7.25.4.9 TechVersion	135
7.25.4.10 Viewport	136
7.26 Afterwarp.DeviceBehavior Struct Reference	136
7.26.1 Detailed Description	136
7.26.2 Member Data Documentation	136
7.26.2.1 Compute	136
7.26.2.2 DepthClearBadPrecision	137
7.26.2.3 DepthClipNegative	137
7.26.2.4 ForceBufferUnbind	137
7.26.2.5 PerSampleShading	137
7.26.2.6 PostDepthCoverage	137
7.26.2.7 SignedNormIntFormatBugged	137
7.26.2.8 Tessellation	137
7.26.2.9 VariableRateRefresh	138
7.27 Afterwarp.DeviceClear Struct Reference	138
7.27.1 Detailed Description	138

7.27.2 Member Data Documentation	138
7.27.2.1 Color	138
7.27.2.2 Depth	138
7.27.2.3 Stencil	139
7.28 Afterwarp.API.Exception Class Reference	139
7.28.1 Detailed Description	140
7.28.2 Constructor & Destructor Documentation	140
7.28.2.1 Exception() [1/3]	140
7.28.2.2 Exception() [2/3]	140
7.28.2.3 Exception() [3/3]	140
7.29 Afterwarp.FloatColor Struct Reference	140
7.29.1 Detailed Description	141
7.29.2 Constructor & Destructor Documentation	141
7.29.2.1 FloatColor() [1/3]	141
7.29.2.2 FloatColor() [2/3]	141
7.29.2.3 FloatColor() [3/3]	142
7.29.3 Member Function Documentation	142
7.29.3.1 operator*() [1/2]	142
7.29.3.2 operator*() [2/2]	142
7.29.3.3 operator+()	142
7.29.3.4 operator-()	142
7.29.3.5 operator/() [1/2]	142
7.29.3.6 operator/() [2/2]	143
7.29.4 Member Data Documentation	143
7.29.4.1 Alpha	143
7.29.4.2 Blue	143
7.29.4.3 Green	143
7.29.4.4 Red	143
7.29.5 Property Documentation	143
7.29.5.1 Black	143
7.29.5.2 RGB	143
7.29.5.3 ToColor	144
7.29.5.4 TranslucentBlack	144
7.29.5.5 TranslucentWhite	144
7.29.5.6 White	144
7.30 Afterwarp.FloatColorRGB Struct Reference	144
7.30.1 Detailed Description	145
7.30.2 Constructor & Destructor Documentation	145
7.30.2.1 FloatColorRGB() [1/2]	145
7.30.2.2 FloatColorRGB() [2/2]	145
7.30.3 Member Function Documentation	146
7.30.3.1 operator*() [1/2]	146

7.30.3.2 operator*() [2/2]	146
7.30.3.3 operator+()	146
7.30.3.4 operator-()	146
7.30.3.5 operator/() [1/2]	146
7.30.3.6 operator/() [2/2]	146
7.30.4 Member Data Documentation	147
7.30.4.1 Blue	147
7.30.4.2 Green	147
7.30.4.3 Red	147
7.30.5 Property Documentation	147
7.30.5.1 Black	147
7.30.5.2 Defined	147
7.30.5.3 ToColor	147
7.30.5.4 ToColorAlpha	147
7.30.5.5 White	148
7.31 Afterwarp.FogParameters Struct Reference	148
7.31.1 Detailed Description	149
7.31.2 Constructor & Destructor Documentation	149
7.31.2.1 FogParameters()	149
7.31.3 Member Data Documentation	149
7.31.3.1 Bias	149
7.31.3.2 Color	149
7.31.3.3 DensityGround	149
7.31.3.4 Distance	150
7.31.3.5 GroundPlane	150
7.31.3.6 Opacity	150
7.31.4 Property Documentation	150
7.31.4.1 Default	150
7.31.4.2 Extinction	150
7.31.4.3 Scattering	150
7.32 Afterwarp.FontAttribute Struct Reference	150
7.32.1 Detailed Description	151
7.32.2 Member Data Documentation	151
7.32.2.1 StrikeOut	151
7.32.2.2 Underline	151
7.33 Afterwarp.FontEffect Struct Reference	151
7.33.1 Detailed Description	152
7.33.2 Constructor & Destructor Documentation	152
7.33.2.1 FontEffect()	152
7.33.3 Member Data Documentation	153
7.33.3.1 BorderBrightness	153
7.33.3.2 BorderOpacity	153

7.33.3.3 BorderThickness	153
7.33.3.4 BorderType	153
7.33.3.5 FillBrightness	153
7.33.3.6 FillOpacity	153
7.33.3.7 ShadowBrightness	153
7.33.3.8 ShadowDistance	154
7.33.3.9 ShadowOpacity	154
7.33.3.10 ShadowSmoothness	154
7.33.3.11 SignedFieldDistance	154
7.33.4 Property Documentation	154
7.33.4.1 Default	154
7.34 Afterwarp.FontParameters Struct Reference	155
7.34.1 Detailed Description	156
7.34.2 Constructor & Destructor Documentation	156
7.34.2.1 FontParameters()	156
7.34.3 Member Data Documentation	156
7.34.3.1 Attributes	156
7.34.3.2 Effect	156
7.34.3.3 FamilyBytes	156
7.34.3.4 Size	156
7.34.3.5 Slant	157
7.34.3.6 Stretch	157
7.34.3.7 Weight	157
7.34.4 Property Documentation	157
7.34.4.1 Family	157
7.35 Afterwarp.TextModeller.FontProvider Class Reference	157
7.35.1 Detailed Description	158
7.36 Afterwarp.GaussianBlur Class Reference	159
7.36.1 Detailed Description	160
7.36.2 Constructor & Destructor Documentation	160
7.36.2.1 GaussianBlur()	160
7.36.3 Member Function Documentation	161
7.36.3.1 Update()	161
7.36.3.2 UpdateAt()	161
7.36.4 Property Documentation	161
7.36.4.1 Chroma	161
7.36.4.2 Device	161
7.36.4.3 FixedSamples	161
7.36.4.4 HardwareFiltering	162
7.36.4.5 Samples	162
7.36.4.6 Sigma	162
7.37 Afterwarp.Grapher Class Reference	162

7.37.1 Detailed Description	164
7.37.2 Constructor & Destructor Documentation	164
7.37.2.1 Grapher()	164
7.37.3 Member Function Documentation	164
7.37.3.1 Arrow()	164
7.37.3.2 Begin()	165
7.37.3.3 BoundingBox()	165
7.37.3.4 DottedLine()	165
7.37.3.5 End()	165
7.37.3.6 Flush()	165
7.37.3.7 Line()	166
7.37.3.8 Lines()	166
7.37.3.9 Point()	166
7.37.3.10 Points()	166
7.37.3.11 Reset()	167
7.37.4 Property Documentation	167
7.37.4.1 BatchCount	167
7.37.4.2 Device	167
7.37.4.3 Transform	167
7.38 Afterwarp.ImageAtlas Class Reference	168
7.38.1 Detailed Description	169
7.38.2 Constructor & Destructor Documentation	170
7.38.2.1 ImageAtlas()	170
7.38.3 Member Function Documentation	170
7.38.3.1 ClearRegions()	170
7.38.3.2 ClearTextures()	170
7.38.3.3 CreateRegion()	170
7.38.3.4 CreateTexture()	170
7.38.3.5 MakeRegions()	170
7.38.3.6 PackRegion()	171
7.38.3.7 PackSurface()	171
7.38.3.8 Region()	171
7.38.3.9 RemoveRegion()	171
7.38.3.10 RemoveTexture()	171
7.38.3.11 Texture()	171
7.38.4 Property Documentation	172
7.38.4.1 Device	172
7.38.4.2 RegionCount	172
7.38.4.3 TextureCount	172
7.39 Afterwarp.ImageRegion Struct Reference	172
7.39.1 Detailed Description	173
7.39.2 Constructor & Destructor Documentation	173

7.39.2.1 ImageRegion()	173
7.39.3 Member Data Documentation	173
7.39.3.1 Flip	173
7.39.3.2 Height	173
7.39.3.3 Index	173
7.39.3.4 Left	174
7.39.3.5 Mirror	174
7.39.3.6 Rotate	174
7.39.3.7 Top	174
7.39.3.8 Width	174
7.40 Afterwarp.ObjectModels.Iterator Class Reference	175
7.40.1 Detailed Description	175
7.40.2 Member Function Documentation	176
7.40.2.1 MoveNext()	176
7.40.2.2 Reset()	176
7.40.3 Property Documentation	176
7.40.3.1 Current	176
7.41 Afterwarp.Widget.Iterator Class Reference	176
7.41.1 Detailed Description	177
7.41.2 Member Function Documentation	177
7.41.2.1 MoveNext()	177
7.41.2.2 Reset()	177
7.41.3 Property Documentation	177
7.41.3.1 Current	177
7.42 Afterwarp.KawaseBlur Class Reference	178
7.42.1 Detailed Description	179
7.42.2 Constructor & Destructor Documentation	179
7.42.2.1 KawaseBlur()	179
7.42.3 Member Function Documentation	179
7.42.3.1 SinglePass()	179
7.42.3.2 Update()	180
7.42.4 Property Documentation	180
7.42.4.1 Device	180
7.42.4.2 Offset	180
7.42.4.3 Passes	180
7.43 Afterwarp.Library Struct Reference	180
7.43.1 Detailed Description	181
7.43.2 Member Function Documentation	181
7.43.2.1 GetVersion()	181
7.43.2.2 SerialCode()	181
7.44 Afterwarp.Margins Struct Reference	181
7.44.1 Detailed Description	182

7.44.2 Constructor & Destructor Documentation	182
7.44.2.1 Margins() [1/4]	182
7.44.2.2 Margins() [2/4]	183
7.44.2.3 Margins() [3/4]	183
7.44.2.4 Margins() [4/4]	183
7.44.3 Member Data Documentation	183
7.44.3.1 Bottom	183
7.44.3.2 Left	183
7.44.3.3 Right	183
7.44.3.4 Top	183
7.44.4 Property Documentation	184
7.44.4.1 BottomLeft	184
7.44.4.2 BottomRight	184
7.44.4.3 Empty	184
7.44.4.4 Horizontal	184
7.44.4.5 TopLeft	184
7.44.4.6 TopRight	184
7.44.4.7 Vertical	184
7.44.4.8 Zero	185
7.45 Afterwarp.MeshAligns Struct Reference	185
7.45.1 Detailed Description	186
7.45.2 Constructor & Destructor Documentation	186
7.45.2.1 MeshAligns()	186
7.45.3 Member Data Documentation	186
7.45.3.1 Bias	186
7.45.3.2 Default	186
7.45.3.3 X	186
7.45.3.4 Y	187
7.45.3.5 Z	187
7.46 Afterwarp.MeshBuffer Class Reference	187
7.46.1 Detailed Description	190
7.46.2 Constructor & Destructor Documentation	191
7.46.2.1 MeshBuffer()	191
7.46.3 Member Function Documentation	191
7.46.3.1 CalculateBounds()	191
7.46.3.2 CalculateFlatNormals()	191
7.46.3.3 CalculateNormals()	191
7.46.3.4 CalculateNormalsWeld()	191
7.46.3.5 Centralize()	192
7.46.3.6 Clear()	192
7.46.3.7 Combine()	192
7.46.3.8 Cone()	192

7.46.3.9 Cube()	192
7.46.3.10 CubeMinimal()	193
7.46.3.11 CubeRound()	193
7.46.3.12 Cylinder()	193
7.46.3.13 Disc()	193
7.46.3.14 EliminateUnusedVertices()	194
7.46.3.15 Extrusion()	194
7.46.3.16 FrustumVolume()	194
7.46.3.17 Geosphere()	194
7.46.3.18 InvertIndexOrder()	195
7.46.3.19 InvertNormals()	195
7.46.3.20 JoinDuplicateVertices()	195
7.46.3.21 LoadFromFile()	195
7.46.3.22 LoadFromFileEx()	195
7.46.3.23 LoadSaveFeedback()	196
7.46.3.24 Model()	196
7.46.3.25 Plane()	196
7.46.3.26 SaveToFile()	196
7.46.3.27 SaveToFileEx()	197
7.46.3.28 SuperEllipse()	197
7.46.3.29 Supertoroid()	197
7.46.3.30 Torus()	198
7.46.3.31 TorusKnot()	198
7.46.3.32 Transfer() [1/2]	198
7.46.3.33 Transfer() [2/2]	199
7.46.3.34 TransferEx() [1/2]	199
7.46.3.35 TransferEx() [2/2]	199
7.46.3.36 TransformVertices()	199
7.46.3.37 TryLoadFromFile()	200
7.46.3.38 TryLoadFromFileEx()	200
7.46.3.39 TrySaveToFile()	200
7.46.3.40 TrySaveToFileEx()	200
7.46.3.41 Voxelize()	201
7.46.4 Property Documentation	201
7.46.4.1 IndexCount	201
7.46.4.2 Indices	201
7.46.4.3 IndicesPtr	201
7.46.4.4 Transform	201
7.46.4.5 VertexCount	201
7.46.4.6 Vertices	201
7.46.4.7 VerticesPtr	202
7.47 Afterwarp.MeshLoadingOption Struct Reference	202

7.47.1 Detailed Description	202
7.47.2 Member Data Documentation	202
7.47.2.1 ExportColors	202
7.47.2.2 ExportNormals	203
7.47.2.3 ExportTangents	203
7.47.2.4 ExportTexCoords	203
7.47.2.5 ExportWeights	203
7.47.2.6 MaterialTextureMissing	203
7.47.2.7 MaterialVertexColor	203
7.47.2.8 MaterialVertexColorPreferred	203
7.47.2.9 StripGeometryNames	204
7.48 Afterwarp.MeshMetaTag Class Reference	204
7.48.1 Detailed Description	206
7.48.2 Member Function Documentation	206
7.48.2.1 GetBounds()	206
7.48.2.2 PortionAdd()	206
7.48.2.3 PortionErase()	206
7.48.2.4 PortionGet()	206
7.48.2.5 PortionsClear()	206
7.48.2.6 PortionsCopy()	206
7.48.3 Property Documentation	207
7.48.3.1 Name	207
7.48.3.2 Parent	207
7.48.3.3 PortionCount	207
7.48.3.4 Type	207
7.49 Afterwarp.MeshMetaTagPortion Struct Reference	207
7.49.1 Detailed Description	208
7.49.2 Constructor & Destructor Documentation	208
7.49.2.1 MeshMetaTagPortion()	208
7.49.3 Member Data Documentation	208
7.49.3.1 BoundsMax	208
7.49.3.2 BoundsMin	208
7.49.3.3 FirstIndex	209
7.49.3.4 FirstVertex	209
7.49.3.5 IndexCount	209
7.49.3.6 VertexCount	209
7.50 Afterwarp.MeshMetaTags Class Reference	209
7.50.1 Detailed Description	211
7.50.2 Constructor & Destructor Documentation	211
7.50.2.1 MeshMetaTags()	211
7.50.3 Member Function Documentation	211
7.50.3.1 Add()	211

7.50.3.2 Clear()	211
7.50.3.3 Copy()	212
7.50.3.4 Erase()	212
7.50.3.5 Tag() [1/2]	212
7.50.3.6 Tag() [2/2]	212
7.50.3.7 TakeAway()	212
7.50.4 Property Documentation	212
7.50.4.1 Count	212
7.51 Afterwarp.MeshMetaTagType Struct Reference	213
7.51.1 Detailed Description	213
7.51.2 Member Data Documentation	213
7.51.2.1 Geometry	213
7.51.2.2 Indeterminate	213
7.51.2.3 Object	213
7.52 Afterwarp.MeshModel Class Reference	214
7.52.1 Detailed Description	215
7.52.2 Constructor & Destructor Documentation	216
7.52.2.1 MeshModel()	216
7.52.3 Member Function Documentation	216
7.52.3.1 Dismantle()	216
7.52.3.2 Draw()	216
7.52.3.3 DrawInstances()	216
7.52.3.4 TakeAway()	217
7.52.4 Property Documentation	217
7.52.4.1 Channel	217
7.52.4.2 IndexBuffer	217
7.52.4.3 IndexCount	217
7.52.4.4 Renderable	217
7.52.4.5 VertexBuffer	217
7.52.4.6 VertexCount	217
7.53 Afterwarp.MeshVoxel Class Reference	218
7.53.1 Detailed Description	219
7.53.2 Constructor & Destructor Documentation	219
7.53.2.1 MeshVoxel()	219
7.53.3 Member Function Documentation	220
7.53.3.1 ComputeParameters()	220
7.53.3.2 Extents()	220
7.53.3.3 Intersect()	220
7.53.3.4 LoadFromFile()	220
7.53.3.5 LoadFromFileInMemory()	220
7.53.3.6 SaveToFile()	221
7.53.3.7 TakeAway()	221

7.53.3.8 TryLoadFromFile()	221
7.53.3.9 TryLoadFromFileInMemory()	221
7.53.3.10 Visualize()	221
7.53.3.11 VisualizeFunc()	222
7.54 Afterwarp.ObjectMaterial Struct Reference	222
7.54.1 Detailed Description	223
7.54.2 Constructor & Destructor Documentation	223
7.54.2.1 ObjectMaterial()	223
7.54.3 Member Data Documentation	224
7.54.3.1 AlbedoColor	224
7.54.3.2 AmbientColor	224
7.54.3.3 Bitmask	224
7.54.3.4 EmissiveColor	224
7.54.3.5 FrostedGlass	224
7.54.3.6 Occlusion	224
7.54.3.7 Payload	224
7.54.3.8 Roughness	225
7.54.3.9 Shadows	225
7.54.3.10 SpecularColor	225
7.54.3.11 SpecularExponent	225
7.54.3.12 Technique	225
7.54.4 Property Documentation	225
7.54.4.1 Default	225
7.55 Afterwarp.ObjectMaterials Class Reference	226
7.55.1 Detailed Description	227
7.55.2 Constructor & Destructor Documentation	227
7.55.2.1 ObjectMaterials()	227
7.55.3 Member Function Documentation	228
7.55.3.1 Add()	228
7.55.3.2 Clear()	228
7.55.3.3 Erase()	228
7.55.3.4 GetMaterial()	228
7.55.3.5 SetMaterial()	228
7.55.4 Property Documentation	228
7.55.4.1 Count	228
7.56 Afterwarp.ObjectModel Class Reference	229
7.56.1 Detailed Description	231
7.56.2 Member Function Documentation	231
7.56.2.1 AABB()	231
7.56.2.2 Child()	232
7.56.2.3 ConnectLatches() [1/2]	232
7.56.2.4 ConnectLatches() [2/2]	232

7.56.2.5 GetConnectedLatches()	232
7.56.2.6 GetLatchTransform() [1/2]	232
7.56.2.7 GetLatchTransform() [2/2]	232
7.56.2.8 GetLatchWaypointCouple()	233
7.56.2.9 GetLatchWaypointCoupleMatrix()	233
7.56.2.10 GetTransform()	233
7.56.2.11 GetWaypointDistance()	233
7.56.2.12 Invalidate()	233
7.56.2.13 SetTransform()	233
7.56.3 Property Documentation	234
7.56.3.1 Alignments	234
7.56.3.2 Attributes	234
7.56.3.3 ChildCount	234
7.56.3.4 Color	234
7.56.3.5 DepthBias	234
7.56.3.6 Description	234
7.56.3.7 Highlight	234
7.56.3.8 ID	235
7.56.3.9 Layers	235
7.56.3.10 Material	235
7.56.3.11 Mesh	235
7.56.3.12 MeshName	235
7.56.3.13 Name	235
7.56.3.14 OrderIndex	235
7.56.3.15 Owner	236
7.56.3.16 Parent	236
7.56.3.17 Payload	236
7.56.3.18 Position	236
7.56.3.19 Size	236
7.56.3.20 Voxel	236
7.57 Afterwarp.ObjectModels Class Reference	237
7.57.1 Detailed Description	239
7.57.2 Constructor & Destructor Documentation	239
7.57.2.1 ObjectModels()	239
7.57.3 Member Function Documentation	239
7.57.3.1 Add() [1/2]	239
7.57.3.2 Add() [2/2]	239
7.57.3.3 Clear()	240
7.57.3.4 ClearViews()	240
7.57.3.5 CreateView()	240
7.57.3.6 Erase()	240
7.57.3.7 EraseView()	240

7.57.3.8 Exists() [1/2]	240
7.57.3.9 Exists() [2/2]	240
7.57.3.10 GetEnumerator()	241
7.57.3.11 Object() [1/2]	241
7.57.3.12 Object() [2/2]	241
7.57.3.13 Payload()	241
7.57.3.14 PayloadExists()	241
7.57.3.15 TryObject() [1/2]	241
7.57.3.16 TryObject() [2/2]	241
7.57.3.17 TryPayload()	242
7.57.4 Property Documentation	242
7.57.4.1 Count	242
7.57.4.2 Meshes	242
7.58 Afterwarp.ObjectModelView Class Reference	242
7.58.1 Detailed Description	244
7.58.2 Member Function Documentation	244
7.58.2.1 AutoDraw()	244
7.58.2.2 CompareFunc()	245
7.58.2.3 Invalidate()	245
7.58.2.4 Object()	245
7.58.2.5 Select()	245
7.58.2.6 SelectAny()	245
7.58.2.7 Sort() [1/2]	245
7.58.2.8 Sort() [2/2]	246
7.58.2.9 Update()	246
7.58.2.10 UpdateNeeded()	246
7.58.3 Property Documentation	246
7.58.3.1 IntersectedObjects	246
7.58.3.2 IntersectedRays	246
7.58.3.3 Layers	246
7.58.3.4 ObjectCount	246
7.58.3.5 ObjectsNotCulled	247
7.58.3.6 Owner	247
7.58.3.7 Projection	247
7.58.3.8 View	247
7.58.3.9 ViewProjection	247
7.59 Afterwarp.OceanMaterial Struct Reference	247
7.59.1 Detailed Description	248
7.59.2 Constructor & Destructor Documentation	248
7.59.2.1 OceanMaterial()	248
7.59.3 Member Data Documentation	249
7.59.3.1 AlbedoColor	249

7.59.3.2 AmbientColor	249
7.59.3.3 Bitmask	249
7.59.3.4 EmissiveColor	249
7.59.3.5 Extinction	249
7.59.3.6 Fresnel	249
7.59.3.7 Shadows	249
7.59.3.8 SpecularColor	250
7.59.3.9 SpecularExponent	250
7.59.4 Property Documentation	250
7.59.4.1 Default	250
7.60 Afterwarp.OceanSimulation Class Reference	250
7.60.1 Detailed Description	252
7.60.2 Constructor & Destructor Documentation	252
7.60.2.1 OceanSimulation()	252
7.60.3 Member Function Documentation	252
7.60.3.1 GetWavesTexture()	252
7.60.3.2 Render()	253
7.60.3.3 Update()	253
7.60.4 Property Documentation	253
7.60.4.1 Attributes	253
7.60.4.2 Device	253
7.60.4.3 Material	253
7.60.4.4 Parameters	253
7.60.4.5 Plane	253
7.60.4.6 Projection	254
7.60.4.7 SamplerShadow	254
7.60.4.8 Scale	254
7.60.4.9 Sections	254
7.60.4.10 View	254
7.60.4.11 ViewDistance	254
7.61 Afterwarp.OceanWavesParameters Struct Reference	254
7.61.1 Detailed Description	255
7.61.2 Constructor & Destructor Documentation	255
7.61.2.1 OceanWavesParameters()	255
7.61.3 Member Data Documentation	255
7.61.3.1 Choppiness	255
7.61.3.2 Resolution	256
7.61.3.3 WaveSize	256
7.61.3.4 Wind	256
7.61.4 Property Documentation	256
7.61.4.1 Default	256
7.62 Afterwarp.ParallaxMappingParameters Struct Reference	256

7.62.1 Detailed Description	257
7.62.2 Constructor & Destructor Documentation	257
7.62.2.1 ParallaxMappingParameters()	257
7.62.3 Member Data Documentation	257
7.62.3.1 Occlusion	257
7.62.3.2 SamplesMax	257
7.62.3.3 SamplesMin	257
7.62.3.4 Scale	257
7.62.4 Property Documentation	258
7.62.4.1 Default	258
7.63 Afterwarp.PathBroker Class Reference	258
7.63.1 Detailed Description	259
7.63.2 Constructor & Destructor Documentation	259
7.63.2.1 PathBroker()	259
7.63.3 Member Function Documentation	260
7.63.3.1 Fill()	260
7.63.3.2 Reset()	260
7.63.3.3 Stroke()	260
7.64 Afterwarp.PathBuilder Class Reference	260
7.64.1 Detailed Description	261
7.64.2 Constructor & Destructor Documentation	261
7.64.2.1 PathBuilder()	261
7.64.3 Member Function Documentation	261
7.64.3.1 Circle()	261
7.64.3.2 Clear()	262
7.64.3.3 ClosePath()	262
7.64.3.4 CurveTo()	262
7.64.3.5 Diamond()	262
7.64.3.6 Flatness()	262
7.64.3.7 Gear()	262
7.64.3.8 LineTo()	263
7.64.3.9 MoveTo()	263
7.64.3.10 QuadCurveTo()	263
7.64.3.11 Rectangle()	263
7.64.3.12 SmoothCurveTo()	263
7.64.3.13 SmoothQuadCurveTo()	263
7.64.4 Property Documentation	264
7.64.4.1 Elements	264
7.65 Afterwarp.PathCommand Struct Reference	264
7.65.1 Detailed Description	264
7.65.2 Member Data Documentation	264
7.65.2.1 Close	264

7.65.2.2 CurveTo	264
7.65.2.3 Flatness	265
7.65.2.4 LimitCusp	265
7.65.2.5 LineTo	265
7.65.2.6 MoveTo	265
7.65.2.7 QuadCurveTo	265
7.65.2.8 Relative	265
7.65.2.9 Smooth	266
7.65.2.10 ToleranceAngle	266
7.66 Afterwarp.PathElement Struct Reference	266
7.66.1 Detailed Description	267
7.66.2 Constructor & Destructor Documentation	267
7.66.2.1 PathElement() [1/2]	267
7.66.2.2 PathElement() [2/2]	267
7.66.3 Member Data Documentation	267
7.66.3.1 Value	267
7.66.4 Property Documentation	267
7.66.4.1 Command	267
7.66.4.2 Length	267
7.67 Afterwarp.Point Struct Reference	268
7.67.1 Detailed Description	269
7.67.2 Constructor & Destructor Documentation	269
7.67.2.1 Point() [1/3]	269
7.67.2.2 Point() [2/3]	269
7.67.2.3 Point() [3/3]	269
7.67.3 Member Function Documentation	270
7.67.3.1 Average()	270
7.67.3.2 Cross()	270
7.67.3.3 Distance()	270
7.67.3.4 Dot()	270
7.67.3.5 Equals()	270
7.67.3.6 GetHashCode()	270
7.67.3.7 operator!=(())	271
7.67.3.8 operator*() [1/2]	271
7.67.3.9 operator*() [2/2]	271
7.67.3.10 operator+()	271
7.67.3.11 operator-()	271
7.67.3.12 operator/() [1/2]	271
7.67.3.13 operator/() [2/2]	272
7.67.3.14 operator==(())	272
7.67.4 Member Data Documentation	272
7.67.4.1 X	272

7.67.4.2 Y	272
7.67.5 Property Documentation	272
7.67.5.1 Angle	272
7.67.5.2 Empty	272
7.67.5.3 Length	273
7.67.5.4 One	273
7.67.5.5 Transpose	273
7.67.5.6 UnitX	273
7.67.5.7 UnitY	273
7.67.5.8 Zero	273
7.68 Afterwarp.Program Class Reference	274
7.68.1 Detailed Description	276
7.68.2 Constructor & Destructor Documentation	276
7.68.2.1 Program()	276
7.68.3 Member Function Documentation	276
7.68.3.1 Begin()	276
7.68.3.2 Bind()	276
7.68.3.3 Commit()	276
7.68.3.4 Draw()	277
7.68.3.5 DrawIndexed()	277
7.68.3.6 DrawInstances()	277
7.68.3.7 DrawInstancesIndexed()	277
7.68.3.8 End()	277
7.68.3.9 ResetBindings()	277
7.68.3.10 ResetCache()	278
7.68.3.11 SetPatchVertices()	278
7.68.3.12 Unbind()	278
7.68.3.13 Update() [1/2]	278
7.68.3.14 Update() [2/2]	278
7.68.4 Property Documentation	278
7.68.4.1 Device	278
7.69 Afterwarp.ProgramElement Struct Reference	279
7.69.1 Detailed Description	279
7.69.2 Constructor & Destructor Documentation	279
7.69.2.1 ProgramElement()	279
7.69.3 Member Data Documentation	280
7.69.3.1 Channel	280
7.69.3.2 Element	280
7.69.3.3 Index	280
7.69.3.4 NameBytes	280
7.69.3.5 Shader	280
7.69.3.6 Size	280

7.69.4 Property Documentation	280
7.69.4.1 Name	280
7.70 Afterwarp.ProgramVariable Struct Reference	281
7.70.1 Detailed Description	281
7.70.2 Constructor & Destructor Documentation	281
7.70.2.1 ProgramVariable()	281
7.70.3 Member Data Documentation	282
7.70.3.1 Count	282
7.70.3.2 Format	282
7.70.3.3 Index	282
7.70.3.4 NameBytes	282
7.70.3.5 Offset	282
7.70.3.6 Shader	282
7.70.3.7 Size	282
7.70.4 Property Documentation	283
7.70.4.1 Name	283
7.71 Afterwarp.Widget.PropertyValue Struct Reference	283
7.71.1 Detailed Description	283
7.71.2 Constructor & Destructor Documentation	283
7.71.2.1 PropertyValue()	283
7.71.3 Property Documentation	284
7.71.3.1 Behavior	284
7.71.3.2 Name	284
7.71.3.3 Type	284
7.71.3.4 Value	284
7.72 Afterwarp.Quad Struct Reference	284
7.72.1 Detailed Description	285
7.72.2 Constructor & Destructor Documentation	285
7.72.2.1 Quad() [1/4]	285
7.72.2.2 Quad() [2/4]	286
7.72.2.3 Quad() [3/4]	286
7.72.2.4 Quad() [4/4]	286
7.72.3 Member Function Documentation	286
7.72.3.1 Offset() [1/2]	286
7.72.3.2 Offset() [2/2]	286
7.72.3.3 Rotated() [1/2]	286
7.72.3.4 Rotated() [2/2]	287
7.72.3.5 RotatedTopLeft()	287
7.72.3.6 Scale()	287
7.72.3.7 Scaled()	287
7.72.3.8 SkewedHoriz()	288
7.72.3.9 SkewedVert()	288

7.72.3.10 Transform()	288
7.72.4 Member Data Documentation	288
7.72.4.1 BottomLeft	288
7.72.4.2 BottomRight	288
7.72.4.3 TopLeft	288
7.72.4.4 TopRight	289
7.72.5 Property Documentation	289
7.72.5.1 Flip	289
7.72.5.2 Mirror	289
7.72.5.3 Unity	289
7.72.5.4 Zero	289
7.73 Afterwarp.RandomSequence Struct Reference	289
7.73.1 Detailed Description	290
7.73.2 Constructor & Destructor Documentation	290
7.73.2.1 RandomSequence() [1/2]	290
7.73.2.2 RandomSequence() [2/2]	290
7.73.3 Member Function Documentation	290
7.73.3.1 Ranged()	290
7.73.4 Property Documentation	291
7.73.4.1 Gaussian	291
7.73.4.2 Raw	291
7.73.4.3 Raw64	291
7.73.4.4 State	291
7.73.4.5 Value	291
7.73.4.6 ValueDouble	291
7.74 Afterwarp.Ray Struct Reference	291
7.74.1 Detailed Description	292
7.74.2 Constructor & Destructor Documentation	292
7.74.2.1 Ray() [1/2]	292
7.74.2.2 Ray() [2/2]	292
7.74.3 Member Function Documentation	292
7.74.3.1 IntersectCubeVolume()	292
7.74.3.2 IntersectPlane()	293
7.74.3.3 IntersectTriangle()	293
7.74.4 Member Data Documentation	293
7.74.4.1 Direction	293
7.74.4.2 Origin	293
7.75 Afterwarp.Rect Struct Reference	293
7.75.1 Detailed Description	295
7.75.2 Constructor & Destructor Documentation	295
7.75.2.1 Rect()	295
7.75.3 Member Function Documentation	295

7.75.3.1 Bounds()	295
7.75.3.2 Contains() [1/2]	295
7.75.3.3 Contains() [2/2]	295
7.75.3.4 Equals()	295
7.75.3.5 GetHashCode()	296
7.75.3.6 Inflate()	296
7.75.3.7 Intersect()	296
7.75.3.8 Join()	296
7.75.3.9 Offset()	296
7.75.3.10 operator"!=()	296
7.75.3.11 operator==(())	297
7.75.3.12 Overlaps()	297
7.75.4 Member Data Documentation	297
7.75.4.1 Height	297
7.75.4.2 Left	297
7.75.4.3 Top	297
7.75.4.4 Width	297
7.75.5 Property Documentation	297
7.75.5.1 Bottom	297
7.75.5.2 Empty	298
7.75.5.3 Right	298
7.75.5.4 Zero	298
7.76 Afterwarp.RectF Struct Reference	298
7.76.1 Detailed Description	299
7.76.2 Constructor & Destructor Documentation	300
7.76.2.1 RectF() [1/3]	300
7.76.2.2 RectF() [2/3]	300
7.76.2.3 RectF() [3/3]	300
7.76.3 Member Function Documentation	300
7.76.3.1 Bounds() [1/2]	300
7.76.3.2 Bounds() [2/2]	300
7.76.3.3 Contains() [1/3]	301
7.76.3.4 Contains() [2/3]	301
7.76.3.5 Contains() [3/3]	301
7.76.3.6 Inflate() [1/2]	301
7.76.3.7 Inflate() [2/2]	301
7.76.3.8 Intersect()	301
7.76.3.9 Join()	301
7.76.3.10 Offset() [1/2]	302
7.76.3.11 Offset() [2/2]	302
7.76.3.12 Overlaps()	302
7.76.4 Member Data Documentation	302

7.76.4.1 Height	302
7.76.4.2 Left	302
7.76.4.3 Top	302
7.76.4.4 Width	302
7.76.5 Property Documentation	303
7.76.5.1 Bottom	303
7.76.5.2 BottomLeft	303
7.76.5.3 BottomRight	303
7.76.5.4 Empty	303
7.76.5.5 Right	303
7.76.5.6 Size	303
7.76.5.7 TopLeft	303
7.76.5.8 TopRight	304
7.76.5.9 Unity	304
7.76.5.10 Zero	304
7.77 Afterwarp.RenderingState Struct Reference	304
7.77.1 Detailed Description	305
7.77.2 Constructor & Destructor Documentation	306
7.77.2.1 RenderingState()	306
7.77.3 Member Data Documentation	306
7.77.3.1 BlendAlpha	306
7.77.3.2 BlendColor	306
7.77.3.3 BlendConstant	306
7.77.3.4 ClampDepthBias	306
7.77.3.5 CullFace	307
7.77.3.6 Default	307
7.77.3.7 DepthBias	307
7.77.3.8 DepthFunc	307
7.77.3.9 SlopeDepthBias	307
7.77.3.10 States	307
7.77.3.11 StencilBack	307
7.77.3.12 StencilFront	307
7.77.3.13 StencilRefMask	308
7.77.3.14 StencilRefValue	308
7.77.3.15 StencilWriteMask	308
7.78 Afterwarp.Sampler Class Reference	308
7.78.1 Detailed Description	310
7.78.2 Constructor & Destructor Documentation	310
7.78.2.1 Sampler()	310
7.78.3 Member Function Documentation	310
7.78.3.1 Bind()	310
7.78.3.2 Unbind()	310

7.78.4 Property Documentation	310
7.78.4.1 Device	310
7.78.4.2 State	311
7.79 Afterwarp.SamplerState Struct Reference	311
7.79.1 Detailed Description	312
7.79.2 Constructor & Destructor Documentation	312
7.79.2.1 SamplerState()	312
7.79.3 Member Data Documentation	313
7.79.3.1 AddressU	313
7.79.3.2 AddressV	313
7.79.3.3 AddressW	313
7.79.3.4 Anisotropy	313
7.79.3.5 BiasLOD	313
7.79.3.6 BorderColor	313
7.79.3.7 CompareFunc	313
7.79.3.8 CompareToRef	314
7.79.3.9 FilterMag	314
7.79.3.10 FilterMin	314
7.79.3.11 FilterMip	314
7.79.3.12 MaxLOD	314
7.79.3.13 MinLOD	314
7.79.4 Property Documentation	314
7.79.4.1 Default	314
7.80 Afterwarp.Scene Class Reference	315
7.80.1 Detailed Description	317
7.80.2 Member Function Documentation	317
7.80.2.1 Begin()	317
7.80.2.2 CreateDepthNormals()	317
7.80.2.3 CreateModeling()	318
7.80.2.4 End()	318
7.80.2.5 GetSampler()	318
7.80.2.6 GetTexture()	318
7.80.2.7 GetVertexElements()	318
7.80.2.8 Instances()	318
7.80.2.9 Prepare()	319
7.80.2.10 ResetCache()	319
7.80.2.11 SetTexture()	319
7.80.2.12 SetVertexElements()	319
7.80.2.13 SetVertexElementsFromTextModeller()	319
7.80.2.14 TryPrepare()	319
7.80.3 Property Documentation	320
7.80.3.1 ActiveVertexElements	320

7.80.3.2 Atlas	320
7.80.3.3 Attributes	320
7.80.3.4 Device	320
7.80.3.5 InstancesCount	320
7.80.3.6 Lights	320
7.80.3.7 Material	320
7.80.3.8 ParallaxMapping	321
7.80.3.9 Program	321
7.80.3.10 Projection	321
7.80.3.11 Rendering	321
7.80.3.12 TextureCabinet	321
7.80.3.13 ToneMappingCoefficients	321
7.80.3.14 View	321
7.80.3.15 World	322
7.81 Afterwarp.SceneLight Struct Reference	322
7.81.1 Detailed Description	323
7.81.2 Constructor & Destructor Documentation	323
7.81.2.1 SceneLight()	323
7.81.3 Member Data Documentation	324
7.81.3.1 AlbedoColor	324
7.81.3.2 AmbientColor	324
7.81.3.3 Angle	324
7.81.3.4 AngleCutoff	324
7.81.3.5 AttenuationEnd	324
7.81.3.6 AttenuationStart	324
7.81.3.7 Bitmask	324
7.81.3.8 Direction	325
7.81.3.9 Intensity	325
7.81.3.10 Payload	325
7.81.3.11 Position	325
7.81.3.12 Reserved	325
7.81.3.13 ShadowCaster	325
7.81.3.14 SpecularColor	325
7.81.3.15 SpecularExponent	326
7.81.4 Property Documentation	326
7.81.4.1 Default	326
7.82 Afterwarp.SceneLights Class Reference	326
7.82.1 Detailed Description	328
7.82.2 Constructor & Destructor Documentation	328
7.82.2.1 SceneLights()	328
7.82.3 Member Function Documentation	328
7.82.3.1 Add()	328

7.82.3.2 Clear()	328
7.82.3.3 Erase()	328
7.82.3.4 Execute()	328
7.82.3.5 GetLight()	329
7.82.3.6 RenderDebug()	329
7.82.3.7 SetLight()	329
7.82.4 Property Documentation	329
7.82.4.1 Clusters	329
7.82.4.2 ClusterSize	329
7.82.4.3 Count	329
7.82.4.4 CullingMode	330
7.82.4.5 DepthSlices	330
7.82.4.6 Device	330
7.82.4.7 Indices	330
7.82.4.8 Size	330
7.83 Afterwarp.SceneMesh Class Reference	331
7.83.1 Detailed Description	333
7.83.2 Member Function Documentation	333
7.83.2.1 AutoDraw()	333
7.83.2.2 AutoDrawSliced()	333
7.83.2.3 Bounds()	333
7.83.2.4 GetSlice()	334
7.83.2.5 SetSlice()	334
7.83.3 Member Data Documentation	334
7.83.3.1 VertexElementsIndexUndefined	334
7.83.4 Property Documentation	334
7.83.4.1 Latches	334
7.83.4.2 Materials	334
7.83.4.3 Model	334
7.83.4.4 Name	335
7.83.4.5 Payload	335
7.83.4.6 Scale	335
7.83.4.7 Size	335
7.83.4.8 Tags	335
7.83.4.9 VertexElementsIndex	335
7.83.4.10 Voxel	335
7.84 Afterwarp.SceneMeshes Class Reference	336
7.84.1 Detailed Description	338
7.84.2 Constructor & Destructor Documentation	338
7.84.2.1 SceneMeshes()	338
7.84.3 Member Function Documentation	338
7.84.3.1 Add()	338

7.84.3.2 AddFromBuffer()	338
7.84.3.3 AddFromFile()	339
7.84.3.4 Clear()	339
7.84.3.5 Erase()	339
7.84.3.6 Exists()	339
7.84.3.7 Mesh() [1/2]	339
7.84.3.8 Mesh() [2/2]	339
7.84.3.9 Payload()	340
7.84.3.10 PayloadExists()	340
7.84.3.11 Slice()	340
7.84.3.12 TryAddFromFile()	340
7.84.3.13 TryMesh()	340
7.84.3.14 TryPayload()	341
7.84.4 Property Documentation	341
7.84.4.1 Count	341
7.84.4.2 Device	341
7.85 Afterwarp.SceneMeshLatch Struct Reference	341
7.85.1 Detailed Description	342
7.85.2 Member Data Documentation	342
7.85.2.1 Group	342
7.85.2.2 NameBytes	342
7.85.2.3 Orientation	342
7.85.2.4 Position	342
7.85.2.5 Type	342
7.85.3 Property Documentation	342
7.85.3.1 Name	342
7.86 Afterwarp.SceneMeshLatches Class Reference	343
7.86.1 Detailed Description	345
7.86.2 Constructor & Destructor Documentation	345
7.86.2.1 SceneMeshLatches()	345
7.86.3 Member Function Documentation	345
7.86.3.1 Add()	345
7.86.3.2 Clear()	345
7.86.3.3 Copy()	345
7.86.3.4 Erase()	345
7.86.3.5 GetLatch()	346
7.86.3.6 GetLatchIndex()	346
7.86.3.7 GetWaypointDistance()	346
7.86.3.8 InvalidateWaypoints()	346
7.86.3.9 LoadFromFile()	346
7.86.3.10 LoadFromFileInMemory()	346
7.86.3.11 SaveToFile()	346

7.86.3.12 SetLatch()	347
7.86.3.13 TakeAway()	347
7.86.3.14 TryLoadFromFile()	347
7.86.3.15 TryLoadFromFileInMemory()	347
7.86.4 Property Documentation	347
7.86.4.1 Count	347
7.87 Afterwarp.SceneMeshLatchType Struct Reference	347
7.87.1 Detailed Description	348
7.87.2 Member Data Documentation	348
7.87.2.1 Joint	348
7.87.2.2 Waypoint	348
7.88 Afterwarp.SceneMeshMaterial Class Reference	348
7.88.1 Detailed Description	350
7.88.2 Member Function Documentation	350
7.88.2.1 AddRange()	350
7.88.2.2 ClearRanges()	350
7.88.2.3 Commit()	350
7.88.2.4 Copy()	351
7.88.2.5 GetRange()	351
7.88.2.6 GetTexture()	351
7.88.2.7 ReleaseTextures()	351
7.88.2.8 SetRange()	351
7.88.2.9 SetTexture()	351
7.88.3 Property Documentation	352
7.88.3.1 Name	352
7.88.3.2 RangeCount	352
7.88.3.3 Shading	352
7.89 Afterwarp.SceneMeshMaterialRange Struct Reference	352
7.89.1 Detailed Description	352
7.89.2 Constructor & Destructor Documentation	353
7.89.2.1 SceneMeshMaterialRange()	353
7.89.3 Member Data Documentation	353
7.89.3.1 IndexCount	353
7.89.3.2 IndexStart	353
7.90 Afterwarp.SceneMeshMaterials Class Reference	353
7.90.1 Detailed Description	355
7.90.2 Constructor & Destructor Documentation	355
7.90.2.1 SceneMeshMaterials()	355
7.90.3 Member Function Documentation	355
7.90.3.1 Add()	355
7.90.3.2 Clear()	355
7.90.3.3 Commit()	356

7.90.3.4 Copy()	356
7.90.3.5 Erase()	356
7.90.3.6 GetMaterial()	356
7.90.3.7 TakeAway()	356
7.90.4 Property Documentation	356
7.90.4.1 Count	356
7.90.4.2 Name	357
7.90.4.3 Texturing	357
7.91 Afterwarp.SceneMeshMaterialShading Struct Reference	357
7.91.1 Detailed Description	358
7.91.2 Constructor & Destructor Documentation	358
7.91.2.1 SceneMeshMaterialShading()	358
7.91.3 Member Data Documentation	358
7.91.3.1 Ambient	358
7.91.3.2 Bloom	358
7.91.3.3 Diffuse	359
7.91.3.4 Emissive	359
7.91.3.5 Lighting	359
7.91.3.6 Reserved	359
7.91.3.7 Roughness	359
7.91.3.8 Specular	359
7.91.3.9 SpecularExponent	359
7.92 Afterwarp.SelectionHighlight Class Reference	360
7.92.1 Detailed Description	361
7.92.2 Constructor & Destructor Documentation	362
7.92.2.1 SelectionHighlight()	362
7.92.3 Member Function Documentation	362
7.92.3.1 Begin()	362
7.92.3.2 Clear()	362
7.92.3.3 End()	362
7.92.3.4 Filter()	362
7.92.3.5 Texture()	362
7.92.4 Property Documentation	363
7.92.4.1 DepthStencil	363
7.92.4.2 Device	363
7.92.4.3 Grayscale	363
7.92.4.4 Parameters	363
7.92.4.5 Rendering	363
7.92.4.6 Samples	363
7.92.4.7 Size	363
7.92.4.8 TextureCoordinates	364
7.93 Afterwarp.SelectionHighlightParameters Struct Reference	364

7.93.1 Detailed Description	365
7.93.2 Constructor & Destructor Documentation	365
7.93.2.1 SelectionHighlightParameters()	365
7.93.3 Member Data Documentation	365
7.93.3.1 BlurOffset	365
7.93.3.2 BlurPasses	365
7.93.3.3 GlowIntensity	365
7.93.3.4 OutlineColor	365
7.93.3.5 OutlineStart	366
7.93.4 Property Documentation	366
7.93.4.1 Default	366
7.94 Afterwarp.ShadowCaster Class Reference	366
7.94.1 Detailed Description	368
7.94.2 Member Function Documentation	368
7.94.2.1 Begin()	368
7.94.2.2 Clear()	368
7.94.2.3 End()	368
7.94.2.4 Filter()	368
7.94.2.5 GetTexture()	368
7.94.3 Property Documentation	369
7.94.3.1 Atlas	369
7.94.3.2 Device	369
7.94.3.3 Position	369
7.94.3.4 Rendering	369
7.94.3.5 Size	369
7.94.3.6 ViewProjection	369
7.95 Afterwarp.ShadowCastingAtlas Class Reference	370
7.95.1 Detailed Description	371
7.95.2 Constructor & Destructor Documentation	372
7.95.2.1 ShadowCastingAtlas()	372
7.95.3 Member Function Documentation	372
7.95.3.1 Add()	372
7.95.3.2 BorderFill()	372
7.95.3.3 Caster()	372
7.95.3.4 Clear()	372
7.95.3.5 Erase()	373
7.95.4 Property Documentation	373
7.95.4.1 Count	373
7.95.4.2 Device	373
7.95.4.3 Padding	373
7.95.4.4 Parameters	373
7.95.4.5 Samples	373

7.95.4.6 Size	373
7.95.4.7 Technique	374
7.95.4.8 Texture	374
7.96 Afterwarp.ShadowParameters Struct Reference	374
7.96.1 Detailed Description	375
7.96.2 Constructor & Destructor Documentation	375
7.96.2.1 ShadowParameters()	375
7.96.3 Member Data Documentation	375
7.96.3.1 Bias	375
7.96.3.2 BleedSigma	375
7.96.3.3 BlurSamples	375
7.96.3.4 BlurSigma	375
7.96.3.5 Exponent1	376
7.96.3.6 Exponent2	376
7.96.3.7 Variance	376
7.96.4 Property Documentation	376
7.96.4.1 Default	376
7.97 Afterwarp.SignedDistanceField Struct Reference	376
7.97.1 Detailed Description	377
7.97.2 Constructor & Destructor Documentation	377
7.97.2.1 SignedDistanceField()	377
7.97.3 Member Data Documentation	377
7.97.3.1 OutlineDistanceMaxSDF	377
7.97.3.2 OutlineDistanceMinSDF	377
7.97.3.3 OutlineOffsetSDF	378
7.97.3.4 SignedFieldDistance	378
7.97.3.5 SuperSampleSDF	378
7.97.4 Property Documentation	378
7.97.4.1 Default	378
7.98 Afterwarp.SpatialFog Class Reference	378
7.98.1 Detailed Description	380
7.98.2 Constructor & Destructor Documentation	380
7.98.2.1 SpatialFog()	380
7.98.3 Member Function Documentation	380
7.98.3.1 Execute()	380
7.98.3.2 ExecuteGlassy()	380
7.98.4 Property Documentation	381
7.98.4.1 Device	381
7.98.4.2 Distance	381
7.98.4.3 Formula	381
7.98.4.4 Parameters	381
7.98.4.5 Projection	381

7.98.4.6 View	381
7.99 Afterwarp.RenderingState.State Struct Reference	381
7.99.1 Detailed Description	382
7.99.2 Member Data Documentation	382
7.99.2.1 AlphaToCoverage	382
7.99.2.2 BlendEnable	382
7.99.2.3 ColorWrite	383
7.99.2.4 CubeMapSeamless	383
7.99.2.5 CullClockwise	383
7.99.2.6 DepthClip	383
7.99.2.7 DepthTest	383
7.99.2.8 DepthWrite	383
7.99.2.9 LineAntialias	383
7.99.2.10 Multisampling	384
7.99.2.11 PerSampleShading	384
7.99.2.12 ScissorClip	384
7.99.2.13 StencilTest	384
7.99.2.14 Wireframe	384
7.100 Afterwarp.RenderingState.StencilState Struct Reference	384
7.100.1 Detailed Description	385
7.100.2 Constructor & Destructor Documentation	385
7.100.2.1 StencilState()	385
7.100.3 Member Data Documentation	385
7.100.3.1 DepthFailOp	385
7.100.3.2 DepthPassOp	385
7.100.3.3 FailOp	386
7.100.3.4 Func	386
7.100.4 Property Documentation	386
7.100.4.1 Default	386
7.101 Afterwarp.Surface Class Reference	386
7.101.1 Detailed Description	389
7.101.2 Constructor & Destructor Documentation	389
7.101.2.1 Surface() [1/3]	389
7.101.2.2 Surface() [2/3]	389
7.101.2.3 Surface() [3/3]	389
7.101.3 Member Function Documentation	389
7.101.3.1 Clear() [1/2]	389
7.101.3.2 Clear() [2/2]	390
7.101.3.3 ConvertFormat()	390
7.101.3.4 Copy() [1/2]	390
7.101.3.5 Copy() [2/2]	390
7.101.3.6 Flip()	390

7.101.3.7 GetPixel()	390
7.101.3.8 GetPixelBilinear()	391
7.101.3.9 GetPixelPtr()	391
7.101.3.10 GetScanline()	391
7.101.3.11 Invert()	391
7.101.3.12 MakeSignedDistanceField()	391
7.101.3.13 Mirror()	391
7.101.3.14 PremultiplyAlpha()	392
7.101.3.15 ResetAlpha()	392
7.101.3.16 Resize()	392
7.101.3.17 SaveToFile()	392
7.101.3.18 SaveToFileInMemory()	392
7.101.3.19 SetPixel()	393
7.101.3.20 ShrinkFrom()	393
7.101.3.21 Stretch()	393
7.101.3.22 StretchBilinear()	393
7.101.3.23 UnpremultiplyAlpha()	393
7.101.4 Property Documentation	394
7.101.4.1 Bits	394
7.101.4.2 ByteSize	394
7.101.4.3 BytesPerPixel	394
7.101.4.4 Empty	394
7.101.4.5 Format	394
7.101.4.6 HasAlphaChannel	394
7.101.4.7 Height	394
7.101.4.8 Pitch	395
7.101.4.9 PremultipliedAlpha	395
7.101.4.10 Width	395
7.102 Afterwarp.SwapChain Class Reference	395
7.102.1 Detailed Description	397
7.102.2 Constructor & Destructor Documentation	397
7.102.2.1 SwapChain()	397
7.102.3 Member Function Documentation	397
7.102.3.1 Begin()	397
7.102.3.2 End()	398
7.102.3.3 Resize()	398
7.102.4 Property Documentation	398
7.102.4.1 DepthStencil	398
7.102.4.2 Device	398
7.102.4.3 Format	398
7.102.4.4 Height	398
7.102.4.5 Multisamples	398

7.102.4.6 VSync	399
7.102.4.7 Width	399
7.102.4.8 WindowHandle	399
7.103 Afterwarp.TextEntryRect Struct Reference	399
7.103.1 Detailed Description	400
7.103.2 Member Data Documentation	400
7.103.2.1 Position	400
7.103.2.2 Rectangle	400
7.104 Afterwarp.TextModeller Class Reference	400
7.104.1 Detailed Description	402
7.104.2 Constructor & Destructor Documentation	403
7.104.2.1 TextModeller()	403
7.104.3 Member Function Documentation	403
7.104.3.1 Clear()	403
7.104.3.2 CopyToMeshBuffer()	403
7.104.3.3 Draw() [1/3]	403
7.104.3.4 Draw() [2/3]	403
7.104.3.5 Draw() [3/3]	404
7.104.3.6 DrawCurved() [1/3]	404
7.104.3.7 DrawCurved() [2/3]	404
7.104.3.8 DrawCurved() [3/3]	405
7.104.3.9 DrawDepthCurved()	405
7.104.3.10 Extent()	405
7.104.3.11 ExtentByShape()	405
7.104.3.12 Prepare()	406
7.104.3.13 Render()	406
7.104.3.14 Reset()	406
7.104.3.15 SetFontParameters()	406
7.104.3.16 TextRects()	406
7.104.4 Property Documentation	406
7.104.4.1 Device	406
7.104.4.2 Provider	407
7.104.4.3 Transform	407
7.105 Afterwarp.TextRenderer Class Reference	407
7.105.1 Detailed Description	409
7.105.2 Constructor & Destructor Documentation	409
7.105.2.1 TextRenderer()	409
7.105.3 Member Function Documentation	409
7.105.3.1 Draw()	409
7.105.3.2 DrawAligned()	410
7.105.3.3 DrawAlignedByPixels()	410
7.105.3.4 DrawCentered()	410

7.105.3.5 DrawCenteredByPixels()	410
7.105.3.6 Extent()	411
7.105.3.7 ExtentByPixels()	411
7.105.3.8 TextRects()	411
7.105.4 Property Documentation	411
7.105.4.1 Canvas	411
7.105.4.2 Parameters	411
7.106 Afterwarp.TextureModifiers Struct Reference	412
7.106.1 Detailed Description	412
7.106.2 Constructor & Destructor Documentation	412
7.106.2.1 TextureModifiers()	412
7.106.3 Member Data Documentation	412
7.106.3.1 Effect	412
7.106.3.2 Interleave	412
7.106.3.3 Scale	413
7.106.3.4 VerticalSpace	413
7.107 Afterwarp.Texture Class Reference	413
7.107.1 Detailed Description	415
7.107.2 Constructor & Destructor Documentation	415
7.107.2.1 Texture() [1/4]	415
7.107.2.2 Texture() [2/4]	416
7.107.2.3 Texture() [3/4]	416
7.107.2.4 Texture() [4/4]	416
7.107.3 Member Function Documentation	416
7.107.3.1 Attach()	416
7.107.3.2 Begin()	417
7.107.3.3 Bind()	417
7.107.3.4 Clear() [1/2]	417
7.107.3.5 Clear() [2/2]	417
7.107.3.6 Copy() [1/2]	417
7.107.3.7 Copy() [2/2]	418
7.107.3.8 CreateNormalsAndOcclusion()	418
7.107.3.9 CreateParallax()	418
7.107.3.10 Detach()	418
7.107.3.11 End()	418
7.107.3.12 GenerateMipMaps()	419
7.107.3.13 LoadFromFile()	419
7.107.3.14 ResetCache()	419
7.107.3.15 Retrieve() [1/2]	419
7.107.3.16 Retrieve() [2/2]	419
7.107.3.17 SaveToFile()	420
7.107.3.18 Unbind()	420

7.107.3.19 Update()	420
7.107.4 Property Documentation	420
7.107.4.1 Device	420
7.107.4.2 Parameters	420
7.107.4.3 PlatformHandle	421
7.108 Afterwarp.TextureCabinet Class Reference	421
7.108.1 Detailed Description	423
7.108.2 Constructor & Destructor Documentation	423
7.108.2.1 TextureCabinet()	423
7.108.3 Member Function Documentation	423
7.108.3.1 Begin()	423
7.108.3.2 Clear()	424
7.108.3.3 End()	424
7.108.3.4 Filter()	424
7.108.3.5 GetTexture()	424
7.108.3.6 Present()	424
7.108.3.7 Resolve()	424
7.108.4 Property Documentation	425
7.108.4.1 AmbientOcclusionParameters	425
7.108.4.2 Attributes	425
7.108.4.3 DepthStencil	425
7.108.4.4 Device	425
7.108.4.5 Fidelity	425
7.108.4.6 Rendering	425
7.108.4.7 Samples	425
7.108.4.8 Size	426
7.108.4.9 Texture	426
7.108.4.10 ToneMappingBloom	426
7.109 Afterwarp.TextureParameters Struct Reference	426
7.109.1 Detailed Description	427
7.109.2 Constructor & Destructor Documentation	427
7.109.2.1 TextureParameters()	427
7.109.3 Member Data Documentation	427
7.109.3.1 Attributes	427
7.109.3.2 DepthStencil	427
7.109.3.3 Format	427
7.109.3.4 Height	427
7.109.3.5 Layers	428
7.109.3.6 Multisamples	428
7.109.3.7 Type	428
7.109.3.8 Width	428
7.110 Afterwarp.Timer Class Reference	428

7.110.1 Detailed Description	430
7.110.2 Constructor & Destructor Documentation	430
7.110.2.1 Timer()	430
7.110.3 Member Function Documentation	430
7.110.3.1 NextSlice()	430
7.110.3.2 Reset()	430
7.110.3.3 TrimSkippedTimeSlices()	430
7.110.3.4 Update()	431
7.110.3.5 UpdateNextSlice()	431
7.110.4 Property Documentation	431
7.110.4.1 ExtractTokens	431
7.110.4.2 FrameRate	431
7.110.4.3 Latency	431
7.110.4.4 SkippedTimeSlices	431
7.110.4.5 Speed	431
7.110.4.6 TimeSlice	432
7.111 Afterwarp.ToneMappingBloom Struct Reference	432
7.111.1 Detailed Description	433
7.111.2 Constructor & Destructor Documentation	433
7.111.2.1 ToneMappingBloom()	433
7.111.3 Member Data Documentation	433
7.111.3.1 BloomBlurSamples	433
7.111.3.2 BloomBlurSigma	433
7.111.3.3 BloomCoefficients	433
7.111.3.4 BloomColorShift	433
7.111.3.5 BloomGamma	434
7.111.3.6 BloomThreshold	434
7.111.3.7 FrostedPower	434
7.111.3.8 GlassyBuckets	434
7.111.3.9 ToneFactors	434
7.111.3.10 ToneWhite	434
7.111.4 Property Documentation	434
7.111.4.1 Default	434
7.112 Afterwarp.Utility Struct Reference	435
7.112.1 Member Function Documentation	436
7.112.1.1 AddColors()	436
7.112.1.2 AssignPayload< T >()	437
7.112.1.3 AverageColors()	437
7.112.1.4 AverageFourColors()	437
7.112.1.5 AverageSixColors()	437
7.112.1.6 BiasTransform()	437
7.112.1.7 BlendColors()	437

7.112.1.8 BlendFourColors()	438
7.112.1.9 ColorToGray()	438
7.112.1.10 ColorToGrayF()	438
7.112.1.11 ComposeColors()	438
7.112.1.12 DisplaceRB()	439
7.112.1.13 FreePayload()	439
7.112.1.14 FromUI()	439
7.112.1.15 GainTransform()	439
7.112.1.16 GetColorAlpha()	439
7.112.1.17 GetColorAlphaF()	440
7.112.1.18 InvertColor()	440
7.112.1.19 MakeColor() [1/2]	440
7.112.1.20 MakeColor() [2/2]	440
7.112.1.21 MakeColorAlpha() [1/2]	441
7.112.1.22 MakeColorAlpha() [2/2]	441
7.112.1.23 MakeColorGray() [1/2]	441
7.112.1.24 MakeColorGray() [2/2]	442
7.112.1.25 MakeColorRGB() [1/2]	442
7.112.1.26 MakeColorRGB() [2/2]	443
7.112.1.27 MakeColorWithGray() [1/2]	443
7.112.1.28 MakeColorWithGray() [2/2]	443
7.112.1.29 MatrixLookAt4x4()	444
7.112.1.30 MatrixOrthographic4x4()	444
7.112.1.31 MatrixPerspective4x4()	444
7.112.1.32 MultiplyColors()	444
7.112.1.33 PremultiplyAlpha()	445
7.112.1.34 RetrievePayload< T >()	445
7.112.1.35 SineAccelerate()	445
7.112.1.36 SineCycle()	445
7.112.1.37 SineDecelerate()	445
7.112.1.38 SineTransform()	445
7.112.1.39 SineTwoCycle()	446
7.112.1.40 SubtractColors()	446
7.112.1.41 ToUI() [1/2]	446
7.112.1.42 ToUI() [2/2]	446
7.112.1.43 UnpremultiplyAlpha()	446
7.112.1.44 VertexElementsEstimatePitch()	446
7.112.2 Property Documentation	447
7.112.2.1 SystemTicks	447
7.113 Afterwarp.VertexElement Struct Reference	447
7.113.1 Detailed Description	448
7.113.2 Constructor & Destructor Documentation	448

7.113.2.1 VertexElement()	448
7.113.3 Member Data Documentation	448
7.113.3.1 Channel	448
7.113.3.2 ChannelIndexBuffer	448
7.113.3.3 ChannelNormalized	448
7.113.3.4 Count	448
7.113.3.5 Format	449
7.113.3.6 NameBytes	449
7.113.3.7 Offset	449
7.113.4 Property Documentation	449
7.113.4.1 Name	449
7.114 Afterwarp.MeshBuffer.VertexEntry Struct Reference	449
7.114.1 Detailed Description	450
7.114.2 Constructor & Destructor Documentation	450
7.114.2.1 VertexEntry()	450
7.114.3 Member Data Documentation	450
7.114.3.1 Color	450
7.114.3.2 Normal	450
7.114.3.3 Position	451
7.114.3.4 Tangent	451
7.114.3.5 TexCoord	451
7.115 Afterwarp.Volume Struct Reference	451
7.115.1 Member Function Documentation	451
7.115.1.1 CalculateNearFarPlanes()	451
7.115.1.2 CalculateVisibleFrame()	452
7.116 Afterwarp.Widget Class Reference	452
7.116.1 Detailed Description	455
7.116.2 Constructor & Destructor Documentation	455
7.116.2.1 Widget()	455
7.116.3 Member Function Documentation	455
7.116.3.1 AcceptKey()	455
7.116.3.2 AcceptMouse()	455
7.116.3.3 Accomodate()	456
7.116.3.4 BringToFront()	456
7.116.3.5 ExternalEventFunc()	456
7.116.3.6 FindAt()	456
7.116.3.7 Get()	456
7.116.3.8 Get< T >()	456
7.116.3.9 GetChild()	456
7.116.3.10 GetChildIndex()	457
7.116.3.11 GetClientRect()	457
7.116.3.12 GetEnumerator()	457

7.116.3.13 GetPadding()	457
7.116.3.14 Invalidate()	457
7.116.3.15 InvokeEvent()	457
7.116.3.16 LocalToScreen()	457
7.116.3.17 ScreenToLocal()	458
7.116.3.18 SendToBack()	458
7.116.3.19 Set()	458
7.116.3.20 Set< T >()	458
7.116.3.21 SetPadding()	458
7.116.3.22 Update()	458
7.116.4 Property Documentation	459
7.116.4.1 Alignment	459
7.116.4.2 ChildCount	459
7.116.4.3 ClassName	459
7.116.4.4 Enabled	459
7.116.4.5 ExternalEvent	459
7.116.4.6 Focused	459
7.116.4.7 Manager	459
7.116.4.8 Margins	460
7.116.4.9 Name	460
7.116.4.10 Parent	460
7.116.4.11 ParentZone	460
7.116.4.12 Payload	460
7.116.4.13 PropertyCount	460
7.116.4.14 Rect	460
7.116.4.15 Style	460
7.116.4.16 Visible	461
7.116.4.17 ZoneCount	461
7.117 Afterwarp.WidgetManager Class Reference	461
7.117.1 Detailed Description	465
7.117.2 Constructor & Destructor Documentation	465
7.117.2.1 WidgetManager()	465
7.117.3 Member Function Documentation	465
7.117.3.1 GetTexture()	465
7.117.3.2 GetWidget()	465
7.117.3.3 Present()	465
7.117.3.4 TryGetWidget() [1/2]	466
7.117.3.5 TryGetWidget() [2/2]	466
7.117.4 Property Documentation	466
7.117.4.1 Application	466
7.117.4.2 Attributes	466
7.117.4.3 BatchCount	466

7.117.4.4 Cursor	466
7.117.4.5 Format	466
7.117.4.6 Multisamples	467
7.117.4.7 Scale	467
7.117.4.8 TextRenderer	467
7.118 Afterwarp.WidgetManagerAttribute Struct Reference	467
7.118.1 Detailed Description	467
7.118.2 Member Data Documentation	467
7.118.2.1 Composition	467
7.118.2.2 Design	468
7.119 Afterwarp.WidgetProperty Struct Reference	468
7.119.1 Detailed Description	468
7.119.2 Member Data Documentation	468
7.119.2.1 Attributes	468
7.119.2.2 Behavior	469
7.119.2.3 DataBytes	469
7.119.2.4 Location	469
7.119.2.5 NameBytes	469
7.119.2.6 Reserved	469
7.119.2.7 Type	469
7.119.3 Property Documentation	469
7.119.3.1 Name	469
8 File Documentation	471
8.1 Afterwarp.API.cs File Reference	471
8.2 Afterwarp.Graphics.cs File Reference	471
8.3 Afterwarp.Types.cs File Reference	474
Index	483

Chapter 1

Afterwarp Framework

[Afterwarp](#) Framework ® is a cross-platform framework for development of scientific, industrial and interactive real-time applications. The framework includes both low-level API with full hardware abstraction, allowing development of custom graphics engine using programmable pipeline and shaders, including compute shaders for GPGPU work, and a high-level framework that can be used to quickly develop 3D applications without knowledge of shaders.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Afterwarp	17
-------------------------------------	----

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Afterwarp.AmbientOcclusionParameters	56
Afterwarp.API	59
Afterwarp.ApplicationConfiguration	68
Afterwarp.Canvas.Attribute	71
Afterwarp.Device.Attribute	72
Afterwarp.ObjectModel.Attribute	73
Afterwarp.Scene.Attribute	74
Afterwarp.Texture.Attribute	77
Afterwarp.TextureCabinet.Attribute	78
Afterwarp.AutoDrawOption	80
Afterwarp.RenderingState.Blend	82
Afterwarp.CameraConstraints	88
Afterwarp.CanvasSamplerState	112
Afterwarp.ColorPair	117
Afterwarp.ColorRect	119
Afterwarp.ComputeBindTextureFormat	122
CriticalFinalizerObject	
Afterwarp.CustomObject	128
Afterwarp.ActorCamera	51
Afterwarp.Application	59
Afterwarp.Buffer	84
Afterwarp.Canvas	89
Afterwarp.CanvasBuffer	109
Afterwarp.ColorDithering	114
Afterwarp.ComputeProgram	124
Afterwarp.Device	132
Afterwarp.GaussianBlur	159
Afterwarp.Grapher	162
Afterwarp.ImageAtlas	168
Afterwarp.KawaseBlur	178
Afterwarp.MeshBuffer	187
Afterwarp.MeshMetaTag	204
Afterwarp.MeshMetaTags	209
Afterwarp.MeshModel	214
Afterwarp.MeshVoxel	218

Afterwarp.ObjectMaterials	226
Afterwarp.ObjectModel	229
Afterwarp.ObjectModelView	242
Afterwarp.ObjectModels	237
Afterwarp.OceanSimulation	250
Afterwarp.PathBroker	258
Afterwarp.Program	274
Afterwarp.Sampler	308
Afterwarp.Scene	315
Afterwarp.SceneLights	326
Afterwarp.SceneMesh	331
Afterwarp.SceneMeshLatches	343
Afterwarp.SceneMeshMaterial	348
Afterwarp.SceneMeshMaterials	353
Afterwarp.SceneMeshes	336
Afterwarp.SelectionHighlight	360
Afterwarp.ShadowCaster	366
Afterwarp.ShadowCastingAtlas	370
Afterwarp.SpatialFog	378
Afterwarp.Surface	386
Afterwarp.SwapChain	395
Afterwarp.TextModeller	400
Afterwarp.TextModeller.FontProvider	157
Afterwarp.TextRenderer	407
Afterwarp.Texture	413
Afterwarp.TextureCabinet	421
Afterwarp.Timer	428
Afterwarp.Widget	452
Afterwarp.WidgetManager	461
Afterwarp.DeviceBehavior	136
Afterwarp.DeviceClear	138
System.Exception	
Afterwarp.API.Exception	139
Afterwarp.FloatColor	140
Afterwarp.FloatColorRGB	144
Afterwarp.FogParameters	148
Afterwarp.FontAttribute	150
Afterwarp.FontEffect	151
Afterwarp.FontParameters	155
IDisposable	
Afterwarp.CustomObject	128
IEnumerable	
Afterwarp.ObjectModels	237
Afterwarp.Widget	452
IEnumerator	
Afterwarp.ObjectModels.Iterator	175
Afterwarp.Widget.Iterator	176
IEquatable	
Afterwarp.CustomObject	128
Afterwarp.ImageRegion	172
Afterwarp.Library	180
Afterwarp.Margins	181
Afterwarp.MeshAligns	185
Afterwarp.MeshLoadingOption	202
Afterwarp.MeshMetaTagPortion	207
Afterwarp.MeshMetaTagType	213
Afterwarp.ObjectMaterial	222
Afterwarp.OceanMaterial	247

Afterwarp.OceanWavesParameters	254
Afterwarp.ParallaxMappingParameters	256
Afterwarp.PathBuilder	260
Afterwarp.PathCommand	264
Afterwarp.PathElement	266
Afterwarp.Point	268
Afterwarp.ProgramElement	279
Afterwarp.ProgramVariable	281
Afterwarp.Widget.PropertyValue	283
Afterwarp.Quad	284
Afterwarp.RandomSequence	289
Afterwarp.Ray	291
Afterwarp.Rect	293
Afterwarp.RectF	298
Afterwarp.RenderingState	304
Afterwarp.SamplerState	311
Afterwarp.SceneLight	322
Afterwarp.SceneMeshLatch	341
Afterwarp.SceneMeshLatchType	347
Afterwarp.SceneMeshMaterialRange	352
Afterwarp.SceneMeshMaterialShading	357
Afterwarp.SelectionHighlightParameters	364
Afterwarp.ShadowParameters	374
Afterwarp.SignedDistanceField	376
Afterwarp.RenderingState.State	381
Afterwarp.RenderingState.StencilState	384
Afterwarp.TextEntryRect	399
Afterwarp.TextRenderModifiers	412
Afterwarp.TextureParameters	426
Afterwarp.ToneMappingBloom	432
Afterwarp.Utility	435
Afterwarp.VertexElement	447
Afterwarp.MeshBuffer.VertexEntry	449
Afterwarp.Volume	451
Afterwarp.WidgetManagerAttribute	467
Afterwarp.WidgetProperty	468

Chapter 4

Class Index

4.1 Class List

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

Afterwarp.ActorCamera	51
Afterwarp.AmbientOcclusionParameters Parameters that define how ambient occlusion is performed	56
Afterwarp.API	59
Afterwarp.Application Application execution manager that handles OS events and controls application's window . . .	59
Afterwarp.ApplicationConfiguration Structure that contains all the parameters necessary to create a new application and its window	68
Afterwarp.Canvas.Attribute Canvas attribute flags that can be combined together	71
Afterwarp.Device.Attribute Device attributes that define its behavior	72
Afterwarp.ObjectModel.Attribute Attribute bit flags for an object that define its status	73
Afterwarp.Scene.Attribute Cumulative attributes that define rendering behavior of the scene	74
Afterwarp.Texture.Attribute Attribute that defines special behavior and/or unique characteristics of the texture	77
Afterwarp.TextureCabinet.Attribute Cumulative attributes that define rendering characteristics of the scene	78
Afterwarp.AutoDrawOption Cumulative options that can be passed to one of scene mesh "AutoDraw" helper functions . .	80
Afterwarp.RenderingState.Blend Blending parameters that are specific to a certain component, or a portion of color	82
Afterwarp.Buffer Generic device buffer object that can contain vertices, indices or shader data	84
Afterwarp.CameraConstraints	88
Afterwarp.Canvas	89
Afterwarp.CanvasBuffer	109
Afterwarp.CanvasSamplerState Sampler parameters that can be easily configured by the canvas	112
Afterwarp.ColorDithering	114
Afterwarp.ColorPair	117
Afterwarp.ColorRect	119
Afterwarp.ComputeBindTextureFormat Compute texture binding parameters	122

Afterwarp.ComputeProgram	124
Compute shader program for general processing on GPU (GPGPU)	
Afterwarp.CustomObject	128
A high-level class that wraps one of Afterwarp 's objects	
Afterwarp.Device	132
Device class that handles initialization, rendering, state management and other common tasks	
Afterwarp.DeviceBehavior	136
Device behavior attributes that define the rendering code path	
Afterwarp.DeviceClear	138
Type of surface layer that should be cleared	
Afterwarp.API.Exception	139
Exception type raised by Afterwarp classes	
Afterwarp.FloatColor	140
Afterwarp.FloatColorRGB	144
Afterwarp.FogParameters	148
Structure containing spatial fog characteristics	
Afterwarp.FontAttribute	150
Font attribute flags that define some visual characteristics	
Afterwarp.FontEffect	151
Attributes and characteristics that define how font glyphs are rasterized	
Afterwarp.FontParameters	155
Parameters that define the appearance and important characteristics of the font	
Afterwarp.TextModeller.FontProvider	157
A provider of font geometry for the modeller	
Afterwarp.GaussianBlur	159
Gaussian Blur module	
Afterwarp.Grapher	162
3D graph plotting module	
Afterwarp.ImageAtlas	168
Image atlas, sub-images of which can be rendered with canvas	
Afterwarp.ImageRegion	172
An image region defined by its texture number and bounding rectangle on that texture	
Afterwarp.ObjectModels.Iterator	175
Iterator for accessing individual objects in the container	
Afterwarp.Widget.Iterator	176
Iterator for accessing properties in the widget	
Afterwarp.KawaseBlur	178
Kawase Blur module	
Afterwarp.Library	180
Shared library service functions	
Afterwarp.Margins	181
Margins defined by top, left, right and bottom edge paddings	
Afterwarp.MeshAligns	185
Alignment for all three axes that determine the placement of mesh around its model	
Afterwarp.MeshBuffer	187
Buffer that contains 3D mesh information	
Afterwarp.MeshLoadingOption	202
Non-exclusive options passed and/or returned from native mesh loader	
Afterwarp.MeshMetaTag	204
Meta-tag, which describes a certain important axis-aligned bounding area inside a mesh	
Afterwarp.MeshMetaTagPortion	207
A portion of the mesh corresponding to a particular tag	
Afterwarp.MeshMetaTags	209
List of meta-tags that describe important axis-aligned bounding areas inside a mesh	
Afterwarp.MeshMetaTagType	213
Type of the mesh meta-tag	
Afterwarp.MeshModel	214
3D mesh model that contains both vertex and index buffers	

Afterwarp.MeshVoxel	
3D mesh voxel representation object	218
Afterwarp.ObjectMaterial	
Material that describes a particular object in 3D world	222
Afterwarp.ObjectMaterials	
Container for storing 3D object materials	226
Afterwarp.ObjectModel	
Object that represents a 3D model in the scene	229
Afterwarp.ObjectModels	
A container of objects and their volume representation in 3D world	237
Afterwarp.ObjectModelView	
A container that represents a particular view that watches objects in the world	242
Afterwarp.OceanMaterial	
Material that describes ocean water surface in a 3D simulation	247
Afterwarp.OceanSimulation	
3D ocean semi-infinite wave field rendering module	250
Afterwarp.OceanWavesParameters	
Parameters that define the appearance of waves simulation	254
Afterwarp.ParallaxMappingParameters	
Parameters that define how parallax mapping is performed	256
Afterwarp.PathBroker	258
Afterwarp.PathBuilder	
A container and helper module to facilitate creation of path elements	260
Afterwarp.PathCommand	
Path commands and attribute bits	264
Afterwarp.PathElement	
A single element in path declaration	266
Afterwarp.Point	
2D integer point	268
Afterwarp.Program	
Shader program that is typically executed on graphics hardware	274
Afterwarp.ProgramElement	
Structure that describes a single physical element of shader program	279
Afterwarp.ProgramVariable	
Structure that describes a single (uniform) variable in a shader program	281
Afterwarp.Widget.PropertyValue	
A widget iteration value, consisting of a property name, type, behavior and its value	283
Afterwarp.Quad	
Floating-point quadrilateral defined by four vertices in clockwise order starting from top/left	284
Afterwarp.RandomSequence	
Pseudo-random number generator (PRNG) utility module	289
Afterwarp.Ray	
A structure that defines a ray in 3D space	291
Afterwarp.Rect	
Rectangle defined by integer top and left margins, width and height	293
Afterwarp.RectF	
Rectangle defined by floating-point top and left margins, width and height	298
Afterwarp.RenderingState	
Parameters that define depth, stencil, rasterizer and blending operations	304
Afterwarp.Sampler	
Sampler object that defines how texture is read by the shaders	308
Afterwarp.SamplerState	
Sampler parameters that are associated with a particular texture unit	311
Afterwarp.Scene	
A module for rendering 3D scenes	315
Afterwarp.SceneLight	
Light source parameters in 3D scene	322

Afterwarp.SceneLights	Module for storing and working with lights in a 3D scene	326
Afterwarp.SceneMesh	A container for a 3D scene mesh model, its optional voxel representation and a bounding volume	331
Afterwarp.SceneMeshes	A collection of 3D scene mesh containers	336
Afterwarp.SceneMeshLatch	A latch in 3D scene mesh, which describes a bone joint connection or a movement waypoint .	341
Afterwarp.SceneMeshLatches	Container for scene mesh latches that define how meshes connect with each other	343
Afterwarp.SceneMeshLatchType	Type of latch used in the 3D scene mesh	347
Afterwarp.SceneMeshMaterial	Material that describes how a portion of a scene mesh geometry should look like	348
Afterwarp.SceneMeshMaterialRange	A range of indices in the mesh that a material applies to	352
Afterwarp.SceneMeshMaterials	Container for scene mesh materials that describe the appearance of scene mesh geometry . .	353
Afterwarp.SceneMeshMaterialShading	Shading parameters for scene mesh material	357
Afterwarp.SelectionHighlight		360
Afterwarp.SelectionHighlightParameters	Parameters that define how selection highlight technique is performed	364
Afterwarp.ShadowCaster	A source that casts shadows that can be shared among multiple light sources	366
Afterwarp.ShadowCastingAtlas	Module that manages shadow maps produced by one or more shadow casters	370
Afterwarp.ShadowParameters	Parameters that define how shadows are rendered	374
Afterwarp.SignedDistanceField	Signed Distance Field (SDF) parameters	376
Afterwarp.SpatialFog	Module that performs both ground fog and distance-based fog simultaneously	378
Afterwarp.RenderingState.State	State flags that can be combined together to form a rendering operation state	381
Afterwarp.RenderingState.StencilState	Stencil state that is independent for each front and back facing	384
Afterwarp.Surface	Surface that contains image in memory for pixel manipulation	386
Afterwarp.SwapChain	Swap-chain that enables double-buffered rendering to a particular window	395
Afterwarp.TextEntryRect	Individual text entry that will appear as rendered	399
Afterwarp.TextModeller	2D and 3D text rendering module	400
Afterwarp.TextRenderer	Text renderer utility that can draw text on the canvas	407
Afterwarp.TextRenderModifiers	Modifier attributes that can be applied to the rendered text	412
Afterwarp.Texture	Texture that contains image data typically stored in GPU memory	413
Afterwarp.TextureCabinet	A container and management module for drawable textures for rendering and processing 3D scene	421
Afterwarp.TextureParameters	Texture parameters and characteristics	426
Afterwarp.Timer		428

Afterwarp.ToneMappingBloom	
Parameters that define behavior of tone-mapping and bloom effects	432
Afterwarp.Utility	435
Afterwarp.VertexElement	
Structure that describes the layout of a single buffer element	447
Afterwarp.MeshBuffer.VertexEntry	
A vertex element of mesh buffer with interleaved data	449
Afterwarp.Volume	451
Afterwarp.Widget	
A widget module serving as a base element of user interface	452
Afterwarp.WidgetManager	
A service module that contains and manages all other widgets	461
Afterwarp.WidgetManagerAttribute	
Widget manager cumulative attribute flags that define its behavior	467
Afterwarp.WidgetProperty	
Information about a widget property	468

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Afterwarp.API.cs	471
Afterwarp.Graphics.cs	471
Afterwarp.Types.cs	474

Chapter 6

Namespace Documentation

6.1 Afterwarp Namespace Reference

Classes

- class [ActorCamera](#)
- struct [AmbientOcclusionParameters](#)
Parameters that define how ambient occlusion is performed.
- struct [API](#)
- class [Application](#)
Application execution manager that handles OS events and controls application's window.
- struct [ApplicationConfiguration](#)
Structure that contains all the parameters necessary to create a new application and its window.
- struct [ApplicationEvents](#)
- struct [AutoDrawOption](#)
Cumulative options that can be passed to one of scene mesh "AutoDraw" helper functions.
- class [Buffer](#)
Generic device buffer object that can contain vertices, indices or shader data.
- struct [CameraConstraints](#)
- class [Canvas](#)
- class [CanvasBuffer](#)
- struct [CanvasSamplerState](#)
Sampler parameters that can be easily configured by the canvas.
- class [ColorDithering](#)
- struct [ColorPair](#)
- struct [ColorRect](#)
- struct [ComputeBindTextureFormat](#)
Compute texture binding parameters.
- class [ComputeProgram](#)
Compute shader program for general processing on GPU (GPGPU).
- class [CustomObject](#)
A high-level class that wraps one of [Afterwarp](#)'s objects.
- class [Device](#)
Device class that handles initialization, rendering, state management and other common tasks.
- struct [DeviceBehavior](#)
Device behavior attributes that define the rendering code path.

- struct [DeviceClear](#)
Type of surface layer that should be cleared.
- struct [FloatColor](#)
- struct [FloatColorRGB](#)
- struct [FogParameters](#)
Structure containing spatial fog characteristics.
- struct [FontAttribute](#)
Font attribute flags that define some visual characteristics.
- struct [FontEffect](#)
Attributes and characteristics that define how font glyphs are rasterized.
- struct [FontParameters](#)
Parameters that define the appearance and important characteristics of the font.
- class [GaussianBlur](#)
Gaussian Blur module.
- class [Grapher](#)
3D graph plotting module.
- class [ImageAtlas](#)
Image atlas, sub-images of which can be rendered with canvas.
- struct [ImageRegion](#)
An image region defined by its texture number and bounding rectangle on that texture.
- class [KawaseBlur](#)
Kawase Blur module.
- struct [Library](#)
Shared library service functions.
- struct [Margins](#)
Margins defined by top, left, right and bottom edge paddings.
- struct [MeshAligns](#)
Alignment for all three axes that determine the placement of mesh around its model.
- class [MeshBuffer](#)
Buffer that contains 3D mesh information.
- struct [MeshLoadingOption](#)
Non-exclusive options passed and/or returned from native mesh loader.
- class [MeshMetaTag](#)
Meta-tag, which describes a certain important axis-aligned bounding area inside a mesh.
- struct [MeshMetaTagPortion](#)
A portion of the mesh corresponding to a particular tag.
- class [MeshMetaTags](#)
List of meta-tags that describe important axis-aligned bounding areas inside a mesh.
- struct [MeshMetaTagType](#)
Type of the mesh meta-tag.
- class [MeshModel](#)
3D mesh model that contains both vertex and index buffers.
- class [MeshVoxel](#)
3D mesh voxel representation object.
- struct [ObjectMaterial](#)
Material that describes a particular object in 3D world.
- class [ObjectMaterials](#)
Container for storing 3D object materials.
- class [ObjectModel](#)
Object that represents a 3D model in the scene.
- class [ObjectModels](#)

- A container of objects and their volume representation in 3D world.*
- class [ObjectModelView](#)
 - A container that represents a particular view that watches objects in the world.*
- struct [OceanMaterial](#)
 - Material that describes ocean water surface in a 3D simulation.*
- class [OceanSimulation](#)
 - 3D ocean semi-infinite wave field rendering module.*
- struct [OceanWavesParameters](#)
 - Parameters that define the appearance of waves simulation.*
- struct [ParallaxMappingParameters](#)
 - Parameters that define how parallax mapping is performed.*
- class [PathBroker](#)
- class [PathBuilder](#)
 - A container and helper module to facilitate creation of path elements.*
- struct [PathCommand](#)
 - Path commands and attribute bits.*
- struct [PathElement](#)
 - A single element in path declaration.*
- struct [Point](#)
 - 2D integer point.*
- class [Program](#)
 - Shader program that is typically executed on graphics hardware.*
- struct [ProgramElement](#)
 - Structure that describes a single physical element of shader program.*
- struct [ProgramVariable](#)
 - Structure that describes a single (uniform) variable in a shader program.*
- struct [Quad](#)
 - Floating-point quadrilateral defined by four vertices in clockwise order starting from top/left.*
- struct [RandomSequence](#)
 - Pseudo-random number generator (PRNG) utility module.*
- struct [Ray](#)
 - A structure that defines a ray in 3D space.*
- struct [Rect](#)
 - Rectangle defined by integer top and left margins, width and height.*
- struct [RectF](#)
 - Rectangle defined by floating-point top and left margins, width and height.*
- struct [RenderingState](#)
 - Parameters that define depth, stencil, rasterizer and blending operations.*
- class [Sampler](#)
 - Sampler object that defines how texture is read by the shaders.*
- struct [SamplerState](#)
 - Sampler parameters that are associated with a particular texture unit.*
- class [Scene](#)
 - A module for rendering 3D scenes.*
- struct [SceneLight](#)
 - Light source parameters in 3D scene.*
- class [SceneLights](#)
 - Module for storing and working with lights in a 3D scene.*
- class [SceneMesh](#)
 - A container for a 3D scene mesh model, its optional voxel representation and a bounding volume.*
- class [SceneMeshes](#)

- A collection of 3D scene mesh containers.*

 - struct [SceneMeshLatch](#)

A latch in 3D scene mesh, which describes a bone joint connection or a movement waypoint.
- class [SceneMeshLatches](#)

Container for scene mesh latches that define how meshes connect with each other.
- struct [SceneMeshLatchType](#)

Type of latch used in the 3D scene mesh.
- class [SceneMeshMaterial](#)

Material that describes how a portion of a scene mesh geometry should look like.
- struct [SceneMeshMaterialRange](#)

A range of indices in the mesh that a material applies to.
- class [SceneMeshMaterials](#)

Container for scene mesh materials that describe the appearance of scene mesh geometry.
- struct [SceneMeshMaterialShading](#)

Shading parameters for scene mesh material.
- class [SelectionHighlight](#)
- struct [SelectionHighlightParameters](#)

Parameters that define how selection highlight technique is performed.
- class [ShadowCaster](#)

A source that casts shadows that can be shared among multiple light sources.
- class [ShadowCastingAtlas](#)

Module that manages shadow maps produced by one or more shadow casters.
- struct [ShadowParameters](#)

Parameters that define how shadows are rendered.
- struct [SignedDistanceField](#)

Signed Distance Field (SDF) parameters.
- class [SpatialFog](#)

Module that performs both ground fog and distance-based fog simultaneously.
- class [Surface](#)

Surface that contains image in memory for pixel manipulation.
- class [SwapChain](#)

Swap-chain that enables double-buffered rendering to a particular window.
- struct [TextEntryRect](#)

Individual text entry that will appear as rendered.
- class [TextModeller](#)

2D and 3D text rendering module.
- class [TextRenderer](#)

Text renderer utility that can draw text on the canvas.
- struct [TextRenderModifiers](#)

Modifier attributes that can be applied to the rendered text.
- class [Texture](#)

Texture that contains image data typically stored in GPU memory.
- class [TextureCabinet](#)

A container and management module for drawable textures for rendering and processing 3D scene.
- struct [TextureParameters](#)

Texture parameters and characteristics.
- class [Timer](#)
- struct [ToneMappingBloom](#)

Parameters that define behavior of tone-mapping and bloom effects.
- struct [Utility](#)
- struct [VertexElement](#)

- *Structure that describes the layout of a single buffer element.*
- struct [Volume](#)
- class [Widget](#)
 - *A widget module serving as a base element of user interface.*
- class [WidgetManager](#)
 - *A service module that contains and manages all other widgets.*
- struct [WidgetManagerAttribute](#)
 - *Widget manager cumulative attribute flags that define its behavior.*
- struct [WidgetProperty](#)
 - *Information about a widget property.*

Enumerations

- enum [DeviceTechnology](#) : byte {
[Unknown](#) , [Direct3D](#) , [OpenGL](#) , [OpenGL_ES](#) ,
[Vulkan](#) , [Metal](#) , [WebGL](#) , [Software](#) ,
[Proprietary](#) }
Type of graphics technology used in the application.
- enum [DevicePlatform](#) : byte {
[Unknown](#) , [Windows](#) , [Linux](#) , [Unix](#) ,
[OSX](#) , [iOS](#) , [Android](#) }
Type of OS platform the application is running on.
- enum [PixelFormat](#) : byte {
[Unknown](#) , [R8](#) , [RG8](#) , [RGBA8](#) ,
[R16](#) , [RG16](#) , [RGBA16](#) , [R8S](#) ,
[RG8S](#) , [RGBA8S](#) , [R16S](#) , [RG16S](#) ,
[RGBA16S](#) , [R8U](#) , [RG8U](#) , [RGBA8U](#) ,
[R16U](#) , [RG16U](#) , [RGBA16U](#) , [R32U](#) ,
[RG32U](#) , [RGBA32U](#) , [R8I](#) , [RG8I](#) ,
[RGBA8I](#) , [R16I](#) , [RG16I](#) , [RGBA16I](#) ,
[R32I](#) , [RG32I](#) , [RGBA32I](#) , [R16F](#) ,
[RG16F](#) , [RGBA16F](#) , [R32F](#) , [RG32F](#) ,
[RGBA32F](#) , [RG11B10F](#) , [RGB9E5F](#) , [RGB10A2](#) ,
[RGB10A2U](#) , [RGBX8](#) , [RGBA8_SRGB](#) , [BGRA8](#) ,
[BGRX8](#) , [BGRA8_SRGB](#) , [BGR10A2](#) , [BGR10X2](#) ,
[BGRA4](#) , [BGRX4](#) , [B5G6R5](#) , [BGR5A1](#) ,
[BGR5X1](#) , [RGB8](#) , [BGR8](#) , [A8](#) ,
[L8](#) , [L16](#) , [LA4](#) , [LA8](#) ,
[I8](#) , [R_BC](#) , [RG_BC](#) , [RGB_BC](#) ,
[RGBA_BC](#) , [RGB_BC_SRGB](#) , [RGBA_BC_SRGB](#) , [D16](#) ,
[D24S8](#) , [D32F](#) , [D32S8F](#) }
Defines how individual pixels and their colors are encoded in images and textures.
- enum [AlphaFormatRequest](#) : byte { [DontCare](#) , [NonPremultiplied](#) , [Premultiplied](#) }
Defines how alpha-channel should be handled in the loaded image.
- enum [ElementFormat](#) : ushort {
[Undefined](#) , [Float](#) , [HalfFloat](#) , [Double](#) ,
[Int](#) , [UnsignedInt](#) , [Short](#) , [UnsignedShort](#) ,
[Byte](#) , [UnsignedByte](#) , [Float_11_11_10](#) }
Format in which a single value of the given element is represented.
- enum [ShaderType](#) : byte {
[Undefined](#) , [Fragment](#) , [Vertex](#) , [Geometry](#) ,
[Compute](#) , [Hull](#) , [Domain](#) }
Type of shader in graphics rendering pipeline.

- enum [ShaderElement](#) : byte {
[Undefined](#) , [Texture](#) , [ConstantBuffer](#) , [TypedBuffer](#) ,
[RWTypedBuffer](#) , [StructuredBuffer](#) , [RWStructuredBuffer](#) }
Type that characterizes shader element.
- enum [TriangleFace](#) : byte { [None](#) , [Back](#) , [Front](#) , [Both](#) }
Type of triangle face that should have operation applied to.
- enum [ComparisonFunc](#) : byte {
[Always](#) , [Less](#) , [LessEqual](#) , [Greater](#) ,
[GreaterEqual](#) , [Equal](#) , [NotEqual](#) , [Never](#) }
Function that defines how depth/stencil operands should be compared.
- enum [StencilOp](#) : byte {
[Keep](#) , [Zero](#) , [Replace](#) , [Invert](#) ,
[Increment](#) , [Decrement](#) , [IncrementWarp](#) , [DecrementWarp](#) }
Operation that should be performed on stencil value.
- enum [BlendFactor](#) : byte {
[One](#) , [Zero](#) , [SourceColor](#) , [InvSourceColor](#) ,
[SourceAlpha](#) , [InvSourceAlpha](#) , [DestColor](#) , [InvDestColor](#) ,
[DestAlpha](#) , [InvDestAlpha](#) , [SourceAlphaSat](#) , [ConstantColor](#) ,
[InvConstantColor](#) , [ConstantAlpha](#) , [InvConstantAlpha](#) , [SourceColor1](#) ,
[InvSourceColor1](#) , [SourceAlpha1](#) , [InvSourceAlpha1](#) }
Factor that determines how alpha-blending operand is considered.
- enum [BlendOp](#) : byte {
[Add](#) , [Subtract](#) , [InvSubtract](#) , [Minimum](#) ,
[Maximum](#) }
Operation that should be performed between two blending operands.
- enum [BufferDataTypes](#) : byte {
[Vertex](#) , [Index](#) , [Constant](#) , [Typed](#) ,
[RWTyped](#) , [Structured](#) , [RWStructured](#) }
Type of data that a generic buffer contains.
- enum [BufferAccessType](#) : byte {
[Default](#) , [Static](#) , [Dynamic](#) , [Compute](#) ,
[Staging](#) }
Type of access that is performed with mesh buffer.
- enum [ComputeTextureAccess](#) : byte { [Read](#) , [Write](#) , [ReadWrite](#) }
Compute program texture access type.
- enum [TextureFilter](#) : byte { [None](#) , [Nearest](#) , [Linear](#) , [Anisotropic](#) }
Filtering mode that defines how colors are sampled from the texture.
- enum [TextureAddress](#) : byte {
[Wrap](#) , [Clamp](#) , [Mirror](#) , [MirrorOnce](#) ,
[Border](#) }
Addressing mode that defines how texture coordinates outside of [0, 1] range are treated.
- enum [PrimitiveTopology](#) : byte {
[Unknown](#) , [Points](#) , [Lines](#) , [LineStrip](#) ,
[Triangles](#) , [TriangleStrip](#) , [LinesAdjacency](#) , [LineStripAdjacency](#) ,
[TrianglesAdjacency](#) , [TriangleStripAdjacency](#) , [Patches](#) }
Rendering primitive topology.
- enum [PathJoint](#) : byte {
[None](#) , [Simple](#) , [Miter](#) , [Bevel](#) ,
[Round](#) , [MiterBevel](#) }
Type of joint for connecting path lines.
- enum [LineCaps](#) : byte { [Butt](#) , [Square](#) , [Round](#) }
Type of caps that covers line end points.
- enum [PointShape](#) : byte {
[Square](#) , [Round](#) , [Triangle](#) , [Star](#) ,
[Cross](#) }

- Shape of point primitive.*

 - enum `TextureType` : byte { `Surface` , `CubeMap` , `Volume` }

Type of texture that defines how it is composed.
- enum `BlendingEffect` : byte {
`Undefined` = 0 , `Normal` , `Shadow` , `Add` ,
`Subtract` , `Multiply` , `InverseMultiply` , `SourceColor` ,
`SourceColorAdd` , `Copy` = 254 , `Custom` = 255 }

Blending effect that defines how primitives are rendered on drawing surface.
- enum `CanvasContextState` : byte { `FlatScene` , `FlatText` , `Projected` , `Undefined` = 255 }

One of possible defined context states that can be pre-configured by the canvas.
- enum `SuperSampleSDF` : byte { `NoSuperSample` , `SuperSample4x` , `SuperSample16x` }

Type of super-sampling used for rendering Signed Distance Field (SDF).
- enum `FontWeight` : byte {
`Thin` , `ExtraLight` , `Light` , `SemiLight` ,
`Normal` , `Medium` , `SemiBold` , `Bold` ,
`ExtraBold` , `Heavy` , `ExtraHeavy` }

Weight of the font (apparent thickness of glyphs).
- enum `FontStretch` : byte {
`UltraCondensed` , `ExtraCondensed` , `Condensed` , `SemiCondensed` ,
`Normal` , `SemiExpanded` , `Expanded` , `ExtraExpanded` ,
`UltraExpanded` }

Relative width of the font.
- enum `FontSlant` : byte { `None` , `Oblique` , `Italic` }

Font slant styles.
- enum `FontBorder` : byte { `None` , `Normal` , `SemiHeavy` , `Heavy` }

Border that outlines text glyph's shapes.
- enum `TextAlignment` : byte { `Start` , `Middle` , `End` }

Text alignment when drawing with certain functions.
- enum `ColorDitheringFormat` : byte {
`RGB8` , `RGB6` , `R5G6B5` , `RGB4` ,
`RG3B2` }

Format to be used as target for color dithering.
- enum `FogFormula` : byte { `Linear` , `Exponential` , `Ground` }

Defines type of formula used in fog calculation.
- enum `DepthFogDistance` : byte { `ZBased` , `EyeRelative` }

Defines how distance is calculated for depth fog.
- enum `TechniqueLighting` : byte { `Phong` , `Minnaert` , `CookTorrance` , `OrenNayer` }

Lighting technique.
- enum `TechniqueShadows` : byte { `None` , `ESM` , `ESM_Warp` , `EVSM` }

Shadow rendering technique.
- enum `ClustersCullingMode` : byte { `Quality` , `Performance` }

Light culling mode for clustered shading.
- enum `TextureFidelity` : byte { `Low` , `Medium` , `High` , `Extreme` }

Determines the choice of pixel formats for the container.
- enum `TextureCabinetType` : byte {
`Depth` = 0 , `Normals` = 1 , `ColorHDR` = 4 , `Color` = 5 ,
`LinearDepths` = 11 , `GlassyColor` = 12 , `GlassyCounts` = 13 , `GlassyComposite` = 14 }

Texture type used in the scene.
- enum `TextureCabinetPass` : byte { `Color` , `Glassy` , `Depth` }

Rendering pass that defines what textures are being used and how they are cleared.
- enum `TextureCabinetFilterType` : byte { `Occlusion` , `Bloom` , `Glassy` , `GlassyFog` }

Filter type that defines how textures are processed.

- enum [SceneTextureType](#) : byte { [Albedo](#) , [NormalMap](#) , [ParallaxMap](#) }
Type of texture used by the scene.
- enum [SceneSamplerType](#) : byte { [Albedo](#) , [NormalMap](#) , [ParallaxMap](#) , [ShadowMap](#) }
Type of sampler used by the scene.
- enum [SelectionHighlightTextureType](#) : byte { [Highlight](#) , [HighlightMask](#) }
Texture type used by the technique.
- enum [SceneMeshTexture](#) : byte { [Diffuse](#) = 0 , [Normals](#) = 1 , [Parallax](#) = 2 }
Type of texture used in scene mesh material.
- enum [ModelTransform](#) : byte {
[Local](#) , [LocalModel](#) , [LocalVolume](#) , [Global](#) ,
[GlobalModel](#) , [GlobalVolume](#) }
Type of transform as used by object models.
- enum [MeshAlign](#) : byte { [Positive](#) , [Origin](#) , [Negative](#) , [Unaligned](#) }
Axis alignment that determines the placement of mesh around its model.
- enum [ObjectModelViewCompare](#) : byte {
[Depth](#) , [DepthReverse](#) , [Ordered](#) , [Meshes](#) ,
[Objects](#) }
Type of comparison applied to an ordered list of objects.
- enum [CameraCommand](#) : byte {
[Movement](#) , [Dragging](#) , [Rotation](#) , [Update](#) ,
[Stop](#) }
Camera movement/rotation command.
- enum [ApplicationWindowState](#) : byte { [Windowed](#) , [Minimized](#) , [Maximized](#) , [FullScreen](#) }
Application's window state enumeration.
- enum [MouseEvent](#) : byte {
[Down](#) , [Move](#) , [Up](#) , [WheelUp](#) ,
[WheelDown](#) , [Enter](#) , [Leave](#) , [DoubleClick](#) }
Type of mouse event corresponding to the notification.
- enum [MouseButton](#) : byte { [None](#) , [Left](#) , [Right](#) , [Middle](#) }
Type of mouse button that corresponds to the notification.
- enum [KeyEvent](#) : byte { [Pressed](#) , [Released](#) , [Typing](#) }
Type of keyboard event corresponding to the notification.
- enum [ApplicationEvent](#) : byte {
[Activate](#) , [Deactivate](#) , [Minimize](#) , [Restore](#) ,
[Appearance](#) }
Type of application event corresponding to the notification.
- enum [FileChooserDialog](#) : byte { [OpenFile](#) , [SaveFile](#) , [Directory](#) }
Type of file chooser dialog type.
- enum [Key](#) : byte {
[Null](#) = 0x00 , [Home](#) = 0x02 , [End](#) = 0x03 , [PrintScreen](#) = 0x05 ,
[Backspace](#) = 0x08 , [Tab](#) = 0x09 , [Linefeed](#) = 0x0A , [Clear](#) = 0x0B ,
[Return](#) = 0x0D , [PageUp](#) = 0x0E , [PageDown](#) = 0x0F , [CapsLock](#) = 0x11 ,
[NumLock](#) = 0x12 , [Pause](#) = 0x13 , [ScrollLock](#) = 0x14 , [SysReq](#) = 0x15 ,
[Cancel](#) = 0x18 , [Insert](#) = 0x1A , [Escape](#) = 0x1B , [Left](#) = 0x1C ,
[Up](#) = 0x1D , [Right](#) = 0x1E , [Down](#) = 0x1F , [Space](#) = 0x20 ,
[KeyA](#) = 0x61 , [KeyB](#) = 0x62 , [KeyC](#) = 0x63 , [KeyD](#) = 0x64 ,
[KeyE](#) = 0x65 , [KeyF](#) = 0x66 , [KeyG](#) = 0x67 , [KeyH](#) = 0x68 ,
[KeyI](#) = 0x69 , [KeyJ](#) = 0x6A , [KeyK](#) = 0x6B , [KeyL](#) = 0x6C ,
[KeyM](#) = 0x6D , [KeyN](#) = 0x6E , [KeyO](#) = 0x6F , [KeyP](#) = 0x70 ,
[KeyQ](#) = 0x71 , [KeyR](#) = 0x72 , [KeyS](#) = 0x73 , [KeyT](#) = 0x74 ,
[KeyU](#) = 0x75 , [KeyV](#) = 0x76 , [KeyW](#) = 0x77 , [KeyX](#) = 0x78 ,
[KeyY](#) = 0x79 , [KeyZ](#) = 0x7A , [Delete](#) = 0x7F , [F1](#) = 0x80 ,
[F2](#) = 0x81 , [F3](#) = 0x82 , [F4](#) = 0x83 , [F5](#) = 0x84 ,
[F6](#) = 0x85 , [F7](#) = 0x86 , [F8](#) = 0x87 , [F9](#) = 0x88 ,

F10 = 0x89 , **F11** = 0x8A , **F12** = 0x8B , **ShiftLeft** = 0x98 ,
ShiftRight = 0x99 , **CtrlLeft** = 0x9A , **CtrlRight** = 0x9B , **AltLeft** = 0x9C ,
AltRight = 0x9D , **SuperLeft** = 0x9E , **SuperRight** = 0x9F , **Numpad0** = 0xA0 ,
Numpad1 = 0xA1 , **Numpad2** = 0xA2 , **Numpad3** = 0xA3 , **Numpad4** = 0xA4 ,
Numpad5 = 0xA5 , **Numpad6** = 0xA6 , **Numpad7** = 0xA7 , **Numpad8** = 0xA8 ,
Numpad9 = 0xA9 , **Multiply** = 0xAA , **Divide** = 0xAB , **Add** = 0xAC ,
Subtract = 0xAD , **Separator** = 0xAE , **Decimal** = 0xAF , **Sleep** = 0xC0 ,
VolumeUp = 0xCC , **VolumeDown** = 0xCE , **VolumeMute** = 0xCF , **MediaNextTrack** = 0xD0 ,
MediaPrevTrack = 0xD1 , **MediaStop** = 0xD2 , **MediaPlayPause** = 0xD3 , **Select** = 0xE0 ,
Print = 0xE1 , **Execute** = 0xE2 , **Help** = 0xE3 , **Apps** = 0xE4 }

Storage type for portable key constants.

- enum **AppCursor** : byte {
Blank = 0x00 , **Arrow** = 0x01 , **TextSelect** , **TextSelectVertical** ,
Wait , **ArrowWait** , **CrossHair** , **Cell** ,
LinkSelect , **NotAllowed** , **Help** , **Move** ,
Drag , **Dragging** , **ResizeVert** , **ResizeHoriz** ,
Resize1 , **Resize2** , **ResizeTop** , **ResizeBottom** ,
ResizeLeft , **ResizeRight** , **ResizeTopLeft** , **ResizeTopRight** ,
ResizeBottomRight , **ResizeBottomLeft** , **Undefined** = 255 }

Application cursor enumeration.

- enum **WidgetAlignment** : byte {
None , **Client** , **Left** , **Top** ,
Right , **Bottom** }

Alignment of the widget respective its parent.

- enum **WidgetPropertyType** : byte {
None , **Integer** , **Int64** , **Float** ,
Real , **Boolean** , **Point** , **Vector** ,
Rect , **Margins** , **Color** , **ColorPair** ,
ColorRect , **Enumeration** , **String** , **Font** }

Type of property that defines its purpose.

- enum **WidgetPropertyBehavior** : byte {
Normal , **Static** , **Indirect** , **Volatile** ,
Event }

Behavior of the property.

- enum **WidgetManagerTextureType** : byte { **Canvas** , **Screen** , **Stripes** }

Texture type used for the compositioning.

6.1.1 Enumeration Type Documentation

6.1.1.1 AlphaFormatRequest

```
enum Afterwarp.AlphaFormatRequest : byte
```

Defines how alpha-channel should be handled in the loaded image.

Enumerator

DontCare	Alpha-channel can be handled either way.
NonPremultiplied	Alpha-channel in the image should not be premultiplied. Under normal circumstances, this is the recommended approach as it preserves RGB color information in its original form. However, when using mipmapping for images that have alpha-channel, Premultiplied gives more accurate results.
Premultiplied	Alpha-channel in the image should be premultiplied. Under normal circumstances, this is not recommended as the image would lose information after RGB components are premultiplied by alpha (and for smaller alpha values, less information is preserved). However, when using mipmapping for images that have alpha-channel, this gives more accurate results.

6.1.1.2 AppCursor

enum `Afterwarp.AppCursor` : byte

[Application](#) cursor enumeration.

Enumerator

Blank	Blank (invisible) cursor.
Arrow	Standard arrow cursor.
TextSelect	Text selection cursor.
TextSelectVertical	
Wait	Wait cursor.
ArrowWait	Arrow cursor with wait icon.
CrossHair	Crosshair cursor.
Cell	Cell cursor.
LinkSelect	Hyperlink selection cursor.
NotAllowed	Not allowed cursor.
Help	Help cursor.
Move	Move cursor.
Drag	Drag cursor.
Dragging	Dragging cursor.
ResizeVert	Vertical resize cursor.
ResizeHoriz	Horizontal resize cursor.
Resize1	Diagonal north-east south-west resize cursor.
Resize2	Diagonal north-west south-east resize cursor.
ResizeTop	Top resize cursor.
ResizeBottom	Bottom resize cursor.
ResizeLeft	Left resize cursor.
ResizeRight	Right resize cursor.
ResizeTopLeft	Top/left resize cursor.
ResizeTopRight	Top/right resize cursor.
ResizeBottomRight	Bottom/right resize cursor.
ResizeBottomLeft	Bottom/left resize cursor.
Undefined	Undefined or unspecified cursor.

6.1.1.3 ApplicationEvent

enum `Afterwarp.ApplicationEvent` : byte

Type of application event corresponding to the notification.

Enumerator

Activate	Application has been activated.
Deactivate	Application has been de-activated.
Minimize	Application has been minimized.
Restore	Application has been restored.
Appearance	DPI changed or user switched between dark/white styles.

6.1.1.4 ApplicationWindowState

```
enum Afterwarp.ApplicationWindowState : byte
```

[Application](#)'s window state enumeration.

Enumerator

Windowed	Windows appears normally with its title and border.
Minimized	Window appears minimized, iconic or hidden.
Maximized	Window appears maximized to fill the screen's client area.
FullScreen	Window appears occupying the entire screen.

6.1.1.5 BlendFactor

```
enum Afterwarp.BlendFactor : byte
```

Factor that determines how alpha-blending operand is considered.

Enumerator

One	Blending factor is a constant that equals to one.
Zero	Blending factor is a constant that equals to zero.
SourceColor	Source color is used as a blending factor.
InvSourceColor	One minus source color is used as a blending factor.
SourceAlpha	Source alpha value is used as a blending factor.
InvSourceAlpha	One minus source alpha value is used as a blending factor.
DestColor	Destination color is used as a blending factor.
InvDestColor	One minus destination color is used as a blending factor.
DestAlpha	Destination alpha value is used as a blending factor.
InvDestAlpha	One minus destination alpha value is used as a blending factor.
SourceAlphaSat	Blending factor is calculated as $\min(\text{SrcAlpha}, \text{InvDestAlpha})$.
ConstantColor	Blending factor is a user-defined color constant.
InvConstantColor	Blending factor is one minus user-defined color constant.
ConstantAlpha	Blending factor is an alpha value from user-defined color constant.
InvConstantAlpha	Blending factor is one minus alpha value from user-defined color constant.
SourceColor1	Source color (from 2nd pixel shader output) is used as a blending factor.
InvSourceColor1	One minus source color (from 2nd pixel shader output) is used as a blending factor.
SourceAlpha1	Source alpha value (from 2nd pixel shader output) is used as a blending factor.
InvSourceAlpha1	One minus source alpha (from 2nd pixel shader output) value is used as a blending factor.

6.1.1.6 BlendingEffect

```
enum Afterwarp.BlendingEffect : byte
```

Blending effect that defines how primitives are rendered on drawing surface.

Enumerator

Undefined	Undefined blending effect. Canvas does not change blending parameters configured in the device context and uses whatever configuration is currently active.
Normal	"Normal" blending effect: $Cd = (Cs * As) + (Cd * (1 - As))$.
Shadow	"Shadow drawing" effect: $Cd = Cd * (1 - As)$.
Add	Additive blending effect: $Cd = (Cs * As) + Cd$.
Subtract	Subtractive blending effect: $Cd = (Cs * As) - Cd$.
Multiply	Multiplication blending effect: $Cd = Cs * Cd$.
InverseMultiply	Inverse multiplication effect: $Cd = (1 - Cs) * Cd$.
SourceColor	Source color blending effect: $Cd = (Cs * Cs) + (Cd * (1 - Cs))$.
SourceColorAdd	Source color additive blending effect: $Cd = (Cs * Cs) + Cd$.
Copy	Alpha-blending disabled, drawing is just a copy operation.
Custom	Custom alpha-blending as set up through device directly.

6.1.1.7 BlendOp

```
enum Afterwarp.BlendOp : byte
```

Operation that should be performed between two blending operands.

Enumerator

Add	Add source and destination values.
Subtract	Subtract source value from destination value.
InvSubtract	Subtract destination value from source value.
Minimum	Calculate minimum between source and destination values.
Maximum	Calculate maximum between source and destination values.

6.1.1.8 BufferAccessType

```
enum Afterwarp.BufferAccessType : byte
```

Type of access that is performed with mesh buffer.

Enumerator

Default	The contents of buffer are updated occasionally.
Static	The contents of buffer are defined during construction and typically remain unchanged.
Dynamic	The contents of buffer are changed very frequently.
Compute	The contents of the buffer are used by GPU exclusively.
Staging	The buffer is used for asynchronous transfers from GPU back to CPU.

6.1.1.9 BufferDataType

```
enum Afterwarp.BufferDataType : byte
```

Type of data that a generic buffer contains.

Enumerator

Vertex	Vertex information is stored in the buffer (Vertex Buffer).
Index	Index information is stored in the buffer (Index Buffer).
Constant	Shader constant information is stored in the buffer (Constant Buffer).
Typed	Read-only shader buffer contained typed information (TBO / Buffer).
RWTyped	Read-only shader buffer contained typed information (TBO / RWBuffer).
Structured	Read-only shader buffer containing structured information (SSBO / StructuredBuffer).
RWStructured	Read/write shader buffer containing structured information (SSBO / RWStructuredBuffer).

6.1.1.10 CameraCommand

```
enum Afterwarp.CameraCommand : byte
```

Camera movement/rotation command.

Enumerator

Movement	Start camera movement on XZ plane.
Dragging	Start camera dragging.
Rotation	Start camera rotation.
Update	Update camera movement/rotation.
Stop	Stop camera movement/rotation.

6.1.1.11 CanvasContextState

```
enum Afterwarp.CanvasContextState : byte
```

One of possible defined context states that can be pre-configured by the canvas.

Enumerator

FlatScene	Geometric shapes and images rendered in a flat 2D scene.
FlatText	Text rendered in a flat 2D scene.
Projected	Geometric shapes, images and text rendered in a 3D surface projection.
Undefined	Undefined context state.

6.1.1.12 ClustersCullingMode

```
enum Afterwarp.ClustersCullingMode : byte
```

Light culling mode for clustered shading.

Enumerator

Quality	High-quality mode supporting up to 1024 simultaneous lights per cluster.
Performance	Performance mode supporting up to 4 simultaneous lights per cluster.

6.1.1.13 ColorDitheringFormat

```
enum Afterwarp.ColorDitheringFormat : byte
```

Format to be used as target for color dithering.

Enumerator

RGB8	Red, green and blue channels have 8 bits each (16M colors).
RGB6	Red, green and blue channels have 6 bits each (256K colors).
R5G6B5	Red and blue channels have 5 bits each, while green channel has 6 bits (64K colors).
RGB4	Red, green and blue channels have 4 bits each (4K colors).
RG3B2	Red and green channels have 3 bits each, while blue channel has 2 bits (256 colors).

6.1.1.14 ComparisonFunc

```
enum Afterwarp.ComparisonFunc : byte
```

Function that defines how depth/stencil operands should be compared.

Enumerator

Always	Comparison is always true.
Less	Comparison results true when source is less than destination.
LessEqual	Comparison results true when source is less than or equal to destination.
Greater	Comparison results true when source is greater than destination.
GreaterEqual	Comparison results true when source is greater than or equal to destination.
Equal	Comparison results true when source equals to destination.
NotEqual	Comparison results true when source is not equal to destination.
Never	Comparison is always false.

6.1.1.15 ComputeTextureAccess

```
enum Afterwarp.ComputeTextureAccess : byte
```

Compute program texture access type.

Enumerator

Read	Texture will be read from, no writes allowed.
Write	Texture will only be written to, no reads allowed.
ReadWrite	Texture will be read from and written to.

6.1.1.16 DepthFogDistance

```
enum Afterwarp.DepthFogDistance : byte
```

Defines how distance is calculated for depth fog.

Enumerator

ZBased	Distance is based on depth.
EyeRelative	Distance is relative to camera position.

6.1.1.17 DevicePlatform

```
enum Afterwarp.DevicePlatform : byte
```

Type of OS platform the application is running on.

Enumerator

Unknown	The platform could not be properly identified.
Windows	Microsoft Windows platform.
Linux	Linux platform.
Unix	Generic Unix platform.
OSX	Apple OS X platform.
iOS	Apple iOS platform.
Android	Google Android platform.

6.1.1.18 DeviceTechnology

```
enum Afterwarp.DeviceTechnology : byte
```

Type of graphics technology used in the application.

Enumerator

Unknown	The technology has not yet been established.
Direct3D	Microsoft Direct3D technology.
OpenGL	OpenGL by Khronos Group.
OpenGL_ES	OpenGL ES by Khronos Group.
Vulkan	Vulkan API by Khronos Group.
Metal	Metal API by Apple.
WebGL	WebGL by Khronos Group.
Software	Software rasterizer.
Proprietary	Private proprietary technology.

6.1.1.19 ElementFormat

```
enum Afterwarp.ElementFormat : ushort
```

Format in which a single value of the given element is represented.

Enumerator

Undefined	Undefined element format (VertexElement::count will define number of actual bytes).
Float	32-bit floating-point format.
HalfFloat	16-bit floating-point format.
Double	64-bit floating-point format.
Int	32-bit signed integer format.
UnsignedInt	32-bit unsigned integer format.
Short	16-bit signed integer format.
UnsignedShort	16-bit unsigned integer format.
Byte	8-bit signed integer format.
UnsignedByte	8-bit unsigned integer format.
Float_11_11_10	32-bit storage for two 11-bit and one 10-bit floating-point numbers. The number of elements for this format must always be set to three (3).

6.1.1.20 FileChooserDialog

```
enum Afterwarp.FileChooserDialog : byte
```

Type of file chooser dialog type.

Enumerator

OpenFile	Open file dialog.
SaveFile	
Directory	Select directory dialog.

6.1.1.21 FogFormula

```
enum Afterwarp.FogFormula : byte
```

Defines type of formula used in fog calculation.

Enumerator

Linear	Fog calculated as $(far - d) / (far - near)$.
Exponential	Fog calculated as $c \times e^{-(d \times b)^2}$.
Ground	Ground fog with linear density and exponential depth.

6.1.1.22 FontBorder

```
enum Afterwarp.FontBorder : byte
```

Border that outlines text glyph's shapes.

Enumerator

None	No border will appear around letters.
Normal	A light border will appear around letters.
SemiHeavy	A semi-heavy border will appear around letters.
Heavy	A heavy border will appear around letters.

6.1.1.23 FontSlant

```
enum Afterwarp.FontSlant : byte
```

Font slant styles.

Enumerator

None	Upright font.
Oblique	Slanted font in a roman style.
Italic	Slanted in an italic style.

6.1.1.24 FontStretch

```
enum Afterwarp.FontStretch : byte
```

Relative width of the font.

Enumerator

UltraCondensed	Ultra-condensed width.
ExtraCondensed	Extra-condensed width.
Condensed	Condensed width.
SemiCondensed	Semi-condensed width.
Normal	Normal width.
SemiExpanded	Semi-expanded width.
Expanded	Expanded width.
ExtraExpanded	Extra-expanded width.
UltraExpanded	Ultra-expanded width.

6.1.1.25 FontWeight

```
enum Afterwarp.FontWeight : byte
```

Weight of the font (apparent thickness of glyphs).

Enumerator

Thin	Letters should appear as thin as technically possible.
ExtraLight	Letters should appear very thin.
Light	Letters should appear even thinner.
SemiLight	Letters should have slightly lighter thickness.
Normal	Letters should appear normal (default thickness).
Medium	Letters should have medium thickness, which is slightly above than normal.
SemiBold	Letters should appear semi-bold.
Bold	Letters should appear as bold.
ExtraBold	Letters should appear as extra thick.
Heavy	Letters should appear very thick.
ExtraHeavy	Letters should appear as thick as technically possible.

6.1.1.26 Key

```
enum Afterwarp.Key : byte
```

Storage type for portable key constants.

Enumerator

Null	
Home	
End	
PrintScreen	
Backspace	
Tab	
Linefeed	
Clear	
Return	
PageUp	
PageDown	
CapsLock	
NumLock	
Pause	
ScrollLock	
SysReq	
Cancel	
Insert	
Escape	
Left	
Up	
Right	
Down	
Space	
KeyA	
KeyB	
KeyC	

Enumerator

KeyD	
KeyE	
KeyF	
KeyG	
KeyH	
KeyI	
KeyJ	
KeyK	
KeyL	
KeyM	
KeyN	
KeyO	
KeyP	
KeyQ	
KeyR	
KeyS	
KeyT	
KeyU	
KeyV	
KeyW	
KeyX	
KeyY	
KeyZ	
Delete	
F1	
F2	
F3	
F4	
F5	
F6	
F7	
F8	
F9	
F10	
F11	
F12	
ShiftLeft	
ShiftRight	
CtrlLeft	
CtrlRight	
AltLeft	
AltRight	
SuperLeft	
SuperRight	
Numpad0	
Numpad1	
Numpad2	
Numpad3	

Enumerator

Numpad4	
Numpad5	
Numpad6	
Numpad7	
Numpad8	
Numpad9	
Multiply	
Divide	
Add	
Subtract	
Separator	
Decimal	
Sleep	
VolumeUp	
VolumeDown	
VolumeMute	
MediaNextTrack	
MediaPrevTrack	
MediaStop	
MediaPlayPause	
Select	
Print	
Execute	
Help	
Apps	

6.1.1.27 KeyEvent

```
enum Afterwarp.KeyEvent : byte
```

Type of keyboard event corresponding to the notification.

Enumerator

Pressed	Key has been pressed.
Released	Key has been released.
Typing	Key is being typed (for Unicode input).

6.1.1.28 LineCaps

```
enum Afterwarp.LineCaps : byte
```

Type of caps that covers line end points.

Enumerator

Butt	Cap that starts and ends exactly at the center of end points.
Square	Cap that extends by half thickness after the end points.
Round	Round cap with a radius equal to half thickness with center being each line end point.

6.1.1.29 MeshAlign

```
enum Afterwarp.MeshAlign : byte
```

Axis alignment that determines the placement of mesh around its model.

Enumerator

Positive	Mesh is aligned from the edge of positive axis.
Origin	Mesh is aligned in the middle of axis origin.
Negative	Mesh is aligned from the edge of negative axis.
Unaligned	Mesh is unaligned on the axis.

6.1.1.30 ModelTransform

```
enum Afterwarp.ModelTransform : byte
```

Type of transform as used by object models.

Enumerator

Local	Local object transform.
LocalModel	Local Model object transform (for rendering).
LocalVolume	Local Volume object transform (OOBB Unity Cube).
Global	Global object transform.
GlobalModel	Global Model object transform (for rendering).
GlobalVolume	Global Volume object transform (OOBB Unity Cube).

6.1.1.31 MouseButton

```
enum Afterwarp.MouseButton : byte
```

Type of mouse button that corresponds to the notification.

Enumerator

None	None or unknown button (typically specified during mouse move event).
Left	Left mouse button.
Right	Right mouse button.
Middle	Middle mouse button.

6.1.1.32 MouseEvent

```
enum Afterwarp.MouseEvent : byte
```

Type of mouse event corresponding to the notification.

Enumerator

Down	Mouse button is pressed.
Move	Mouse cursor is being moved.
Up	Mouse button is de-pressed.
WheelUp	Mouse wheel has been rotated up.
WheelDown	Mouse wheel has been rotated down.
Enter	Mouse has entered component's area.
Leave	Mouse has left component's area.
DoubleClick	Mouse double-click event.

6.1.1.33 ObjectModelViewCompare

```
enum Afterwarp.ObjectModelViewCompare : byte
```

Type of comparison applied to an ordered list of objects.

Enumerator

Depth	Objects are compared by their depth.
DepthReverse	Objects are compared by their depth in reverse (for rendering transparency without OIT).
Ordered	Objects are compared first by their order index, then by depth.
Meshes	Objects are compared first by their mesh and then by depth.
Objects	Objects are sorted by their instance type id, mesh and then material.

6.1.1.34 PathJoint

```
enum Afterwarp.PathJoint : byte
```

Type of joint for connecting path lines.

Enumerator

None	No joint is produced, lines on each joint end with a butt cap.
Simple	Simple interpolated joint that does not guarantee correct line width on sharp corners.
Miter	Miter joint with angular corners.
Bevel	Beveled joint.
Round	Round joint.
MiterBevel	Miter joint unless the miter would extend beyond its limit multiplied by half thickness.

6.1.1.35 PixelFormat

```
enum Afterwarp.PixelFormat : byte
```

Defines how individual pixels and their colors are encoded in images and textures.

Enumerator

Unknown	Unknown pixel format.
R8	8-bit unsigned normalized format with red channel only.
RG8	16-bit format with 8-bit unsigned normalized red and green channels.
RGBA8	32-bit format with 8-bit unsigned normalized red, green, blue and alpha channels.
R16	8-bit unsigned normalized format with red channel only.
RG16	32-bit format with 8-bit unsigned normalized red and green channels.
RGBA16	64-bit ARGB pixel format with each channel having 16 bits.
R8S	8-bit signed normalized format with red channel only.
RG8S	16-bit format with 8-bit signed normalized red and green channels.
RGBA8S	32-bit format with 8-bit signed normalized red, green, blue and alpha channels.
R16S	16-bit signed normalized format with red channel only.
RG16S	32-bit format with 16-bit signed normalized red and green channels.
RGBA16S	64-bit format with 16-bit signed normalized red, green, blue and alpha channels.
R8U	8-bit format with 8-bit unsigned integer red channel.
RG8U	16-bit format with 8-bit unsigned integer red and green channels.
RGBA8U	32-bit format with 8-bit unsigned integer red and green channels.
R16U	16-bit format with 16-bit unsigned integer red channel.
RG16U	32-bit format with 16-bit unsigned integer red and green channels.
RGBA16U	64-bit format with 16-bit unsigned integer red and green channels.
R32U	32-bit format with 32-bit unsigned integer red channel.
RG32U	64-bit format with 32-bit unsigned integer red and green channels.
RGBA32U	128-bit format with 32-bit unsigned integer red and green channels.
R8I	8-bit format with 8-bit signed integer red channel.
RG8I	16-bit format with 8-bit signed integer red and green channels.
RGBA8I	32-bit format with 8-bit signed integer red and green channels.
R16I	16-bit format with 16-bit signed integer red channel.
RG16I	32-bit format with 16-bit signed integer red and green channels.
RGBA16I	64-bit format with 16-bit signed integer red and green channels.
R32I	32-bit format with 32-bit signed integer red channel.
RG32I	64-bit format with 32-bit signed integer red and green channels.
RGBA32I	128-bit format with 16-bit unsigned integer red and green channels.
R16F	16-bit floating-point pixel format, which has only one component.
RG16F	32-bit floating-point pixel format containing two components with 16 bits each.
RGBA16F	64-bit floating-point ARGB pixel format with each component having 16 bits.
R32F	32-bit floating-point pixel format, which has only one component.
RG32F	64-bit floating-point pixel format containing two components with 32 bits each.
RGBA32F	128-bit floating-point ARGB pixel format with each component having 32 bits.
RG11B10F	32-bit format with 11-bit unsigned floating-point red and green channels, and 10-bit blue channel.
RGB9E5F	32-bit format with 9-bit floating-point red, green and blue channels with shared 5-bit exponent.

Enumerator

RGB10A2	32-bit format with 10-bit unsigned normalized red, green and blue channels, and 2-bit alpha-channel.
RGB10A2U	32-bit format with 10-bit unsigned integer red, green and blue channels, and 2-bit alpha-channel.
RGBX8	32-bit format with 8-bit unsigned normalized red, green and blue channels (8 bits unused).
RGBA8_SRGB	32-bit format with 8-bit unsigned normalized red, green, blue and alpha channels with sRGB curve.
BGRA8	32-bit format with 8-bit unsigned normalized blue, green, red and alpha channels.
BGRX8	32-bit format with 8-bit unsigned normalized blue, green and red channels (8 bits unused).
BGRA8_SRGB	32-bit format with 8-bit unsigned normalized blue, green, red and alpha channels with sRGB curve.
BGR10A2	32-bit format with 10-bit unsigned normalized blue, green and red channels, and 2-bit alpha-channel.
BGR10X2	32-bit format with 10-bit unsigned normalized blue, green and red channels (2 bits unused).
BGRA4	16-bit format with 4-bit unsigned normalized blue, green, red and alpha channels.
BGRX4	16-bit format with 4-bit unsigned normalized blue, green and red channels (4 bits unused).
B5G6R5	16-bit format with 5-bit unsigned normalized blue and red, and 6 bits for green channels.
BGR5A1	16-bit format with 5-bit unsigned normalized blue, green and red, 1 bit for alpha channels.
BGR5X1	16-bit format with 5-bit unsigned normalized blue, green and red channels (1 bit unused).
RGB8	24-bit format with 8-bit unsigned normalized red, green and blue channels.
BGR8	24-bit format with 8-bit unsigned normalized blue, green and red channels.
A8	8-bit alpha-channel pixel format.
L8	8-bit luminance pixel format.
L16	16-bit luminance pixel format.
LA4	8-bit luminance/alpha pixel format with 4 bits per channel.
LA8	16-bit luminance/alpha pixel format with 8 bits per channel.
I8	8-bit palette indexed format.
R_BC	Compressed format with red channel only.
RG_BC	Compressed format with red and green channels.
RGB_BC	Compressed format with red, green and blue channels.
RGBA_BC	Compressed format with red, green, blue and alpha channels.
RGB_BC_SRGB	Compressed format with red, green and blue channels with sRGB curve.
RGBA_BC_SRGB	Compressed format with red, green and blue channels with sRGB curve, and an alpha-channel.
D16	Depth/stencil format with 16-bit (unsigned normalized) depth values and no storage for stencil.
D24S8	Depth/stencil format with 24-bit (unsigned normalized) depth values and 8 bits for stencil.
D32F	Depth/stencil format with 32-bit (floating-point) depth values and no storage for stencil.
D32S8F	Depth/stencil format with 32-bit (floating-point) depth values and 8 bits for stencil.

6.1.1.36 PointShape

```
enum Afterwarp.PointShape : byte
```

Shape of point primitive.

Enumerator

Square	Square point.
Round	Round point.
Triangle	Equilateral triangle point.
Star	5-sided star point.
Cross	Cross point.

6.1.1.37 PrimitiveTopology

```
enum Afterwarp.PrimitiveTopology : byte
```

Rendering primitive topology.

Enumerator

Unknown	Unknown (undefined) topology.
Points	Points or point lists.
Lines	Lines or line lists.
LineStrip	Line strips.
Triangles	Triangles or triangle lists.
TriangleStrip	Triangle strips.
LinesAdjacency	Lines or line lists with adjacency information.
LineStripAdjacency	Line strips with adjacency information.
TrianglesAdjacency	Triangles or triangle lists with adjacency information.
TriangleStripAdjacency	Triangle strips with adjacency information.
Patches	Tessellation patches.

6.1.1.38 SceneMeshTexture

```
enum Afterwarp.SceneMeshTexture : byte
```

Type of texture used in scene mesh material.

Enumerator

Diffuse	Diffuse color texture.
Normals	Normal-mapping texture.
Parallax	Parallax-mapping texture.

6.1.1.39 SceneSamplerType

```
enum Afterwarp.SceneSamplerType : byte
```

Type of sampler used by the scene.

Enumerator

Albedo	Albedo sampler.
NormalMap	Normal Map sampler.
ParallaxMap	Displacement Map sampler for Parallax Mapping technique.
ShadowMap	Shadow Map sampler.

6.1.1.40 SceneTextureType

```
enum Afterwarp.SceneTextureType : byte
```

Type of texture used by the scene.

Enumerator

Albedo	Albedo texture.
NormalMap	Normal Map texture.
ParallaxMap	Displacement Map texture for Parallax Mapping technique.

6.1.1.41 SelectionHighlightTextureType

```
enum Afterwarp.SelectionHighlightTextureType : byte
```

[Texture](#) type used by the technique.

Enumerator

Highlight	Texture that contains the selection highlight, to be rendered on top of the scene.
HighlightMask	Texture that contains the highlight mask, that can be optionally rendered on top of the scene.

6.1.1.42 ShaderElement

```
enum Afterwarp.ShaderElement : byte
```

Type that characterizes shader element.

Enumerator

Undefined	Undefined element type.
Texture	Element is a texture.
ConstantBuffer	Element is a constant buffer (uniform buffer object).
TypedBuffer	Element is a typed buffer (TBO / Buffer). Shares channels with textures.
RWTypedBuffer	Element is a typed buffer (TBO / RWBuffer). Shares channels with textures. For writing under OpenGL, this is bound as ImageBuffer (not TBO). For writing under Direct3D, this is bound as an Unordered Access View (UAV).
StructuredBuffer	Element is a compute shader buffer (SSBO / StructuredBuffer).
RWStructuredBuffer	Element is a compute shader buffer (SSBO / RWStructuredBuffer).

6.1.1.43 ShaderType

```
enum Afterwarp.ShaderType : byte
```

Type of shader in graphics rendering pipeline.

Enumerator

Undefined	Undefined or unknown shader.
Fragment	Fragment (pixel) shader.
Vertex	Vertex shader.
Geometry	Geometry shader.
Compute	Compute shader.
Hull	Hull shader.
Domain	Domain shader.

6.1.1.44 StencilOp

```
enum Afterwarp.StencilOp : byte
```

Operation that should be performed on stencil value.

Enumerator

Keep	Preserve destination value.
Zero	Set destination value to zero.
Replace	Replace destination value with the source one.
Invert	Invert the destination value.
Increment	Increment the destination value by the source one and clamp, if necessary.
Decrement	Decrement the destination value by the source one and clamp, if necessary.
IncrementWarp	Increment the destination value by the source one with wrap-around.
DecrementWarp	Decrement the destination value by the source one with wrap-around.

6.1.1.45 SuperSampleSDF

```
enum Afterwarp.SuperSampleSDF : byte
```

Type of super-sampling used for rendering Signed Distance Field (SDF).

Enumerator

NoSuperSample	Non-supersampled rendering (for low-performance systems). Note that if outline is enabled, its color will be calculated using simplified scheme, where luma is calculated as usual, chroma is calculated in a simple fashion and hue is not affected.
SuperSample4x	4x super-sampled rendering.
SuperSample16x	16x super-sampled rendering.

6.1.1.46 TechniqueLighting

```
enum Afterwarp.TechniqueLighting : byte
```

Lighting technique.

Enumerator

Phong	Traditional Phong lighting.
Minnaert	Minnaert lighting.
CookTorrance	Cook-Torrance lighting.
OrenNayer	Oren-Nayer lighting.

6.1.1.47 TechniqueShadows

```
enum Afterwarp.TechniqueShadows : byte
```

Shadow rendering technique.

Enumerator

None	No shadows are rendered or used.
ESM	Exponential Shadow Maps (ESM) using a single depth component.
ESM_Warp	Exponential Shadow Maps (ESM) using two depth components for positive and negative parts.
EVSM	Exponential Variance Shadow Maps (EVSM) using four depth components.

6.1.1.48 TextAlignment

```
enum Afterwarp.TextAlignment : byte
```

Text alignment when drawing with certain functions.

Enumerator

Start	Text should be aligned to the beginning (either top or left depending on context).
Middle	Text should be centered in the middle.
End	Text should be aligned to the end (either bottom or right depending on context).

6.1.1.49 TextureAddress

```
enum Afterwarp.TextureAddress : byte
```

Addressing mode that defines how texture coordinates outside of [0, 1] range are treated.

Enumerator

Wrap	Texture coordinates are wrapped; that is, only fractional part is considered.
------	---

Enumerator

Clamp	Texture coordinates are clamped to stay within [0, 1] range.
Mirror	Texture coordinates are mirrored in each odd cycle (e.g. from 1 to 2, from 3 to 4 and so on).
MirrorOnce	Texture coordinates are mirrored for one cycle and then clamped.
Border	When texture coordinates are outside of [0, 1] range, a separate border color will be used instead.

6.1.1.50 TextureCabinetFilterType

```
enum Afterwarp.TextureCabinetFilterType : byte
```

Filter type that defines how textures are processed.

Enumerator

Occlusion	Ambient occlusion computation.
Bloom	Bloom (glare) effect processing.
Glassy	Order-independent transparency (OIT) composition.
GlassyFog	Spatial fog effects for order-independent transparency (OIT) composition.

6.1.1.51 TextureCabinetPass

```
enum Afterwarp.TextureCabinetPass : byte
```

Rendering pass that defines what textures are being used and how they are cleared.

Enumerator

Color	Main color scene pass.
Glassy	Order-independent transparency (OIT) post-pass.
Depth	Depth (and normals) pre-pass.

6.1.1.52 TextureCabinetType

```
enum Afterwarp.TextureCabinetType : byte
```

[Texture](#) type used in the scene.

Enumerator

Depth	Depth buffer of the scene.
Normals	Texture containing information about screen-space normals.
ColorHDR	High dynamic range color accumulation buffer.
Color	Color buffer of the scene (non-MSAA).
LinearDepths	Linear depths (MSAA).
GlassyColor	Color accumulation buffer.
GlassyCounts	Count accumulation buffer.
GlassyComposite	Auxiliary texture used for compositioning.

6.1.1.53 TextureFidelity

enum `Afterwarp.TextureFidelity` : byte

Determines the choice of pixel formats for the container.

Enumerator

Low	Use lowest quality texture formats, which may reduce GPU memory consumption and potentially improve performance slightly at expense of visual quality.
Medium	Use standard quality texture formats suitable for most common usage scenarios.
High	High quality texture formats, producing accurate and exceptionally looking images.
Extreme	Excessively high quality texture formats, typically useful for generating images to be printed, screenshots and presentations. This may increase GPU memory consumption and impact performance.

6.1.1.54 TextureFilter

enum `Afterwarp.TextureFilter` : byte

Filtering mode that defines how colors are sampled from the texture.

Enumerator

None	No filtering is performed (applies only to mipmap filter, in which case it gets disabled).
Nearest	Nearest integer sample filtering.
Linear	Linear interpolation filtering.
Anisotropic	Linear anisotropic filtering.

6.1.1.55 TextureType

enum `Afterwarp.TextureType` : byte

Type of texture that defines how it is composed.

Enumerator

Surface	Standard 2D texture surface.
CubeMap	Cube Map consisting of 6 individual 2D faces.
Volume	Layered 3D volume texture.

6.1.1.56 TriangleFace

enum `Afterwarp.TriangleFace` : byte

Type of triangle face that should have operation applied to.

Enumerator

None	Neither back nor front faces should be processed.
Back	Back faces should be processed.
Front	Front faces should be processed.
Both	Both back and front faces should be processed.

6.1.1.57 WidgetAlignment

```
enum Afterwarp.WidgetAlignment : byte
```

Alignment of the widget respective its parent.

Enumerator

None	No alignment is applied to widget.
Client	Widget occupies the entire client area.
Left	Widget is placed on the left occupying the entire client height.
Top	Widget is placed on the top occupying the entire client width.
Right	Widget is placed on the right occupying the entire client height.
Bottom	Widget is placed on the bottom occupying the entire client width.

6.1.1.58 WidgetManagerTextureType

```
enum Afterwarp.WidgetManagerTextureType : byte
```

[Texture](#) type used for the compositioning.

Enumerator

Canvas	Texture with multisampling that is used for rendering using canvas.
Screen	Final compositioning texture.
Stripes	Texture containing relief stripes.

6.1.1.59 WidgetPropertyBehavior

```
enum Afterwarp.WidgetPropertyBehavior : byte
```

Behavior of the property.

Enumerator

Normal	Standard property behavior.
Static	Property wraps an existing variable of the parent.
Indirect	Property is accessed through callback events.
Volatile	Property is considered temporary or "volatile".
Event	Property represents an event.

6.1.1.60 WidgetPropertyType

enum `Afterwarp.WidgetPropertyType` : `byte`

Type of property that defines its purpose.

Enumerator

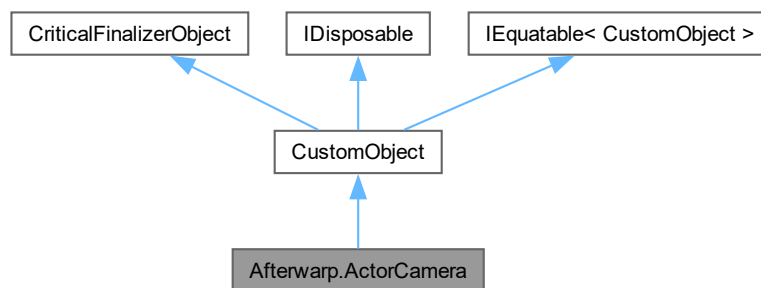
None	Unknown or undefined type. This may also refer to an event rather than property.
Integer	32-bit signed integer.
Int64	64-bit signed integer.
Float	32-bit floating-point type.
Real	64-bit floating-point type.
Boolean	Boolean value (either yes/no or true/false).
Point	Floating-point 2D vector.
Vector	Floating-point 3D vector.
Rect	Rectangle defined by four floating-point values: left, top, width and height.
Margins	Margins defined by four floating-point values: left, top, right and bottom.
Color	Color value.
ColorPair	A pair of two colors.
ColorRect	A set of four colors.
Enumeration	Enumeration (e.g. alignment) that is represented as a single byte.
String	UTF-8 encoded string.
Font	Text font.

Chapter 7

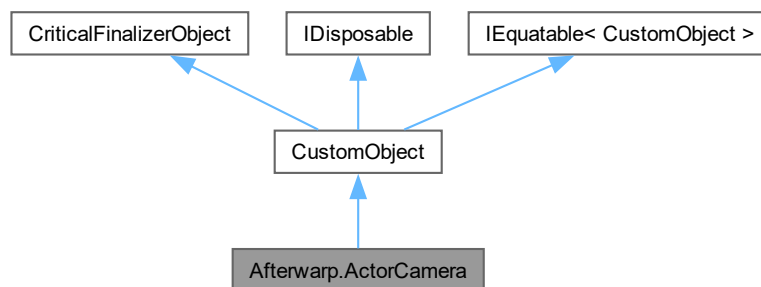
Class Documentation

7.1 Afterwarp.ActorCamera Class Reference

Inheritance diagram for Afterwarp.ActorCamera:



Collaboration diagram for Afterwarp.ActorCamera:



Public Member Functions

- [ActorCamera](#) (Vector3 position, Vector3 rotation, float distance)
Creates new instance of camera module.
- [ActorCamera](#) (float distance)
Creates new instance of camera module.
- Vector3 [GetPosition](#) ()
Returns camera's position.
- bool [SetPosition](#) (Vector3 position)
Updates camera's position. Returns "true" if position has actually changed.
- Vector4 [GetRotation](#) ()
Returns current camera rotation. "w" component is a rotation for Lenticular printing.
- bool [SetRotation](#) (Vector4 rotation)
- Quaternion [GetQuaternion](#) ()
Returns current camera rotation as quaternion.
- bool [SetQuaternion](#) (Quaternion quaternion)
Changes current camera's rotation as quaternion. Returns "true" if rotation has actually changed.
- void [Zoom](#) (Vector2 position, Vector2 size, Matrix4x4 projection, float delta)
Changes camera's zoom by adjusting its distance and position around the specified point.
- void [ZoomOrtho](#) (Vector2 position, Vector2 size, float distance, float adjustedDistance, Matrix4x4 projection, Matrix4x4 adjustedProjection)
Adjusts camera position while performing orthographic zoom.
- bool [Command](#) ([CameraCommand](#) command, Vector3 position, Vector2 size, Vector3? sense=null, Matrix4x4? projection=null)
Changes camera's zoom by adjusting its distance and position around the specified point.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- Vector3 [Forward](#) [get]
Returns camera's forward vector (looking direction).
- Vector3 [Right](#) [get]
Returns camera's right vector (sideway direction).
- Vector3 [Ceiling](#) [get]
Returns camera's ceiling vector (up direction).
- float [Distance](#) [get, set]
Returns or changes the distance between camera's position and the viewer (for third-person view).
- Matrix4x4 [View](#) [get]
Returns camera's resulting 3D view matrix.
- [CameraConstraints Constraints](#) [get, set]
Returns or changes 3D camera constraints.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.1.1 Detailed Description

A camera module that can represent both first-person and third-person views for 3D world exploration and object manipulation.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 ActorCamera() [1/2]

```
Afterwarp.ActorCamera.ActorCamera (
    Vector3 position,
    Vector3 rotation,
    float distance ) [inline]
```

Creates new instance of camera module.

7.1.2.2 ActorCamera() [2/2]

```
Afterwarp.ActorCamera.ActorCamera (
    float distance ) [inline]
```

Creates new instance of camera module.

7.1.3 Member Function Documentation

7.1.3.1 Command()

```
bool Afterwarp.ActorCamera.Command (
    CameraCommand command,
    Vector3 position,
    Vector2 size,
    Vector3? sense = null,
    Matrix4x4? projection = null ) [inline]
```

Changes camera's zoom by adjusting its distance and position around the specified point.

7.1.3.2 GetPosition()

```
Vector3 Afterwarp.ActorCamera.GetPosition ( ) [inline]
```

Returns camera's position.

7.1.3.3 GetQuaternion()

```
Quaternion Afterwarp.ActorCamera.GetQuaternion ( ) [inline]
```

Returns current camera rotation as quaternion.

7.1.3.4 GetRotation()

```
Vector4 Afterwarp.ActorCamera.GetRotation ( ) [inline]
```

Returns current camera rotation. "w" component is a rotation for Lenticular printing.

7.1.3.5 SetPosition()

```
bool Afterwarp.ActorCamera.SetPosition (
    Vector3 position )
```

Updates camera's position. Returns "true" if position has actually changed.

7.1.3.6 SetQuaternion()

```
bool Afterwarp.ActorCamera.SetQuaternion (
    Quaternion quaternion ) [inline]
```

Changes current camera's rotation as quaternion. Returns "true" if rotation has actually changed.

7.1.3.7 SetRotation()

```
bool Afterwarp.ActorCamera.SetRotation (
    Vector4 rotation )
```

Sets new camera rotation. "w" component is a rotation for Lenticular printing. Returns "true" if rotation has actually changed.

7.1.3.8 Zoom()

```
void Afterwarp.ActorCamera.Zoom (
    Vector2 position,
    Vector2 size,
    Matrix4x4 projection,
    float delta ) [inline]
```

Changes camera's zoom by adjusting its distance and position around the specified point.

7.1.3.9 ZoomOrtho()

```
void Afterwarp.ActorCamera.ZoomOrtho (
    Vector2 position,
    Vector2 size,
    float distance,
    float adjustedDistance,
    Matrix4x4 projection,
    Matrix4x4 adjustedProjection ) [inline]
```

Adjusts camera position while performing orthographic zoom.

7.1.4 Property Documentation

7.1.4.1 Ceiling

```
Vector3 Afterwarp.ActorCamera.Ceiling [get]
```

Returns camera's ceiling vector (up direction).

7.1.4.2 Constraints

```
CameraConstraints Afterwarp.ActorCamera.Constraints [get], [set]
```

Returns or changes 3D camera constraints.

7.1.4.3 Distance

```
float Afterwarp.ActorCamera.Distance [get], [set]
```

Returns or changes the distance between camera's position and the viewer (for third-person view).

7.1.4.4 Forward

```
Vector3 Afterwarp.ActorCamera.Forward [get]
```

Returns camera's forward vector (looking direction).

7.1.4.5 Right

```
Vector3 Afterwarp.ActorCamera.Right [get]
```

Returns camera's right vector (sideway direction).

7.1.4.6 View

Matrix4x4 Afterwarp.ActorCamera.View [get]

Returns camera's resulting 3D view matrix.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.2 Afterwarp.AmbientOcclusionParameters Struct Reference

Parameters that define how ambient occlusion is performed.

Public Member Functions

- [AmbientOcclusionParameters](#) (float radius, int samples, float attenuation, float contrast, float depthBias, float kernelBias, float blurSigma, float blurFallOff, float strength)
Creates new ambient occlusion parameters with the given values.

Public Attributes

- float [Radius](#)
Radius of the occlusion.
- int [Samples](#)
Number of kernel samples.
- float [Attenuation](#)
Linear attenuation.
- float [Contrast](#)
Contrast of the occlusion.
- float [DepthBias](#)
Bias value to resolve depth precision issues.
- float [KernelBias](#)
Bias value to prevent sampling close to tangent surface.
- float [BlurSigma](#)
Sigma coefficient for blur filter.
- float [BlurFallOff](#)
Fall-off coefficient for detecting depth discontinuities in blur filter.
- float [Strength](#)
Strength of the effect.

Properties

- static [AmbientOcclusionParameters Default](#) [get]
Returns default ambient occlusion parameters.

7.2.1 Detailed Description

Parameters that define how ambient occlusion is performed.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 AmbientOcclusionParameters()

```
Afterwarp.AmbientOcclusionParameters.AmbientOcclusionParameters (
    float radius,
    int samples,
    float attenuation,
    float contrast,
    float depthBias,
    float kernelBias,
    float blurSigma,
    float blurFallOff,
    float strength ) [inline]
```

Creates new ambient occlusion parameters with the given values.

7.2.3 Member Data Documentation

7.2.3.1 Attenuation

```
float Afterwarp.AmbientOcclusionParameters.Attenuation
```

Linear attenuation.

7.2.3.2 BlurFallOff

```
float Afterwarp.AmbientOcclusionParameters.BlurFallOff
```

Fall-off coefficient for detecting depth discontinuities in blur filter.

7.2.3.3 BlurSigma

```
float Afterwarp.AmbientOcclusionParameters.BlurSigma
```

Sigma coefficient for blur filter.

7.2.3.4 Contrast

```
float Afterwarp.AmbientOcclusionParameters.Contrast
```

Contrast of the occlusion.

7.2.3.5 DepthBias

```
float Afterwarp.AmbientOcclusionParameters.DepthBias
```

Bias value to resolve depth precision issues.

7.2.3.6 KernelBias

```
float Afterwarp.AmbientOcclusionParameters.KernelBias
```

Bias value to prevent sampling close to tangent surface.

7.2.3.7 Radius

```
float Afterwarp.AmbientOcclusionParameters.Radius
```

Radius of the occlusion.

7.2.3.8 Samples

```
int Afterwarp.AmbientOcclusionParameters.Samples
```

Number of kernel samples.

7.2.3.9 Strength

```
float Afterwarp.AmbientOcclusionParameters.Strength
```

Strength of the effect.

7.2.4 Property Documentation

7.2.4.1 Default

```
AmbientOcclusionParameters Afterwarp.AmbientOcclusionParameters.Default [static], [get]
```

Returns default ambient occlusion parameters.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.3 Afterwarp.API Struct Reference

Classes

- class [Exception](#)

Exception type raised by [Afterwarp](#) classes.

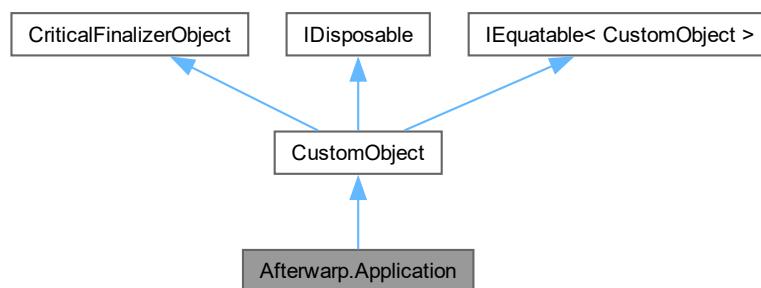
The documentation for this struct was generated from the following files:

- [Afterwarp.API.cs](#)
- [Afterwarp.Types.cs](#)

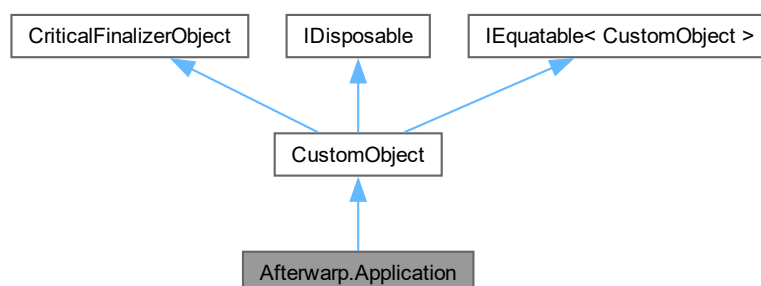
7.4 Afterwarp.Application Class Reference

Application execution manager that handles OS events and controls application's window.

Inheritance diagram for Afterwarp.Application:



Collaboration diagram for Afterwarp.Application:



Public Member Functions

- delegate void [EventCallback](#) (object sender)
Basic application event.
- delegate bool [BoolCallback](#) (object sender)
Application creation event.
- delegate void [MouseButtonCallback](#) (object sender, [MouseEvent](#) mouseEvent, [MouseButton](#) button, [Point](#) position)
Application mouse handling event.
- delegate void [KeyboardCallback](#) (object sender, [KeyEvent](#) keyEvent, int virtualKey, ushort keyCode)
Application keyboard handling event.
- delegate void [StatusCallback](#) (object sender, [ApplicationEvent](#) appEvent)
Event function for application status change events.
- delegate void [EventHookCallback](#) (object sender, IntPtr eventData)
Custom application message handling event.
- [Application](#) ([ApplicationConfiguration](#) configuration)
Creates new instance of application manager.
- void [Invalidate](#) ()
Requests the application window to be repainted.
- bool [Execute](#) ()
- void [Terminate](#) ()
Finishes execution of the application.
- [Key](#) [TranslateVirtualKey](#) (int virtualKey)
Converts virtual key code to portable key code. Returns Key::Null if no equivalent is found.
- int [ConvertPortableKey](#) ([Key](#) key)
Converts portable key code to virtual key code. Returns 0 if no equivalent is found.
- string [ReadTextFromClipboard](#) (int textBufferSize=4096)
Attempts to read text from clipboard.
- void [WriteTextToClipboard](#) (string text)
Attempts to write text to clipboard.
- void [CaptureMouseInput](#) ()
Captures mouse input for the application window.
- bool [ReleaseMouseInput](#) ()
Releases mouse input for the application window.
- void [SetIcons](#) ([Surface](#)[] surfaces)
Specifies one or more application icons.
- void [SetIcons](#) (string[] fileNames)
- bool [FileChooserDialog](#) (ref string filePath, [FileChooserDialog](#) dialog, string filters, string caption, string accept, string reject, int filePathBufferSize=4096)
Captures mouse input for the application window.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- IntPtr [WindowHandle](#) [get]
Retrieves the handle of main application window.
- string [Title](#) [get, set]
Application's title.
- string [IconTitle](#) [get, set]
Application icon's title.
- string [ExecutablePath](#) [get]
Path to application's executable.
- Rect [WindowRect](#) [get, set]
Current application's window rectangle.
- Rect [ClientRect](#) [get]
Current application's window client rectangle.
- Point [ClientSize](#) [set]
Sets new application's window client size.
- Point [MinimalSize](#) [get, set]
Current minimal size of the window. Values of zero means that there is no limit.
- double [WindowScale](#) [get]
Returns the scale of application window.
- [ApplicationWindowState](#) [WindowState](#) [get, set]
Current application's window state.
- [AppCursor](#) [Cursor](#) [get, set]
Current application window's cursor.
- bool [MouseInputCaptured](#) [get]
Indicates whether the mouse input is currently captured.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Events

- [BoolCallback OnCreate](#)
Event invoked when application is created.
- [EventCallback OnDestroy](#)
Event invoked when application is destroyed.
- [EventCallback OnRender](#)
Event invoked when application is rendered.
- [MouseCallback OnMouse](#)
Event invoked when mouse has performed an action.
- [KeyboardCallback OnKeyboard](#)
Event invoked when keyboard has performed an action.
- [StatusCallback OnStatus](#)
Event invoked when application and/or window status has changed.
- [BoolCallback OnIdle](#)
Event invoked when application is idle.
- [EventCallback OnResize](#)
Event invoked when application's window has been resized.
- [EventHookCallback OnHook](#)
Event invoked to handle application's internal processes.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.4.1 Detailed Description

Application execution manager that handles OS events and controls application's window.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 Application()

```
Afterwarp.Application.Application (
    ApplicationConfiguration configuration ) [inline]
```

Creates new instance of application manager.

7.4.3 Member Function Documentation

7.4.3.1 BoolCallback()

```
delegate bool Afterwarp.Application.BoolCallback (
    object sender )
```

Application creation event.

7.4.3.2 CaptureMouseInput()

```
void Afterwarp.Application.CaptureMouseInput ( ) [inline]
```

Captures mouse input for the application window.

7.4.3.3 ConvertPortableKey()

```
int Afterwarp.Application.ConvertPortableKey (
    Key key ) [inline]
```

Converts portable key code to virtual key code. Returns 0 if no equivalent is found.

7.4.3.4 EventCallback()

```
delegate void Afterwarp.Application.EventCallback (
    object sender )
```

Basic application event.

7.4.3.5 EventHookCallback()

```
delegate void Afterwarp.Application.EventHookCallback (
    object sender,
    IntPtr eventData )
```

Custom application message handling event.

7.4.3.6 Execute()

```
bool Afterwarp.Application.Execute ( )
```

Executes application's main loop (or returns "false" when there were errors during application create event).

7.4.3.7 FileChooserDialog()

```
bool Afterwarp.Application.FileChooserDialog (
    ref string filePath,
    FileChooserDialog dialog,
    string filters,
    string caption,
    string accept,
    string reject,
    int filePathBufferSize = 4096 ) [inline]
```

Captures mouse input for the application window.

7.4.3.8 Invalidate()

```
void Afterwarp.Application.Invalidate ( )
```

Requests the application window to be repainted.

7.4.3.9 KeyboardCallback()

```
delegate void Afterwarp.Application.KeyboardCallback (
    object sender,
    KeyEvent keyEvent,
    int virtualKey,
    ushort keyCode )
```

Application keyboard handling event.

7.4.3.10 MouseCallback()

```
delegate void Afterwarp.Application.MouseCallback (
    object sender,
    MouseEventArgs mouseEvent,
    MouseButton button,
    Point position )
```

Application mouse handling event.

7.4.3.11 ReadTextFromClipboard()

```
string Afterwarp.Application.ReadTextFromClipboard (
    int textBufferSize = 4096 ) [inline]
```

Attempts to read text from clipboard.

7.4.3.12 ReleaseMouseInput()

```
bool Afterwarp.Application.ReleaseMouseInput ( )
```

Releases mouse input for the application window.

7.4.3.13 SetIcons() [1/2]

```
void Afterwarp.Application.SetIcons (
    string[] fileNames ) [inline]
```

Changes application icons by loading them from external files. Note: this method will only work while at least one device instance currently exists.

7.4.3.14 SetIcons() [2/2]

```
void Afterwarp.Application.SetIcons (
    Surface[] surfaces ) [inline]
```

Specifies one or more application icons.

7.4.3.15 StatusCallback()

```
delegate void Afterwarp.Application.StatusCallback (
    object sender,
    ApplicationEvent appEvent )
```

Event function for application status change events.

7.4.3.16 Terminate()

```
void Afterwarp.Application.Terminate ( )
```

Finishes execution of the application.

7.4.3.17 TranslateVirtualKey()

```
Key Afterwarp.Application.TranslateVirtualKey (
    int virtualKey ) [inline]
```

Converts virtual key code to portable key code. Returns Key::Null if no equivalent is found.

7.4.3.18 WriteTextToClipboard()

```
void Afterwarp.Application.WriteTextToClipboard (
    string text ) [inline]
```

Attempts to write text to clipboard.

7.4.4 Property Documentation

7.4.4.1 ClientRect

```
Rect Afterwarp.Application.ClientRect [get]
```

Current application's window client rectangle.

7.4.4.2 ClientSize

```
Point Afterwarp.Application.ClientSize [set]
```

Sets new application's window client size.

7.4.4.3 Cursor

```
AppCursor Afterwarp.Application.Cursor [get], [set]
```

Current application window's cursor.

7.4.4.4 ExecutablePath

```
string Afterwarp.Application.ExecutablePath [get]
```

Path to application's executable.

7.4.4.5 IconTitle

```
string Afterwarp.Application.IconTitle [get], [set]
```

Application icon's title.

7.4.4.6 MinimalSize

```
Point Afterwarp.Application.MinimalSize [get], [set]
```

Current minimal size of the window. Values of zero means that there is no limit.

7.4.4.7 MouseInputCaptured

```
bool Afterwarp.Application.MouseInputCaptured [get]
```

Indicates whether the mouse input is currently captured.

7.4.4.8 Title

```
string Afterwarp.Application.Title [get], [set]
```

Application's title.

7.4.4.9 WindowHandle

```
IntPtr Afterwarp.Application.WindowHandle [get]
```

Retrieves the handle of main application window.

7.4.4.10 WindowRect

```
Rect Afterwarp.Application.WindowRect [get], [set]
```

Current application's window rectangle.

7.4.4.11 WindowScale

```
double Afterwarp.Application.WindowScale [get]
```

Returns the scale of application window.

7.4.4.12 WindowState

```
ApplicationWindowState Afterwarp.Application.WindowState [get], [set]
```

Current application's window state.

7.4.5 Event Documentation

7.4.5.1 OnCreate

`BoolCallback Afterwarp.Application.OnCreate`

Event invoked when application is created.

7.4.5.2 OnDestroy

`EventCallback Afterwarp.Application.OnDestroy`

Event invoked when application is destroyed.

7.4.5.3 OnHook

`EventHookCallback Afterwarp.Application.OnHook`

Event invoked to handle application's internal processes.

7.4.5.4 OnIdle

`BoolCallback Afterwarp.Application.OnIdle`

Event invoked when application is idle.

7.4.5.5 OnKeyboard

`KeyboardCallback Afterwarp.Application.OnKeyboard`

Event invoked when keyboard has performed an action.

7.4.5.6 OnMouse

`MouseButtonCallback Afterwarp.Application.OnMouse`

Event invoked when mouse has performed an action.

7.4.5.7 OnRender

`EventCallback Afterwarp.Application.OnRender`

Event invoked when application is rendered.

7.4.5.8 OnResize

`EventCallback Afterwarp.Application.OnResize`

Event invoked when application's window has been resized.

7.4.5.9 OnStatus

`StatusCallback Afterwarp.Application.OnStatus`

Event invoked when application and/or window status has changed.

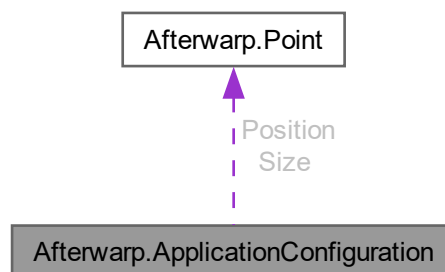
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.5 Afterwarp.ApplicationConfiguration Struct Reference

Structure that contains all the parameters necessary to create a new application and its window.

Collaboration diagram for Afterwarp.ApplicationConfiguration:



Public Member Functions

- [ApplicationConfiguration](#) (string windowClassName, [Point](#) size, [ApplicationWindowState](#) state=[ApplicationWindowState.Window](#)
[Point?](#) position=null, IntPtr instance=new IntPtr(), IntPtr iconSmall=new IntPtr(), IntPtr iconBig=new IntPtr())
Creates a structure with the given values.

Public Attributes

- [Point Size](#)
Initial size of the application window.
- [ApplicationWindowState State](#)
Initial state of application window.
- [Point Position](#)
- [IntPtr Instance](#)
Handle of application's instance.
- [IntPtr IconSmall](#)
Handle of application's "small" icon.
- [IntPtr IconBig](#)
Handle of application's "big" icon.
- [byte\[\] WindowClassNameBytes](#)
Nul-terminated string for application class name.

Properties

- [string WindowClassName](#) [get, set]
Name of application window class.

7.5.1 Detailed Description

Structure that contains all the parameters necessary to create a new application and its window.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 ApplicationConfiguration()

```
Afterwarp.ApplicationConfiguration.ApplicationConfiguration (  
    string windowClassName,  
    Point size,  
    ApplicationWindowState state = ApplicationWindowState::Windowed,  
    Point? position = null,  
    IntPtr instance = new IntPtr(),  
    IntPtr iconSmall = new IntPtr(),  
    IntPtr iconBig = new IntPtr() ) [inline]
```

Creates a structure with the given values.

7.5.3 Member Data Documentation

7.5.3.1 IconBig

```
IntPtr Afterwarp.ApplicationConfiguration.IconBig
```

Handle of application's "big" icon.

7.5.3.2 IconSmall

```
IntPtr Afterwarp.ApplicationConfiguration.IconSmall
```

Handle of application's "small" icon.

7.5.3.3 Instance

```
IntPtr Afterwarp.ApplicationConfiguration.Instance
```

Handle of application's instance.

7.5.3.4 Position

```
Point Afterwarp.ApplicationConfiguration.Position
```

Initial position of application window. If both X and Y are set to INT32_MAX, then a default position will be used.

7.5.3.5 Size

```
Point Afterwarp.ApplicationConfiguration.Size
```

Initial size of the application window.

7.5.3.6 State

```
ApplicationWindowState Afterwarp.ApplicationConfiguration.State
```

Initial state of application window.

7.5.3.7 WindowClassNameBytes

```
byte [] Afterwarp.ApplicationConfiguration.WindowClassNameBytes
```

Nul-terminated string for application class name.

7.5.4 Property Documentation

7.5.4.1 WindowClassName

```
string Afterwarp.ApplicationConfiguration.WindowClassName [get], [set]
```

Name of application window class.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.6 Afterwarp.Canvas.Attribute Struct Reference

Canvas attribute flags that can be combined together.

Static Public Attributes

- const uint **Projected** = 0x01u
- const uint **SDF** = 0x02u
Preprocess the texture by using its alpha-channel as a signed distance field.
- const uint **Outline** = 0x04u
Include outline when rendering signed distance field (implies SDF).
- const uint **Cubic** = 0x08u
Sample texture using bicubic interpolation for higher-quality results.
- const uint **ColorAdjust** = 0x10u
Adjusts texture color using YCH instead of RGB (only works for textures).
- const uint **Transform** = 0x20u
Applies 3x2 world transformation matrix to non-holographic primitives.

7.6.1 Detailed Description

Canvas attribute flags that can be combined together.

7.6.2 Member Data Documentation

7.6.2.1 ColorAdjust

```
const uint Afterwarp.Canvas.Attribute.ColorAdjust = 0x10u [static]
```

Adjusts texture color using YCH instead of RGB (only works for textures).

7.6.2.2 Cubic

```
const uint Afterwarp.Canvas.Attribute.Cubic = 0x08u [static]
```

Sample texture using bicubic interpolation for higher-quality results.

7.6.2.3 Outline

```
const uint Afterwarp.Canvas.Attribute.Outline = 0x04u [static]
```

Include outline when rendering signed distance field (implies SDF).

7.6.2.4 Projected

```
const uint Afterwarp.Canvas.Attribute.Projected = 0x01u [static]
```

Transform 2D coordinates as 3D vectors (with Z = 0) by the appropriate 4x4 matrix as if drawing to an arbitrary plane in 3D.

7.6.2.5 SDF

```
const uint Afterwarp.Canvas.Attribute.SDF = 0x02u [static]
```

Preprocess the texture by using its alpha-channel as a signed distance field.

7.6.2.6 Transform

```
const uint Afterwarp.Canvas.Attribute.Transform = 0x20u [static]
```

Applies 3x2 world transformation matrix to non-holographic primitives.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.7 Afterwarp.Device.Attribute Struct Reference

Device attributes that define its behavior.

Static Public Attributes

- const uint [Debug](#) = 0x01
Debug-type context that helps to resolve issues in rendering code.
- const uint [Software](#) = 0x02
Software-based solution is used for rendering instead of GPU acceleration.
- const uint [Legacy](#) = 0x04
Compatibility device that is meant for older technologies.
- const uint [LimitedExtensions](#) = 0x80

7.7.1 Detailed Description

Device attributes that define its behavior.

7.7.2 Member Data Documentation

7.7.2.1 Debug

```
const uint Afterwarp.Device.Attribute.Debug = 0x01 [static]
```

Debug-type context that helps to resolve issues in rendering code.

7.7.2.2 Legacy

```
const uint Afterwarp.Device.Attribute.Legacy = 0x04 [static]
```

Compatibility device that is meant for older technologies.

7.7.2.3 LimitedExtensions

```
const uint Afterwarp.Device.Attribute.LimitedExtensions = 0x80 [static]
```

Device should only enable extensions determined by their appropriate level (deviceAttributeExtensions), even if a higher core version is available that inherently supports higher level extensions.

7.7.2.4 Software

```
const uint Afterwarp.Device.Attribute.Software = 0x02 [static]
```

Software-based solution is used for rendering instead of GPU acceleration.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.8 Afterwarp.ObjectModel.Attribute Struct Reference

Attribute bit flags for an object that define its status.

Static Public Attributes

- const uint [Visible](#) = 0x01u
Object is visible so can be rendered and selected (if selected flag is set).
- const uint [Selectable](#) = 0x02u
Object can be selected.
- const uint [Transparent](#) = 0x04u
Object is semi-transparent (might be rendered through a different mechanism).
- const uint [Hierarchyless](#) = 0x08u
- const uint [DisableRealign](#) = 0x10u
Object model should not automatically realign itself.
- const uint [NonViewable](#) = 0x20u

7.8.1 Detailed Description

Attribute bit flags for an object that define its status.

7.8.2 Member Data Documentation

7.8.2.1 DisableRealign

```
const uint Afterwarp.ObjectModel.Attribute.DisableRealign = 0x10u [static]
```

Object model should not automatically realign itself.

7.8.2.2 Hierarchyless

```
const uint Afterwarp.ObjectModel.Attribute.Hierarchyless = 0x08u [static]
```

Object does not participate in bounding volume hierarchy (BVH) calculation. This is particularly useful for constantly moving objects or those objects that have unusually large dimensions overlapping with other objects.

7.8.2.3 NonViewable

```
const uint Afterwarp.ObjectModel.Attribute.NonViewable = 0x20u [static]
```

Object does not appear among visible objects in the view. This is useful for abstract objects that have no associated mesh, so aren't rendered in any way, but can still be selected.

7.8.2.4 Selectable

```
const uint Afterwarp.ObjectModel.Attribute.Selectable = 0x02u [static]
```

Object can be selected.

7.8.2.5 Transparent

```
const uint Afterwarp.ObjectModel.Attribute.Transparent = 0x04u [static]
```

Object is semi-transparent (might be rendered through a different mechanism).

7.8.2.6 Visible

```
const uint Afterwarp.ObjectModel.Attribute.Visible = 0x01u [static]
```

Object is visible so can be rendered and selected (if selected flag is set).

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.9 Afterwarp.Scene.Attribute Struct Reference

Cumulative attributes that define rendering behavior of the scene.

Static Public Attributes

- const uint [Instancing](#) = 0x01u
Instancing technique will be used during rendering.
- const uint [Normals](#) = 0x02u
Vertex normals will be processed and taken into account.
- const uint [Coloring](#) = 0x04u
Vertex colors will be processed and taken into account.
- const uint [TexturingCubic](#) = 0x08u
Cubic interpolation will be used for texture mapping.
- const uint [ShadowsCubic](#) = 0x10u
Cubic interpolation will be used for shadow mapping.
- const uint [Glassy](#) = 0x20u
Perform adjustments for Order-independent transparency (OIT) rendering.
- const uint [SinglePass](#) = 0x40u
The whole scene is rendered in a single pass with ambient occlusion.
- const uint [DepthPrePass](#) = 0x80u
- const uint [LinearDepths](#) = 0x100u
- const uint [Modeling](#) = 0x1000u
Scene is a modeling scene, which means it supports textures and materials.

7.9.1 Detailed Description

Cumulative attributes that define rendering behavior of the scene.

7.9.2 Member Data Documentation

7.9.2.1 Coloring

```
const uint Afterwarp.Scene.Attribute.Coloring = 0x04u [static]
```

Vertex colors will be processed and taken into account.

7.9.2.2 DepthPrePass

```
const uint Afterwarp.Scene.Attribute.DepthPrePass = 0x80u [static]
```

Generate output for depth pre-pass, which at minimum includes normals. Note: it is possible to do depth pre-pass without normals by excluding this attribute.

7.9.2.3 Glassy

```
const uint Afterwarp.Scene.Attribute.Glassy = 0x20u [static]
```

Perform adjustments for Order-independent transparency (OIT) rendering.

7.9.2.4 Instancing

```
const uint Afterwarp.Scene.Attribute.Instancing = 0x01u [static]
```

Instancing technique will be used during rendering.

7.9.2.5 LinearDepths

```
const uint Afterwarp.Scene.Attribute.LinearDepths = 0x100u [static]
```

Produce linear depths as part of depth pre-pass for spatial fog or water effects. Note: this requires DepthPrePass flag to be enabled and thus always includes normals.

7.9.2.6 Modeling

```
const uint Afterwarp.Scene.Attribute.Modeling = 0x1000u [static]
```

Scene is a modeling scene, which means it supports textures and materials.

7.9.2.7 Normals

```
const uint Afterwarp.Scene.Attribute.Normals = 0x02u [static]
```

Vertex normals will be processed and taken into account.

7.9.2.8 ShadowsCubic

```
const uint Afterwarp.Scene.Attribute.ShadowsCubic = 0x10u [static]
```

Cubic interpolation will be used for shadow mapping.

7.9.2.9 SinglePass

```
const uint Afterwarp.Scene.Attribute.SinglePass = 0x40u [static]
```

The whole scene is rendered in a single pass with ambient occlusion.

7.9.2.10 TexturingCubic

```
const uint Afterwarp.Scene.Attribute.TexturingCubic = 0x08u [static]
```

Cubic interpolation will be used for texture mapping.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.10 Afterwarp.Texture.Attribute Struct Reference

Attribute that defines special behavior and/or unique characteristics of the texture.

Static Public Attributes

- const uint [MipMapping](#) = 0x01u
- const uint [Drawable](#) = 0x02u
- const uint [Dynamic](#) = 0x04u
- const uint [PremultipliedAlpha](#) = 0x08u
- const uint [Compute](#) = 0x40u
 - Texture will be used for Compute shader access (Direct3D-only, makes texture UAV-capable).*
- const uint [Scratch](#) = 0x8000u

7.10.1 Detailed Description

Attribute that defines special behavior and/or unique characteristics of the texture.

7.10.2 Member Data Documentation

7.10.2.1 Compute

```
const uint Afterwarp.Texture.Attribute.Compute = 0x40u [static]
```

Texture will be used for Compute shader access (Direct3D-only, makes texture UAV-capable).

7.10.2.2 Drawable

```
const uint Afterwarp.Texture.Attribute.Drawable = 0x02u [static]
```

Texture contains a special type of surface that can be rendered to. In other words, the texture can be considered a render target (Direct3D) or render buffer object (OpenGL).

7.10.2.3 Dynamic

```
const uint Afterwarp.Texture.Attribute.Dynamic = 0x04u [static]
```

Texture is meant for frequent write-only access. Dynamic textures cannot contain mipmap surface chain attached and cannot be drawable either.

7.10.2.4 MipMapping

```
const uint Afterwarp.Texture.Attribute.MipMapping = 0x01u [static]
```

Texture has a full mipmap surface chain attached. MipMapping technique can improve visual quality when the texture is drawn in different sizes, especially in smaller ones.

7.10.2.5 PremultipliedAlpha

```
const uint Afterwarp.Texture.Attribute.PremultipliedAlpha = 0x08u [static]
```

Texture has its color components premultiplied by alpha-channel. This implies permanent loss of information as the components are multiplied by alpha value and stored (so, for example, pixels with alpha value of zero permanently lose all color information), however this can improve visual quality on mipmaps with translucent pixels. The parameter is merely a hint for rendering system, it does not change the actual pixels - an action that should be performed separately.

7.10.2.6 Scratch

```
const uint Afterwarp.Texture.Attribute.Scratch = 0x8000u [static]
```

Texture is of "scratch" type. That is, it uses system memory for storage and, when used in context of Direct3D and/or OpenGL, it cannot be used directly.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.11 Afterwarp.TextureCabinet.Attribute Struct Reference

Cumulative attributes that define rendering characteristics of the scene.

Static Public Attributes

- const uint [AmbientOcclusion](#) = 0x0001u
Screen-Space Ambient Occlusion technique.
- const uint [AmbientOcclusionDownscale](#) = 0x0002u
Downscales SSAO texture to enable 4x faster performance on low and mid-end GPUs.
- const uint [Bloom](#) = 0x0004u
Bloom (glare) technique.
- const uint [BloomDownscale](#) = 0x0008u
Bloom is rendered at half of resolution to accomodate for higher display scales.
- const uint [BloomCubic](#) = 0x0010u
Bloom is applied using cubic interpolation for higher quality look.
- const uint [LinearDepths](#) = 0x0020u
Linear depths are provided for techniques such as spatial fog or water.
- const uint [Glassy](#) = 0x0040u
Accurate order-independent transparency (OIT) single-pass technique.
- const uint [GlassyFast](#) = 0x0080u
Approximate order-independent transparency (OIT) single-pass technique.
- const uint [GlassyFog](#) = 0x0100u
Add to order-independent transparency (OIT) technique support for spatial fog effects.
- const uint [GlassyFrosted](#) = 0x0200u
Frosted Glass effect added to order-independent transparency (OIT) techniques.

7.11.1 Detailed Description

Cumulative attributes that define rendering characteristics of the scene.

7.11.2 Member Data Documentation

7.11.2.1 AmbientOcclusion

```
const uint Afterwarp.TextureCabinet.Attribute.AmbientOcclusion = 0x0001u [static]
```

Screen-Space Ambient Occlusion technique.

7.11.2.2 AmbientOcclusionDownscale

```
const uint Afterwarp.TextureCabinet.Attribute.AmbientOcclusionDownscale = 0x0002u [static]
```

Downscales SSAO texture to enable 4x faster performance on low and mid-end GPUs.

7.11.2.3 Bloom

```
const uint Afterwarp.TextureCabinet.Attribute.Bloom = 0x0004u [static]
```

Bloom (glare) technique.

7.11.2.4 BloomCubic

```
const uint Afterwarp.TextureCabinet.Attribute.BloomCubic = 0x0010u [static]
```

Bloom is applied using cubic interpolation for higher quality look.

7.11.2.5 BloomDownscale

```
const uint Afterwarp.TextureCabinet.Attribute.BloomDownscale = 0x0008u [static]
```

Bloom is rendered at half of resolution to accomodate for higher display scales.

7.11.2.6 Glassy

```
const uint Afterwarp.TextureCabinet.Attribute.Glassy = 0x0040u [static]
```

Accurate order-independent transparency (OIT) single-pass technique.

7.11.2.7 GlassyFast

```
const uint Afterwarp.TextureCabinet.Attribute.GlassyFast = 0x0080u [static]
```

Approximate order-independent transparency (OIT) single-pass technique.

7.11.2.8 GlassyFog

```
const uint Afterwarp.TextureCabinet.Attribute.GlassyFog = 0x0100u [static]
```

Add to order-independent transparency (OIT) technique support for spatial fog effects.

7.11.2.9 GlassyFrosted

```
const uint Afterwarp.TextureCabinet.Attribute.GlassyFrosted = 0x0200u [static]
```

Frosted Glass effect added to order-independent transparency (OIT) techniques.

7.11.2.10 LinearDepths

```
const uint Afterwarp.TextureCabinet.Attribute.LinearDepths = 0x0020u [static]
```

Linear depths are provided for techniques such as spatial fog or water.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.12 Afterwarp.AutoDrawOption Struct Reference

Cumulative options that can be passed to one of scene mesh "AutoDraw" helper functions.

Static Public Attributes

- const uint [PostUnbind](#) = 0x01
- const uint [ResetTextures](#) = 0x02
- const uint [MaterialIgnore](#) = 0x04
- const uint [MaterialTransparency](#) = 0x08
- const uint [MaterialSkipTransparent](#) = 0x10
Portions of the mesh that have semi-transparent materials should be skipped.
- const uint [MaterialSkipOpaque](#) = 0x20
Portions of the mesh that have opaque materials should be skipped.
- const uint [ObjectSkipTransparent](#) = 0x40
Skips objects that have transparent attribute set.
- const uint [ObjectSkipNonTransparent](#) = 0x80
Skips objects that do not have transparent attribute set.
- const uint [ObjectSkipHavingMaterials](#) = 0x100
Skips objects that have mesh materials present.
- const uint [ObjectSkipNotHavingMaterials](#) = 0x200
Skips objects that do not have mesh materials present.
- const uint [ObjectHighlighted](#) = 0x400
Draws objects that have a highlight color assigned.
- const uint [ObjectDisableInstancing](#) = 0x800
Do not use instancing to render multiple objects simultaneously.

7.12.1 Detailed Description

Cumulative options that can be passed to one of scene mesh "AutoDraw" helper functions.

7.12.2 Member Data Documentation

7.12.2.1 MaterialIgnore

```
const uint Afterwarp.AutoDrawOption.MaterialIgnore = 0x04 [static]
```

Ignore materials integrated into the mesh if such are present, using object material and/or vertex colors instead. This has no impact on "ObjectSkip____HavingMaterials" options functionality.

7.12.2.2 MaterialSkipOpaque

```
const uint Afterwarp.AutoDrawOption.MaterialSkipOpaque = 0x20 [static]
```

Portions of the mesh that have opaque materials should be skipped.

7.12.2.3 MaterialSkipTransparent

```
const uint Afterwarp.AutoDrawOption.MaterialSkipTransparent = 0x10 [static]
```

Portions of the mesh that have semi-transparent materials should be skipped.

7.12.2.4 MaterialTransparency

```
const uint Afterwarp.AutoDrawOption.MaterialTransparency = 0x08 [static]
```

Alpha-channel from the scene mesh materials should be assigned to alpha-channel of the scene material (which normally corresponds to bloom factor). This is typically used when rendering semi-transparent objects.

7.12.2.5 ObjectDisableInstancing

```
const uint Afterwarp.AutoDrawOption.ObjectDisableInstancing = 0x800 [static]
```

Do not use instancing to render multiple objects simultaneously.

7.12.2.6 ObjectHighlighted

```
const uint Afterwarp.AutoDrawOption.ObjectHighlighted = 0x400 [static]
```

Draws objects that have a highlight color assigned.

7.12.2.7 ObjectSkipHavingMaterials

```
const uint Afterwarp.AutoDrawOption.ObjectSkipHavingMaterials = 0x100 [static]
```

Skips objects that have mesh materials present.

7.12.2.8 ObjectSkipNonTransparent

```
const uint Afterwarp.AutoDrawOption.ObjectSkipNonTransparent = 0x80 [static]
```

Skips objects that do not have transparent attribute set.

7.12.2.9 ObjectSkipNotHavingMaterials

```
const uint Afterwarp.AutoDrawOption.ObjectSkipNotHavingMaterials = 0x200 [static]
```

Skips objects that do not have mesh materials present.

7.12.2.10 ObjectSkipTransparent

```
const uint Afterwarp.AutoDrawOption.ObjectSkipTransparent = 0x40 [static]
```

Skips objects that have transparent attribute set.

7.12.2.11 PostUnbind

```
const uint Afterwarp.AutoDrawOption.PostUnbind = 0x01 [static]
```

Unbind the rendering buffers after issuing draw call. This can help resolve racing conditions under buggy drivers (mostly OpenGL) at the expense of reduced performance.

7.12.2.12 ResetTextures

```
const uint Afterwarp.AutoDrawOption.ResetTextures = 0x02 [static]
```

Reset scene textures after issuing draw call. For performance reasons, the textures should be reset after rendering all objects.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.13 Afterwarp.RenderingState.Blend Struct Reference

Blending parameters that are specific to a certain component, or a portion of color.

Public Member Functions

- [Blend](#) ([BlendFactor](#) source, [BlendFactor](#) dest, [BlendOp](#) op)
Creates blending parameters structure with the given values.

Public Attributes

- [BlendFactor Source](#)
Blending factor used for the source component.
- [BlendFactor Dest](#)
Blending factor used for the destination component.
- [BlendOp Op](#)
Blending operation used between source and destination components.

Properties

- static [Blend Default](#) [get]
Returns default blending parameters.

7.13.1 Detailed Description

Blending parameters that are specific to a certain component, or a portion of color.

7.13.2 Constructor & Destructor Documentation

7.13.2.1 Blend()

```
Afterwarp.RenderingState.Blend.Blend (
    BlendFactor source,
    BlendFactor dest,
    BlendOp op ) [inline]
```

Creates blending parameters structure with the given values.

7.13.3 Member Data Documentation

7.13.3.1 Dest

[BlendFactor](#) Afterwarp.RenderingState.Blend.Dest

Blending factor used for the destination component.

7.13.3.2 Op

[BlendOp](#) Afterwarp.RenderingState.Blend.Op

Blending operation used between source and destination components.

7.13.3.3 Source

`BlendFactor` `Afterwarp.RenderingState.Blend.Source`

Blending factor used for the source component.

7.13.4 Property Documentation

7.13.4.1 Default

`Blend` `Afterwarp.RenderingState.Blend.Default` `[static]`, `[get]`

Returns default blending parameters.

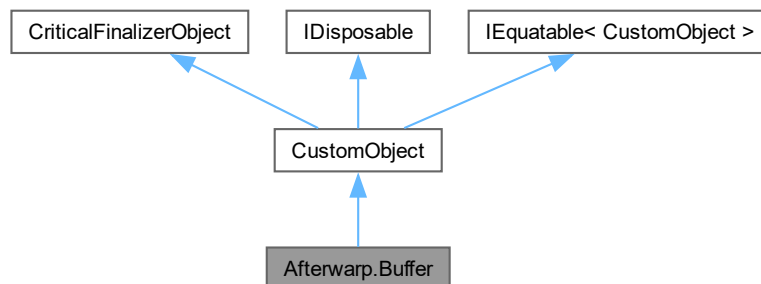
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

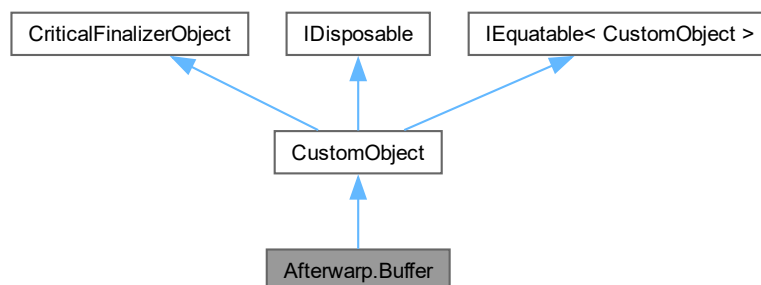
7.14 Afterwarp.Buffer Class Reference

Generic device buffer object that can contain vertices, indices or shader data.

Inheritance diagram for `Afterwarp.Buffer`:



Collaboration diagram for `Afterwarp.Buffer`:



Public Member Functions

- [Buffer](#) ([Device](#) device, [BufferDataType](#) dataType, [BufferAccessType](#) accessType, uint size, uint pitch=16u, [PixelFormat](#) format=[PixelFormat.Unknown](#), Array initialData=null)
- void [Update](#) (Array data, uint offset=0u)
- void [Retrieve](#) (Array data, uint offset=0)

Retrieves the contents of the buffer from a particular offset.
- void [Copy](#) ([Buffer](#) source, uint offsetDest=0, uint offsetSource=0, uint size=0)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]

Returns device associated with this buffer.
- IntPtr [PlatformHandle](#) [get]

Returns platform-specific buffer handle.
- [BufferDataType](#) [DataType](#) [get]

Returns the type of data that buffer contains.
- [BufferAccessType](#) [AccessType](#) [get]

Returns the access type of for buffer data.
- uint [Size](#) [get]

Returns buffer size in bytes.
- uint [Pitch](#) [get]

Returns the number of bytes for offset to move for advancing the pointer from one element to another.
- [PixelFormat](#) [Format](#) [get]

Returns pixel format that represents the buffer contents.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]

Wrapped object's handle.

Additional Inherited Members**Protected Member Functions inherited from [Afterwarp.CustomObject](#)**

- virtual void [Dispose](#) (bool disposing)

Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)

Wrapped object's handle.

7.14.1 Detailed Description

Generic device buffer object that can contain vertices, indices or shader data.

7.14.2 Constructor & Destructor Documentation

7.14.2.1 Buffer()

```
Afterwarp.Buffer.Buffer (
    Device device,
    BufferDataType dataType,
    BufferAccessType accessType,
    uint size,
    uint pitch = 16u,
    PixelFormat format = PixelFormat::Unknown,
    Array initialData = null ) [inline]
```

Creates a new instance of hardware buffer associated with a particular device with the given parameters and (optional) initial data.

7.14.3 Member Function Documentation

7.14.3.1 Copy()

```
void Afterwarp.Buffer.Copy (
    Buffer source,
    uint offsetDest = 0,
    uint offsetSource = 0,
    uint size = 0 ) [inline]
```

Issues a memory barrier to make sure that data written by shaders to the buffer is valid before it is accessed in another dispatch call.

7.14.3.2 Retrieve()

```
void Afterwarp.Buffer.Retrieve (
    Array data,
    uint offset = 0 ) [inline]
```

Retrieves the contents of the buffer from a particular offset.

7.14.3.3 Update()

```
void Afterwarp.Buffer.Update (
    Array data,
    uint offset = 0u ) [inline]
```

Updates the contents of the buffer with a new data at a particular offset. Note that if buffer has been created with an access type that is non-default, then typically a whole buffer should be updated, attempting to update a portion of it may be implementation-dependent.

7.14.4 Property Documentation

7.14.4.1 AccessType

`BufferAccessType` Afterwarp.Buffer.AccessType [get]

Returns the access type of for buffer data.

7.14.4.2 DataType

`BufferDataType` Afterwarp.Buffer.DataType [get]

Returns the type of data that buffer contains.

7.14.4.3 Device

`Device` Afterwarp.Buffer.Device [get]

Returns device associated with this buffer.

7.14.4.4 Format

`PixelFormat` Afterwarp.Buffer.Format [get]

Returns pixel format that represents the buffer contents.

7.14.4.5 Pitch

`uint` Afterwarp.Buffer.Pitch [get]

Returns the number of bytes for offset to move for advancing the pointer from one element to another.

7.14.4.6 PlatformHandle

`IntPtr` Afterwarp.Buffer.PlatformHandle [get]

Returns platform-specific buffer handle.

7.14.4.7 Size

`uint` Afterwarp.Buffer.Size [get]

Returns buffer size in bytes.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.15 Afterwarp.CameraConstraints Struct Reference

Public Member Functions

- [CameraConstraints](#) (Vector4 positionMin, Vector4 positionMax, Vector4 rotationMin, Vector4 rotationMax)
Creates new camera constraints with the given values.

Public Attributes

- Vector4 [PositionMin](#)
Minimum position allowed for camera positioning. "w" component is a minimum distance constraint.
- Vector4 [PositionMax](#)
Maximum position allowed for camera positioning. "w" component is a maximum distance constraint.
- Vector4 [RotationMin](#)
Minimum camera rotation angles. "w" component is a minimum angle for Lenticular printing rotation.
- Vector4 [RotationMax](#)
Maximum camera rotation angles. "w" component is a maximum angle for Lenticular printing rotation.

7.15.1 Detailed Description

Camera position, rotation and distance constraints. If the difference between min and max constraint on a particular axis is almost zero or less, then no constraint is applied.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 CameraConstraints()

```
Afterwarp.CameraConstraints.CameraConstraints (
    Vector4 positionMin,
    Vector4 positionMax,
    Vector4 rotationMin,
    Vector4 rotationMax ) [inline]
```

Creates new camera constraints with the given values.

7.15.3 Member Data Documentation

7.15.3.1 PositionMax

```
Vector4 Afterwarp.CameraConstraints.PositionMax
```

Maximum position allowed for camera positioning. "w" component is a maximum distance constraint.

7.15.3.2 PositionMin

```
Vector4 Afterwarp.CameraConstraints.PositionMin
```

Minimum position allowed for camera positioning. "w" component is a minimum distance constraint.

7.15.3.3 RotationMax

`Vector4 Afterwarp.CameraConstraints.RotationMax`

Maximum camera rotation angles. "w" component is a maximum angle for Lenticular printing rotation.

7.15.3.4 RotationMin

`Vector4 Afterwarp.CameraConstraints.RotationMin`

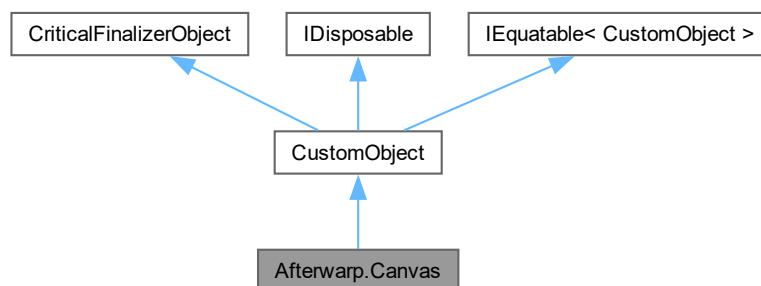
Minimum camera rotation angles. "w" component is a minimum angle for Lenticular printing rotation.

The documentation for this struct was generated from the following file:

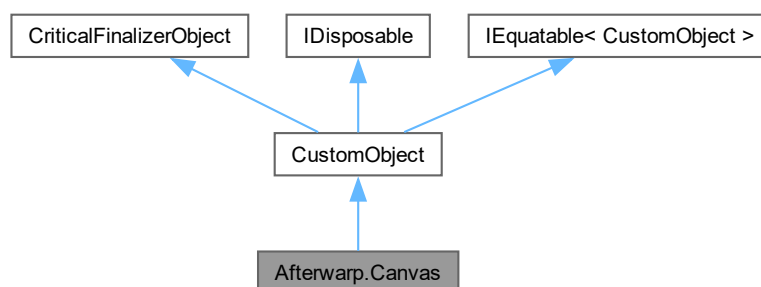
- [Afterwarp.Types.cs](#)

7.16 Afterwarp.Canvas Class Reference

Inheritance diagram for `Afterwarp.Canvas`:



Collaboration diagram for `Afterwarp.Canvas`:



Classes

- struct [Attribute](#)
Canvas attribute flags that can be combined together.

Public Member Functions

- [Canvas](#) ([Device](#) device)
Creates new 2D rendering canvas.
- void [Begin](#) ()
Starts rendering with the canvas.
- void [End](#) ()
Finishes rendering with the canvas.
- void [Pixels](#) (Vector2[] positions, uint[] colors, int elementCount, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws pixels with the given positions and colors.
- void [Lines](#) (Vector2[] vertices, uint[] colors, int[] indices, int vertexCount, int primitiveCount, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws lines with the given vertices, colors and indices.
- void [Triangles](#) (Vector2[] vertices, uint[] colors, int[] indices, int vertexCount, int primitiveCount, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws triangles with the given vertices, colors and indices.
- void [TexturedTriangles](#) ([Texture](#) texture, Vector2[] vertices, Vector2[] texCoords, uint[] colors, int[] indices, int vertexCount, int primitiveCount, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws triangles with the given vertices, texture coordinates, colors and indices.
- void [Pixel](#) (Vector2 position, uint color, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a pixel with the given position and color.
- void [Line](#) (Vector2 srcPos, Vector2 destPos, [ColorPair](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a line with the given positions and colors.
- void [ThickLine](#) (Vector2 srcPos, Vector2 destPos, [ColorPair](#) colors, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a line of varying thickness with the given positions and colors.
- void [LineEllipse](#) (Vector2 origin, Vector2 radius, int steps, uint color, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws ellipse with the given origin, radiuses, color and steps using "line" primitive.
- void [LineCircle](#) (Vector2 origin, float radius, int steps, uint color, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws circle with the given origin, radius, color and steps using "line" primitive.
- void [ThickLineEllipse](#) (Vector2 origin, Vector2 radius, int steps, uint color, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws ellipse of varying thickness with the given origin, radiuses, color and steps.
- void [ThickLineCircle](#) (Vector2 origin, float radius, int steps, uint color, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws circle of varying thickness with the given origin, radius, color and steps.
- void [LineTriangle](#) (Vector2 vertex1, Vector2 vertex2, Vector2 vertex3, uint color1, uint color2, uint color3, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws lines between specified triangle vertices and colors.
- void [ThickLineTriangle](#) (Vector2 vertex1, Vector2 vertex2, Vector2 vertex3, uint color1, uint color2, uint color3, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws line path of varying thickness between specified triangle vertices and colors.
- void [LineQuad](#) ([Quad](#) vertices, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a wireframe quadrilateral with the given vertices and colors using "line" primitive.
- void [ThickLineQuad](#) ([Quad](#) vertices, [ColorRect](#) colors, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a wireframe quadrilateral of varying thickness with the given vertices and colors.

- void [LineHexagon](#) (uint color1, uint color2, uint color3, uint color4, uint color5, uint color6, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [LineHexagon](#) ([ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [ThickLineHexagon](#) (uint color1, uint color2, uint color3, uint color4, uint color5, uint color6, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [ThickLineHexagon](#) ([ColorRect](#) colors, float thickness, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Triangle](#) ([Vector2](#) vertex1, [Vector2](#) vertex2, [Vector2](#) vertex3, uint color1, uint color2, uint color3, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws triangle filled with color gradient specified by given positions and colors.
- void [Quad](#) ([Quad](#) vertices, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled quadrilateral with the given vertices and colors.
- void [FillRect](#) ([RectF](#) rect, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled rectangle with the given vertices and colors.
- void [FillRect](#) (float left, float top, float width, float height, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled rectangle with the given vertices and colors.
- void [FrameRect](#) ([RectF](#) rect, [ColorRect](#) colors, float thickness=1.0f, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a rectangle of varying thickness with the given vertices and colors.
- void [FrameRect](#) (float left, float top, float width, float height, [ColorRect](#) colors, float thickness=1.0f, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a rectangle of varying thickness with the given vertices and colors.
- void [FillRoundRect](#) ([RectF](#) rect, [ColorRect](#) colors, float radius, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled rectangle with the given gradient and round corners.
- void [FrameRoundRect](#) ([RectF](#) rect, [ColorRect](#) colors, float radius, float thickness, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a round frame for the given rectangle area with varying thickness and gradient.
- void [FillRoundRectTop](#) ([RectF](#) rect, [ColorRect](#) colors, float radius, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled rectangle with the given gradient and round corners on top.
- void [FillRoundRectBottom](#) ([RectF](#) rect, [ColorRect](#) colors, float radius, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled rectangle with the given gradient and round corners on bottom.
- void [FillRoundRectTopInverse](#) ([RectF](#) rect, [ColorRect](#) colors, float radius, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Highlight](#) ([RectF](#) rect, float cornerRadius, float luma, float distance, int steps)
- void [Hexagon](#) (uint color1, uint color2, uint color3, uint color4, uint color5, uint color6, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Hexagon](#) ([ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Arc](#) ([Vector2](#) origin, [Vector2](#) radius, float initAngle, float endAngle, int steps, uint colorInside, uint color↔Outside, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Arc](#) (float x, float y, float radius, float initAngle, float endAngle, int steps, uint colorInside, uint colorOutside, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Arc](#) ([Vector2](#) origin, [Vector2](#) radius, float initAngle, float endAngle, int steps, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled arc with the given origin, radiuses, angles, steps and colors.
- void [Arc](#) (float x, float y, float radius, float initAngle, float endAngle, int steps, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled arc with the given coordinates, radius, angles, steps and colors.
- void [Ellipse](#) ([Vector2](#) origin, [Vector2](#) radius, int steps, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Circle](#) (float x, float y, float radius, int steps, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Ribbon](#) ([Vector2](#) origin, [Vector2](#) insideRadius, [Vector2](#) outsideRadius, int steps, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws a filled ribbon between inner and outer radiuses filled with four-color gradient.
- void [Ribbon](#) ([Vector2](#) origin, [Vector2](#) insideRadius, [Vector2](#) outsideRadius, int steps, [ColorPair](#) color1, [ColorPair](#) color2, [ColorPair](#) color3, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))

- Draws a filled ribbon between inner and outer radiuses filled with continuous three-pair gradient.*
- void [Ribbon](#) (Vector2 origin, Vector2 insideRadius, Vector2 outsideRadius, int steps, uint color, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a filled ribbon between inner and outer radiuses filled with a single color.*
- void [Tape](#) (Vector2 origin, Vector2 insideRadius, Vector2 outsideRadius, float initAngle, float endAngle, int steps, [ColorRect](#) colors, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a filled tape between the given radiuses and angles, filled with four-color gradient.*
- void [Tape](#) (Vector2 origin, Vector2 insideRadius, Vector2 outsideRadius, float initAngle, float endAngle, int steps, [ColorPair](#) color1, [ColorPair](#) color2, [ColorPair](#) color3, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a filled tape between the given radiuses and angles, filled with continuous three-pair gradient.*
- void [Tape](#) (Vector2 origin, Vector2 insideRadius, Vector2 outsideRadius, float initAngle, float endAngle, int steps, uint color, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a filled tape between the given radiuses and angles, filled with a single color.*
- void [RectWithHole](#) ([RectF](#) rect, Vector2 holeOrigin, Vector2 holeRadius, [ColorPair](#) colors, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a filled rectangle with a hole inside of it, filled with the given colors.*
- void [Draw](#) ([CanvasBuffer](#) buffer, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws triangles from an existing buffer.*
- void [Triangle](#) ([Texture](#) texture, Vector2 vertex1, Vector2 vertex2, Vector2 vertex3, Vector2 tex↵Coord1, Vector2 texCoord2, Vector2 texCoord3, uint color1, uint color2, uint color3, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [TriangleRegion](#) ([Texture](#) texture, Vector2 vertex1, Vector2 vertex2, Vector2 vertex3, Vector2 tex↵Coord1, Vector2 texCoord2, Vector2 texCoord3, uint color1, uint color2, uint color3, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [Quad](#) ([Texture](#) texture, [Quad](#) vertices, [ColorRect?](#) colors=null, [Quad?](#) texCoords=null, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [QuadRegion](#) ([Texture](#) texture, [Quad](#) vertices, [ColorRect?](#) colors, [Quad](#) texCoords, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [RoundRect](#) ([Texture](#) texture, [RectF](#) rect, [Quad](#) texCoords, [ColorRect?](#) colors, float radius, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a textured rectangle with round corners and the given gradient and texture coordinates.*
- void [RoundRectRegion](#) ([Texture](#) texture, [RectF](#) rect, [Quad](#) texCoords, [ColorRect?](#) colors, float radius, int steps, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [QuadImage](#) ([ImageAtlas](#) atlas, [Quad](#) vertices, int regionIndex, [ColorRect?](#) colors=null, [RectF?](#) source↵Rect=null, uint modifiers=0, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- Draws a textured quadrilateral from source portion of image atlas, multiplied by a color gradient.*
- void [ResetSamplerState](#) ()
- Resets any canvas sampler state that is currently set.*
- void [Flush](#) ()
- Flushes the canvas by rendering primitives that are still in batch buffers.*
- void [Reset](#) ()
- Resets canvas internal resources and releases any non-critical memory buffers.*

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Static Public Attributes

- const float [HexagonDelta](#) = 1.154700538f
- Distance between hexagon center and each of its vertices (used for drawing hexagons).*

Properties

- [Device Device](#) [get]
Returns device associated with this canvas.
- [Matrix4x4 World](#) [get, set]
3D world transformation matrix (applies only "Projected" rendering mode).
- [Matrix4x4 ViewProjection](#) [get, set]
Combined 3D view/projection transformation matrix (applies only "Projected" rendering mode).
- [Matrix3x2 Transform](#) [get, set]
2D transformation matrix.
- [SignedDistanceField SignedDistanceField](#) [get, set]
- [CanvasContextState ContextState](#) [get, set]
Device rendering state pre-configured to one of high-level canvas states.
- [CanvasSamplerState SamplerState](#) [get, set]
Canvas texture sampling parameters.
- [uint Attributes](#) [get, set]
Canvas attributes that define rendering characteristics and features.
- [int BatchCount](#) [get]
- [Rect ClipRect](#) [get, set]
Clipping rectangle when performing 2D rendering.

Properties inherited from [Afterwarp.CustomObject](#)

- [IntPtr Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- [IntPtr _handle](#)
Wrapped object's handle.

7.16.1 Detailed Description

Canvas interface that provides utility functions for rendering a variety of 2D shapes filled with solid color gradients and/or textures.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 Canvas()

```
Afterwarp.Canvas.Canvas (
    Device device ) [inline]
```

Creates new 2D rendering canvas.

7.16.3 Member Function Documentation

7.16.3.1 Arc() [1/4]

```
void Afterwarp.Canvas.Arc (
    float x,
    float y,
    float radius,
    float initAngle,
    float endAngle,
    int steps,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled arc with the given coordinates, radius, angles, steps and colors.

7.16.3.2 Arc() [2/4]

```
void Afterwarp.Canvas.Arc (
    float x,
    float y,
    float radius,
    float initAngle,
    float endAngle,
    int steps,
    uint colorInside,
    uint colorOutside,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled arc with the given coordinates, radius, angles, steps and color gradient going away from the center of the arc.

7.16.3.3 Arc() [3/4]

```
void Afterwarp.Canvas.Arc (
    Vector2 origin,
    Vector2 radius,
    float initAngle,
    float endAngle,
    int steps,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled arc with the given origin, radiuses, angles, steps and colors.

7.16.3.4 Arc() [4/4]

```
void Afterwarp.Canvas.Arc (
    Vector2 origin,
    Vector2 radius,
    float initAngle,
    float endAngle,
    int steps,
    uint colorInside,
    uint colorOutside,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled arc with the given origin, radiuses, angles, steps and color gradient going away from the center of the arc.

7.16.3.5 Begin()

```
void Afterwarp.Canvas.Begin ( ) [inline]
```

Starts rendering with the canvas.

7.16.3.6 Circle()

```
void Afterwarp.Canvas.Circle (
    float x,
    float y,
    float radius,
    int steps,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws filled circle at given position and radius. The circle is subdivided into a number of triangles specified in steps. The shape of circle is filled with four-color gradient.

7.16.3.7 Draw()

```
void Afterwarp.Canvas.Draw (
    CanvasBuffer buffer,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws triangles from an existing buffer.

7.16.3.8 Ellipse()

```
void Afterwarp.Canvas.Ellipse (
    Vector2 origin,
    Vector2 radius,
    int steps,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws filled ellipse at given position and radius. The ellipse is subdivided into a number of triangles specified in steps. The shape of ellipse is filled with four-color gradient.

7.16.3.9 End()

```
void Afterwarp.Canvas.End ( )
```

Finishes rendering with the canvas.

7.16.3.10 FillRect() [1/2]

```
void Afterwarp.Canvas.FillRect (
    float left,
    float top,
    float width,
    float height,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with the given vertices and colors.

7.16.3.11 FillRect() [2/2]

```
void Afterwarp.Canvas.FillRect (
    RectF rect,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with the given vertices and colors.

7.16.3.12 FillRoundRect()

```
void Afterwarp.Canvas.FillRoundRect (
    RectF rect,
    ColorRect colors,
    float radius,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with the given gradient and round corners.

7.16.3.13 FillRoundRectBottom()

```
void Afterwarp.Canvas.FillRoundRectBottom (
    RectF rect,
    ColorRect colors,
    float radius,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with the given gradient and round corners on bottom.

7.16.3.14 FillRoundRectTop()

```
void Afterwarp.Canvas.FillRoundRectTop (
    RectF rect,
    ColorRect colors,
    float radius,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with the given gradient and round corners on top.

7.16.3.15 FillRoundRectTopInverse()

```
void Afterwarp.Canvas.FillRoundRectTopInverse (
    RectF rect,
    ColorRect colors,
    float radius,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with the given gradient and round corners, but top is reverted. This can be used to render rounded window background behind caption.

7.16.3.16 Flush()

```
void Afterwarp.Canvas.Flush ( )
```

Flushes the canvas by rendering primitives that are still in batch buffers.

7.16.3.17 FrameRect() [1/2]

```
void Afterwarp.Canvas.FrameRect (
    float left,
    float top,
    float width,
    float height,
    ColorRect colors,
    float thickness = 1::Of,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a rectangle of varying thickness with the given vertices and colors.

7.16.3.18 FrameRect() [2/2]

```
void Afterwarp.Canvas.FrameRect (
    RectF rect,
    ColorRect colors,
    float thickness = 1::Of,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a rectangle of varying thickness with the given vertices and colors.

7.16.3.19 FrameRoundRect()

```
void Afterwarp.Canvas.FrameRoundRect (
    RectF rect,
    ColorRect colors,
    float radius,
    float thickness,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a round frame for the given rectangle area with varying thickness and gradient.

7.16.3.20 Hexagon() [1/2]

```
void Afterwarp.Canvas.Hexagon (
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws hexagon of unity size centered along axis origin and filled with four-color gradient interpolated to the corresponding vertices. The bounding width of hexagon is exactly one, which simplifies placement of multiple hexagons. The final size, position and rotation of hexagon can be given using one or a combination of several 3x2 transformation matrices multiplied together.

7.16.3.21 Hexagon() [2/2]

```
void Afterwarp.Canvas.Hexagon (
    uint color1,
    uint color2,
    uint color3,
    uint color4,
    uint color5,
    uint color6,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws hexagon of unity size centered along axis origin and filled with gradient of six colors at the corresponding vertices. The bounding width of hexagon is exactly one, which simplifies placement of multiple hexagons. The final size, position and rotation of hexagon can be given using one or a combination of several 3x2 transformation matrices multiplied together.

7.16.3.22 Highlight()

```
void Afterwarp.Canvas.Highlight (
    RectF rect,
    float cornerRadius,
    float luma,
    float distance,
    int steps ) [inline]
```

Draws either a highlight or a shadow around the specified rectangle with the given parameters, mimicking effect of gaussian highlight.

7.16.3.23 Line()

```
void Afterwarp.Canvas.Line (
    Vector2 srcPos,
    Vector2 destPos,
    ColorPair colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a line with the given positions and colors.

7.16.3.24 LineCircle()

```
void Afterwarp.Canvas.LineCircle (
    Vector2 origin,
    float radius,
    int steps,
    uint color,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws circle with the given origin, radius, color and steps using "line" primitive.

7.16.3.25 LineEllipse()

```
void Afterwarp.Canvas.LineEllipse (
    Vector2 origin,
    Vector2 radius,
    int steps,
    uint color,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws ellipse with the given origin, radiuses, color and steps using "line" primitive.

7.16.3.26 LineHexagon() [1/2]

```
void Afterwarp.Canvas.LineHexagon (
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws lines between each vertex in a hexagon of unity size centered along axis origin and filled with four-color gradient interpolated to the corresponding vertices. The bounding width of hexagon is exactly one, which simplifies placement of multiple hexagons. The final size, position and rotation of hexagon can be given using one or a combination of several 3x2 transformation matrices multiplied together. This method uses `line` primitive.

7.16.3.27 LineHexagon() [2/2]

```
void Afterwarp.Canvas.LineHexagon (
    uint color1,
    uint color2,
    uint color3,
    uint color4,
    uint color5,
    uint color6,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws lines between each vertex in a hexagon of unity size centered along axis origin and drawn with gradient of six colors at the corresponding vertices. The bounding width of hexagon is exactly one, which simplifies placement of multiple hexagons. The final size, position and rotation of hexagon can be given using one or a combination of several 3x2 transformation matrices multiplied together. This method uses `line` primitive.

7.16.3.28 LineQuad()

```
void Afterwarp.Canvas.LineQuad (
    Quad vertices,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a wireframe quadrilateral with the given vertices and colors using "line" primitive.

7.16.3.29 Lines()

```
void Afterwarp.Canvas.Lines (
    Vector2[] vertices,
    uint[] colors,
    int[] indices,
    int vertexCount,
    int primitiveCount,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws lines with the given vertices, colors and indices.

7.16.3.30 LineTriangle()

```
void Afterwarp.Canvas.LineTriangle (
    Vector2 vertex1,
    Vector2 vertex2,
    Vector2 vertex3,
    uint color1,
    uint color2,
    uint color3,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws lines between specified triangle vertices and colors.

7.16.3.31 Pixel()

```
void Afterwarp.Canvas.Pixel (
    Vector2 position,
    uint color,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a pixel with the given position and color.

7.16.3.32 Pixels()

```
void Afterwarp.Canvas.Pixels (
    Vector2[] positions,
    uint[] colors,
    int elementCount,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws pixels with the given positions and colors.

7.16.3.33 Quad() [1/2]

```
void Afterwarp.Canvas.Quad (
    Quad vertices,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled quadrilateral with the given vertices and colors.

7.16.3.34 Quad() [2/2]

```
void Afterwarp.Canvas.Quad (
    Texture texture,
    Quad vertices,
    ColorRect? colors = null,
    Quad? texCoords = null,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a textured quadrilateral multiplied with a color gradient specified by the given vertices, colors and texture coordinates (in range from 0 to 1).

7.16.3.35 QuadImage()

```
void Afterwarp.Canvas.QuadImage (
    ImageAtlas atlas,
    Quad vertices,
    int regionIndex,
    ColorRect? colors = null,
    RectF? sourceRect = null,
    uint modifiers = 0,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a textured quadrilateral from source portion of image atlas, multiplied by a color gradient.

7.16.3.36 QuadRegion()

```
void Afterwarp.Canvas.QuadRegion (
    Texture texture,
    Quad vertices,
    ColorRect? colors,
    Quad texCoords,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a textured quadrilateral multiplied with a color gradient specified by the given vertices, colors and texture coordinates (in pixels).

7.16.3.37 RectWithHole()

```
void Afterwarp.Canvas.RectWithHole (
    RectF rect,
    Vector2 holeOrigin,
    Vector2 holeRadius,
    ColorPair colors,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled rectangle with a hole inside of it, filled with the given colors.

7.16.3.38 Reset()

```
void Afterwarp.Canvas.Reset ( )
```

Resets canvas internal resources and releases any non-critical memory buffers.

7.16.3.39 ResetSamplerState()

```
void Afterwarp.Canvas.ResetSamplerState ( )
```

Resets any canvas sampler state that is currently set.

7.16.3.40 Ribbon() [1/3]

```
void Afterwarp.Canvas.Ribbon (
    Vector2 origin,
    Vector2 insideRadius,
    Vector2 outsideRadius,
    int steps,
    ColorPair color1,
    ColorPair color2,
    ColorPair color3,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled ribbon between inner and outer radiuses filled with continuous three-pair gradient.

7.16.3.41 Ribbon() [2/3]

```
void Afterwarp.Canvas.Ribbon (
    Vector2 origin,
    Vector2 insideRadius,
    Vector2 outsideRadius,
    int steps,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled ribbon between inner and outer radiuses filled with four-color gradient.

7.16.3.42 Ribbon() [3/3]

```
void Afterwarp.Canvas.Ribbon (
    Vector2 origin,
    Vector2 insideRadius,
    Vector2 outsideRadius,
    int steps,
    uint color,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled ribbon between inner and outer radiuses filled with a single color.

7.16.3.43 RoundRect()

```
void Afterwarp.Canvas.RoundRect (
    Texture texture,
    RectF rect,
    Quad texCoords,
    ColorRect? colors,
    float radius,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a textured rectangle with round corners and the given gradient and texture coordinates.

7.16.3.44 RoundRectRegion()

```
void Afterwarp.Canvas.RoundRectRegion (
    Texture texture,
    RectF rect,
    Quad texCoords,
    ColorRect? colors,
    float radius,
    int steps,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a textured rectangle with round corners and the given gradient and texture coordinates (in pixels).

7.16.3.45 Tape() [1/3]

```
void Afterwarp.Canvas.Tape (
    Vector2 origin,
    Vector2 insideRadius,
    Vector2 outsideRadius,
    float initAngle,
    float endAngle,
    int steps,
    ColorPair color1,
    ColorPair color2,
    ColorPair color3,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled tape between the given radiuses and angles, filled with continuous three-pair gradient.

7.16.3.46 Tape() [2/3]

```
void Afterwarp.Canvas.Tape (
    Vector2 origin,
    Vector2 insideRadius,
    Vector2 outsideRadius,
    float initAngle,
    float endAngle,
    int steps,
    ColorRect colors,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled tape between the given radiuses and angles, filled with four-color gradient.

7.16.3.47 Tape() [3/3]

```
void Afterwarp.Canvas.Tape (
    Vector2 origin,
    Vector2 insideRadius,
    Vector2 outsideRadius,
    float initAngle,
    float endAngle,
    int steps,
    uint color,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a filled tape between the given radiuses and angles, filled with a single color.

7.16.3.48 TexturedTriangles()

```
void Afterwarp.Canvas.TexturedTriangles (
    Texture texture,
    Vector2[] vertices,
    Vector2[] texCoords,
    uint[] colors,
    int[] indices,
    int vertexCount,
    int primitiveCount,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws triangles with the given vertices, texture coordinates, colors and indices.

7.16.3.49 ThickLine()

```
void Afterwarp.Canvas.ThickLine (
    Vector2 srcPos,
    Vector2 destPos,
    ColorPair colors,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a line of varying thickness with the given positions and colors.

7.16.3.50 ThickLineCircle()

```
void Afterwarp.Canvas.ThickLineCircle (
    Vector2 origin,
    float radius,
    int steps,
    uint color,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws circle of varying thickness with the given origin, radius, color and steps.

7.16.3.51 ThickLineEllipse()

```
void Afterwarp.Canvas.ThickLineEllipse (
    Vector2 origin,
    Vector2 radius,
    int steps,
    uint color,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws ellipse of varying thickness with the given origin, radiuses, color and steps.

7.16.3.52 ThickLineHexagon() [1/2]

```
void Afterwarp.Canvas.ThickLineHexagon (
    ColorRect colors,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws lines of varying thickness between each vertex in a hexagon of unity size centered along axis origin and filled with four-color gradient interpolated to the corresponding vertices. The final size, position and rotation of hexagon can be given using one or a combination of several 3x2 transformation matrices multiplied together.

7.16.3.53 ThickLineHexagon() [2/2]

```
void Afterwarp.Canvas.ThickLineHexagon (
    uint color1,
    uint color2,
    uint color3,
    uint color4,
    uint color5,
    uint color6,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws lines of varying thickness between each vertex in a hexagon of unity size centered along axis origin and drawn with gradient of six colors at the corresponding corners. The final size, position and rotation of hexagon can be given using one or a combination of several 3x2 transformation matrices multiplied together.

7.16.3.54 ThickLineQuad()

```
void Afterwarp.Canvas.ThickLineQuad (
    Quad vertices,
    ColorRect colors,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a wireframe quadrilateral of varying thickness with the given vertices and colors.

7.16.3.55 ThickLineTriangle()

```
void Afterwarp.Canvas.ThickLineTriangle (
    Vector2 vertex1,
    Vector2 vertex2,
    Vector2 vertex3,
    uint color1,
    uint color2,
    uint color3,
    float thickness,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws line path of varying thickness between specified triangle vertices and colors.

7.16.3.56 Triangle() [1/2]

```
void Afterwarp.Canvas.Triangle (
    Texture texture,
    Vector2 vertex1,
    Vector2 vertex2,
    Vector2 vertex3,
    Vector2 texCoord1,
    Vector2 texCoord2,
    Vector2 texCoord3,
    uint color1,
    uint color2,
    uint color3,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a textured triangle multiplied with a color gradient specified by given positions, colors and texture coordinates (in range from 0 to 1).

7.16.3.57 Triangle() [2/2]

```
void Afterwarp.Canvas.Triangle (
    Vector2 vertex1,
    Vector2 vertex2,
    Vector2 vertex3,
    uint color1,
    uint color2,
    uint color3,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws triangle filled with color gradient specified by given positions and colors.

7.16.3.58 TriangleRegion()

```
void Afterwarp.Canvas.TriangleRegion (
    Texture texture,
    Vector2 vertex1,
    Vector2 vertex2,
    Vector2 vertex3,
    Vector2 texCoord1,
```

```

Vector2 texCoord2,
Vector2 texCoord3,
uint color1,
uint color2,
uint color3,
BlendingEffect effect = BlendingEffect::Normal ) [inline]

```

Draws a textured triangle multiplied with a color gradient specified by given positions, colors and texture coordinates (in pixels).

7.16.3.59 Triangles()

```

void Afterwarp.Canvas.Triangles (
    Vector2[] vertices,
    uint[] colors,
    int[] indices,
    int vertexCount,
    int primitiveCount,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]

```

Draws triangles with the given vertices, colors and indices.

7.16.4 Member Data Documentation

7.16.4.1 HexagonDelta

```
const float Afterwarp.Canvas.HexagonDelta = 1.154700538f [static]
```

Distance between hexagon center and each of its vertices (used for drawing hexagons).

7.16.5 Property Documentation

7.16.5.1 Attributes

```
uint Afterwarp.Canvas.Attributes [get], [set]
```

Canvas attributes that define rendering characteristics and features.

7.16.5.2 BatchCount

```
int Afterwarp.Canvas.BatchCount [get]
```

Number of batches that were rendered between current set of begin/end block. Drawing each individual batch involves some overhead, so when this parameter happens to be considerably high at some point, the rendering code should be revised for better grouping of images, shapes and blending types.

7.16.5.3 ClipRect

`Rect` Afterwarp.Canvas.ClipRect [get], [set]

Clipping rectangle when performing 2D rendering.

7.16.5.4 ContextState

`CanvasContextState` Afterwarp.Canvas.ContextState [get], [set]

Device rendering state pre-configured to one of high-level canvas states.

7.16.5.5 Device

`Device` Afterwarp.Canvas.Device [get]

Returns device associated with this canvas.

7.16.5.6 SamplerState

`CanvasSamplerState` Afterwarp.Canvas.SamplerState [get], [set]

Canvas texture sampling parameters.

7.16.5.7 SignedDistanceField

`SignedDistanceField` Afterwarp.Canvas.SignedDistanceField [get], [set]

Signed Distance Field (SDF) parameters. The actual SDF rendering needs to be enabled by specifying one of the appropriate canvas attributes.

7.16.5.8 Transform

`Matrix3x2` Afterwarp.Canvas.Transform [get], [set]

2D transformation matrix.

7.16.5.9 ViewProjection

`Matrix4x4` Afterwarp.Canvas.ViewProjection [get], [set]

Combined 3D view/projection transformation matrix (applies only "Projected" rendering mode).

7.16.5.10 World

Matrix4x4 Afterwarp.Canvas.World [get], [set]

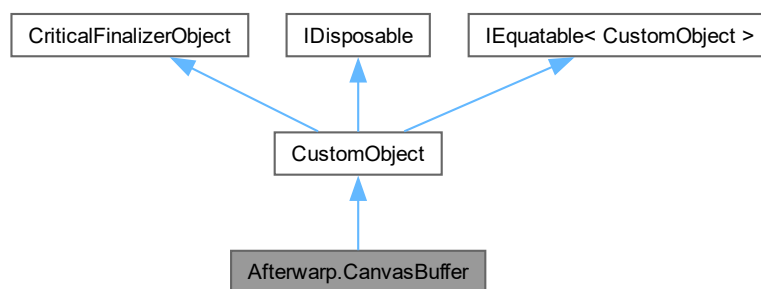
3D world transformation matrix (applies only "Projected" rendering mode).

The documentation for this class was generated from the following file:

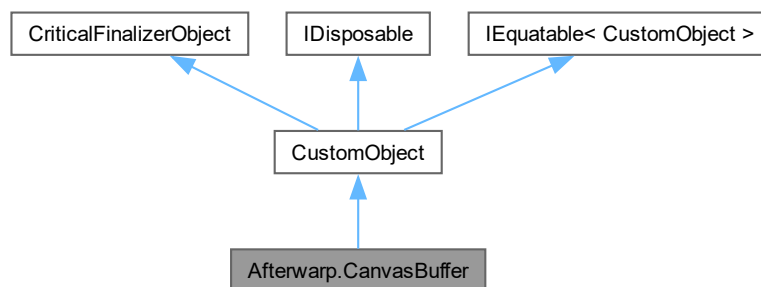
- [Afterwarp.Graphics.cs](#)

7.17 Afterwarp.CanvasBuffer Class Reference

Inheritance diagram for Afterwarp.CanvasBuffer:



Collaboration diagram for Afterwarp.CanvasBuffer:



Public Member Functions

- `CanvasBuffer ()`
Creates new instance of canvas buffer.
- `void Clear ()`
Clears all buffers.
- `void ClearAndShrink ()`
Clears all buffers and releases their memory.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- int [VertexCount](#) [get, set]
Total number of vertices in the buffer.
- int [IndexCount](#) [get, set]
Total number of indices in the buffer.
- Vector2[] [Vertices](#) [get, set]
Returns or changes existing vertices in the buffer.
- uint[] [Colors](#) [get, set]
Returns or changes existing colors in the buffer.
- int[] [Indices](#) [get, set]
Returns or changes existing indices in the buffer.
- [RectF Extents](#) [get]
Calculates boundaries for the existing vertices.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.17.1 Detailed Description

A container for vertices and indices defining one or more triangular meshes that can be suitable for rendering with a 2D canvas.

7.17.2 Constructor & Destructor Documentation

7.17.2.1 CanvasBuffer()

```
Afterwarp.CanvasBuffer.CanvasBuffer ( ) [inline]
```

Creates new instance of canvas buffer.

7.17.3 Member Function Documentation

7.17.3.1 Clear()

```
void Afterwarp.CanvasBuffer.Clear ( )
```

Clears all buffers.

7.17.3.2 ClearAndShrink()

```
void Afterwarp.CanvasBuffer.ClearAndShrink ( )
```

Clears all buffers and releases their memory.

7.17.4 Property Documentation

7.17.4.1 Colors

```
uint [ ] Afterwarp.CanvasBuffer.Colors [get], [set]
```

Returns or changes existing colors in the buffer.

7.17.4.2 Extents

```
RectF Afterwarp.CanvasBuffer.Extents [get]
```

Calculates boundaries for the existing vertices.

7.17.4.3 IndexCount

```
int Afterwarp.CanvasBuffer.IndexCount [get], [set]
```

Total number of indices in the buffer.

7.17.4.4 Indices

```
int [ ] Afterwarp.CanvasBuffer.Indices [get], [set]
```

Returns or changes existing indices in the buffer.

7.17.4.5 VertexCount

```
int Afterwarp.CanvasBuffer.VertexCount [get], [set]
```

Total number of vertices in the buffer.

7.17.4.6 Vertices

```
Vector2 [] Afterwarp.CanvasBuffer.Vertices [get], [set]
```

Returns or changes existing vertices in the buffer.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.18 Afterwarp.CanvasSamplerState Struct Reference

Sampler parameters that can be easily configured by the canvas.

Public Member Functions

- [CanvasSamplerState](#) ([TextureFilter](#) filterMin, [TextureFilter](#) filterMag, [TextureFilter](#) filterMip=[TextureFilter.None](#), [TextureAddress](#) addressU=[TextureAddress.Wrap](#), [TextureAddress](#) addressV=[TextureAddress.Wrap](#), uint borderColor=0xFFFFFFFFu)

Creates canvas sampler parameters with the given values.

Public Attributes

- [TextureFilter FilterMin](#)
Minification texture filtering.
- [TextureFilter FilterMag](#)
Magnification texture filtering.
- [TextureFilter FilterMip](#)
Mipmapping texture filtering.
- [TextureAddress AddressU](#)
Horizontal texture coordinate addressing.
- [TextureAddress AddressV](#)
Vertical texture coordinate addressing.
- uint [BorderColor](#)
Special color constant to be used for TextureAddress::Border addressing mode.

Properties

- static [CanvasSamplerState Default](#) [get]
Returns default canvas sampler parameters.

7.18.1 Detailed Description

Sampler parameters that can be easily configured by the canvas.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 CanvasSamplerState()

```
Afterwarp.CanvasSamplerState.CanvasSamplerState (
    TextureFilter filterMin,
    TextureFilter filterMag,
    TextureFilter filterMip = TextureFilter::None,
    TextureAddress addressU = TextureAddress::Wrap,
    TextureAddress addressV = TextureAddress::Wrap,
    uint borderColor = 0xFFFFFFFFu ) [inline]
```

Creates canvas sampler parameters with the given values.

7.18.3 Member Data Documentation

7.18.3.1 AddressU

```
TextureAddress Afterwarp.CanvasSamplerState.AddressU
```

Horizontal texture coordinate addressing.

7.18.3.2 AddressV

```
TextureAddress Afterwarp.CanvasSamplerState.AddressV
```

Vertical texture coordinate addressing.

7.18.3.3 BorderColor

```
uint Afterwarp.CanvasSamplerState.BorderColor
```

Special color constant to be used for TextureAddress::Border addressing mode.

7.18.3.4 FilterMag

```
TextureFilter Afterwarp.CanvasSamplerState.FilterMag
```

Magnification texture filtering.

7.18.3.5 FilterMin

`TextureFilter Afterwarp.CanvasSamplerState.FilterMin`

Minification texture filtering.

7.18.3.6 FilterMip

`TextureFilter Afterwarp.CanvasSamplerState.FilterMip`

Mipmapping texture filtering.

7.18.4 Property Documentation

7.18.4.1 Default

`CanvasSamplerState Afterwarp.CanvasSamplerState.Default [static], [get]`

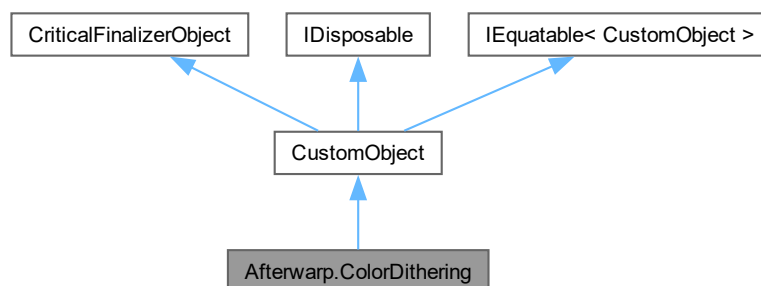
Returns default canvas sampler parameters.

The documentation for this struct was generated from the following file:

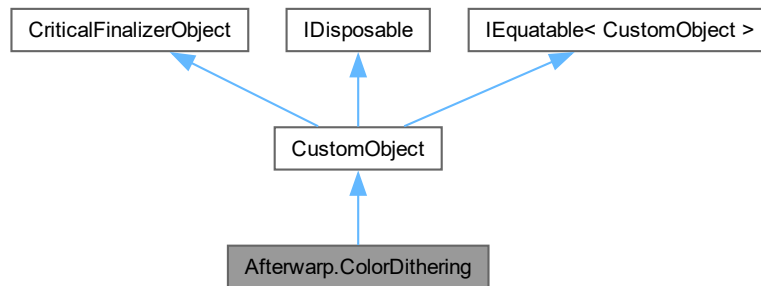
- [Afterwarp.Types.cs](#)

7.19 Afterwarp.ColorDithering Class Reference

Inheritance diagram for `Afterwarp.ColorDithering`:



Collaboration diagram for Afterwarp.ColorDithering:



Public Member Functions

- [ColorDithering](#) ([Device](#) device)
Creates new instance of Color Dithering module.
- void [Execute](#) ([Texture](#) source)
Performs color dithering of the source texture rendering a full-screen quad.
- void [Execute](#) ([Texture](#) destination, [Texture](#) source)
Performs color dithering of the source texture rendering to destination texture.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device associated with the module.
- [ColorDitheringFormat](#) [Format](#) [get, set]
Returns or changes a target color quantization format for dithering.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)

Wrapped object's handle.

7.19.1 Detailed Description

Color dithering module that can be used when rendering from higher precision color formats (e.g. HDR) to lower precision, effectively reducing banding effect.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 ColorDithering()

```
Afterwarp.ColorDithering.ColorDithering (
    Device device ) [inline]
```

Creates new instance of Color Dithering module.

7.19.3 Member Function Documentation

7.19.3.1 Execute() [1/2]

```
void Afterwarp.ColorDithering.Execute (
    Texture destination,
    Texture source ) [inline]
```

Performs color dithering of the source texture rendering to destination texture.

7.19.3.2 Execute() [2/2]

```
void Afterwarp.ColorDithering.Execute (
    Texture source ) [inline]
```

Performs color dithering of the source texture rendering a full-screen quad.

7.19.4 Property Documentation

7.19.4.1 Device

```
Device Afterwarp.ColorDithering.Device [get]
```

Returns device associated with the module.

7.19.4.2 Format

`ColorDitheringFormat` Afterwarp.ColorDithering.Format [get], [set]

Returns or changes a target color quantization format for dithering.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.20 Afterwarp.ColorPair Struct Reference

Public Member Functions

- `ColorPair` (uint color)
Creates a new color pair with a given color.
- `ColorPair` (uint first, uint second)
Creates a new color pair with a given values.

Public Attributes

- uint `First`
First color entry, which can be reinterpreted as top or left color depending on context.
- uint `Second`
Second color entry, which can be reinterpreted as bottom or right color depending on context.

Properties

- static `ColorPair Black` [get]
Returns a pair of opaque Black colors.
- static `ColorPair White` [get]
Returns a pair of opaque White colors.
- static `ColorPair TranslucentBlack` [get]
Returns a pair of translucent Black colors.
- static `ColorPair TranslucentWhite` [get]
Returns a pair of translucent White colors.

7.20.1 Detailed Description

A combination of two colors, primarily used for displaying text with the first color being on top and the second being on bottom. The format for specifying colors is defined as ARGB8).

7.20.2 Constructor & Destructor Documentation

7.20.2.1 ColorPair() [1/2]

```
Afterwarp.ColorPair.ColorPair (
    uint color ) [inline]
```

Creates a new color pair with a given color.

7.20.2.2 ColorPair() [2/2]

```
Afterwarp.ColorPair.ColorPair (
    uint first,
    uint second ) [inline]
```

Creates a new color pair with a given values.

7.20.3 Member Data Documentation

7.20.3.1 First

```
uint Afterwarp.ColorPair.First
```

First color entry, which can be reinterpreted as top or left color depending on context.

7.20.3.2 Second

```
uint Afterwarp.ColorPair.Second
```

Second color entry, which can be reinterpreted as bottom or right color depending on context.

7.20.4 Property Documentation

7.20.4.1 Black

```
ColorPair Afterwarp.ColorPair.Black [static], [get]
```

Returns a pair of opaque Black colors.

7.20.4.2 TranslucentBlack

```
ColorPair Afterwarp.ColorPair.TranslucentBlack [static], [get]
```

Returns a pair of translucent Black colors.

7.20.4.3 TranslucentWhite

`ColorPair` Afterwarp.ColorPair.TranslucentWhite [static], [get]

Returns a pair of translucent White colors.

7.20.4.4 White

`ColorPair` Afterwarp.ColorPair.White [static], [get]

Returns a pair of opaque White colors.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.21 Afterwarp.ColorRect Struct Reference

Public Member Functions

- `ColorRect` (uint color)
Creates a color rectangle with all values set to zero.
- `ColorRect` (uint topLeft, uint topRight, uint bottomRight, uint bottomLeft)
Creates a color rectangle with a given values.

Static Public Member Functions

- static `ColorRect GradientX` (`ColorPair` colors)
Creates a color rectangle of four colors from two color pair to create horizontal gradient.
- static `ColorRect GradientX` (uint left, uint right)
Creates a color rectangle of four colors from two color values to create horizontal gradient.
- static `ColorRect GradientY` (`ColorPair` colors)
Creates a color rectangle of four colors from two color pair to create vertical gradient.
- static `ColorRect GradientY` (uint top, uint bottom)
Creates a color rectangle of four colors from two color values to create vertical gradient.

Public Attributes

- uint `TopLeft`
Color corresponding to top / left corner.
- uint `TopRight`
Color corresponding to top / right corner.
- uint `BottomRight`
Color corresponding to bottom / right corner.
- uint `BottomLeft`
Color corresponding to bottom / left corner.

Properties

- static [ColorRect Black](#) [get]
Returns a color rectangle with opaque Black color.
- static [ColorRect White](#) [get]
Returns a color rectangle with opaque White color.
- static [ColorRect TranslucentBlack](#) [get]
Returns a color rectangle with translucent Black color.
- static [ColorRect TranslucentWhite](#) [get]
Returns a color rectangle with translucent White color.

7.21.1 Detailed Description

A rectangle of four colors, primarily used for displaying colored quads, where each color corresponds to top/left, top/right, bottom/right and bottom/left accordingly (clockwise). The format for specifying colors is defined as ARGB8.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 ColorRect() [1/2]

```
Afterwarp.ColorRect.ColorRect (
    uint color ) [inline]
```

Creates a color rectangle with all values set to zero.

Creates a color rectangle with four colors having the same component in each corner.

7.21.2.2 ColorRect() [2/2]

```
Afterwarp.ColorRect.ColorRect (
    uint topLeft,
    uint topRight,
    uint bottomRight,
    uint bottomLeft ) [inline]
```

Creates a color rectangle with a given values.

7.21.3 Member Function Documentation

7.21.3.1 GradientX() [1/2]

```
static ColorRect Afterwarp.ColorRect.GradientX (
    ColorPair colors ) [static]
```

Creates a color rectangle of four colors from two color pair to create horizontal gradient.

7.21.3.2 GradientX() [2/2]

```
static ColorRect Afterwarp.ColorRect.GradientX (  
    uint left,  
    uint right ) [static]
```

Creates a color rectangle of four colors from two color values to create horizontal gradient.

7.21.3.3 GradientY() [1/2]

```
static ColorRect Afterwarp.ColorRect.GradientY (  
    ColorPair colors ) [static]
```

Creates a color rectangle of four colors from two color pair to create vertical gradient.

7.21.3.4 GradientY() [2/2]

```
static ColorRect Afterwarp.ColorRect.GradientY (  
    uint top,  
    uint bottom ) [static]
```

Creates a color rectangle of four colors from two color values to create vertical gradient.

7.21.4 Member Data Documentation

7.21.4.1 BottomLeft

```
uint Afterwarp.ColorRect.BottomLeft
```

Color corresponding to bottom / left corner.

7.21.4.2 BottomRight

```
uint Afterwarp.ColorRect.BottomRight
```

Color corresponding to bottom / right corner.

7.21.4.3 TopLeft

```
uint Afterwarp.ColorRect.TopLeft
```

Color corresponding to top / left corner.

7.21.4.4 TopRight

```
uint Afterwarp.ColorRect.TopRight
```

Color corresponding to top / right corner.

7.21.5 Property Documentation

7.21.5.1 Black

`ColorRect` `Afterwarp.ColorRect.Black` [static], [get]

Returns a color rectangle with opaque Black color.

7.21.5.2 TranslucentBlack

`ColorRect` `Afterwarp.ColorRect.TranslucentBlack` [static], [get]

Returns a color rectangle with translucent Black color.

7.21.5.3 TranslucentWhite

`ColorRect` `Afterwarp.ColorRect.TranslucentWhite` [static], [get]

Returns a color rectangle with translucent White color.

7.21.5.4 White

`ColorRect` `Afterwarp.ColorRect.White` [static], [get]

Returns a color rectangle with opaque White color.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.22 Afterwarp.ComputeBindTextureFormat Struct Reference

Compute texture binding parameters.

Public Member Functions

- `ComputeBindTextureFormat` (uint channel, int mipLevel, int layer, `ComputeTextureAccess` access, `PixelFormat` format)

Creates new binding parameters with the given values.

Public Attributes

- uint [Channel](#)
Channel to which the texture is bound to.
- int [MipLevel](#)
Number of mipmap level that should be attached.
- int [Layer](#)
Layer number from a volume or array texture to bound (-1 means an entire texture should be bound).
- [ComputeTextureAccess](#) [Access](#)
Type of access that will be used with the texture.
- [PixelFormat](#) [Format](#)
Format of the texture's pixels.

7.22.1 Detailed Description

Compute texture binding parameters.

7.22.2 Constructor & Destructor Documentation

7.22.2.1 ComputeBindTextureFormat()

```
Afterwarp.ComputeBindTextureFormat.ComputeBindTextureFormat (
    uint channel,
    int mipLevel,
    int layer,
    ComputeTextureAccess access,
    PixelFormat format ) [inline]
```

Creates new binding parameters with the given values.

7.22.3 Member Data Documentation

7.22.3.1 Access

[ComputeTextureAccess](#) Afterwarp.ComputeBindTextureFormat.Access

Type of access that will be used with the texture.

7.22.3.2 Channel

uint Afterwarp.ComputeBindTextureFormat.Channel

Channel to which the texture is bound to.

7.22.3.3 Format

`PixelFormat Afterwarp.ComputeBindTextureFormat.Format`

Format of the texture's pixels.

7.22.3.4 Layer

`int Afterwarp.ComputeBindTextureFormat.Layer`

Layer number from a volume or array texture to bound (-1 means an entire texture should be bound).

7.22.3.5 MipLevel

`int Afterwarp.ComputeBindTextureFormat.MipLevel`

Number of mipmap level that should be attached.

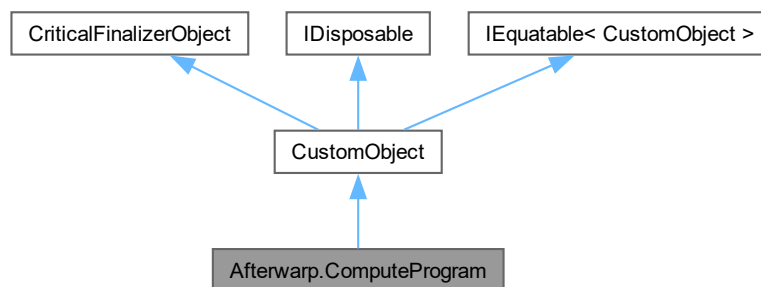
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

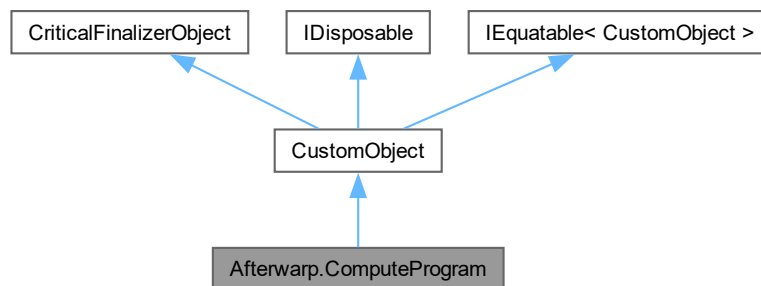
7.23 Afterwarp.ComputeProgram Class Reference

Compute shader program for general processing on GPU (GPGPU).

Inheritance diagram for `Afterwarp.ComputeProgram`:



Collaboration diagram for Afterwarp.ComputeProgram:



Public Member Functions

- **ComputeProgram** (**Device** device, **ProgramElement**[] programElements, byte[] shader)
Creates a new instance of compute shader program.
- void **Bind** (**Buffer** buffer, uint channel=0u, uint offset=0u)
- void **Unbind** (**Buffer** buffer, uint channel=0)
- void **Bind** (**Texture** texture, **ComputeBindTextureFormat** bindFormat)
Assigns texture to a particular channel.
- void **Unbind** (**Texture** texture, **ComputeBindTextureFormat** bindFormat)
Removes existing texture association with the particular channel.
- void **ResetBindings** ()
Resets any bindings that were previously made as if Unbind() would be called for each of them.
- void **Commit** ()
- void **Begin** ()
Activates the compute shader program and prepares for rendering.
- void **End** ()
Deactivates the compute shader program, resets previously active input streams and bindings.
- void **Dispatch** (int groupsX, int groupsY, int groupsZ)
Launches one or more compute shader work groups.

Public Member Functions inherited from **Afterwarp.CustomObject**

- void **Dispose** ()
- bool **Equals** (**CustomObject** other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()

Properties

- **Device Device** [get]
Returns device associated with this compute program.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.23.1 Detailed Description

Compute shader program for general processing on GPU (GPGPU).

7.23.2 Constructor & Destructor Documentation

7.23.2.1 ComputeProgram()

```
Afterwarp.ComputeProgram.ComputeProgram (
    Device device,
    ProgramElement[] programElements,
    byte[] shader ) [inline]
```

Creates a new instance of compute shader program.

7.23.3 Member Function Documentation

7.23.3.1 Begin()

```
void Afterwarp.ComputeProgram.Begin ( ) [inline]
```

Activates the compute shader program and prepares for rendering.

7.23.3.2 Bind() [1/2]

```
void Afterwarp.ComputeProgram.Bind (
    Buffer buffer,
    uint channel = 0u,
    uint offset = 0u ) [inline]
```

Assigns a generic buffer (or a portion of starting from the given offset in bytes) to a particular channel. The current bindings can be considered valid until End() or ResetBindings() is called.

7.23.3.3 Bind() [2/2]

```
void Afterwarp.ComputeProgram.Bind (
    Texture texture,
    ComputeBindTextureFormat bindFormat ) [inline]
```

Assigns texture to a particular channel.

7.23.3.4 Commit()

```
void Afterwarp.ComputeProgram.Commit ( ) [inline]
```

Applies any binding changes to constant buffers, which usually occur when a dispatch call is issued, to occur immediately.

7.23.3.5 Dispatch()

```
void Afterwarp.ComputeProgram.Dispatch (
    int groupsX,
    int groupsY,
    int groupsZ ) [inline]
```

Launches one or more compute shader work groups.

7.23.3.6 End()

```
void Afterwarp.ComputeProgram.End ( )
```

Deactivates the compute shader program, resets previously active input streams and bindings.

7.23.3.7 ResetBindings()

```
void Afterwarp.ComputeProgram.ResetBindings ( )
```

Resets any bindings that were previously made as if Unbind() would be called for each of them.

7.23.3.8 Unbind() [1/2]

```
void Afterwarp.ComputeProgram.Unbind (
    Buffer buffer,
    uint channel = 0 ) [inline]
```

Removes association that was previously made with Bind() between the buffer and the given channel.

7.23.3.9 Unbind() [2/2]

```
void Afterwarp.ComputeProgram.Unbind (
    Texture texture,
    ComputeBindTextureFormat bindFormat ) [inline]
```

Removes existing texture association with the particular channel.

7.23.4 Property Documentation

7.23.4.1 Device

```
Device Afterwarp.ComputeProgram.Device [get]
```

Returns device associated with this compute program.

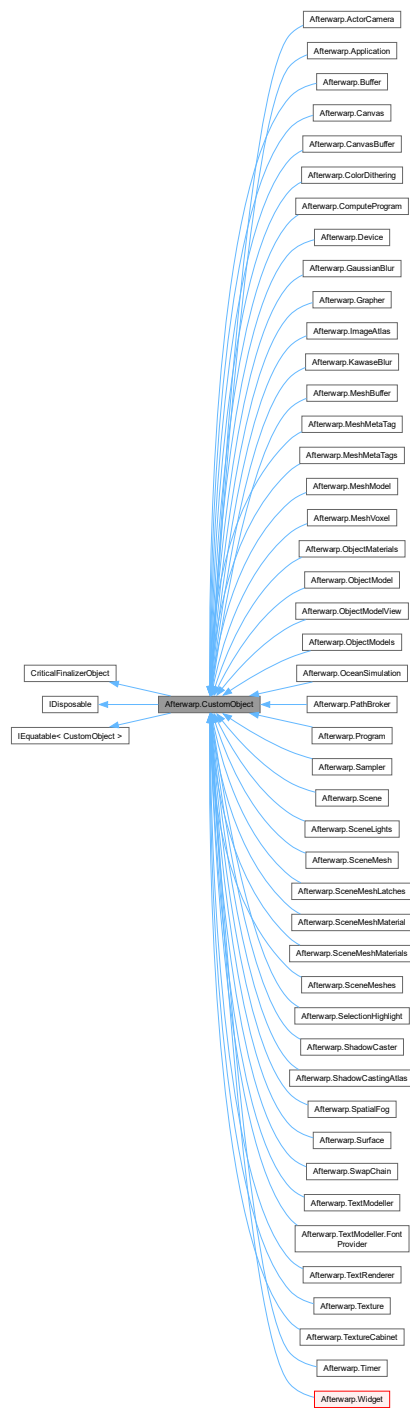
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

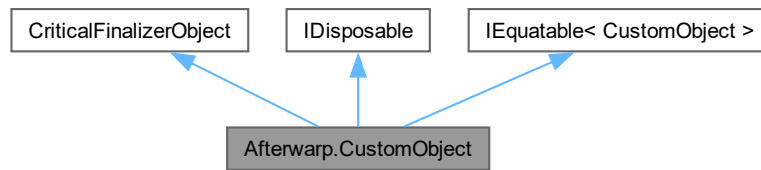
7.24 Afterwarp.CustomObject Class Reference

A high-level class that wraps one of [Afterwarp](#)'s objects.

Inheritance diagram for Afterwarp.CustomObject:



Collaboration diagram for Afterwarp.CustomObject:



Public Member Functions

- void `Dispose` ()
- bool `Equals` (`CustomObject` other)
- override bool `Equals` (object obj)
- override int `GetHashCode` ()

Protected Member Functions

- virtual void `Dispose` (bool disposing)
Releases the object's resources.

Protected Attributes

- IntPtr `_handle`
Wrapped object's handle.

Properties

- IntPtr `Handle` [get]
Wrapped object's handle.

7.24.1 Detailed Description

A high-level class that wraps one of `Afterwarp`'s objects.

7.24.2 Member Function Documentation

7.24.2.1 `Dispose()` [1/2]

```
void Afterwarp.CustomObject.Dispose ( ) [inline]
```

7.24.2.2 Dispose() [2/2]

```
virtual void Afterwarp.CustomObject.Dispose (
    bool disposing ) [inline], [protected], [virtual]
```

Releases the object's resources.

7.24.2.3 Equals() [1/2]

```
bool Afterwarp.CustomObject.Equals (
    CustomObject other ) [inline]
```

7.24.2.4 Equals() [2/2]

```
override bool Afterwarp.CustomObject.Equals (
    object obj ) [inline]
```

7.24.2.5 GetHashCode()

```
override int Afterwarp.CustomObject.GetHashCode ( ) [inline]
```

7.24.3 Member Data Documentation

7.24.3.1 _handle

```
IntPtr Afterwarp.CustomObject._handle [protected]
```

Wrapped object's handle.

7.24.4 Property Documentation

7.24.4.1 Handle

```
IntPtr Afterwarp.CustomObject.Handle [get]
```

Wrapped object's handle.

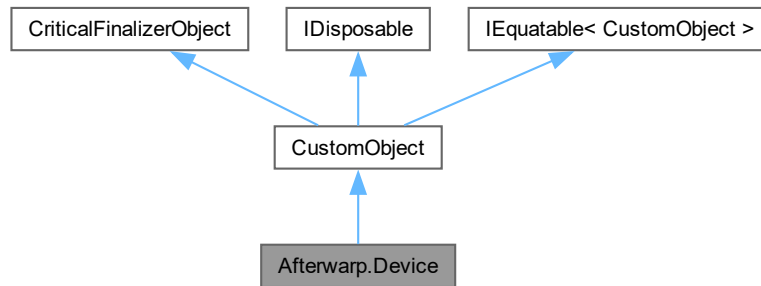
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

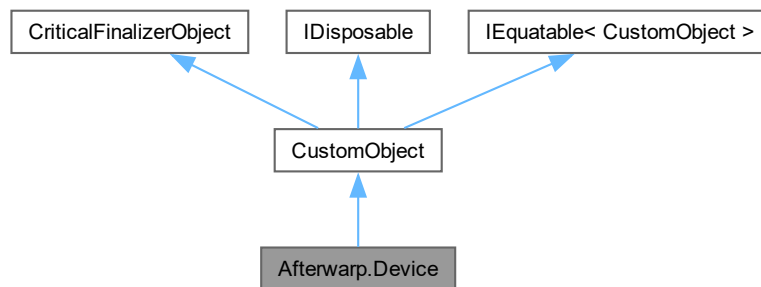
7.25 Afterwarp.Device Class Reference

Device class that handles initialization, rendering, state management and other common tasks.

Inheritance diagram for Afterwarp.Device:



Collaboration diagram for Afterwarp.Device:



Classes

- struct [Attribute](#)
Device attributes that define its behavior.

Public Member Functions

- [Device](#) ([DeviceTechnology](#) technology, uint attributes=0)
Creates a new graphics device of the given technology and attributes.
- void [ResetCache](#) ()
Releases any cached device resources.
- void [Clear](#) (byte layers, [FloatColor](#) color, float depth=1.0f, uint stencil=0)
Clears rendering surface that is currently active.
- void [MemoryBarrier](#) (uint bits)
Issues a memory barrier according to the specified bit flags.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- uint [Attributes](#) [get]
Returns attributes with which the device has been created.
- uint [Behavior](#) [get]
Returns device behavior attributes.
- uint [LegacyBits](#) [get]
Returns legacy feature bits that define older hardware features.
- [DeviceTechnology Technology](#) [get]
Returns graphics technology type that is currently being used.
- [DevicePlatform Platform](#) [get]
Returns OS platform the device application is currently running on.
- uint [TechVersion](#) [get]
- uint [TechFeatureVersion](#) [get]
- [RenderingState RenderingState](#) [get, set]
Rendering parameters and operation characteristics of the device.
- [Rect Viewport](#) [get, set]
Current viewport parameters.
- [Rect Scissor](#) [get, set]
Current scissor parameters.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.25.1 Detailed Description

Device class that handles initialization, rendering, state management and other common tasks.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 Device()

```
Afterwarp.Device.Device (
    DeviceTechnology technology,
    uint attributes = 0 ) [inline]
```

Creates a new graphics device of the given technology and attributes.

7.25.3 Member Function Documentation

7.25.3.1 Clear()

```
void Afterwarp.Device.Clear (
    byte layers,
    FloatColor color,
    float depth = 1::0f,
    uint stencil = 0 ) [inline]
```

Clears rendering surface that is currently active.

7.25.3.2 MemoryBarrier()

```
void Afterwarp.Device.MemoryBarrier (
    uint bits )
```

Issues a memory barrier according to the specified bit flags.

7.25.3.3 ResetCache()

```
void Afterwarp.Device.ResetCache ( )
```

Releases any cached device resources.

7.25.4 Property Documentation

7.25.4.1 Attributes

```
uint Afterwarp.Device.Attributes [get]
```

Returns attributes with which the device has been created.

7.25.4.2 Behavior

```
uint Afterwarp.Device.Behavior [get]
```

Returns device behavior attributes.

7.25.4.3 LegacyBits

```
uint Afterwarp.Device.LegacyBits [get]
```

Returns legacy feature bits that define older hardware features.

7.25.4.4 Platform

```
DevicePlatform Afterwarp.Device.Platform [get]
```

Returns OS platform the device application is currently running on.

7.25.4.5 RenderingState

```
RenderingState Afterwarp.Device.RenderingState [get], [set]
```

Rendering parameters and operation characteristics of the device.

7.25.4.6 Scissor

```
Rect Afterwarp.Device.Scissor [get], [set]
```

Current scissor parameters.

7.25.4.7 TechFeatureVersion

```
uint Afterwarp.Device.TechFeatureVersion [get]
```

Returns the feature level version of technology that is currently being used. The difference between this and technology version is that the last one indicates type of technology being used (for example, DirectX 3D), while this one indicates the level of features available (for example, DirectX 9.0c). The values here are specified in hexadecimal format. That is, a value of 0x213 would indicate version 2.1.3.

7.25.4.8 Technology

```
DeviceTechnology Afterwarp.Device.Technology [get]
```

Returns graphics technology type that is currently being used.

7.25.4.9 TechVersion

```
uint Afterwarp.Device.TechVersion [get]
```

Returns the version of technology that is currently being used. The values are specified in hexadecimal format. That is, a value of 0x100 indicates version 1.0, while a value of 0x247 would indicate version 2.4.7. The value is used in combination with device technology, so if technology is DirectX3D, and this function returns 0xA10, it means that DirectX3D 10.1 is being used.

7.25.4.10 Viewport

`Rect Afterwarp.Device.Viewport [get], [set]`

Current viewport parameters.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.26 Afterwarp.DeviceBehavior Struct Reference

Device behavior attributes that define the rendering code path.

Static Public Attributes

- const uint [PerSampleShading](#) = 0x00000001
Graphics hardware supports executing fragment shader per each MSAA sample.
- const uint [DepthClipNegative](#) = 0x00000002
- const uint [Compute](#) = 0x00000004
Compute shader support is available.
- const uint [Tessellation](#) = 0x00000008
Tessellation shader support is available.
- const uint [VariableRateRefresh](#) = 0x00000100
Direct3D 11: Variable Rate Refresh (VRR) display support is available.
- const uint [PostDepthCoverage](#) = 0x00000200
OpenGL: ARB_post_depth_coverage, Direct3D 11: PostZCoverageEnable through NvAPI.
- const uint [ForceBufferUnbind](#) = 0x00001000
OpenGL / Intel HD Graphics bug workaround: always unbind buffers before updating their contents.
- const uint [DepthClearBadPrecision](#) = 0x00002000
Direct3D 11 / Parallels bug workaround: clear depth buffer with value lower than one.
- const uint [SignedNormIntFormatBugged](#) = 0x00004000
Direct3D 11 / VMware bug workaround: avoid signed normalized integer formats.

7.26.1 Detailed Description

Device behavior attributes that define the rendering code path.

7.26.2 Member Data Documentation

7.26.2.1 Compute

`const uint Afterwarp.DeviceBehavior.Compute = 0x00000004 [static]`

Compute shader support is available.

7.26.2.2 DepthClearBadPrecision

```
const uint Afterwarp.DeviceBehavior.DepthClearBadPrecision = 0x00002000 [static]
```

Direct3D 11 / Parallels bug workaround: clear depth buffer with value lower than one.

7.26.2.3 DepthClipNegative

```
const uint Afterwarp.DeviceBehavior.DepthClipNegative = 0x00000002 [static]
```

3D clip space uses negative values for depth in range of [-1, 1] instead of [0, 1]. This typically occurs on OpenGL (ES) backend that do not support clip control functions.

7.26.2.4 ForceBufferUnbind

```
const uint Afterwarp.DeviceBehavior.ForceBufferUnbind = 0x00001000 [static]
```

OpenGL / Intel HD Graphics bug workaround: always unbind buffers before updating their contents.

7.26.2.5 PerSampleShading

```
const uint Afterwarp.DeviceBehavior.PerSampleShading = 0x00000001 [static]
```

Graphics hardware supports executing fragment shader per each MSAA sample.

7.26.2.6 PostDepthCoverage

```
const uint Afterwarp.DeviceBehavior.PostDepthCoverage = 0x00000200 [static]
```

OpenGL: ARB_post_depth_coverage, Direct3D 11: PostZCoverageEnable through NvAPI.

7.26.2.7 SignedNormIntFormatBugged

```
const uint Afterwarp.DeviceBehavior.SignedNormIntFormatBugged = 0x00004000 [static]
```

Direct3D 11 / VMware bug workaround: avoid signed normalized integer formats.

7.26.2.8 Tessellation

```
const uint Afterwarp.DeviceBehavior.Tessellation = 0x00000008 [static]
```

Tessellation shader support is available.

7.26.2.9 VariableRateRefresh

```
const uint Afterwarp.DeviceBehavior.VariableRateRefresh = 0x00000100 [static]
```

Direct3D 11: Variable Rate Refresh (VRR) display support is available.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.27 Afterwarp.DeviceClear Struct Reference

Type of surface layer that should be cleared.

Static Public Attributes

- const byte [Color](#) = 0x01
Color buffer.
- const byte [Depth](#) = 0x02
Depth buffer.
- const byte [Stencil](#) = 0x04
Stencil buffer.

7.27.1 Detailed Description

Type of surface layer that should be cleared.

7.27.2 Member Data Documentation

7.27.2.1 Color

```
const byte Afterwarp.DeviceClear.Color = 0x01 [static]
```

Color buffer.

7.27.2.2 Depth

```
const byte Afterwarp.DeviceClear.Depth = 0x02 [static]
```

Depth buffer.

7.27.2.3 Stencil

```
const byte Afterwarp.DeviceClear.Stencil = 0x04 [static]
```

Stencil buffer.

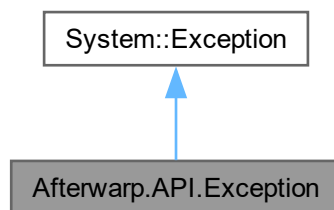
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

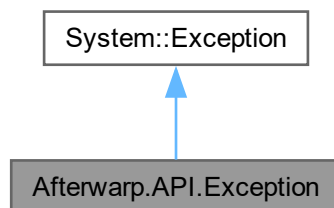
7.28 Afterwarp.API.Exception Class Reference

Exception type raised by [Afterwarp](#) classes.

Inheritance diagram for Afterwarp.API.Exception:



Collaboration diagram for Afterwarp.API.Exception:



Public Member Functions

- [Exception](#) ()
- [Exception](#) (string message)
- [Exception](#) (string message, System.Exception innerException)

7.28.1 Detailed Description

Exception type raised by [Afterwarp](#) classes.

7.28.2 Constructor & Destructor Documentation

7.28.2.1 Exception() [1/3]

```
Afterwarp.API.Exception.Exception ( ) [inline]
```

7.28.2.2 Exception() [2/3]

```
Afterwarp.API.Exception.Exception (
    string message ) [inline]
```

7.28.2.3 Exception() [3/3]

```
Afterwarp.API.Exception.Exception (
    string message,
    System::Exception innerException ) [inline]
```

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.29 Afterwarp.FloatColor Struct Reference

Public Member Functions

- [FloatColor](#) (float red, float green, float blue, float alpha=1.0f)
Creates new floating-color RGBA structure with the given values.
- [FloatColor](#) (uint color)
Creates new floating-color RGBA structure from an unsigned 32-bit color value.
- [FloatColor](#) ([FloatColorRGB](#) color, float alpha=1.0f)
Creates new floating-color RGBA structure from an unsigned 32-bit color value.

Static Public Member Functions

- static [FloatColor operator+](#) ([FloatColor](#) color1, [FloatColor](#) color2)
Returns a sum of two colors.
- static [FloatColor operator-](#) ([FloatColor](#) color1, [FloatColor](#) color2)
Subtracts one color from another.
- static [FloatColor operator*](#) ([FloatColor](#) color1, [FloatColor](#) color2)
Multiplies one color by another.
- static [FloatColor operator/](#) ([FloatColor](#) color1, [FloatColor](#) color2)
Divides one color by another.
- static [FloatColor operator*](#) ([FloatColor](#) color, float theta)
Multiplies color by a coefficient.
- static [FloatColor operator/](#) ([FloatColor](#) color, float theta)
Divides color by a coefficient.

Public Attributes

- float [Red](#)
Red value ranging from 0.0 (no intensity) to 1.0 (fully intense).
- float [Green](#)
Green value ranging from 0.0 (no intensity) to 1.0 (fully intense).
- float [Blue](#)
Blue value ranging from 0.0 (no intensity) to 1.0 (fully intense).
- float [Alpha](#)
Alpha-channel value ranging from 0.0 (translucent) to 1.0 (opaque).

Properties

- static [FloatColor Black](#) [get]
Returns a black color.
- static [FloatColor White](#) [get]
Returns a white color.
- static [FloatColor TranslucentBlack](#) [get]
Returns a translucent black color.
- static [FloatColor TranslucentWhite](#) [get]
Returns a translucent white color.
- readonly [FloatColorRGB RGB](#) [get]
Returns RGB portion of the color, discarding alpha-channel.
- readonly uint [ToColor](#) [get]
Converts floating-point color to 32-bit integer RGBA color.

7.29.1 Detailed Description

A special high-precision color value that has each individual component represented as 32-bit floating-point value in range of [0, 1]. Although components may have values outside of aforementioned range, such colors cannot be reliably displayed on the screen.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 FloatColor() [1/3]

```
Afterwarp.FloatColor.FloatColor (
    float red,
    float green,
    float blue,
    float alpha = 1::0f ) [inline]
```

Creates new floating-color RGBA structure with the given values.

7.29.2.2 FloatColor() [2/3]

```
Afterwarp.FloatColor.FloatColor (
    uint color ) [inline]
```

Creates new floating-color RGBA structure from an unsigned 32-bit color value.

7.29.2.3 FloatColor() [3/3]

```
Afterwarp.FloatColor.FloatColor (
    FloatColorRGB color,
    float alpha = 1::0f ) [inline]
```

Creates new floating-color RGBA structure from an unsigned 32-bit color value.

7.29.3 Member Function Documentation

7.29.3.1 operator*() [1/2]

```
static FloatColor Afterwarp.FloatColor.operator* (
    FloatColor color,
    float theta ) [inline], [static]
```

Multiplies color by a coefficient.

7.29.3.2 operator*() [2/2]

```
static FloatColor Afterwarp.FloatColor.operator* (
    FloatColor color1,
    FloatColor color2 ) [inline], [static]
```

Multiplies one color by another.

7.29.3.3 operator+()

```
static FloatColor Afterwarp.FloatColor.operator+ (
    FloatColor color1,
    FloatColor color2 ) [inline], [static]
```

Returns a sum of two colors.

7.29.3.4 operator-()

```
static FloatColor Afterwarp.FloatColor.operator- (
    FloatColor color1,
    FloatColor color2 ) [inline], [static]
```

Subtracts one color from another.

7.29.3.5 operator/() [1/2]

```
static FloatColor Afterwarp.FloatColor.operator/ (
    FloatColor color,
    float theta ) [inline], [static]
```

Divides color by a coefficient.

7.29.3.6 operator/() [2/2]

```
static FloatColor Afterwarp.FloatColor.operator/ (
    FloatColor color1,
    FloatColor color2 ) [inline], [static]
```

Divides one color by another.

7.29.4 Member Data Documentation

7.29.4.1 Alpha

```
float Afterwarp.FloatColor.Alpha
```

Alpha-channel value ranging from 0.0 (translucent) to 1.0 (opaque).

7.29.4.2 Blue

```
float Afterwarp.FloatColor.Blue
```

Blue value ranging from 0.0 (no intensity) to 1.0 (fully intense).

7.29.4.3 Green

```
float Afterwarp.FloatColor.Green
```

Green value ranging from 0.0 (no intensity) to 1.0 (fully intense).

7.29.4.4 Red

```
float Afterwarp.FloatColor.Red
```

Red value ranging from 0.0 (no intensity) to 1.0 (fully intense).

7.29.5 Property Documentation

7.29.5.1 Black

```
FloatColor Afterwarp.FloatColor.Black [static], [get]
```

Returns a black color.

7.29.5.2 RGB

```
readonly FloatColorRGB Afterwarp.FloatColor.RGB [get]
```

Returns RGB portion of the color, discarding alpha-channel.

7.29.5.3 ToColor

`readonly uint Afterwarp.FloatColor.ToColor [get]`

Converts floating-point color to 32-bit integer RGBA color.

7.29.5.4 TranslucentBlack

`FloatColor Afterwarp.FloatColor.TranslucentBlack [static], [get]`

Returns a translucent black color.

7.29.5.5 TranslucentWhite

`FloatColor Afterwarp.FloatColor.TranslucentWhite [static], [get]`

Returns a translucent white color.

7.29.5.6 White

`FloatColor Afterwarp.FloatColor.White [static], [get]`

Returns a white color.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.30 Afterwarp.FloatColorRGB Struct Reference

Public Member Functions

- [FloatColorRGB](#) (float red, float green, float blue)
Creates new floating-color RGB structure with the given values.
- [FloatColorRGB](#) (uint color)
Creates new floating-color RGB structure from an unsigned 24-bit color value.

Static Public Member Functions

- static [FloatColorRGB operator+](#) ([FloatColorRGB](#) color1, [FloatColorRGB](#) color2)
Returns a sum of two colors.
- static [FloatColorRGB operator-](#) ([FloatColorRGB](#) color1, [FloatColorRGB](#) color2)
Subtracts one color from another.
- static [FloatColorRGB operator*](#) ([FloatColorRGB](#) color1, [FloatColorRGB](#) color2)
Multiplies one color by another.
- static [FloatColorRGB operator/](#) ([FloatColorRGB](#) color1, [FloatColorRGB](#) color2)
Divides one color by another.
- static [FloatColorRGB operator*](#) ([FloatColorRGB](#) color, float theta)
Multiplies color by a coefficient.
- static [FloatColorRGB operator/](#) ([FloatColorRGB](#) color, float theta)
Divides color by a coefficient.

Public Attributes

- float [Red](#)
Red value ranging from 0.0 (no intensity) to 1.0 (fully intense).
- float [Green](#)
Green value ranging from 0.0 (no intensity) to 1.0 (fully intense).
- float [Blue](#)
Blue value ranging from 0.0 (no intensity) to 1.0 (fully intense).

Properties

- readonly uint [ToColor](#) [get]
Converts floating-point color to 32-bit integer RGB color (where A bits are set to zero).
- readonly bool [Defined](#) [get]
Tests whether all values of the color are not NaNs or infinities.
- readonly uint [ToColorAlpha](#) [get]
Converts floating-point color to 32-bit integer RGBA color (where A is 255).
- static [FloatColorRGB Black](#) [get]
Returns a Black color.
- static [FloatColorRGB White](#) [get]
Returns a White color.

7.30.1 Detailed Description

A three-component special high-precision color value that has each individual component represented as 32-bit floating-point value in range of [0, 1]. Although components may have values outside of aforementioned range, such colors cannot be reliably displayed on the screen.

7.30.2 Constructor & Destructor Documentation

7.30.2.1 FloatColorRGB() [1/2]

```
Afterwarp.FloatColorRGB.FloatColorRGB (  
    float red,  
    float green,  
    float blue ) [inline]
```

Creates new floating-color RGB structure with the given values.

7.30.2.2 FloatColorRGB() [2/2]

```
Afterwarp.FloatColorRGB.FloatColorRGB (  
    uint color ) [inline]
```

Creates new floating-color RGB structure from an unsigned 24-bit color value.

7.30.3 Member Function Documentation

7.30.3.1 operator*() [1/2]

```
static FloatColorRGB Afterwarp.FloatColorRGB.operator* (  
    FloatColorRGB color,  
    float theta ) [inline], [static]
```

Multiplies color by a coefficient.

7.30.3.2 operator*() [2/2]

```
static FloatColorRGB Afterwarp.FloatColorRGB.operator* (  
    FloatColorRGB color1,  
    FloatColorRGB color2 ) [inline], [static]
```

Multiplies one color by another.

7.30.3.3 operator+()

```
static FloatColorRGB Afterwarp.FloatColorRGB.operator+ (  
    FloatColorRGB color1,  
    FloatColorRGB color2 ) [inline], [static]
```

Returns a sum of two colors.

7.30.3.4 operator-()

```
static FloatColorRGB Afterwarp.FloatColorRGB.operator- (  
    FloatColorRGB color1,  
    FloatColorRGB color2 ) [inline], [static]
```

Subtracts one color from another.

7.30.3.5 operator/() [1/2]

```
static FloatColorRGB Afterwarp.FloatColorRGB.operator/ (  
    FloatColorRGB color,  
    float theta ) [inline], [static]
```

Divides color by a coefficient.

7.30.3.6 operator/() [2/2]

```
static FloatColorRGB Afterwarp.FloatColorRGB.operator/ (  
    FloatColorRGB color1,  
    FloatColorRGB color2 ) [inline], [static]
```

Divides one color by another.

7.30.4 Member Data Documentation

7.30.4.1 Blue

```
float Afterwarp.FloatColorRGB.Blue
```

Blue value ranging from 0.0 (no intensity) to 1.0 (fully intense).

7.30.4.2 Green

```
float Afterwarp.FloatColorRGB.Green
```

Green value ranging from 0.0 (no intensity) to 1.0 (fully intense).

7.30.4.3 Red

```
float Afterwarp.FloatColorRGB.Red
```

Red value ranging from 0.0 (no intensity) to 1.0 (fully intense).

7.30.5 Property Documentation

7.30.5.1 Black

```
FloatColorRGB Afterwarp.FloatColorRGB.Black [static], [get]
```

Returns a Black color.

7.30.5.2 Defined

```
readonly bool Afterwarp.FloatColorRGB.Defined [get]
```

Tests whether all values of the color are not NaNs or infinities.

7.30.5.3 ToColor

```
readonly uint Afterwarp.FloatColorRGB.ToColor [get]
```

Converts floating-point color to 32-bit integer RGB color (where A bits are set to zero).

7.30.5.4 ToColorAlpha

```
readonly uint Afterwarp.FloatColorRGB.ToColorAlpha [get]
```

Converts floating-point color to 32-bit integer RGBA color (where A is 255).

7.30.5.5 White

`FloatColorRGB Afterwarp.FloatColorRGB.White [static], [get]`

Returns a White color.

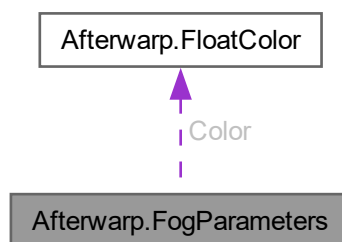
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.31 Afterwarp.FogParameters Struct Reference

Structure containing spatial fog characteristics.

Collaboration diagram for Afterwarp.FogParameters:



Public Member Functions

- [FogParameters](#) ([FloatColor](#) color, float opacity, Vector4 groundPlane, float densityGround, float distance, float bias)
Creates new fog parameters structure with the given values.

Public Attributes

- [FloatColor](#) [Color](#)
Fog color.
- float [Opacity](#)
Fog opacity.
- Vector4 [GroundPlane](#)
Equation of the plane for ground fog.
- float [DensityGround](#)
Ground fog density factor.
- float [Distance](#)
A distance (far plane) for linear fog.
- float [Bias](#)
A bias for linear fog.

Properties

- static [FogParameters Default](#) [get]
Returns default fog parameters.
- float [Scattering](#) [get, set]
An in-scattering parameter for exponential fog.
- float [Extinction](#) [get, set]
An extinction parameter for exponential fog.

7.31.1 Detailed Description

Structure containing spatial fog characteristics.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 FogParameters()

```
Afterwarp.FogParameters.FogParameters (
    FloatColor color,
    float opacity,
    Vector4 groundPlane,
    float densityGround,
    float distance,
    float bias ) [inline]
```

Creates new fog parameters structure with the given values.

7.31.3 Member Data Documentation

7.31.3.1 Bias

```
float Afterwarp.FogParameters.Bias
```

A bias for linear fog.

7.31.3.2 Color

```
FloatColor Afterwarp.FogParameters.Color
```

Fog color.

7.31.3.3 DensityGround

```
float Afterwarp.FogParameters.DensityGround
```

Ground fog density factor.

7.31.3.4 Distance

```
float Afterwarp.FogParameters.Distance
```

A distance (far plane) for linear fog.

7.31.3.5 GroundPlane

```
Vector4 Afterwarp.FogParameters.GroundPlane
```

Equation of the plane for ground fog.

7.31.3.6 Opacity

```
float Afterwarp.FogParameters.Opacity
```

Fog opacity.

7.31.4 Property Documentation

7.31.4.1 Default

```
FogParameters Afterwarp.FogParameters.Default [static], [get]
```

Returns default fog parameters.

7.31.4.2 Extinction

```
float Afterwarp.FogParameters.Extinction [get], [set]
```

An extinction parameter for exponential fog.

7.31.4.3 Scattering

```
float Afterwarp.FogParameters.Scattering [get], [set]
```

An in-scattering parameter for exponential fog.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.32 Afterwarp.FontAttribute Struct Reference

Font attribute flags that define some visual characteristics.

Static Public Attributes

- const byte [Underline](#) = 0x01
Flip region vertically.
- const byte [StrikeOut](#) = 0x02
Font with strike-out text.

7.32.1 Detailed Description

Font attribute flags that define some visual characteristics.

7.32.2 Member Data Documentation

7.32.2.1 StrikeOut

```
const byte Afterwarp.FontAttribute.StrikeOut = 0x02 [static]
```

Font with strike-out text.

7.32.2.2 Underline

```
const byte Afterwarp.FontAttribute.Underline = 0x01 [static]
```

Flip region vertically.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.33 Afterwarp.FontEffect Struct Reference

Attributes and characteristics that define how font glyphs are rasterized.

Public Member Functions

- [FontEffect](#) (float fillBrightness, float fillOpacity, [FontBorder](#) borderType, float borderThickness, float border↵Brightness, float borderOpacity, float shadowSmoothness, Vector2 shadowDistance, float shadowBrightness, float shadowOpacity, float signedFieldDistance)
Creates font effect structure with the given values.

Public Attributes

- float [FillBrightness](#)
- float [FillOpacity](#)
- [FontBorder](#) [BorderType](#)
Type of border that will appear around the text.
- float [BorderThickness](#)
Thickness of the font's border in pixels.
- float [BorderBrightness](#)
- float [BorderOpacity](#)
- float [ShadowSmoothness](#)
How strong the blur should be applied to letter's shadow (number of steps).
- [Vector2](#) [ShadowDistance](#)
Distance between letter and its shadow in pixels.
- float [ShadowBrightness](#)
- float [ShadowOpacity](#)
- float [SignedFieldDistance](#)

Properties

- static [FontEffect Default](#) `[get]`
Returns default font effect parameters.

7.33.1 Detailed Description

Attributes and characteristics that define how font glyphs are rasterized.

7.33.2 Constructor & Destructor Documentation

7.33.2.1 [FontEffect\(\)](#)

```
Afterwarp.FontEffect.FontEffect (
    float fillBrightness,
    float fillOpacity,
    FontBorder borderType,
    float borderThickness,
    float borderBrightness,
    float borderOpacity,
    float shadowSmoothness,
    Vector2 shadowDistance,
    float shadowBrightness,
    float shadowOpacity,
    float signedFieldDistance ) [inline]
```

Creates font effect structure with the given values.

7.33.3 Member Data Documentation

7.33.3.1 BorderBrightness

```
float Afterwarp.FontEffect.BorderBrightness
```

Brightness of the font's border in range of [0, 1], where zero means the border will appear completely dark and one means that the border will appear completely white.

7.33.3.2 BorderOpacity

```
float Afterwarp.FontEffect.BorderOpacity
```

Opacity of the font's border in range of [0, 1], where zero means the border will appear completely transparent (that is, not appear at all) and one means that the border will appear completely opaque.

7.33.3.3 BorderThickness

```
float Afterwarp.FontEffect.BorderThickness
```

Thickness of the font's border in pixels.

7.33.3.4 BorderType

```
FontBorder Afterwarp.FontEffect.BorderType
```

Type of border that will appear around the text.

7.33.3.5 FillBrightness

```
float Afterwarp.FontEffect.FillBrightness
```

Brightness of the letter's fill in range of [0, 1], where zero means the letter will appear completely dark and one means that letter will appear completely white.

7.33.3.6 FillOpacity

```
float Afterwarp.FontEffect.FillOpacity
```

Opacity of the letter's fill in range of [0, 1], where zero means the letter will appear completely transparent (that is, not appear at all) and one means that the letter will appear completely opaque.

7.33.3.7 ShadowBrightness

```
float Afterwarp.FontEffect.ShadowBrightness
```

Brightness of the letter's shadow in range of [0, 1], where zero means the shadow will appear completely dark and one means that the shadow will appear completely white.

7.33.3.8 ShadowDistance

```
Vector2 Afterwarp.FontEffect.ShadowDistance
```

Distance between letter and its shadow in pixels.

7.33.3.9 ShadowOpacity

```
float Afterwarp.FontEffect.ShadowOpacity
```

Opacity of the letter's shadow in range of [0, 1], where zero means the shadow will appear completely transparent (that is, not appear at all) and one means that the shadow will appear completely opaque.

7.33.3.10 ShadowSmoothness

```
float Afterwarp.FontEffect.ShadowSmoothness
```

How strong the blur should be applied to letter's shadow (number of steps).

7.33.3.11 SignedFieldDistance

```
float Afterwarp.FontEffect.SignedFieldDistance
```

Distance in pixels for generation of Signed Distance Field (SDF) font glyphs. This parameter must be greater or equal to zero. If the value is not zero, then the font is considered SDF.

7.33.4 Property Documentation

7.33.4.1 Default

```
FontEffect Afterwarp.FontEffect.Default [static], [get]
```

Returns default font effect parameters.

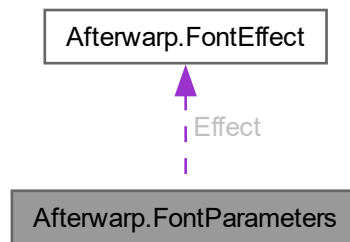
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.34 Afterwarp.FontParameters Struct Reference

Parameters that define the appearance and important characteristics of the font.

Collaboration diagram for Afterwarp.FontParameters:



Public Member Functions

- [FontParameters](#) (string family, float size, [FontWeight](#) weight=[FontWeight.Normal](#), [FontStretch](#) stretch=[FontStretch.Normal](#), [FontSlant](#) slant=[FontSlant.None](#), byte attributes=0)
Creates new font parameters structure with the given values.

Public Attributes

- byte[] [FamilyBytes](#)
Font family described as an array of UTF-8 encoded bytes.
- float [Size](#)
Size of the font.
- [FontWeight](#) [Weight](#)
Font letter weight.
- [FontStretch](#) [Stretch](#)
Font letter stretch.
- [FontSlant](#) [Slant](#)
Font letter slant.
- byte [Attributes](#)
Additional font attributes.
- [FontEffect](#) [Effect](#)
Effect parameters that contribute to font rasterization.

Properties

- string [Family](#) [get, set]
Font family (e.g. "Helvetica").

7.34.1 Detailed Description

Parameters that define the appearance and important characteristics of the font.

7.34.2 Constructor & Destructor Documentation

7.34.2.1 FontParameters()

```
Afterwarp.FontParameters.FontParameters (
    string family,
    float size,
    FontWeight weight = FontWeight::Normal,
    FontStretch stretch = FontStretch::Normal,
    FontSlant slant = FontSlant::None,
    byte attributes = 0 ) [inline]
```

Creates new font parameters structure with the given values.

7.34.3 Member Data Documentation

7.34.3.1 Attributes

```
byte Afterwarp.FontParameters.Attributes
```

Additional font attributes.

7.34.3.2 Effect

```
FontEffect Afterwarp.FontParameters.Effect
```

Effect parameters that contribute to font rasterization.

7.34.3.3 FamilyBytes

```
byte [ ] Afterwarp.FontParameters.FamilyBytes
```

Font family described as an array of UTF-8 encoded bytes.

7.34.3.4 Size

```
float Afterwarp.FontParameters.Size
```

Size of the font.

7.34.3.5 Slant

`FontSlant` Afterwarp.FontParameters.Slant

Font letter slant.

7.34.3.6 Stretch

`FontStretch` Afterwarp.FontParameters.Stretch

Font letter stretch.

7.34.3.7 Weight

`FontWeight` Afterwarp.FontParameters.Weight

Font letter weight.

7.34.4 Property Documentation

7.34.4.1 Family

```
string Afterwarp.FontParameters.Family [get], [set]
```

Font family (e.g. "Helvetica").

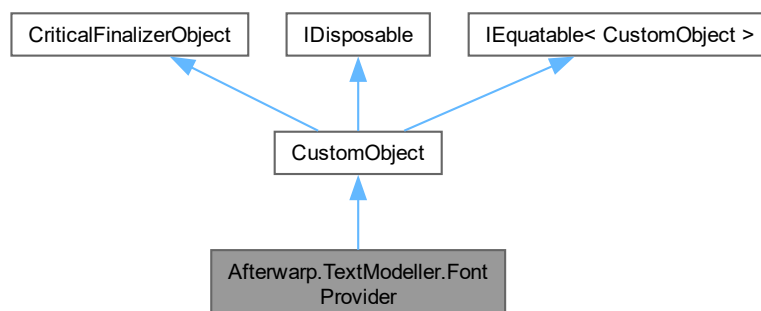
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

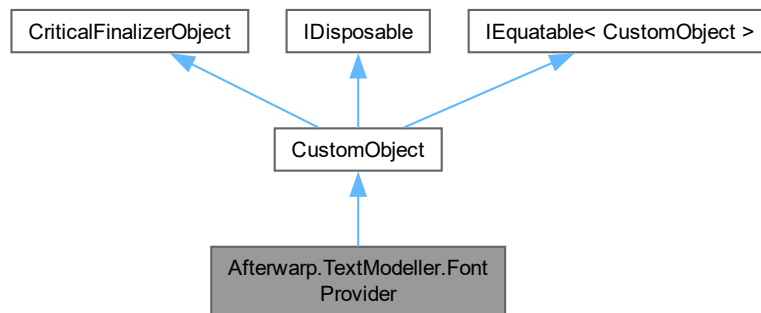
7.35 Afterwarp.TextModeller.FontProvider Class Reference

A provider of font geometry for the modeller.

Inheritance diagram for Afterwarp.TextModeller.FontProvider:



Collaboration diagram for Afterwarp.TextModeller.FontProvider:



Additional Inherited Members

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

7.35.1 Detailed Description

A provider of font geometry for the modeller.

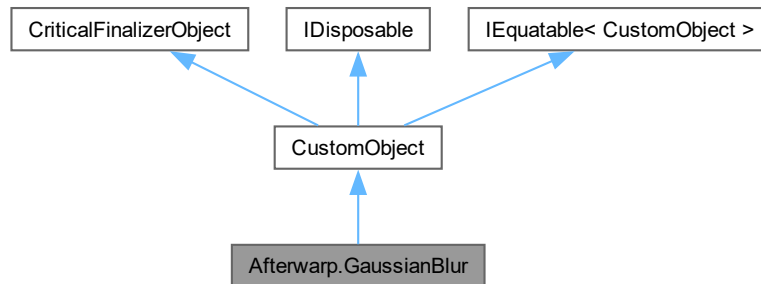
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

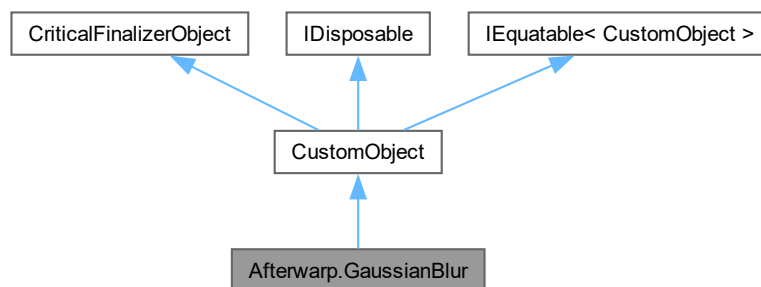
7.36 Afterwarp.GaussianBlur Class Reference

Gaussian Blur module.

Inheritance diagram for Afterwarp.GaussianBlur:



Collaboration diagram for Afterwarp.GaussianBlur:



Public Member Functions

- `GaussianBlur` (`Device` device, float sigma=3.0f, int samples=9, bool hardwareFiltering=true)
Creates new instance of Gaussian Blur module.
- void `Update` (`Texture` destination, `Texture` intermediary, `Texture` source)
- void `UpdateAt` (`Texture` destination, `Texture` intermediary, `Texture` source, `Point?` destPos=null, `Rect?` intermRect=null)

Public Member Functions inherited from `Afterwarp.CustomObject`

- void `Dispose` ()
- bool `Equals` (`CustomObject` other)
- override bool `Equals` (object obj)
- override int `GetHashCode` ()

Properties

- [Device Device](#) [get]
Returns device associated with the module.
- int [FixedSamples](#) [get]
Returns maximum number of samples supported by a fixed-sample blur, which enables higher performance.
- int [Samples](#) [get, set]
Returns or changes a number of samples in the blur kernel. The value must be odd, between 5 and 15.
- float [Sigma](#) [get, set]
- float [Chroma](#) [get, set]
- bool [HardwareFiltering](#) [get, set]
Returns or changes whether hardware filtering is used to accelerate the blur.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.36.1 Detailed Description

Gaussian Blur module.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 GaussianBlur()

```
Afterwarp.GaussianBlur.GaussianBlur (
    Device device,
    float sigma = 3::0f,
    int samples = 9,
    bool hardwareFiltering = true ) [inline]
```

Creates new instance of Gaussian Blur module.

7.36.3 Member Function Documentation

7.36.3.1 Update()

```
void Afterwarp.GaussianBlur.Update (
    Texture destination,
    Texture intermediary,
    Texture source ) [inline]
```

Applies blur through use of an an intermediate texture. Both destination and intermediary textures must have drawable attribute. If source texture has drawable attribute, then it can be passed as destination texture as well.

7.36.3.2 UpdateAt()

```
void Afterwarp.GaussianBlur.UpdateAt (
    Texture destination,
    Texture intermediary,
    Texture source,
    Point? destPos = null,
    Rect? intermRect = null ) [inline]
```

Applies a horizontal blur from source to intermediary texture, then a vertical blur from intermediary to destination texture. Both destination and intermediary textures must have drawable attribute. If source texture has drawable attribute, then it can be passed as destination texture as well. If destination position and/or intermediate rectangle are specified, then only a portion of intermediary texture will be copied to destination at the given offset.

7.36.4 Property Documentation

7.36.4.1 Chroma

```
float Afterwarp.GaussianBlur.Chroma [get], [set]
```

Returns or changes chroma offset for an optional color adjustment. Chroma offset must be in range of [0, inf), where 1 means no adjustment is to be performed.

7.36.4.2 Device

```
Device Afterwarp.GaussianBlur.Device [get]
```

Returns device associated with the module.

7.36.4.3 FixedSamples

```
int Afterwarp.GaussianBlur.FixedSamples [get]
```

Returns maximum number of samples supported by a fixed-sample blur, which enables higher performance.

7.36.4.4 HardwareFiltering

```
bool Afterwarp.GaussianBlur.HardwareFiltering [get], [set]
```

Returns or changes whether hardware filtering is used to accelerate the blur.

7.36.4.5 Samples

```
int Afterwarp.GaussianBlur.Samples [get], [set]
```

Returns or changes a number of samples in the blur kernel. The value must be odd, between 5 and 15.

7.36.4.6 Sigma

```
float Afterwarp.GaussianBlur.Sigma [get], [set]
```

Returns or changes a value of sigma, which defines the curve distribution of the blur. It must be equal or higher than zero.

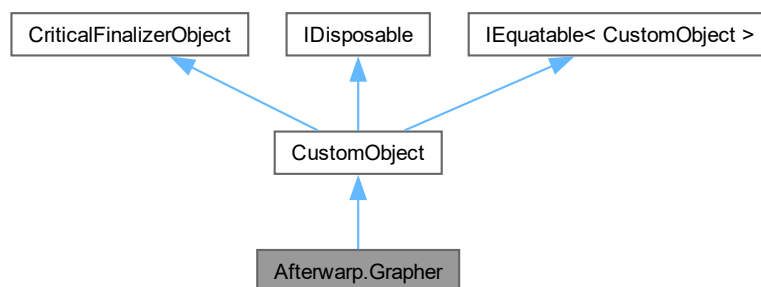
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

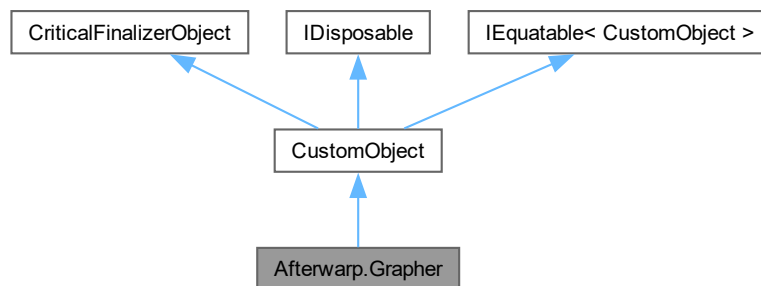
7.37 Afterwarp.Grapher Class Reference

3D graph plotting module.

Inheritance diagram for Afterwarp.Grapher:



Collaboration diagram for Afterwarp.Grapher:



Public Member Functions

- [Grapher](#) ([Device](#) device)
Creates new instance of 3D graph plotting module associated with a particular device.
- void [Begin](#) ()
Prepares grapher for rendering.
- void [End](#) ()
Finishes grapher rendering.
- void [Points](#) ([Vector4](#)[] vertices, [uint](#)[] colors, [float](#)[] angles, [int](#) elementCount, [PointShape](#) shape=[PointShape.Square](#))
- void [Lines](#) ([Vector4](#)[] vertices, [uint](#)[] colors, [int](#)[] indices, [int](#) vertexCount, [int](#) indexCount, [LineCaps](#) caps=[LineCaps.Butt](#))
Draws lines at their given vertices, thicknesses ("w" component of each vertex), colors and indices.
- void [Point](#) ([Vector3](#) position, [uint](#) color, [float](#) size, [PointShape](#) shape=[PointShape.Square](#), [float](#) angle=0.0f)
Draws a single point with the given parameters.
- void [Line](#) ([Vector3](#) position1, [Vector3](#) position2, [uint](#) color1, [uint](#) color2, [float](#) thickness1, [float](#) thickness2, [LineCaps](#) caps=[LineCaps.Butt](#))
Draws a single line with the given parameters.
- void [DottedLine](#) ([Vector3](#) position1, [Vector3](#) position2, [uint](#) color1, [uint](#) color2, [float](#) thickness1, [float](#) thickness2, [float](#) sparsity, [PointShape](#) shape=[PointShape.Square](#), [float](#) angle=0.0f)
Draws a line of dots with the given parameters.
- void [Arrow](#) ([Vector2](#) targetSize, [Vector3](#) origin, [Vector3](#) destination, [uint](#) color, [float](#) thickness, [float](#) size, [LineCaps](#) caps=[LineCaps.Butt](#))
- void [BoundingBox](#) ([Matrix4x4](#) volume, [uint](#) color, [float](#) thickness, [float](#) length=0.5f, [LineCaps](#) caps=[LineCaps.Butt](#))
Draws lines around a bounding box for the given volume matrix.
- void [Flush](#) ()
- void [Reset](#) ()

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override [int](#) [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device associated with this module.
- [Matrix4x4](#) [Transform](#) [get, set]
3D transformation matrix that combines world, view and projection matrices.
- [int](#) [BatchCount](#) [get]

Properties inherited from [Afterwarp.CustomObject](#)

- [IntPtr](#) [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- [IntPtr](#) [_handle](#)
Wrapped object's handle.

7.37.1 Detailed Description

3D graph plotting module.

7.37.2 Constructor & Destructor Documentation

7.37.2.1 Grapher()

```
Afterwarp.Grapher.Grapher (
    Device device ) [inline]
```

Creates new instance of 3D graph plotting module associated with a particular device.

7.37.3 Member Function Documentation

7.37.3.1 Arrow()

```
void Afterwarp.Grapher.Arrow (
    Vector2 targetSize,
    Vector3 origin,
    Vector3 destination,
    uint color,
    float thickness,
    float size,
    LineCaps caps = LineCaps::Butt ) [inline]
```

Draws an arrow from origin to destination and the given parameters. "targetSize" is the rendering surface size (required for proper arrow angle calculation).

7.37.3.2 Begin()

```
void Afterwarp.Grapher.Begin ( ) [inline]
```

Prepares grapher for rendering.

7.37.3.3 BoundingBox()

```
void Afterwarp.Grapher.BoundingBox (
    Matrix4x4 volume,
    uint color,
    float thickness,
    float length = 0.5f,
    LineCaps caps = LineCaps::Butt ) [inline]
```

Draws lines around a bounding box for the given volume matrix.

7.37.3.4 DottedLine()

```
void Afterwarp.Grapher.DottedLine (
    Vector3 position1,
    Vector3 position2,
    uint color1,
    uint color2,
    float thickness1,
    float thickness2,
    float sparsity,
    PointShape shape = PointShape::Square,
    float angle = 0.0f ) [inline]
```

Draws a line of dots with the given parameters.

7.37.3.5 End()

```
void Afterwarp.Grapher.End ( )
```

Finishes grapher rendering.

7.37.3.6 Flush()

```
void Afterwarp.Grapher.Flush ( )
```

Flushes grapher cache and draws any pending primitives on the rendering surface. This can be useful to make sure that nothing remains in graph cache before changing any device states.

7.37.3.7 Line()

```
void Afterwarp.Grapher.Line (
    Vector3 position1,
    Vector3 position2,
    uint color1,
    uint color2,
    float thickness1,
    float thickness2,
    LineCaps caps = LineCaps::Butt ) [inline]
```

Draws a single line with the given parameters.

7.37.3.8 Lines()

```
void Afterwarp.Grapher.Lines (
    Vector4[] vertices,
    uint[] colors,
    int[] indices,
    int vertexCount,
    int indexCount,
    LineCaps caps = LineCaps::Butt ) [inline]
```

Draws lines at their given vertices, thicknesses ("w" component of each vertex), colors and indices.

7.37.3.9 Point()

```
void Afterwarp.Grapher.Point (
    Vector3 position,
    uint color,
    float size,
    PointShape shape = PointShape::Square,
    float angle = 0::0f ) [inline]
```

Draws a single point with the given parameters.

7.37.3.10 Points()

```
void Afterwarp.Grapher.Points (
    Vector4[] vertices,
    uint[] colors,
    float[] angles,
    int elementCount,
    PointShape shape = PointShape::Square ) [inline]
```

Draws points at their given vertices with their respective sizes ("w" component of each vertex), colors, angles and the appropriate shape in 3D space.

7.37.3.11 Reset()

```
void Afterwarp.Grapher.Reset ( )
```

Resets grapher internal resources and releases any non-critical memory buffers. This can reduce overall memory consumption, when switching from one complex visual scene to another, to allow graph to "adapt" to a new scene from a fresh start.

7.37.4 Property Documentation

7.37.4.1 BatchCount

```
int Afterwarp.Grapher.BatchCount [get]
```

Returns number of batches that were sent to rendering pipeline. Drawing each individual batch involves some overhead, which is implementation-specific. If this parameter happens to be considerably high at some point, the rendering code should be revised for better grouping of plotting primitives.

7.37.4.2 Device

```
Device Afterwarp.Grapher.Device [get]
```

Returns device associated with this module.

7.37.4.3 Transform

```
Matrix4x4 Afterwarp.Grapher.Transform [get], [set]
```

3D transformation matrix that combines world, view and projection matrices.

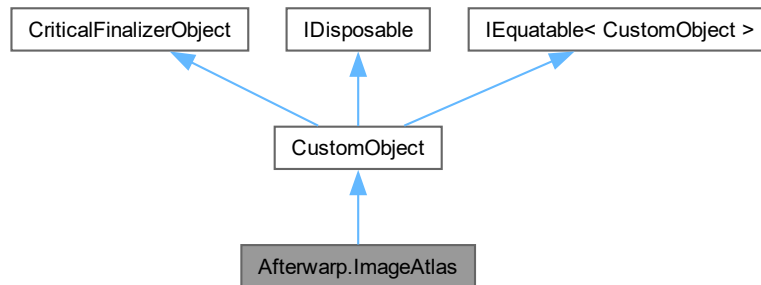
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

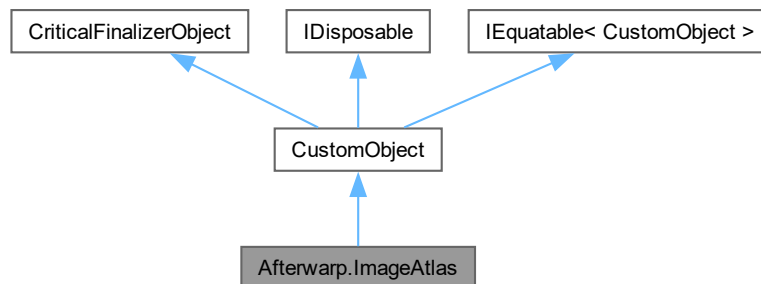
7.38 Afterwarp.ImageAtlas Class Reference

Image atlas, sub-images of which can be rendered with canvas.

Inheritance diagram for Afterwarp.ImageAtlas:



Collaboration diagram for Afterwarp.ImageAtlas:



Public Member Functions

- [ImageAtlas](#) ([Device](#) device)
Creates new instance of image atlas associated with a particular device.
- [Texture](#) [Texture](#) (int textureIndex)
Returns texture associated with a particular index or an empty object if such is not available.
- [ImageRegion](#) [Region](#) (int regionIndex)
Returns region information for the given index.
- int [CreateRegion](#) ([Rect](#) rect, int textureIndex)
Creates and adds a new region to the list.
- void [RemoveRegion](#) (int regionIndex)
Removes region with given index from the list.
- void [ClearRegions](#) ()

- *Removes all regions.*
- void [MakeRegions](#) ([Point](#) patternSize, [Point](#) visibleSize, int patternCount)
Creates list of regions based on repeated rectangular pattern dimensions.
- void [ClearTextures](#) ()
Removes all existing textures.
- void [RemoveTexture](#) (int textureIndex)
Removes texture with the given index.
- int [CreateTexture](#) ([Point](#) size, [PixelFormat](#) format=[PixelFormat.Unknown](#), uint attributes=0)
- int [PackRegion](#) ([Point](#) size, int padding=0)
- int [PackSurface](#) ([Surface](#) surface, [Rect?](#) sourceRect=null, int padding=0)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with this image atlas.
- int [TextureCount](#) [get]
Number of textures contained within the atlas.
- int [RegionCount](#) [get]
Number of regions contained within the atlas.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.38.1 Detailed Description

Image atlas, sub-images of which can be rendered with canvas.

7.38.2 Constructor & Destructor Documentation

7.38.2.1 ImageAtlas()

```
Afterwarp.ImageAtlas.ImageAtlas (
    Device device ) [inline]
```

Creates new instance of image atlas associated with a particular device.

7.38.3 Member Function Documentation

7.38.3.1 ClearRegions()

```
void Afterwarp.ImageAtlas.ClearRegions ( )
```

Removes all regions.

7.38.3.2 ClearTextures()

```
void Afterwarp.ImageAtlas.ClearTextures ( )
```

Removes all existing textures.

7.38.3.3 CreateRegion()

```
int Afterwarp.ImageAtlas.CreateRegion (
    Rect rect,
    int textureIndex ) [inline]
```

Creates and adds a new region to the list.

7.38.3.4 CreateTexture()

```
int Afterwarp.ImageAtlas.CreateTexture (
    Point size,
    PixelFormat format = PixelFormat::Unknown,
    uint attributes = 0 ) [inline]
```

Attempts to create and initialize a texture with the given parameters, which is then added to . the list. In case of failure, -1 is returned.

7.38.3.5 MakeRegions()

```
void Afterwarp.ImageAtlas.MakeRegions (
    Point patternSize,
    Point visibleSize,
    int patternCount ) [inline]
```

Creates list of regions based on repeated rectangular pattern dimensions.

7.38.3.6 PackRegion()

```
int Afterwarp.ImageAtlas.PackRegion (
    Point size,
    int padding = 0 ) [inline]
```

Finds a suitable location to accomodate the given rectangle in existing available space, adds the appropriate region to the list and returns its index. If no space is available, -1 is returned.

7.38.3.7 PackSurface()

```
int Afterwarp.ImageAtlas.PackSurface (
    Surface surface,
    Rect? sourceRect = null,
    int padding = 0 ) [inline]
```

Finds a suitable location to accomodate the given surface in existing available space, adds the appropriate region to the list and copies the contents of surface to the appropriate texture, returning the final region index. If no space is available or copying fails, -1 is returned.

7.38.3.8 Region()

```
ImageRegion Afterwarp.ImageAtlas.Region (
    int regionIndex ) [inline]
```

Returns region information for the given index.

7.38.3.9 RemoveRegion()

```
void Afterwarp.ImageAtlas.RemoveRegion (
    int regionIndex )
```

Removes region with given index from the list.

7.38.3.10 RemoveTexture()

```
void Afterwarp.ImageAtlas.RemoveTexture (
    int textureIndex )
```

Removes texture with the given index.

7.38.3.11 Texture()

```
Texture Afterwarp.ImageAtlas.Texture (
    int textureIndex )
```

Returns texture associated with a particular index or an empty object if such is not available.

7.38.4 Property Documentation

7.38.4.1 Device

`Device` `Afterwarp.ImageAtlas.Device` [get]

Returns device associated with this image atlas.

7.38.4.2 RegionCount

`int` `Afterwarp.ImageAtlas.RegionCount` [get]

Number of regions contained within the atlas.

7.38.4.3 TextureCount

`int` `Afterwarp.ImageAtlas.TextureCount` [get]

Number of textures contained within the atlas.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.39 Afterwarp.ImageRegion Struct Reference

An image region defined by its texture number and bounding rectangle on that texture.

Public Member Functions

- [ImageRegion](#) (int left, int top, int width, int height, int index)
Creates new image region structure with the given values.

Public Attributes

- ushort [Left](#)
Left position of the region.
- ushort [Top](#)
Top position of the region.
- ushort [Width](#)
Width of the region.
- ushort [Height](#)
Height of the region.
- ushort [Index](#)
Texture index associated with this region.

Static Public Attributes

- const uint [Flip](#) = 0x01u
Flip region vertically.
- const uint [Mirror](#) = 0x02u
Mirror region horizontally.
- const uint [Rotate](#) = 0x04u
Rotate region by 90 degrees clockwise.

7.39.1 Detailed Description

An image region defined by its texture number and bounding rectangle on that texture.

7.39.2 Constructor & Destructor Documentation

7.39.2.1 ImageRegion()

```
Afterwarp.ImageRegion.ImageRegion (  
    int left,  
    int top,  
    int width,  
    int height,  
    int index ) [inline]
```

Creates new image region structure with the given values.

7.39.3 Member Data Documentation

7.39.3.1 Flip

```
const uint Afterwarp.ImageRegion.Flip = 0x01u [static]
```

Flip region vertically.

7.39.3.2 Height

```
ushort Afterwarp.ImageRegion.Height
```

Height of the region.

7.39.3.3 Index

```
ushort Afterwarp.ImageRegion.Index
```

Texture index associated with this region.

7.39.3.4 Left

```
ushort Afterwarp.ImageRegion.Left
```

Left position of the region.

7.39.3.5 Mirror

```
const uint Afterwarp.ImageRegion.Mirror = 0x02u [static]
```

Mirror region horizontally.

7.39.3.6 Rotate

```
const uint Afterwarp.ImageRegion.Rotate = 0x04u [static]
```

Rotate region by 90 degrees clockwise.

7.39.3.7 Top

```
ushort Afterwarp.ImageRegion.Top
```

Top position of the region.

7.39.3.8 Width

```
ushort Afterwarp.ImageRegion.Width
```

Width of the region.

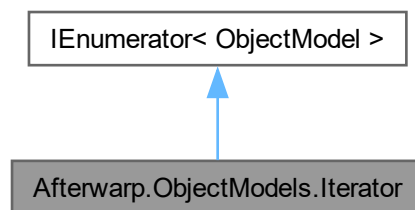
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

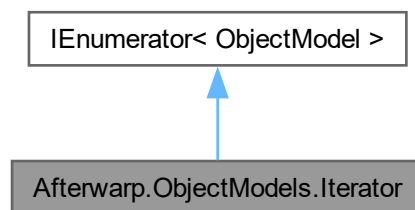
7.40 Afterwarp.ObjectModels.Iterator Class Reference

Iterator for accessing individual objects in the container.

Inheritance diagram for Afterwarp.ObjectModels.Iterator:



Collaboration diagram for Afterwarp.ObjectModels.Iterator:



Public Member Functions

- bool [MoveNext](#) ()
- void [Reset](#) ()

Properties

- [ObjectModel Current](#) [get]

7.40.1 Detailed Description

Iterator for accessing individual objects in the container.

7.40.2 Member Function Documentation

7.40.2.1 MoveNext()

```
bool Afterwarp.ObjectModels.Iterator.MoveNext ( ) [inline]
```

7.40.2.2 Reset()

```
void Afterwarp.ObjectModels.Iterator.Reset ( ) [inline]
```

7.40.3 Property Documentation

7.40.3.1 Current

```
ObjectModel Afterwarp.ObjectModels.Iterator.Current [get]
```

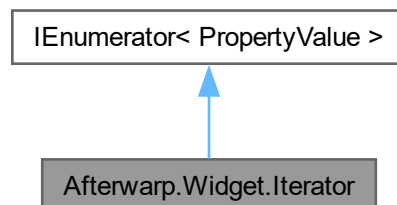
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

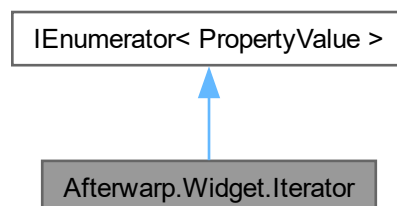
7.41 Afterwarp.Widget.Iterator Class Reference

Iterator for accessing properties in the widget.

Inheritance diagram for Afterwarp.Widget.Iterator:



Collaboration diagram for Afterwarp.Widget.Iterator:



Public Member Functions

- bool [MoveNext](#) ()
- void [Reset](#) ()

Properties

- [PropertyValue Current](#) [get]

7.41.1 Detailed Description

Iterator for accessing properties in the widget.

7.41.2 Member Function Documentation

7.41.2.1 MoveNext()

```
bool Afterwarp.Widget.Iterator.MoveNext ( ) [inline]
```

7.41.2.2 Reset()

```
void Afterwarp.Widget.Iterator.Reset ( ) [inline]
```

7.41.3 Property Documentation

7.41.3.1 Current

```
PropertyValue Afterwarp.Widget.Iterator.Current [get]
```

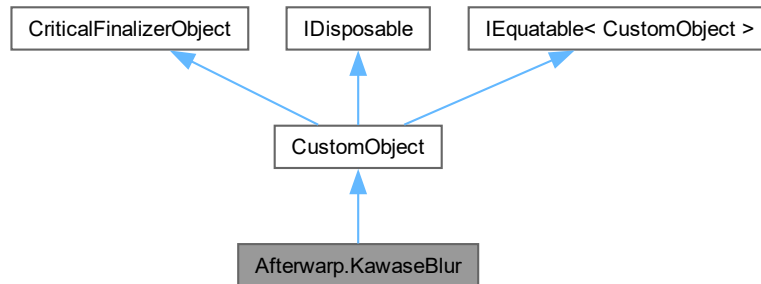
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

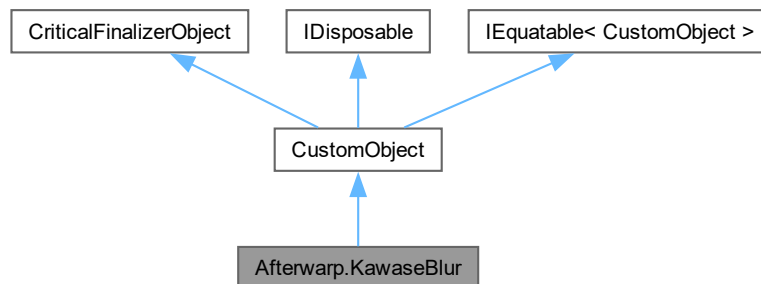
7.42 Afterwarp.KawaseBlur Class Reference

Kawase Blur module.

Inheritance diagram for Afterwarp.KawaseBlur:



Collaboration diagram for Afterwarp.KawaseBlur:



Public Member Functions

- [KawaseBlur](#) ([Device](#) device)
Creates new instance of Kawase Blur module.
- void [Update](#) ([Texture](#) destination, [Texture](#) intermediary, [Texture](#) source)
- void [SinglePass](#) ([Texture](#) destination, [Texture](#) source, [Vector2](#) offset)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the module.
- int [Passes](#) [get, set]
Returns or changes a number of blur passes (each pass includes two processing steps).
- [Vector2 Offset](#) [get, set]
Returns or changes pixel offset for each individual step.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.42.1 Detailed Description

Kawase Blur module.

7.42.2 Constructor & Destructor Documentation

7.42.2.1 KawaseBlur()

```
Afterwarp.KawaseBlur.KawaseBlur (
    Device device ) [inline]
```

Creates new instance of Kawase Blur module.

7.42.3 Member Function Documentation

7.42.3.1 SinglePass()

```
void Afterwarp.KawaseBlur.SinglePass (
    Texture destination,
    Texture source,
    Vector2 offset ) [inline]
```

Applies blur in a single pass with the given custom offset. This can be used simultaneously when performing downsampling.

7.42.3.2 Update()

```
void Afterwarp.KawaseBlur.Update (
    Texture destination,
    Texture intermediary,
    Texture source ) [inline]
```

Applies an advancing blur kernel from source to intermediary texture, then a vertical blur from intermediary to destination texture. Both destination and intermediary textures must have drawable attribute. If source texture has drawable attribute, then it can be passed as destination texture as well.

7.42.4 Property Documentation

7.42.4.1 Device

```
Device Afterwarp.KawaseBlur.Device [get]
```

Returns device associated with the module.

7.42.4.2 Offset

```
Vector2 Afterwarp.KawaseBlur.Offset [get], [set]
```

Returns or changes pixel offset for each individual step.

7.42.4.3 Passes

```
int Afterwarp.KawaseBlur.Passes [get], [set]
```

Returns or changes a number of blur passes (each pass includes two processing steps).

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.43 Afterwarp.Library Struct Reference

Shared library service functions.

Static Public Member Functions

- static bool [GetVersion](#) (out int major, out int minor, out uint build)
Returns current version of the shared library.
- static void [SerialCode](#) (string code)
Specifies a business serial key for the shared library.

7.43.1 Detailed Description

Shared library service functions.

7.43.2 Member Function Documentation

7.43.2.1 GetVersion()

```
static bool Afterwarp.Library.GetVersion (
    out int major,
    out int minor,
    out uint build ) [inline], [static]
```

Returns current version of the shared library.

7.43.2.2 SerialCode()

```
static void Afterwarp.Library.SerialCode (
    string code ) [inline], [static]
```

Specifies a business serial key for the shared library.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.44 Afterwarp.Margins Struct Reference

Margins defined by top, left, right and bottom edge paddings.

Public Member Functions

- [Margins](#) (float left, float top, float right, float bottom)
Creates margins by specifying its paddings individually.
- [Margins](#) (Vector2 topLeft, Vector2 bottomRight)
Creates margins by specifying its top/left and bottom/right corners.
- [Margins](#) (float horizontal, float vertical)
Creates margins by specifying horizontal and vertical paddings.
- [Margins](#) (float padding)
Creates margins by specifying same padding for all edges.

Public Attributes

- float [Left](#)
Left edge padding.
- float [Top](#)
Top edge padding.
- float [Right](#)
Right edge padding.
- float [Bottom](#)
Bottom edge padding.

Properties

- readonly bool [Empty](#) [get]
Indicates whether all values are zero.
- readonly float [Horizontal](#) [get]
Returns a sum of left and right paddings.
- readonly float [Vertical](#) [get]
Returns a sum of top and bottom paddings.
- Vector2 [TopLeft](#) [get, set]
Top and left edge.
- Vector2 [TopRight](#) [get, set]
Top and right edge.
- Vector2 [BottomRight](#) [get, set]
Bottom and right edge.
- Vector2 [BottomLeft](#) [get, set]
Bottom and left edge.
- static [Margins Zero](#) [get]
Returns an empty margins, where all elements are set to zero.

7.44.1 Detailed Description

Margins defined by top, left, right and bottom edge paddings.

7.44.2 Constructor & Destructor Documentation

7.44.2.1 [Margins\(\)](#) [1/4]

```
Afterwarp.Margins.Margins (  
    float left,  
    float top,  
    float right,  
    float bottom ) [inline]
```

Creates margins by specifying its paddings individually.

7.44.2.2 Margins() [2/4]

```
Afterwarp.Margins.Margins (
    Vector2 topLeft,
    Vector2 bottomRight ) [inline]
```

Creates margins by specifying its top/left and bottom/right corners.

7.44.2.3 Margins() [3/4]

```
Afterwarp.Margins.Margins (
    float horizontal,
    float vertical ) [inline]
```

Creates margins by specifying horizontal and vertical paddings.

7.44.2.4 Margins() [4/4]

```
Afterwarp.Margins.Margins (
    float padding ) [inline]
```

Creates margins by specifying same padding for all edges.

7.44.3 Member Data Documentation

7.44.3.1 Bottom

```
float Afterwarp.Margins.Bottom
```

Bottom edge padding.

7.44.3.2 Left

```
float Afterwarp.Margins.Left
```

Left edge padding.

7.44.3.3 Right

```
float Afterwarp.Margins.Right
```

Right edge padding.

7.44.3.4 Top

```
float Afterwarp.Margins.Top
```

Top edge padding.

7.44.4 Property Documentation

7.44.4.1 BottomLeft

`Vector2 Afterwarp.Margins.BottomLeft [get], [set]`

Bottom and left edge.

7.44.4.2 BottomRight

`Vector2 Afterwarp.Margins.BottomRight [get], [set]`

Bottom and right edge.

7.44.4.3 Empty

`readonly bool Afterwarp.Margins.Empty [get]`

Indicates whether all values are zero.

7.44.4.4 Horizontal

`readonly float Afterwarp.Margins.Horizontal [get]`

Returns a sum of left and right paddings.

7.44.4.5 TopLeft

`Vector2 Afterwarp.Margins.TopLeft [get], [set]`

Top and left edge.

7.44.4.6 TopRight

`Vector2 Afterwarp.Margins.TopRight [get], [set]`

Top and right edge.

7.44.4.7 Vertical

`readonly float Afterwarp.Margins.Vertical [get]`

Returns a sum of top and bottom paddings.

7.44.4.8 Zero

`Margins` `Afterwarp.Margins.Zero` `[static]`, `[get]`

Returns an empty margins, where all elements are set to zero.

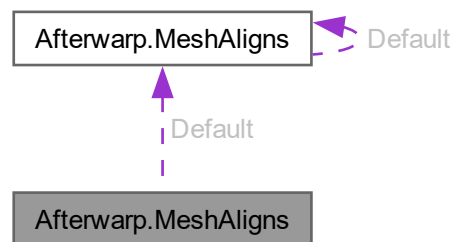
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.45 Afterwarp.MeshAligns Struct Reference

Alignment for all three axes that determine the placement of mesh around its model.

Collaboration diagram for `Afterwarp.MeshAligns`:



Public Member Functions

- [MeshAligns](#) ([MeshAlign](#) x, [MeshAlign](#) y, [MeshAlign](#) z, float bias)
Creates mesh alignment structure with the given values.

Public Attributes

- [MeshAlign X](#)
Alignment for X axis.
- [MeshAlign Y](#)
Alignment for Y axis.
- [MeshAlign Z](#)
Alignment for Z axis.
- float [Bias](#)
Bias value used for volume size adjustment.

Static Public Attributes

- static readonly [MeshAligns Default](#)
Returns default mesh alignment values.

7.45.1 Detailed Description

Alignment for all three axes that determine the placement of mesh around its model.

7.45.2 Constructor & Destructor Documentation

7.45.2.1 MeshAligns()

```
Afterwarp.MeshAligns.MeshAligns (
    MeshAlign x,
    MeshAlign y,
    MeshAlign z,
    float bias ) [inline]
```

Creates mesh alignment structure with the given values.

7.45.3 Member Data Documentation

7.45.3.1 Bias

```
float Afterwarp.MeshAligns.Bias
```

Bias value used for volume size adjustment.

7.45.3.2 Default

```
readonly MeshAligns Afterwarp.MeshAligns.Default [static]
```

Initial value:

```
= new(MeshAlign.Origin, MeshAlign.Positive,
    MeshAlign.Origin, 0.1f)
```

Returns default mesh alignment values.

7.45.3.3 X

```
MeshAlign Afterwarp.MeshAligns.X
```

Alignment for X axis.

7.45.3.4 Y

`MeshAlign` `Afterwarp.MeshAligns.Y`

Alignment for Y axis.

7.45.3.5 Z

`MeshAlign` `Afterwarp.MeshAligns.Z`

Alignment for Z axis.

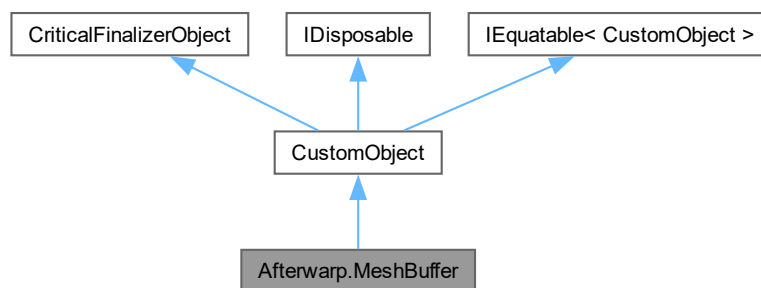
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

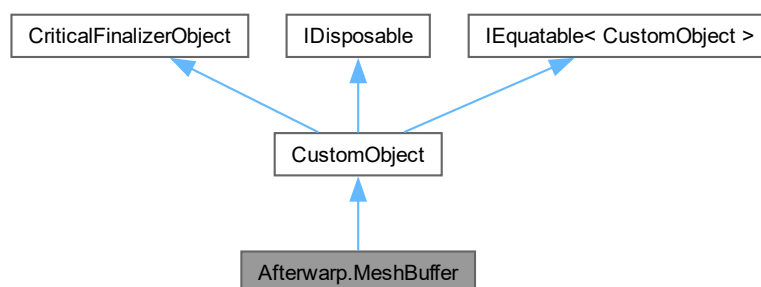
7.46 Afterwarp.MeshBuffer Class Reference

Buffer that contains 3D mesh information.

Inheritance diagram for `Afterwarp.MeshBuffer`:



Collaboration diagram for `Afterwarp.MeshBuffer`:



Classes

- struct [VertexEntry](#)

A vertex element of mesh buffer with interleaved data.

Public Member Functions

- delegate int [LoadSaveFeedback](#) (string message, int progress)
Mesh loading and saving feedback delegate.
- [MeshBuffer](#) ()
Creates new instance of 3D mesh buffer container.
- [MeshModel Model](#) ([Device](#) device, [VertexElement](#)[] vertexElements, int firstVertex=0, int vertexCount=0, int firstIndex=0, int indexCount=0, uint channel=0)
- void [SuperEllipse](#) (int longitudeSections, int latitudeSections, [Vector3](#) origin, [Vector3](#) radius, float initLongitudeAngle=0.0f, float endLongitudeAngle=(float) Math.PI *2.0f, float longitudeShape=1.0f, float initLatitudeAngle=(float) Math.PI *-0.5f, float endLatitudeAngle=(float) Math.PI *0.5f, float latitudeShape=1.0f, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, [FloatColor](#)? color=null, bool indicesClockwise=true)
- void [Geosphere](#) (int subdivisions=3, [FloatColor](#)? color=null, bool flatNormals=false, bool indicesClockwise=true)
Creates a geosphere with the given parameters.
- void [Cylinder](#) (int radialSections, int heightSections, [Vector3](#) origin, [Vector3](#) horizAxis, [Vector3](#) vertAxis, [Vector3](#) heightAxis, float initAngle=0.0f, float endAngle=(float) Math.PI *2.0f, float shape=1.0f, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, float bendAngle=0.0f, float lateralAngle=0.0f, [FloatColor](#)? color=null, bool indicesClockwise=true)
- void [Cone](#) (int sections, [Vector3](#) origin, float radius, float height, [FloatColor](#)? color=null, bool indicesClockwise=true)
Creates a simplistic cone with the given parameters in XZ plane with its base at origin.
- void [Disc](#) (int radialSections, int innerSections, [Vector3](#) origin, [Vector3](#) horizAxis, [Vector3](#) vertAxis, [Vector3](#) normal, float initAngle=0.0f, float endAngle=(float) Math.PI *2.0f, float shape=1.0f, float initRadius=0.0f, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, float lateralAngle=0.0f, [FloatColor](#)? color=null, bool indicesClockwise=true)
- void [Plane](#) (int horizSections, int vertSections, [Vector3](#) origin, [Vector3](#) horizAxis, [Vector3](#) vertAxis, [Vector3](#) normal, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, [FloatColor](#)? color=null, bool indicesClockwise=true)
Creates a 3D plane with the given parameters. Axis vectors define both orientation and size.
- void [Cube](#) (int horizSections, int vertSections, int depthSections, [Vector3](#) origin, [Vector3](#) horizAxis, [Vector3](#) vertAxis, [Vector3](#) depthAxis, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, [FloatColor](#)? color=null, bool indicesClockwise=true)
Creates a 3D cube with the given parameters. Axis vectors define both orientation and size.
- void [CubeMinimal](#) ([Vector3](#) origin, [Vector3](#) size, [FloatColor](#)? color=null, bool indicesClockwise=true)
- void [CubeRound](#) ([Vector3](#) origin, [Vector3](#) size, float roundness=0.1f, [FloatColor](#)? color=null, bool indicesClockwise=true)
Creates a 3D cube with round corners.
- void [Torus](#) (int outerSections, int innerSections, [Vector3](#) origin, [Vector3](#) horizAxis, [Vector3](#) vertAxis, [Vector2](#) innerRadius, float initInnerAngle=0.0f, float endInnerAngle=(float) Math.PI *2.0f, float innerShape=1.0f, float initOuterAngle=0.0f, float endOuterAngle=(float) Math.PI *2.0f, float outerShape=1.0f, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, float lateralAngle=0.0f, [FloatColor](#)? color=null, bool indicesClockwise=true)
- void [TorusKnot](#) (int outerSections, int innerSections, [Vector3](#) origin, int p, int q, [Vector2](#) innerRadius, float outerRadius, float initInnerAngle=0.0f, float endInnerAngle=(float) Math.PI *2.0f, float innerShape=1.0f, float initOuterAngle=0.0f, float endOuterAngle=(float) Math.PI *2.0f, [Vector2](#)? initTexCoord=null, [Vector2](#)? endTexCoord=null, float lateralAngle=0.0f, [FloatColor](#)? color=null, bool indicesClockwise=true)

- void [Supertoroid](#) (int outerSections, int innerSections, Vector3 origin, float innerRadius, float outerRadius, float initInnerAngle=0.0f, float endInnerAngle=(float) Math.PI *2.0f, float innerShape=1.0f, float initOuterAngle=0.0f, float endOuterAngle=(float) Math.PI *2.0f, float outerShape=1.0f, Vector2? initTexCoord=null, Vector2? endTexCoord=null, [FloatColor](#)? color=null, bool indicesClockwise=true)
- void [FrustumVolume](#) (Matrix4x4 viewProjectionInverse, bool depthClipNegative=false)
Creates a 3D volume of a view frustum from the given view/projection matrix.
- void [Extrusion](#) ([PathElement](#)[] path, float depth, Vector2? texCoords=null, [FloatColor](#)[] colors=null, bool quality=true)
Creates an extruded mesh around the given path. The mesh is colored with a gradient of four colors.
- void [TransformVertices](#) (int firstVertex=0, int vertexCount=0)
Multiplies vertices in a given range by currently set transformation matrix.
- void [CalculateFlatNormals](#) (float epsilon=API.Epsilon)
- void [CalculateNormals](#) (int firstVertex, int vertexCount, int firstIndex, int indexCount)
- void [CalculateNormalsWeld](#) (int firstVertex, int vertexCount, int firstIndex, int indexCount, float weldEpsilon=API.Epsilon)
- void [InvertNormals](#) (int firstVertex=0, int vertexCount=0)
- void [InvertIndexOrder](#) (int firstIndex=0, int indexCount=0)
Inverts order of indices (e.g. from clockwise to counter-clockwise and vice-versa).
- void [CalculateBounds](#) (out Vector3 vertexMin, out Vector3 vertexMax, int firstVertex=0, int vertexCount=0)
Calculates minimum and maximum vertex coordinate boundaries in the given range of mesh.
- void [Centralize](#) (int firstVertex=0, int vertexCount=0)
Calculates coordinate boundaries and centralizes the vertices in the given range of mesh.
- void [EliminateUnusedVertices](#) (int firstVertex=0, int vertexCount=0)
- void [JoinDuplicateVertices](#) (int firstVertex=0, int vertexCount=0, float threshold=API.Epsilon)
- void [Combine](#) ([MeshBuffer](#) source, int firstVertex=0, int vertexCount=0, int firstIndex=0, int indexCount=0)
Includes vertices and indices from source mesh buffer into the current one.
- void [Clear](#) ()
Removes all vertices and indices in the mesh buffer.
- bool [TryLoadFromFile](#) (string fileName, [SceneMeshMaterials](#) materials, [MeshMetaTags](#) tags, ref uint options, [LoadSaveFeedback](#) feedback, out string debug)
- void [LoadFromFile](#) (string fileName, ref uint options, [SceneMeshMaterials](#) materials=null, [MeshMetaTags](#) tags=null, [LoadSaveFeedback](#) feedback=null)
- bool [TrySaveToFile](#) (string fileName, [SceneMeshMaterials](#) materials, [MeshMetaTags](#) tags, uint options, [LoadSaveFeedback](#) feedback, out string debug)
- void [SaveToFile](#) (string fileName, [SceneMeshMaterials](#) materials=null, [MeshMetaTags](#) tags=null, uint options=0, [LoadSaveFeedback](#) feedback=null)
- bool [TryLoadFromFileEx](#) (string fileName, [MeshMetaTags](#) tags, out Vector3 minBounds, out Vector3 maxBounds, bool normals, out string debug)
- void [LoadFromFileEx](#) (string fileName, [MeshMetaTags](#) tags, out Vector3 minBounds, out Vector3 maxBounds, bool normals)
- bool [TrySaveToFileEx](#) (string fileName, uint options, out string debug)
Attempts to save the mesh to file on disk using Assimp DLL from an existing mesh buffer.
- void [SaveToFileEx](#) (string fileName, uint options)
Saves the mesh to file on disk using Assimp DLL from an existing mesh buffer.
- void [Transfer](#) ([Buffer](#) buffer, [VertexElement](#)[] vertexElements, int firstVertex, int vertexCount, uint channel=0, uint offset=0, uint semanticIndex=0)
- void [Transfer](#) ([Buffer](#) buffer, int firstVertex, int firstIndex, int indexCount, uint offset=0)
- uint [TransferEx](#) (Array vertices, [VertexElement](#)[] vertexElements, int firstVertex, int vertexCount, uint channel=0, uint semanticIndex=0)
- uint [TransferEx](#) (Array indices, uint pitch, int firstVertex, int firstIndex, int indexCount)
- [MeshVoxel Voxelize](#) (byte levels, bool colors=false)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- int [VertexCount](#) [get, set]
Total number of vertices in 3D mesh buffer.
- int [IndexCount](#) [get, set]
Total number of indices in 3D mesh buffer.
- Matrix4x4 [Transform](#) [get, set]
Transformation matrix that is applied to generated vertices during construction.
- [VertexEntry](#)[] [Vertices](#) [get, set]
Either returns or updates mesh buffer vertices.
- int[] [Indices](#) [get, set]
Either returns or updates mesh buffer indices.
- IntPtr [VerticesPtr](#) [get]
Returns pointer to unmanaged memory containing vertices stored in 3D mesh buffer.
- IntPtr [IndicesPtr](#) [get]
Returns pointer to unmanaged memory containing indices stored in the 3D mesh buffer.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.46.1 Detailed Description

Buffer that contains 3D mesh information.

7.46.2 Constructor & Destructor Documentation

7.46.2.1 MeshBuffer()

```
Afterwarp.MeshBuffer.MeshBuffer ( ) [inline]
```

Creates new instance of 3D mesh buffer container.

7.46.3 Member Function Documentation

7.46.3.1 CalculateBounds()

```
void Afterwarp.MeshBuffer.CalculateBounds (
    out Vector3 vertexMin,
    out Vector3 vertexMax,
    int firstVertex = 0,
    int vertexCount = 0 ) [inline]
```

Calculates minimum and maximum vertex coordinate boundaries in the given range of mesh.

7.46.3.2 CalculateFlatNormals()

```
void Afterwarp.MeshBuffer.CalculateFlatNormals (
    float epsilon = API::Epsilon ) [inline]
```

Recalculates mesh normals for the given range of indices by using normals of triangles. This rebuilds vertex and index lists entirely.

7.46.3.3 CalculateNormals()

```
void Afterwarp.MeshBuffer.CalculateNormals (
    int firstVertex,
    int vertexCount,
    int firstIndex,
    int indexCount ) [inline]
```

Recalculates mesh normals for the given range of indices by calculating face normals first and then averaging the resulting values to face vertices.

7.46.3.4 CalculateNormalsWeld()

```
void Afterwarp.MeshBuffer.CalculateNormalsWeld (
    int firstVertex,
    int vertexCount,
    int firstIndex,
    int indexCount,
    float weldEpsilon = API::Epsilon ) [inline]
```

Recalculates mesh normals for the given range of indices. Vertices that have their positions almost the same (depends on "weldEpsilon") will be considered duplicated and thus will receive weighted normals from neighbor faces as if it would be the same shared vertex.

7.46.3.5 Centralize()

```
void Afterwarp.MeshBuffer.Centralize (
    int firstVertex = 0,
    int vertexCount = 0 ) [inline]
```

Calculates coordinate boundaries and centralizes the vertices in the given range of mesh.

7.46.3.6 Clear()

```
void Afterwarp.MeshBuffer.Clear ( )
```

Removes all vertices and indices in the mesh buffer.

7.46.3.7 Combine()

```
void Afterwarp.MeshBuffer.Combine (
    MeshBuffer source,
    int firstVertex = 0,
    int vertexCount = 0,
    int firstIndex = 0,
    int indexCount = 0 ) [inline]
```

Includes vertices and indices from source mesh buffer into the current one.

7.46.3.8 Cone()

```
void Afterwarp.MeshBuffer.Cone (
    int sections,
    Vector3 origin,
    float radius,
    float height,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a simplistic cone with the given parameters in XZ plane with its base at origin.

7.46.3.9 Cube()

```
void Afterwarp.MeshBuffer.Cube (
    int horizSections,
    int vertSections,
    int depthSections,
    Vector3 origin,
    Vector3 horizAxis,
    Vector3 vertAxis,
    Vector3 depthAxis,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D cube with the given parameters. Axis vectors define both orientation and size.

7.46.3.10 CubeMinimal()

```
void Afterwarp.MeshBuffer.CubeMinimal (
    Vector3 origin,
    Vector3 size,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D cube with minimal number of vertices and indices. Texture coordinates, normals and tangents are not generated.

7.46.3.11 CubeRound()

```
void Afterwarp.MeshBuffer.CubeRound (
    Vector3 origin,
    Vector3 size,
    float roundness = 0::1f,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D cube with round corners.

7.46.3.12 Cylinder()

```
void Afterwarp.MeshBuffer.Cylinder (
    int radialSections,
    int heightSections,
    Vector3 origin,
    Vector3 horizAxis,
    Vector3 vertAxis,
    Vector3 heightAxis,
    float initAngle = 0::0f,
    float endAngle = (float)Math::PI * 2::0f,
    float shape = 1::0f,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    float bendAngle = 0::0f,
    float lateralAngle = 0::0f,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a cylinder with the given parameters. Axis vectors define both orientation and size. Initial and ending angles are specified in $[0, 2 * \text{PI}]$ range.

7.46.3.13 Disc()

```
void Afterwarp.MeshBuffer.Disc (
    int radialSections,
    int innerSections,
    Vector3 origin,
    Vector3 horizAxis,
    Vector3 vertAxis,
    Vector3 normal,
```

```

float initAngle = 0::0f,
float endAngle = (float)Math::PI * 2::0f,
float shape = 1::0f,
float initRadius = 0::0f,
Vector2? initTexCoord = null,
Vector2? endTexCoord = null,
float lateralAngle = 0::0f,
FloatColor? color = null,
bool indicesClockwise = true ) [inline]

```

Creates a disc with the given parameters. Axis vectors define both orientation and size. Initial and ending angles are specified in $[0, 2 * \text{PI}]$ range. `initRadius` defines initial radius in proportion to overall disc size.

7.46.3.14 EliminateUnusedVertices()

```

void Afterwarp.MeshBuffer.EliminateUnusedVertices (
    int firstVertex = 0,
    int vertexCount = 0 ) [inline]

```

Processes given range of indices, marking vertices that are used and then eliminating vertices that were left unused, updating the entire array of indices.

7.46.3.15 Extrusion()

```

void Afterwarp.MeshBuffer.Extrusion (
    PathElement[] path,
    float depth,
    Vector2? texCoords = null,
    FloatColor[] colors = null,
    bool quality = true ) [inline]

```

Creates an extruded mesh around the given path. The mesh is colored with a gradient of four colors.

7.46.3.16 FrustumVolume()

```

void Afterwarp.MeshBuffer.FrustumVolume (
    Matrix4x4 viewProjectionInverse,
    bool depthClipNegative = false ) [inline]

```

Creates a 3D volume of a view frustum from the given view/projection matrix.

7.46.3.17 Geosphere()

```

void Afterwarp.MeshBuffer.Geosphere (
    int subdivisions = 3,
    FloatColor? color = null,
    bool flatNormals = false,
    bool indicesClockwise = true ) [inline]

```

Creates a geosphere with the given parameters.

7.46.3.18 InvertIndexOrder()

```
void Afterwarp.MeshBuffer.InvertIndexOrder (
    int firstIndex = 0,
    int indexCount = 0 ) [inline]
```

Inverts order of indices (e.g. from clockwise to counter-clockwise and vice-versa).

7.46.3.19 InvertNormals()

```
void Afterwarp.MeshBuffer.InvertNormals (
    int firstVertex = 0,
    int vertexCount = 0 ) [inline]
```

Invert the normals for vertices in the given range. If vertex count is not specified, then remaining vertices starting at the given vertex and up to the end of the list are processed.

7.46.3.20 JoinDuplicateVertices()

```
void Afterwarp.MeshBuffer.JoinDuplicateVertices (
    int firstVertex = 0,
    int vertexCount = 0,
    float treshold = API::Epsilon ) [inline]
```

Joins vertices that have duplicate positions, averaging their normals, tangents and colors. This has complexity of $O(n^2)$, where n is number of vertices to process.

7.46.3.21 LoadFromFile()

```
void Afterwarp.MeshBuffer.LoadFromFile (
    string fileName,
    ref uint options,
    SceneMeshMaterials materials = null,
    MeshMetaTags tags = null,
    LoadSaveFeedback feedback = null ) [inline]
```

Loads natively Wavefront OBJ mesh file format (or a proprietary mesh binary format) from disk to the mesh buffer. Optionally loads materials included with the mesh file and/or meta tags (important boundaries of sub-meshes). Invokes "feedback" delegate regularly to inform on loading progres.

7.46.3.22 LoadFromFileEx()

```
void Afterwarp.MeshBuffer.LoadFromFileEx (
    string fileName,
    MeshMetaTags tags,
    out Vector3 minBounds,
    out Vector3 maxBounds,
    bool normals ) [inline]
```

Loads the mesh file from disk using Assimp DLL to mesh buffer, calculates its boundaries and provides debug information. Optionally loads mesh meta tags (important boundaries of sub-meshes). If "normals" parameter is true, then normals will be optionally generated when such are not present.

7.46.3.23 LoadSaveFeedback()

```
delegate int Afterwarp.MeshBuffer.LoadSaveFeedback (
    string message,
    int progress )
```

Mesh loading and saving feedback delegate.

7.46.3.24 Model()

```
MeshModel Afterwarp.MeshBuffer.Model (
    Device device,
    VertexElement[] vertexElements,
    int firstVertex = 0,
    int vertexCount = 0,
    int firstIndex = 0,
    int indexCount = 0,
    uint channel = 0 ) [inline]
```

Creates new instance of 3D mesh model with the given vertex elements and actual data from this buffer.

7.46.3.25 Plane()

```
void Afterwarp.MeshBuffer.Plane (
    int horizSections,
    int vertSections,
    Vector3 origin,
    Vector3 horizAxis,
    Vector3 vertAxis,
    Vector3 normal,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D plane with the given parameters. Axis vectors define both orientation and size.

7.46.3.26 SaveToFile()

```
void Afterwarp.MeshBuffer.SaveToFile (
    string fileName,
    SceneMeshMaterials materials = null,
    MeshMetaTags tags = null,
    uint options = 0,
    LoadSaveFeedback feedback = null ) [inline]
```

Saves mesh buffer contents to a Wavefront OBJ mesh format natively to file on disk. Materials and tags are optional and will also be saved, if provided. If saving fails, optionally provides debug information.

7.46.3.27 SaveToFileEx()

```
void Afterwarp.MeshBuffer.SaveToFileEx (
    string fileName,
    uint options ) [inline]
```

Saves the mesh to file on disk using Assimp DLL from an existing mesh buffer.

7.46.3.28 SuperEllipse()

```
void Afterwarp.MeshBuffer.SuperEllipse (
    int longitudeSections,
    int latitudeSections,
    Vector3 origin,
    Vector3 radius,
    float initLongitudeAngle = 0::0f,
    float endLongitudeAngle = (float)Math::PI * 2::0f,
    float longitudeShape = 1::0f,
    float initLatitudeAngle = (float)Math::PI * -0::5f,
    float endLatitudeAngle = (float)Math::PI * 0::5f,
    float latitudeShape = 1::0f,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D superellipse with the given parameters. Longitude's initial and ending angles are specified in $[0, 2 * \text{PI}]$ ranges, whereas latitude's initial and ending angles are specified in a different range of $[-\text{PI} / 2, \text{PI} / 2]$.

7.46.3.29 Supertoroid()

```
void Afterwarp.MeshBuffer.Supertoroid (
    int outerSections,
    int innerSections,
    Vector3 origin,
    float innerRadius,
    float outerRadius,
    float initInnerAngle = 0::0f,
    float endInnerAngle = (float)Math::PI * 2::0f,
    float innerShape = 1::0f,
    float initOuterAngle = 0::0f,
    float endOuterAngle = (float)Math::PI * 2::0f,
    float outerShape = 1::0f,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D supertoroid located in XY plane with the given parameters. Initial and ending angles for both inner and outer rings are specified in $[0, 2 * \text{PI}]$ range.

7.46.3.30 Torus()

```
void Afterwarp.MeshBuffer.Torus (
    int outerSections,
    int innerSections,
    Vector3 origin,
    Vector3 horizAxis,
    Vector3 vertAxis,
    Vector2 innerRadius,
    float initInnerAngle = 0::0f,
    float endInnerAngle = (float)Math::PI * 2::0f,
    float innerShape = 1::0f,
    float initOuterAngle = 0::0f,
    float endOuterAngle = (float)Math::PI * 2::0f,
    float outerShape = 1::0f,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    float lateralAngle = 0::0f,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D torus with the given parameters. Axis vectors define both orientation and size. Initial and ending angles for both inner and outer rings are specified in $[0, 2 * \text{PI}]$ range.

7.46.3.31 TorusKnot()

```
void Afterwarp.MeshBuffer.TorusKnot (
    int outerSections,
    int innerSections,
    Vector3 origin,
    int p,
    int q,
    Vector2 innerRadius,
    float outerRadius,
    float initInnerAngle = 0::0f,
    float endInnerAngle = (float)Math::PI * 2::0f,
    float innerShape = 1::0f,
    float initOuterAngle = 0::0f,
    float endOuterAngle = (float)Math::PI * 2::0f,
    Vector2? initTexCoord = null,
    Vector2? endTexCoord = null,
    float lateralAngle = 0::0f,
    FloatColor? color = null,
    bool indicesClockwise = true ) [inline]
```

Creates a 3D torus with the given parameters. Axis vectors define both orientation and size. Initial and ending angles for both inner and outer rings are specified in $[0, 2 * \text{PI}]$ range.

7.46.3.32 Transfer() [1/2]

```
void Afterwarp.MeshBuffer.Transfer (
    Buffer buffer,
    int firstVertex,
    int firstIndex,
    int indexCount,
    uint offset = 0 ) [inline]
```

Transfers mesh index elements to an index buffer object with the given format at the specified offset.

7.46.3.33 Transfer() [2/2]

```
void Afterwarp.MeshBuffer.Transfer (
    Buffer buffer,
    VertexElement[] vertexElements,
    int firstVertex,
    int vertexCount,
    uint channel = 0,
    uint offset = 0,
    uint semanticIndex = 0 ) [inline]
```

Transfers mesh vertex elements to an interleaved vertex buffer object with the given format at the specified channel and offset.

7.46.3.34 TransferEx() [1/2]

```
uint Afterwarp.MeshBuffer.TransferEx (
    Array indices,
    uint pitch,
    int firstVertex,
    int firstIndex,
    int indexCount ) [inline]
```

Transfers mesh indices to an array. Returns number of bytes that the resulting data occupies. If "indices" is zero, this function only calculates how many bytes the data will occupy.

7.46.3.35 TransferEx() [2/2]

```
uint Afterwarp.MeshBuffer.TransferEx (
    Array vertices,
    VertexElement[] vertexElements,
    int firstVertex,
    int vertexCount,
    uint channel = 0,
    uint semanticIndex = 0 ) [inline]
```

Transfers mesh vertex elements to an interleaved array, with the given format at the specified channel. Returns number of bytes that the resulting data occupies. If "vertices" is null, this function only calculates how many bytes the data will occupy.

7.46.3.36 TransformVertices()

```
void Afterwarp.MeshBuffer.TransformVertices (
    int firstVertex = 0,
    int vertexCount = 0 ) [inline]
```

Multiplies vertices in a given range by currently set transformation matrix.

7.46.3.37 TryLoadFromFile()

```
bool Afterwarp.MeshBuffer.TryLoadFromFile (
    string fileName,
    SceneMeshMaterials materials,
    MeshMetaTags tags,
    ref uint options,
    LoadSaveFeedback feedback,
    out string debug ) [inline]
```

Loads natively Wavefront OBJ mesh file format (or a proprietary mesh binary format) from disk to the mesh buffer. Optionally loads materials included with the mesh file and/or meta tags (important boundaries of sub-meshes). Invokes "feedback" delegate regularly to inform on loading progres. If loading fails, optionally provides debug information.

7.46.3.38 TryLoadFromFileEx()

```
bool Afterwarp.MeshBuffer.TryLoadFromFileEx (
    string fileName,
    MeshMetaTags tags,
    out Vector3 minBounds,
    out Vector3 maxBounds,
    bool normals,
    out string debug ) [inline]
```

Attempts to load the mesh file from disk using Assimp DLL to mesh buffer, calculates its boundaries and provides debug information. Optionally loads mesh meta tags (important boundaries of sub-meshes). If "normals" parameter is true, then normals will be optionally generated when such are not present.

7.46.3.39 TrySaveToFile()

```
bool Afterwarp.MeshBuffer.TrySaveToFile (
    string fileName,
    SceneMeshMaterials materials,
    MeshMetaTags tags,
    uint options,
    LoadSaveFeedback feedback,
    out string debug ) [inline]
```

Saves mesh buffer contents to a Wavefront OBJ mesh format natively to file on disk. Materials and tags are optional and will also be saved, if provided. If saving fails, optionally provides debug information.

7.46.3.40 TrySaveToFileEx()

```
bool Afterwarp.MeshBuffer.TrySaveToFileEx (
    string fileName,
    uint options,
    out string debug ) [inline]
```

Attempts to save the mesh to file on disk using Assimp DLL from an existing mesh buffer.

7.46.3.41 Voxelize()

```
MeshVoxel Afterwarp.MeshBuffer.Voxelize (
    byte levels,
    bool colors = false ) [inline]
```

Creates and builds a 3D voxel representation for geometry in the mesh buffer. A special care must be taken to define levels, as using high values may exhaust available RAM resources; typical values should be between 3 and 4.

7.46.4 Property Documentation

7.46.4.1 IndexCount

```
int Afterwarp.MeshBuffer.IndexCount [get], [set]
```

Total number of indices in 3D mesh buffer.

7.46.4.2 Indices

```
int [] Afterwarp.MeshBuffer.Indices [get], [set]
```

Either returns or updates mesh buffer indices.

7.46.4.3 IndicesPtr

```
IntPtr Afterwarp.MeshBuffer.IndicesPtr [get]
```

Returns pointer to unmanaged memory containing indices stored in the 3D mesh buffer.

7.46.4.4 Transform

```
Matrix4x4 Afterwarp.MeshBuffer.Transform [get], [set]
```

Transformation matrix that is applied to generated vertices during construction.

7.46.4.5 VertexCount

```
int Afterwarp.MeshBuffer.VertexCount [get], [set]
```

Total number of vertices in 3D mesh buffer.

7.46.4.6 Vertices

```
VertexEntry [] Afterwarp.MeshBuffer.Vertices [get], [set]
```

Either returns or updates mesh buffer vertices.

7.46.4.7 VerticesPtr

```
IntPtr Afterwarp.MeshBuffer.VerticesPtr [get]
```

Returns pointer to unmanaged memory containing vertices stored in 3D mesh buffer.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.47 Afterwarp.MeshLoadingOption Struct Reference

Non-exclusive options passed and/or returned from native mesh loader.

Static Public Attributes

- const uint [MaterialVertexColor](#) = 0x01
Material diffuse colors should be mapped to vertex colors.
- const uint [MaterialVertexColorPreferred](#) = 0x02
- const uint [MaterialTextureMissing](#) = 0x80
One or more material texture could not be loaded.
- const uint [ExportNormals](#) = 0x100
Include vertex normals in the exported file.
- const uint [ExportTangents](#) = 0x200
Include vertex tangents in the exported file.
- const uint [ExportTexCoords](#) = 0x400
Include vertex texture coordinates in the exported file.
- const uint [ExportColors](#) = 0x800
Include vertex colors in the exported file.
- const uint [ExportWeights](#) = 0x1000
Include vertex weights in the exported file.
- const uint [StripGeometryNames](#) = 0x2000
Strip names for geometry and object nodes when saving in OBJ format.

7.47.1 Detailed Description

Non-exclusive options passed and/or returned from native mesh loader.

7.47.2 Member Data Documentation

7.47.2.1 ExportColors

```
const uint Afterwarp.MeshLoadingOption.ExportColors = 0x800 [static]
```

Include vertex colors in the exported file.

7.47.2.2 ExportNormals

```
const uint Afterwarp.MeshLoadingOption.ExportNormals = 0x100 [static]
```

Include vertex normals in the exported file.

7.47.2.3 ExportTangents

```
const uint Afterwarp.MeshLoadingOption.ExportTangents = 0x200 [static]
```

Include vertex tangents in the exported file.

7.47.2.4 ExportTexCoords

```
const uint Afterwarp.MeshLoadingOption.ExportTexCoords = 0x400 [static]
```

Include vertex texture coordinates in the exported file.

7.47.2.5 ExportWeights

```
const uint Afterwarp.MeshLoadingOption.ExportWeights = 0x1000 [static]
```

Include vertex weights in the exported file.

7.47.2.6 MaterialTextureMissing

```
const uint Afterwarp.MeshLoadingOption.MaterialTextureMissing = 0x80 [static]
```

One or more material texture could not be loaded.

7.47.2.7 MaterialVertexColor

```
const uint Afterwarp.MeshLoadingOption.MaterialVertexColor = 0x01 [static]
```

Material diffuse colors should be mapped to vertex colors.

7.47.2.8 MaterialVertexColorPreferred

```
const uint Afterwarp.MeshLoadingOption.MaterialVertexColorPreferred = 0x02 [static]
```

Assume "MaterialVertexColor" option only when the model would reasonably look the same as if it would have been rendered with its materials. This requires that all model materials do not use texturing, have no emissive color, no lighting technique, no bloom factor specified or have these at their default values.

7.47.2.9 StripGeometryNames

```
const uint Afterwarp.MeshLoadingOption.StripGeometryNames = 0x2000 [static]
```

Strip names for geometry and object nodes when saving in OBJ format.

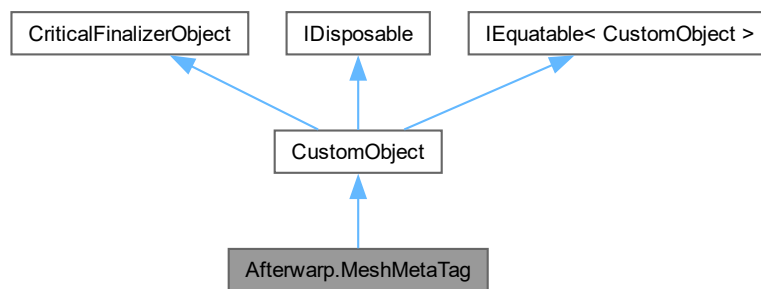
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

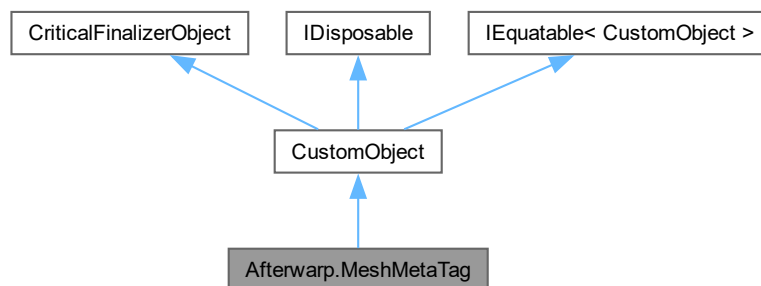
7.48 Afterwarp.MeshMetaTag Class Reference

Meta-tag, which describes a certain important axis-aligned bounding area inside a mesh.

Inheritance diagram for Afterwarp.MeshMetaTag:



Collaboration diagram for Afterwarp.MeshMetaTag:



Public Member Functions

- [MeshMetaTagPortion PortionGet](#) (int index)
Returns a portion that corresponds to the tag with a given index.
- void [PortionAdd](#) ([MeshMetaTagPortion](#) portion)
Adds a new portion to the tag.
- void [PortionErase](#) (int index)
Erases a portion with the given index.
- void [PortionsClear](#) ()
Removes all existing portions.
- void [PortionsCopy](#) ([MeshMetaTag](#) tag)
Removes any existing portions and then copies them from an existing tag.
- void [GetBounds](#) (out Vector3 boundsMin, out Vector3 boundsMax)
Calculates and returns boundaries for all existing tag's portions, if such exist.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [MeshMetaTags Parent](#) [get]
Returns mesh meta tag's parent container.
- string [Name](#) [get]
Returns name of the tag.
- byte [Type](#) [get]
Returns type of the tag.
- int [PortionCount](#) [get]
Returns number of existing portions associated with the tag.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members**Protected Member Functions inherited from [Afterwarp.CustomObject](#)**

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.48.1 Detailed Description

Meta-tag, which describes a certain important axis-aligned bounding area inside a mesh.

7.48.2 Member Function Documentation

7.48.2.1 GetBounds()

```
void Afterwarp.MeshMetaTag.GetBounds (
    out Vector3 boundsMin,
    out Vector3 boundsMax ) [inline]
```

Calculates and returns boundaries for all existing tag's portions, if such exist.

7.48.2.2 PortionAdd()

```
void Afterwarp.MeshMetaTag.PortionAdd (
    MeshMetaTagPortion portion ) [inline]
```

Adds a new portion to the tag.

7.48.2.3 PortionErase()

```
void Afterwarp.MeshMetaTag.PortionErase (
    int index )
```

Erases a portion with the given index.

7.48.2.4 PortionGet()

```
MeshMetaTagPortion Afterwarp.MeshMetaTag.PortionGet (
    int index ) [inline]
```

Returns a portion that corresponds to the tag with a given index.

7.48.2.5 PortionsClear()

```
void Afterwarp.MeshMetaTag.PortionsClear ( )
```

Removes all existing portions.

7.48.2.6 PortionsCopy()

```
void Afterwarp.MeshMetaTag.PortionsCopy (
    MeshMetaTag tag ) [inline]
```

Removes any existing portions and then copies them from an existing tag.

7.48.3 Property Documentation

7.48.3.1 Name

```
string Afterwarp.MeshMetaTag.Name [get]
```

Returns name of the tag.

7.48.3.2 Parent

```
MeshMetaTags Afterwarp.MeshMetaTag.Parent [get]
```

Returns mesh meta tag's parent container.

7.48.3.3 PortionCount

```
int Afterwarp.MeshMetaTag.PortionCount [get]
```

Returns number of existing portions associated with the tag.

7.48.3.4 Type

```
byte Afterwarp.MeshMetaTag.Type [get]
```

Returns type of the tag.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.49 Afterwarp.MeshMetaTagPortion Struct Reference

A portion of the mesh corresponding to a particular tag.

Public Member Functions

- [MeshMetaTagPortion](#) (int firstIndex, int indexCount, int firstVertex, int vertexCount, Vector3 boundsMin, Vector3 boundsMax)

Creates a new portion of mesh meta-tag.

Public Attributes

- int [FirstIndex](#)
Starting index of the tag.
- int [IndexCount](#)
Number of indices of the tag.
- int [FirstVertex](#)
Starting vertex of the tag.
- int [VertexCount](#)
Number of vertices of the tag.
- Vector3 [BoundsMin](#)
Tag's minimum boundaries.
- Vector3 [BoundsMax](#)
Tag's maximum boundaries.

7.49.1 Detailed Description

A portion of the mesh corresponding to a particular tag.

7.49.2 Constructor & Destructor Documentation

7.49.2.1 MeshMetaTagPortion()

```
Afterwarp.MeshMetaTagPortion.MeshMetaTagPortion (
    int firstIndex,
    int indexCount,
    int firstVertex,
    int vertexCount,
    Vector3 boundsMin,
    Vector3 boundsMax ) [inline]
```

Creates a new portion of mesh meta-tag.

7.49.3 Member Data Documentation

7.49.3.1 BoundsMax

```
Vector3 Afterwarp.MeshMetaTagPortion.BoundsMax
```

Tag's maximum boundaries.

7.49.3.2 BoundsMin

```
Vector3 Afterwarp.MeshMetaTagPortion.BoundsMin
```

Tag's minimum boundaries.

7.49.3.3 FirstIndex

```
int Afterwarp.MeshMetaTagPortion.FirstIndex
```

Starting index of the tag.

7.49.3.4 FirstVertex

```
int Afterwarp.MeshMetaTagPortion.FirstVertex
```

Starting vertex of the tag.

7.49.3.5 IndexCount

```
int Afterwarp.MeshMetaTagPortion.IndexCount
```

Number of indices of the tag.

7.49.3.6 VertexCount

```
int Afterwarp.MeshMetaTagPortion.VertexCount
```

Number of vertices of the tag.

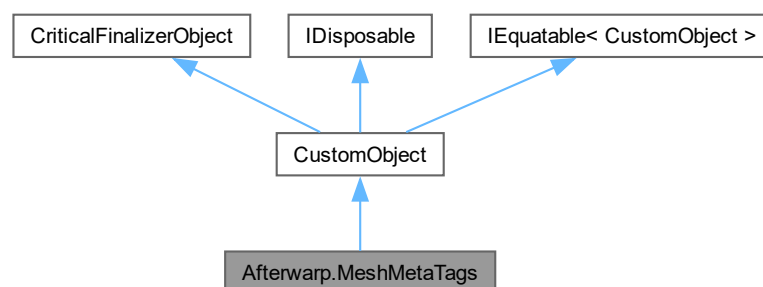
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

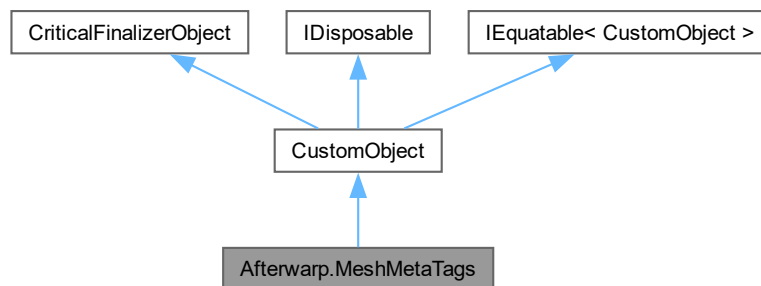
7.50 Afterwarp.MeshMetaTags Class Reference

List of meta-tags that describe important axis-aligned bounding areas inside a mesh.

Inheritance diagram for Afterwarp.MeshMetaTags:



Collaboration diagram for Afterwarp.MeshMetaTags:



Public Member Functions

- [MeshMetaTags](#) ()
Creates new instance of mesh meta-tags container.
- [MeshMetaTag Tag](#) (int index)
Returns a particular tag with the given index.
- [MeshMetaTag Tag](#) (string name, byte type=0)
Returns a particular tag with the given name (case-insensitive) and type.
- [MeshMetaTag Add](#) (string name, byte type=0)
Creates a new tag with the given name (case-insensitive) and type.
- void [Copy](#) ([MeshMetaTags](#) other)
Copies tags from another container.
- void [TakeAway](#) ([MeshMetaTags](#) other)
Takes away the internal contents from another tags container without doing any copying.
- void [Erase](#) (int index)
Removes an existing tag from container list.
- void [Clear](#) ()
Removes all existing tags from container list.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- int [Count](#) [get]
Returns number of existing tags.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.50.1 Detailed Description

List of meta-tags that describe important axis-aligned bounding areas inside a mesh.

7.50.2 Constructor & Destructor Documentation

7.50.2.1 MeshMetaTags()

```
Afterwarp.MeshMetaTags.MeshMetaTags ( ) [inline]
```

Creates new instance of mesh meta-tags container.

7.50.3 Member Function Documentation

7.50.3.1 Add()

```
MeshMetaTag Afterwarp.MeshMetaTags.Add (
    string name,
    byte type = 0 ) [inline]
```

Creates a new tag with the given name (case-insensitive) and type.

7.50.3.2 Clear()

```
void Afterwarp.MeshMetaTags.Clear ( )
```

Removes all existing tags from container list.

7.50.3.3 Copy()

```
void Afterwarp.MeshMetaTags.Copy (
    MeshMetaTags other ) [inline]
```

Copies tags from another container.

7.50.3.4 Erase()

```
void Afterwarp.MeshMetaTags.Erase (
    int index )
```

Removes an existing tag from container list.

7.50.3.5 Tag() [1/2]

```
MeshMetaTag Afterwarp.MeshMetaTags.Tag (
    int index ) [inline]
```

Returns a particular tag with the given index.

7.50.3.6 Tag() [2/2]

```
MeshMetaTag Afterwarp.MeshMetaTags.Tag (
    string name,
    byte type = 0 ) [inline]
```

Returns a particular tag with the given name (case-insensitive) and type.

7.50.3.7 TakeAway()

```
void Afterwarp.MeshMetaTags.TakeAway (
    MeshMetaTags other ) [inline]
```

Takes away the internal contents from another tags container without doing any copying.

7.50.4 Property Documentation

7.50.4.1 Count

```
int Afterwarp.MeshMetaTags.Count [get]
```

Returns number of existing tags.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.51 Afterwarp.MeshMetaTagType Struct Reference

Type of the mesh meta-tag.

Static Public Attributes

- const byte [Indeterminate](#) = 0
Tag type is not determined.
- const byte [Geometry](#) = 1
Tag represents a geometry group.
- const byte [Object](#) = 2
Tag represents an object group.

7.51.1 Detailed Description

Type of the mesh meta-tag.

7.51.2 Member Data Documentation

7.51.2.1 Geometry

```
const byte Afterwarp.MeshMetaTagType.Geometry = 1 [static]
```

Tag represents a geometry group.

7.51.2.2 Indeterminate

```
const byte Afterwarp.MeshMetaTagType.Indeterminate = 0 [static]
```

Tag type is not determined.

7.51.2.3 Object

```
const byte Afterwarp.MeshMetaTagType.Object = 2 [static]
```

Tag represents an object group.

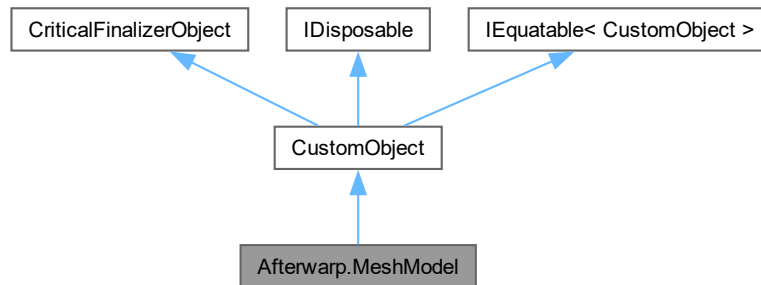
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

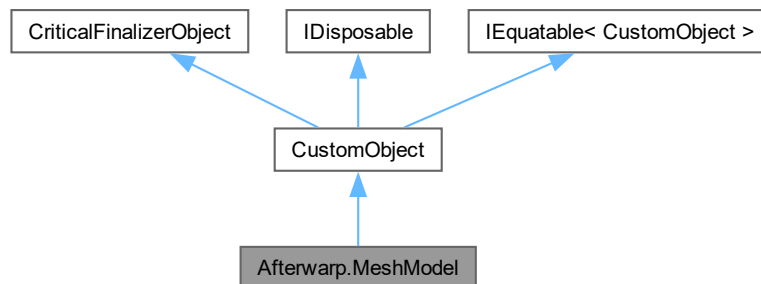
7.52 Afterwarp.MeshModel Class Reference

3D mesh model that contains both vertex and index buffers.

Inheritance diagram for Afterwarp.MeshModel:



Collaboration diagram for Afterwarp.MeshModel:



Public Member Functions

- `MeshModel` (`Device` device, int vertexCount, int vertexElementPitch, int indexCount, uint channel=0, Array initialVertexData=null, Array initialIndexData=null)
- void `Dismantle` ()
- void `TakeAway` (`MeshModel` other)

Takes away the internal contents from another 3D mesh model container without doing any copying.
- void `Draw` (`Program` program, `PrimitiveTopology` topology=`PrimitiveTopology.Triangles`, int elementCount=0, int firstIndex=0, int baseVertex=0, bool postUnbind=true)

Renders the mesh model with the given program.
- void `DrawInstances` (`Program` program, int instanceCount, `PrimitiveTopology` topology=`PrimitiveTopology.Triangles`, int elementCount=0, int firstIndex=0, int baseVertex=0, bool postUnbind=true)

Renders multiple instances of the mesh model with a given program.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- bool [Renderable](#) [get]
- int [VertexCount](#) [get]
Returns total number of vertices stored in the model.
- int [IndexCount](#) [get]
Returns total number of indices stored in the model.
- uint [Channel](#) [get]
Returns channel associated with vertex buffer format.
- [Buffer VertexBuffer](#) [get]
Returns an integrated vertex buffer.
- [Buffer IndexBuffer](#) [get]
Returns pointer to integrated index buffer.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.52.1 Detailed Description

3D mesh model that contains both vertex and index buffers.

7.52.2 Constructor & Destructor Documentation

7.52.2.1 MeshModel()

```
Afterwarp.MeshModel.MeshModel (
    Device device,
    int vertexCount,
    int vertexElementPitch,
    int indexCount,
    uint channel = 0,
    Array initialVertexData = null,
    Array initialIndexData = null ) [inline]
```

Creates new instance of 3D mesh model container associated with a particular device and pre-initialized vertex and/or index buffers.

7.52.3 Member Function Documentation

7.52.3.1 Dismantle()

```
void Afterwarp.MeshModel.Dismantle ( )
```

Releases vertex and/or index buffers, setting number of vertices and indices to zero. This basically turns an existing model into a "dummy" or a placeholder.

7.52.3.2 Draw()

```
void Afterwarp.MeshModel.Draw (
    Program program,
    PrimitiveTopology topology = PrimitiveTopology::Triangles,
    int elementCount = 0,
    int firstIndex = 0,
    int baseVertex = 0,
    bool postUnbind = true ) [inline]
```

Renders the mesh model with the given program.

7.52.3.3 DrawInstances()

```
void Afterwarp.MeshModel.DrawInstances (
    Program program,
    int instanceCount,
    PrimitiveTopology topology = PrimitiveTopology::Triangles,
    int elementCount = 0,
    int firstIndex = 0,
    int baseVertex = 0,
    bool postUnbind = true ) [inline]
```

Renders multiple instances of the mesh model with a given program.

7.52.3.4 TakeAway()

```
void Afterwarp.MeshModel.TakeAway (
    MeshModel other )
```

Takes away the internal contents from another 3D mesh model container without doing any copying.

7.52.4 Property Documentation

7.52.4.1 Channel

```
uint Afterwarp.MeshModel.Channel [get]
```

Returns channel associated with vertex buffer format.

7.52.4.2 IndexBuffer

```
Buffer Afterwarp.MeshModel.IndexBuffer [get]
```

Returns pointer to integrated index buffer.

7.52.4.3 IndexCount

```
int Afterwarp.MeshModel.IndexCount [get]
```

Returns total number of indices stored in the model.

7.52.4.4 Renderable

```
bool Afterwarp.MeshModel.Renderable [get]
```

Tests whether the model can actually be rendered. For that, it must contain at least a non-empty vertex buffer. Non-renderable mesh is basically a "dummy" that serves as a placeholder.

7.52.4.5 VertexBuffer

```
Buffer Afterwarp.MeshModel.VertexBuffer [get]
```

Returns an integrated vertex buffer.

7.52.4.6 VertexCount

```
int Afterwarp.MeshModel.VertexCount [get]
```

Returns total number of vertices stored in the model.

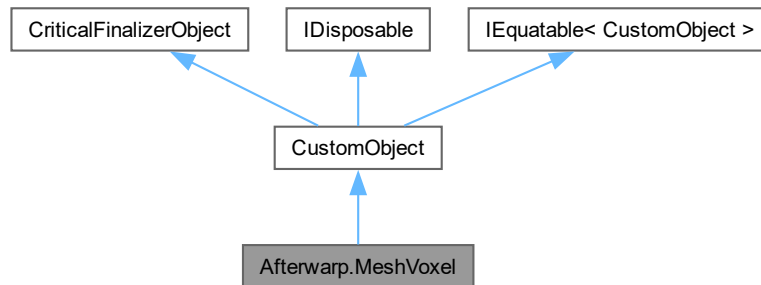
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

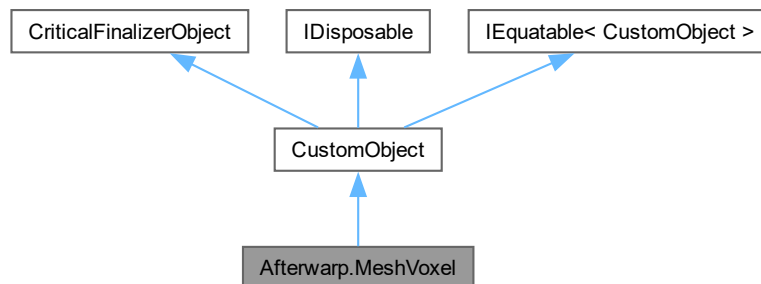
7.53 Afterwarp.MeshVoxel Class Reference

3D mesh voxel representation object.

Inheritance diagram for Afterwarp.MeshVoxel:



Collaboration diagram for Afterwarp.MeshVoxel:



Public Member Functions

- delegate void `VisualizeFunc` (Vector3 position, Vector3 size, `FloatColor` color)
Delegate function that is called for voxel visualization.
- `MeshVoxel` ()
Creates an empty 3D mesh voxel representation.
- bool `TryLoadFromFile` (string fileName, out int[] dimensions, out byte levels, out bool colors)
Loads an existing 3D mesh voxel representation from a file on disk.
- void `LoadFromFile` (string fileName, out int[] dimensions, out byte levels, out bool colors)
Loads an existing 3D mesh voxel representation from a file on disk.
- bool `TryLoadFromFileInMemory` (Array buffer, out int[] dimensions, out byte levels, out bool colors)
Loads an existing 3D mesh voxel representation from a file in memory.
- void `LoadFromFileInMemory` (Array buffer, out int[] dimensions, out byte levels, out bool colors)

- *Loads an existing 3D mesh voxel representation from a file in memory.*
- void [SaveToFile](#) (string fileName, int[] dimensions, out byte levels, out bool colors)
Saves 3D mesh voxel representation to a file on disk.
- void [Extents](#) (out Vector3 position, out Vector3 size)
Returns position and size of 3D mesh voxel representation.
- void [ComputeParameters](#) (out int[] dimensions, out byte levels, out bool colors)
Calculates existing voxel dimensions, levels and whether colors are present.
- void [Visualize](#) (byte levelMax, [VisualizeFunc](#) visualize)
Visualizes 3D mesh voxel representation by invoking delegate function for each cube.
- bool [Intersect](#) (Vector3 origin, Vector3 direction, Matrix4x4 world, Matrix4x4 view, out float distance, out int testsCount)
Performs intersection between voxel representation and ray.
- void [TakeAway](#) ([MeshVoxel](#) other)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

7.53.1 Detailed Description

3D mesh voxel representation object.

7.53.2 Constructor & Destructor Documentation

7.53.2.1 MeshVoxel()

```
Afterwarp.MeshVoxel.MeshVoxel ( ) [inline]
```

Creates an empty 3D mesh voxel representation.

7.53.3 Member Function Documentation

7.53.3.1 ComputeParameters()

```
void Afterwarp.MeshVoxel.ComputeParameters (
    out int[] dimensions,
    out byte levels,
    out bool colors ) [inline]
```

Calculates existing voxel dimensions, levels and whether colors are present.

7.53.3.2 Extents()

```
void Afterwarp.MeshVoxel.Extents (
    out Vector3 position,
    out Vector3 size ) [inline]
```

Returns position and size of 3D mesh voxel representation.

7.53.3.3 Intersect()

```
bool Afterwarp.MeshVoxel.Intersect (
    Vector3 origin,
    Vector3 direction,
    Matrix4x4 world,
    Matrix4x4 view,
    out float distance,
    out int testsCount ) [inline]
```

Performs intersection between voxel representation and ray.

7.53.3.4 LoadFromFile()

```
void Afterwarp.MeshVoxel.LoadFromFile (
    string fileName,
    out int[] dimensions,
    out byte levels,
    out bool colors ) [inline]
```

Loads an existing 3D mesh voxel representation from a file on disk.

7.53.3.5 LoadFromFileInMemory()

```
void Afterwarp.MeshVoxel.LoadFromFileInMemory (
    Array buffer,
    out int[] dimensions,
    out byte levels,
    out bool colors ) [inline]
```

Loads an existing 3D mesh voxel representation from a file in memory.

7.53.3.6 SaveToFile()

```
void Afterwarp.MeshVoxel.SaveToFile (
    string fileName,
    int[] dimensions,
    out byte levels,
    out bool colors ) [inline]
```

Saves 3D mesh voxel representation to a file on disk.

7.53.3.7 TakeAway()

```
void Afterwarp.MeshVoxel.TakeAway (
    MeshVoxel other ) [inline]
```

Takes away the internal contents from another 3D mesh voxel representation replacing any existing contents, without doing any copying.

7.53.3.8 TryLoadFromFile()

```
bool Afterwarp.MeshVoxel.TryLoadFromFile (
    string fileName,
    out int[] dimensions,
    out byte levels,
    out bool colors ) [inline]
```

Loads an existing 3D mesh voxel representation from a file on disk.

7.53.3.9 TryLoadFromFileInMemory()

```
bool Afterwarp.MeshVoxel.TryLoadFromFileInMemory (
    Array buffer,
    out int[] dimensions,
    out byte levels,
    out bool colors ) [inline]
```

Loads an existing 3D mesh voxel representation from a file in memory.

7.53.3.10 Visualize()

```
void Afterwarp.MeshVoxel.Visualize (
    byte levelMax,
    VisualizeFunc visualize ) [inline]
```

Visualizes 3D mesh voxel representation by invoking delegate function for each cube.

7.53.3.11 VisualizeFunc()

```
delegate void Afterwarp.MeshVoxel.VisualizeFunc (
    Vector3 position,
    Vector3 size,
    FloatColor color )
```

Delegate function that is called for voxel visualization.

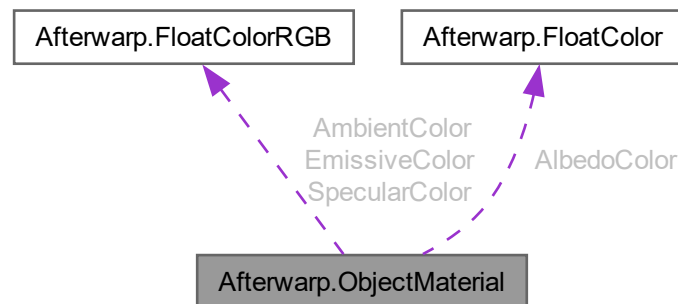
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.54 Afterwarp.ObjectMaterial Struct Reference

Material that describes a particular object in 3D world.

Collaboration diagram for Afterwarp.ObjectMaterial:



Public Member Functions

- `ObjectMaterial` (`TechniqueLighting` technique, `FloatColorRGB` ambientColor, `FloatColor` albedoColor, `FloatColorRGB` specularColor, float specularExponent, `FloatColorRGB` emissiveColor, `Vector3` roughness, float shadows=0.5f, float occlusion=0.0f, uint bitmask=0, float frostedGlass=0.0f, `IntPtr` payload=new `IntPtr`())
Creates new instance of object material with the given values.

Public Attributes

- [TechniqueLighting Technique](#)
Technique that describes how light is reflected off the object's surface.
- [FloatColorRGB AmbientColor](#)
Ambient color that is always received by the object regardless the position of light.
- float [Shadows](#)
Strength of received shadows. A value of zero means no shadows will be received.
- [FloatColor AlbedoColor](#)
Albedo color, where alpha-channel is typically a Bloom Factor (glare from strong reflections).
- [FloatColorRGB SpecularColor](#)
Specular color, which is added depending on reflected vector from material's surface.
- float [SpecularExponent](#)
Specular Exponent that defines how short or wide specular reflection appears.
- [FloatColorRGB EmissiveColor](#)
Emissive color, which is always added before any lights.
- float [Occlusion](#)
Strength of ambient occlusion. A value of zero means no occlusion will be received.
- Vector3 [Roughness](#)
Roughness of the object's surface that defines how light is reflected.
- uint [Bitmask](#)
- float [FrostedGlass](#)
Strength of Frosted Glass effect when rendering order-independent transparency.
- IntPtr [Payload](#)
Custom payload value.

Properties

- static [ObjectMaterial Default](#) [get]
Returns material with default parameters.

7.54.1 Detailed Description

Material that describes a particular object in 3D world.

7.54.2 Constructor & Destructor Documentation**7.54.2.1 ObjectMaterial()**

```
Afterwarp.ObjectMaterial.ObjectMaterial (
    TechniqueLighting technique,
    FloatColorRGB ambientColor,
    FloatColor albedoColor,
    FloatColorRGB specularColor,
    float specularExponent,
    FloatColorRGB emissiveColor,
    Vector3 roughness,
    float shadows = 0::5f,
    float occlusion = 0::0f,
    uint bitmask = 0,
    float frostedGlass = 0::0f,
    IntPtr payload = new IntPtr() ) [inline]
```

Creates new instance of object material with the given values.

7.54.3 Member Data Documentation

7.54.3.1 AlbedoColor

`FloatColor` Afterwarp.ObjectMaterial.AlbedoColor

Albedo color, where alpha-channel is typically a Bloom Factor (glare from strong reflections).

7.54.3.2 AmbientColor

`FloatColorRGB` Afterwarp.ObjectMaterial.AmbientColor

Ambient color that is always received by the object regardless the position of light.

7.54.3.3 Bitmask

`uint` Afterwarp.ObjectMaterial.Bitmask

A bitmask that defines what lights affect this object. A value of zero means that this object is affected by all lights.

7.54.3.4 EmissiveColor

`FloatColorRGB` Afterwarp.ObjectMaterial.EmissiveColor

Emissive color, which is always added before any lights.

7.54.3.5 FrostedGlass

`float` Afterwarp.ObjectMaterial.FrostedGlass

Strength of Frosted Glass effect when rendering order-independent transparency.

7.54.3.6 Occlusion

`float` Afterwarp.ObjectMaterial.Occlusion

Strength of ambient occlusion. A value of zero means no occlusion will be received.

7.54.3.7 Payload

`IntPtr` Afterwarp.ObjectMaterial.Payload

Custom payload value.

7.54.3.8 Roughness

`Vector3 Afterwarp.ObjectMaterial.Roughness`

Roughness of the object's surface that defines how light is reflected.

7.54.3.9 Shadows

`float Afterwarp.ObjectMaterial.Shadows`

Strength of received shadows. A value of zero means no shadows will be received.

7.54.3.10 SpecularColor

`FloatColorRGB Afterwarp.ObjectMaterial.SpecularColor`

Specular color, which is added depending on reflected vector from material's surface.

7.54.3.11 SpecularExponent

`float Afterwarp.ObjectMaterial.SpecularExponent`

Specular Exponent that defines how short or wide specular reflection appears.

7.54.3.12 Technique

`TechniqueLighting Afterwarp.ObjectMaterial.Technique`

Technique that describes how light is reflected off the object's surface.

7.54.4 Property Documentation

7.54.4.1 Default

`ObjectMaterial Afterwarp.ObjectMaterial.Default [static], [get]`

Returns material with default parameters.

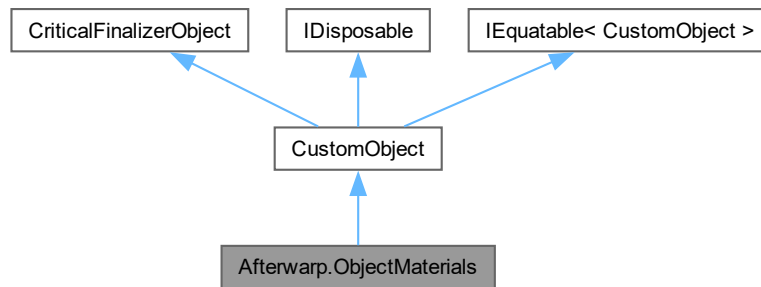
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

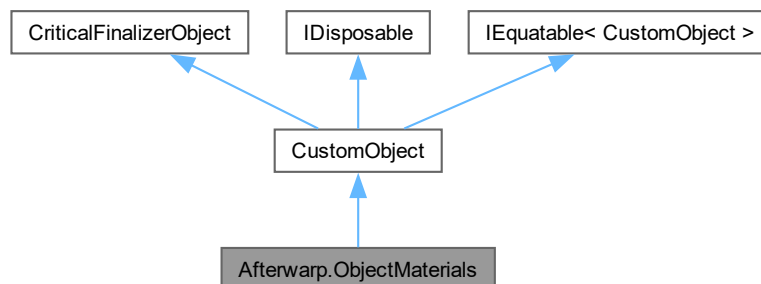
7.55 Afterwarp.ObjectMaterials Class Reference

Container for storing 3D object materials.

Inheritance diagram for Afterwarp.ObjectMaterials:



Collaboration diagram for Afterwarp.ObjectMaterials:



Public Member Functions

- [ObjectMaterials](#) ()
Creates new instance of 3D object materials container.
- void [Add](#) ([ObjectMaterial?](#) material)
Adds and returns a new 3D object material with default parameters.
- [ObjectMaterial](#) [GetMaterial](#) (int index)
Retrieves an existing 3D object material associated with the given index.
- void [SetMaterial](#) (int index, [ObjectMaterial](#) material)
Changes an existing 3D object material associated with the given index.
- void [Erase](#) (int index)
Erases material with the given index.
- void [Clear](#) ()
Removes all existing materials.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- int [Count](#) [get]
Returns total number of materials.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.55.1 Detailed Description

Container for storing 3D object materials.

7.55.2 Constructor & Destructor Documentation

7.55.2.1 [ObjectMaterials\(\)](#)

```
Afterwarp.ObjectMaterials.ObjectMaterials ( ) [inline]
```

Creates new instance of 3D object materials container.

7.55.3 Member Function Documentation

7.55.3.1 Add()

```
void Afterwarp.ObjectMaterials.Add (
    ObjectMaterial? material ) [inline]
```

Adds and returns a new 3D object material with default parameters.

7.55.3.2 Clear()

```
void Afterwarp.ObjectMaterials.Clear ( )
```

Removes all existing materials.

7.55.3.3 Erase()

```
void Afterwarp.ObjectMaterials.Erase (
    int index ) [inline]
```

Erases material with the given index.

7.55.3.4 GetMaterial()

```
ObjectMaterial Afterwarp.ObjectMaterials.GetMaterial (
    int index ) [inline]
```

Retrieves an existing 3D object material associated with the given index.

7.55.3.5 SetMaterial()

```
void Afterwarp.ObjectMaterials.SetMaterial (
    int index,
    ObjectMaterial material ) [inline]
```

Changes an existing 3D object material associated with the given index.

7.55.4 Property Documentation

7.55.4.1 Count

```
int Afterwarp.ObjectMaterials.Count [get]
```

Returns total number of materials.

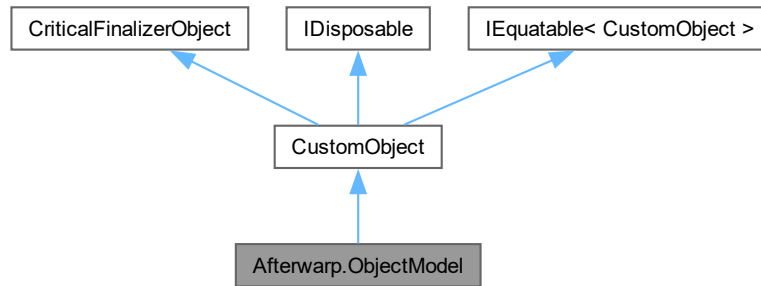
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

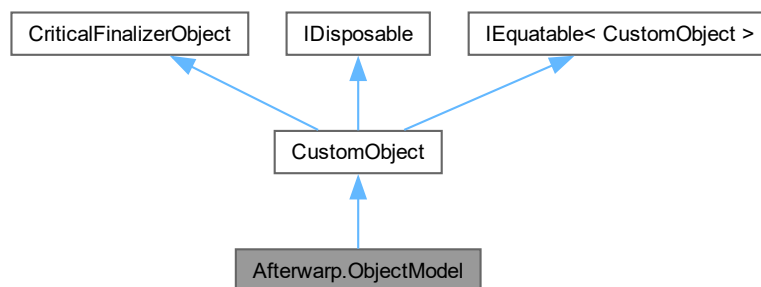
7.56 Afterwarp.ObjectModel Class Reference

Object that represents a 3D model in the scene.

Inheritance diagram for Afterwarp.ObjectModel:



Collaboration diagram for Afterwarp.ObjectModel:



Classes

- struct [Attribute](#)
Attribute bit flags for an object that define its status.

Public Member Functions

- [ObjectModel Child](#) (int index)
Returns object's child with the given index.
- Matrix4x4 [GetTransform](#) ([ModelTransform](#) transform)
Retrieves one of current object model's transforms.
- bool [SetTransform](#) ([ModelTransform](#) transform, Matrix4x4 matrix)
Updates one of current object model's transforms. Returns "true" if the transform really changed.

- void [AABB](#) (out Vector3 boxMin, out Vector3 boxMax)
Updates and returns axis-aligned bounding box of the object.
- void [Invalidate](#) ()
Marks local transform, volume and position as dirty, including children.
- void [ConnectLatches](#) (int latchParent, int latchLocal)
Connects a local bone joint with the given index to a corresponding joint of the parent.
- void [ConnectLatches](#) (string latchParentName, string latchLocalName)
- void [GetConnectedLatches](#) (out int latchParent, out int latchLocal)
Returns indexes of currently connected bone joints.
- float [GetWaypointDistance](#) (int group)
Returns calculated waypoint traveling distance for the given latch group.
- void [GetLatchWaypointCouple](#) (int group, float distance, out Vector3 position, out Quaternion orientation)
- Matrix4x4 [GetLatchWaypointCoupleMatrix](#) (int group, float distance)
- Matrix4x4 [GetLatchTransform](#) (int latchIndex, bool local=false)
Retrieves either local or global transformation matrix for a given latch index.
- Matrix4x4 [GetLatchTransform](#) (string latchName, bool local=false)
Retrieves either local or global transformation matrix for a latch with the given name.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [ObjectModels Owner](#) [get]
Returns a container that owns the object.
- IntPtr [ID](#) [get]
Returns object's unique identification number.
- string [Name](#) [get, set]
Returns or changes object's name (case-insensitive).
- string [Description](#) [get, set]
Returns or changes object's description.
- IntPtr [Payload](#) [get]
Returns object's payload.
- [MeshVoxel Voxel](#) [get, set]
Returns or changes object model's voxel representation.
- [SceneMesh Mesh](#) [get, set]
Returns or changes object's rendereable mesh.
- string [MeshName](#) [get, set]
- [ObjectModel Parent](#) [get, set]
Returns or changes object model's parent.
- Vector3 [Position](#) [get]
Extracts and returns current object position.
- float [DepthBias](#) [get, set]
Returns or changes current depth bias, which is used during object selection.
- uint [Attributes](#) [get, set]
Returns or changes object model's attributes.

- int [OrderIndex](#) [get, set]
Returns or changes object's priority index.
- ulong [Layers](#) [get, set]
Returns or changes layers on which the object is visible.
- Vector3 [Size](#) [get, set]
Returns or changes object's size.
- [MeshAligns Alignments](#) [get, set]
Returns or changes object's mesh alignments.
- int [Material](#) [get, set]
Returns or changes object's material.
- [FloatColor Color](#) [get, set]
Returns or changes object's color.
- [FloatColor Highlight](#) [get, set]
Returns or changes object's highlight color.
- int [ChildCount](#) [get]
Returns number of object model's children.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.56.1 Detailed Description

Object that represents a 3D model in the scene.

7.56.2 Member Function Documentation

7.56.2.1 AABB()

```
void Afterwarp.ObjectModel.AABB (
    out Vector3 boxMin,
    out Vector3 boxMax ) [inline]
```

Updates and returns axis-aligned bounding box of the object.

7.56.2.2 Child()

```
ObjectModel Afterwarp.ObjectModel.Child (
    int index ) [inline]
```

Returns object's child with the given index.

7.56.2.3 ConnectLatches() [1/2]

```
void Afterwarp.ObjectModel.ConnectLatches (
    int latchParent,
    int latchLocal ) [inline]
```

Connects a local bone joint with the given index to a corresponding joint of the parent.

7.56.2.4 ConnectLatches() [2/2]

```
void Afterwarp.ObjectModel.ConnectLatches (
    string latchParentName,
    string latchLocalName ) [inline]
```

Connects a local bone joint with the given name (case-insensitive) to a corresponding joint of the parent.

7.56.2.5 GetConnectedLatches()

```
void Afterwarp.ObjectModel.GetConnectedLatches (
    out int latchParent,
    out int latchLocal ) [inline]
```

Returns indexes of currently connected bone joints.

7.56.2.6 GetLatchTransform() [1/2]

```
Matrix4x4 Afterwarp.ObjectModel.GetLatchTransform (
    int latchIndex,
    bool local = false ) [inline]
```

Retrieves either local or global transformation matrix for a given latch index.

7.56.2.7 GetLatchTransform() [2/2]

```
Matrix4x4 Afterwarp.ObjectModel.GetLatchTransform (
    string latchName,
    bool local = false ) [inline]
```

Retrieves either local or global transformation matrix for a latch with the given name.

7.56.2.8 GetLatchWaypointCouple()

```
void Afterwarp.ObjectModel.GetLatchWaypointCouple (
    int group,
    float distance,
    out Vector3 position,
    out Quaternion orientation ) [inline]
```

Calculates interpolated position and orientation based on the given distance for the given latch group.

7.56.2.9 GetLatchWaypointCoupleMatrix()

```
Matrix4x4 Afterwarp.ObjectModel.GetLatchWaypointCoupleMatrix (
    int group,
    float distance ) [inline]
```

Calculates transformation matrix with interpolated position and orientation based on the given distance for the given latch group.

7.56.2.10 GetTransform()

```
Matrix4x4 Afterwarp.ObjectModel.GetTransform (
    ModelTransform transform ) [inline]
```

Retrieves one of current object model's transforms.

7.56.2.11 GetWaypointDistance()

```
float Afterwarp.ObjectModel.GetWaypointDistance (
    int group )
```

Returns calculated waypoint traveling distance for the given latch group.

7.56.2.12 Invalidate()

```
void Afterwarp.ObjectModel.Invalidate ( )
```

Marks local transform, volume and position as dirty, including children.

7.56.2.13 SetTransform()

```
bool Afterwarp.ObjectModel.SetTransform (
    ModelTransform transform,
    Matrix4x4 matrix ) [inline]
```

Updates one of current object model's transforms. Returns "true" if the transform really changed.

7.56.3 Property Documentation

7.56.3.1 Alignments

`MeshAligns` Afterwarp.ObjectModel.Alignments [get], [set]

Returns or changes object's mesh alignments.

7.56.3.2 Attributes

`uint` Afterwarp.ObjectModel.Attributes [get], [set]

Returns or changes object model's attributes.

7.56.3.3 ChildCount

`int` Afterwarp.ObjectModel.ChildCount [get]

Returns number of object model's children.

7.56.3.4 Color

`FloatColor` Afterwarp.ObjectModel.Color [get], [set]

Returns or changes object's color.

7.56.3.5 DepthBias

`float` Afterwarp.ObjectModel.DepthBias [get], [set]

Returns or changes current depth bias, which is used during object selection.

7.56.3.6 Description

`string` Afterwarp.ObjectModel.Description [get], [set]

Returns or changes object's description.

7.56.3.7 Highlight

`FloatColor` Afterwarp.ObjectModel.Highlight [get], [set]

Returns or changes object's highlight color.

7.56.3.8 ID

```
IntPtr Afterwarp.ObjectModel.ID [get]
```

Returns object's unique identification number.

7.56.3.9 Layers

```
ulong Afterwarp.ObjectModel.Layers [get], [set]
```

Returns or changes layers on which the object is visible.

7.56.3.10 Material

```
int Afterwarp.ObjectModel.Material [get], [set]
```

Returns or changes object's material.

7.56.3.11 Mesh

```
SceneMesh Afterwarp.ObjectModel.Mesh [get], [set]
```

Returns or changes object's rendereable mesh.

7.56.3.12 MeshName

```
string Afterwarp.ObjectModel.MeshName [get], [set]
```

Returns or changes object's rendereable mesh name (must be one of the mesh names from the associated container).

7.56.3.13 Name

```
string Afterwarp.ObjectModel.Name [get], [set]
```

Returns or changes object's name (case-insensitive).

7.56.3.14 OrderIndex

```
int Afterwarp.ObjectModel.OrderIndex [get], [set]
```

Returns or changes object's priority index.

7.56.3.15 Owner

`ObjectModels Afterwarp.ObjectModel.Owner [get]`

Returns a container that owns the object.

7.56.3.16 Parent

`ObjectModel Afterwarp.ObjectModel.Parent [get], [set]`

Returns or changes object model's parent.

7.56.3.17 Payload

`IntPtr Afterwarp.ObjectModel.Payload [get]`

Returns object's payload.

7.56.3.18 Position

`Vector3 Afterwarp.ObjectModel.Position [get]`

Extracts and returns current object position.

7.56.3.19 Size

`Vector3 Afterwarp.ObjectModel.Size [get], [set]`

Returns or changes object's size.

7.56.3.20 Voxel

`MeshVoxel Afterwarp.ObjectModel.Voxel [get], [set]`

Returns or changes object model's voxel representation.

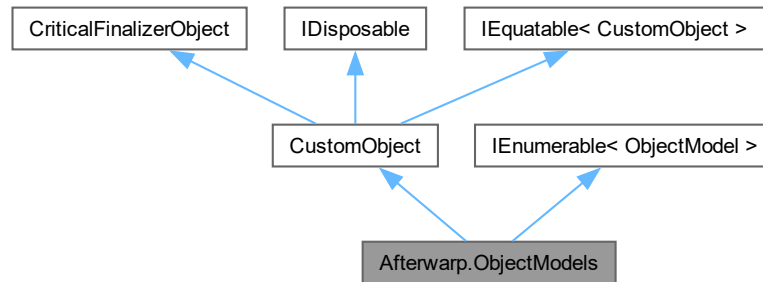
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

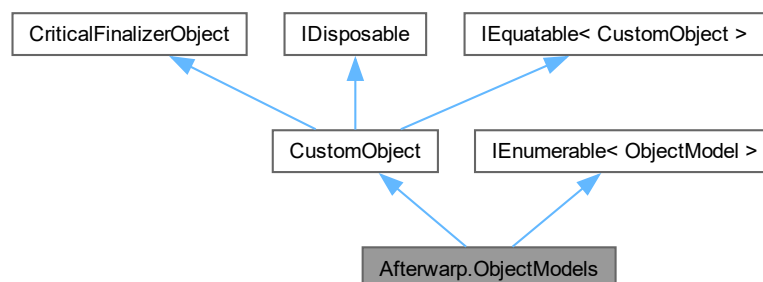
7.57 Afterwarp.ObjectModels Class Reference

A container of objects and their volume representation in 3D world.

Inheritance diagram for Afterwarp.ObjectModels:



Collaboration diagram for Afterwarp.ObjectModels:



Classes

- class [Iterator](#)
Iterator for accessing individual objects in the container.

Public Member Functions

- [ObjectModels](#) ([SceneMeshes](#) meshes=null)
Creates new instance of object container.
- [ObjectModel Add](#) (IntPtr id, string name=null, IntPtr payload=default, uint attributes=[ObjectModel.Attribute.Visible](#)|[ObjectModel.Attribute...](#))
- [ObjectModel Add](#) (string name=null, IntPtr payload=default, uint attributes=[ObjectModel.Attribute.Visible](#)|[ObjectModel.Attribute...](#))
Creates a new object with the given name and/or payload association.
- void [Erase](#) ([ObjectModel](#) objectModel)

- Removes the given object.*
 - void [Clear](#) ()
- Removes all objects.*
 - [ObjectModel TryObject](#) (IntPtr id)*Returns an object with the given ID if such exists.*
- [ObjectModel Object](#) (IntPtr id)*Returns an object with the given ID.*
- [ObjectModel TryObject](#) (string name)*Returns an object with the given name (case-insensitive) if such exists.*
- [ObjectModel Object](#) (string name)*Returns an object with the given name (case-insensitive).*
- [ObjectModel TryPayload](#) (IntPtr payload)*Returns an object with the given payload, if such exists.*
- [ObjectModel Payload](#) (IntPtr payload)*Returns an object with the given payload.*
- bool [Exists](#) (IntPtr id)*Tests whether an object with the given ID exists.*
- bool [Exists](#) (string name)*Tests whether an object with the given name exists.*
- bool [PayloadExists](#) (IntPtr payload)*Tests whether an object with the given payload exists.*
- [ObjectModelView CreateView](#) ()*Creates a new view associated with this container.*
- void [EraseView](#) ([ObjectModelView](#) view)*Removes a given view from the container.*
- void [ClearViews](#) ()*Removes all existing views from the container.*
- IEnumerator< [ObjectModel](#) > [GetEnumerator](#) ()*Creates enumerator for iterating through the container.*

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [SceneMeshes Meshes](#) [get]
- Returns the associated mesh container.*
- int [Count](#) [get]
- Returns number of existing objects.*

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
- Wrapped object's handle.*

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.57.1 Detailed Description

A container of objects and their volume representation in 3D world.

7.57.2 Constructor & Destructor Documentation

7.57.2.1 ObjectModels()

```
Afterwarp.ObjectModels.ObjectModels (
    SceneMeshes meshes = null ) [inline]
```

Creates new instance of object container.

7.57.3 Member Function Documentation

7.57.3.1 Add() [1/2]

```
ObjectModel Afterwarp.ObjectModels.Add (
    IntPtr id,
    string name = null,
    IntPtr payload = default,
    uint attributes = ObjectModel::Attribute::Visible | ObjectModel::Attribute::Selectable
) [inline]
```

Creates a new object with a given id, name and/or payload association. If the given ID already exists, a new unique one will be assigned to the object. Unique IDs are generated incrementally with roll-over when highest possible ID is reached, so eventually lower order IDs might be reused; the function ensures, however, that a given ID is unique in sense that it is not assigned to another other object.

7.57.3.2 Add() [2/2]

```
ObjectModel Afterwarp.ObjectModels.Add (
    string name = null,
    IntPtr payload = default,
    uint attributes = ObjectModel::Attribute::Visible | ObjectModel::Attribute::Selectable
) [inline]
```

Creates a new object with the given name and/or payload association.

7.57.3.3 Clear()

```
void Afterwarp.ObjectModels.Clear ( )
```

Removes all objects.

7.57.3.4 ClearViews()

```
void Afterwarp.ObjectModels.ClearViews ( )
```

Removes all existing views from the container.

7.57.3.5 CreateView()

```
ObjectModelView Afterwarp.ObjectModels.CreateView ( ) [inline]
```

Creates a new view associated with this container.

7.57.3.6 Erase()

```
void Afterwarp.ObjectModels.Erase (
    ObjectModel objectModel ) [inline]
```

Removes the given object.

7.57.3.7 EraseView()

```
void Afterwarp.ObjectModels.EraseView (
    ObjectModelView view ) [inline]
```

Removes a given view from the container.

7.57.3.8 Exists() [1/2]

```
bool Afterwarp.ObjectModels.Exists (
    IntPtr id )
```

Tests whether an object with the given ID exists.

7.57.3.9 Exists() [2/2]

```
bool Afterwarp.ObjectModels.Exists (
    string name )
```

Tests whether an object with the given name exists.

7.57.3.10 GetEnumerator()

```
IEnumerator< ObjectModel > Afterwarp.ObjectModels.GetEnumerator ( )
```

Creates enumerator for iterating through the container.

7.57.3.11 Object() [1/2]

```
ObjectModel Afterwarp.ObjectModels.Object (
    IntPtr id ) [inline]
```

Returns an object with the given ID.

7.57.3.12 Object() [2/2]

```
ObjectModel Afterwarp.ObjectModels.Object (
    string name ) [inline]
```

Returns an object with the given name (case-insensitive).

7.57.3.13 Payload()

```
ObjectModel Afterwarp.ObjectModels.Payload (
    IntPtr payload ) [inline]
```

Returns an object with the given payload.

7.57.3.14 PayloadExists()

```
bool Afterwarp.ObjectModels.PayloadExists (
    IntPtr payload )
```

Tests whether an object with the given payload exists.

7.57.3.15 TryObject() [1/2]

```
ObjectModel Afterwarp.ObjectModels.TryObject (
    IntPtr id ) [inline]
```

Returns an object with the given ID if such exists.

7.57.3.16 TryObject() [2/2]

```
ObjectModel Afterwarp.ObjectModels.TryObject (
    string name ) [inline]
```

Returns an object with the given name (case-insensitive) if such exists.

7.57.3.17 TryPayload()

```
ObjectModel Afterwarp.ObjectModels.TryPayload (
    IntPtr payload ) [inline]
```

Returns an object with the given payload, if such exists.

7.57.4 Property Documentation

7.57.4.1 Count

```
int Afterwarp.ObjectModels.Count [get]
```

Returns number of existing objects.

7.57.4.2 Meshes

```
SceneMeshes Afterwarp.ObjectModels.Meshes [get]
```

Returns the associated mesh container.

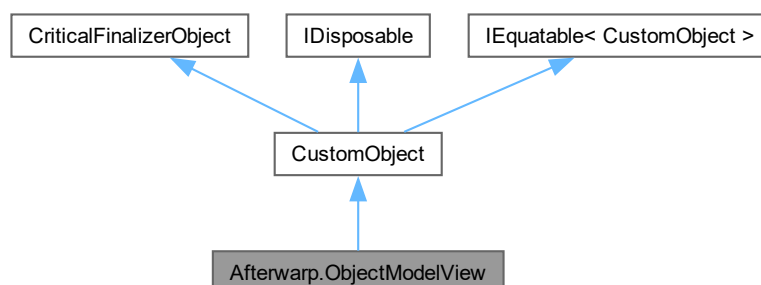
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

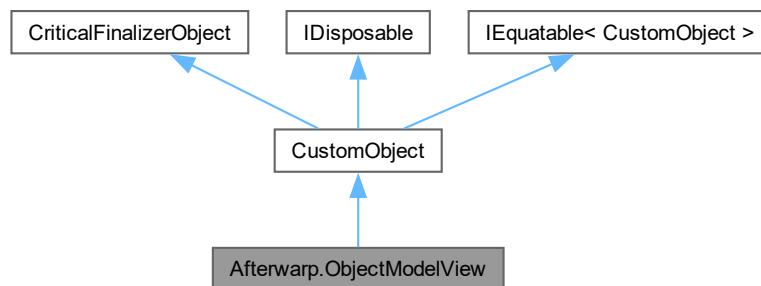
7.58 Afterwarp.ObjectModelView Class Reference

A container that represents a particular view that watches objects in the world.

Inheritance diagram for Afterwarp.ObjectModelView:



Collaboration diagram for Afterwarp.ObjectModelView:



Public Member Functions

- delegate int [CompareFunc](#) ([ObjectModel](#) object1, [ObjectModel](#) object2)
Comparison function for a pair of objects in a 3D scene.
- void [UpdateNeeded](#) ()
Notifies the container that all visible objects must be recalculated.
- void [Invalidate](#) ()
Notifies the container that it needs to be repainted.
- [ObjectModel](#) [Object](#) (int index)
Returns an object with the given index from the view container.
- bool [Update](#) ()
- void [Sort](#) ([ObjectModelViewCompare](#) compare)
Sorts the list of objects using a predefined comparison function.
- void [Sort](#) ([CompareFunc](#) compare)
Sorts the list of objects using a custom comparison function.
- [ObjectModel](#) [Select](#) ([Ray](#) ray, out float distance)
- [ObjectModel](#) [SelectAny](#) ([Ray](#) ray, out float distance)
- int [AutoDraw](#) ([Scene](#) scene, [ObjectMaterials](#) materials=null, uint options=0)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [ObjectModels](#) [Owner](#) [get]
Returns parent of the object model's view.
- Matrix4x4 [View](#) [get, set]
Returns or changes "View" transformation matrix of the container.
- Matrix4x4 [Projection](#) [get, set]

- Returns or changes "Projection" transformation matrix of the container.*
- Matrix4x4 [ViewProjection](#) [get]
Returns combined "View/Projection" matrix of the container.
- ulong [Layers](#) [get, set]
Returns or changes layers that determine object visibility in the view.
- int [ObjectCount](#) [get]
Returns number of existing objects that are visible in the view.
- int [ObjectsNotCulled](#) [get]
- int [IntersectedRays](#) [get]
Returns number of ray vs AABB intersections performed last time selection was performed.
- int [IntersectedObjects](#) [get]
Returns number of objects that passed intersection during last selection attempt.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.58.1 Detailed Description

A container that represents a particular view that watches objects in the world.

7.58.2 Member Function Documentation

7.58.2.1 AutoDraw()

```
int Afterwarp.ObjectModelView.AutoDraw (
    Scene scene,
    ObjectMaterials materials = null,
    uint options = 0 ) [inline]
```

Renders objects that are visible in the view according to the provided options. If *scene* is `null`, this does not perform instancing or issue actual draw calls, instead performing a simple non-instancing run and counting any tentative draw calls that would be issued; this can be used to determine if any objects would be rendered at all or not to decide whether to perform a certain rendering pass.

7.58.2.2 CompareFunc()

```
delegate int Afterwarp.ObjectModelView.CompareFunc (
    ObjectModel object1,
    ObjectModel object2 )
```

Comparison function for a pair of objects in a 3D scene.

7.58.2.3 Invalidate()

```
void Afterwarp.ObjectModelView.Invalidate ( )
```

Notifies the container that it needs to be repainted.

7.58.2.4 Object()

```
ObjectModel Afterwarp.ObjectModelView.Object (
    int index ) [inline]
```

Returns an object with the given index from the view container.

7.58.2.5 Select()

```
ObjectModel Afterwarp.ObjectModelView.Select (
    Ray ray,
    out float distance ) [inline]
```

Performs object selection either through object's voxel hierarchy if such is present and if not, through ray-volume intersection. All selectable and visible objects are tested and the object with closest intersection is finally chosen.

7.58.2.6 SelectAny()

```
ObjectModel Afterwarp.ObjectModelView.SelectAny (
    Ray ray,
    out float distance ) [inline]
```

Performs object selection either through object's voxel hierarchy if such is present and if not, through ray-volume intersection. All visible objects are tested and the object with closest intersection is finally chosen.

7.58.2.7 Sort() [1/2]

```
void Afterwarp.ObjectModelView.Sort (
    CompareFunc compare ) [inline]
```

Sorts the list of objects using a custom comparison function.

7.58.2.8 Sort() [2/2]

```
void Afterwarp.ObjectModelView.Sort (
    ObjectModelViewCompare compare )
```

Sorts the list of objects using a predefined comparison function.

7.58.2.9 Update()

```
bool Afterwarp.ObjectModelView.Update ( )
```

Creates a list of objects for selection and rendering from the view container. Returns "true" if there are any objects visible in the view.

7.58.2.10 UpdateNeeded()

```
void Afterwarp.ObjectModelView.UpdateNeeded ( )
```

Notifies the container that all visible objects must be recalculated.

7.58.3 Property Documentation

7.58.3.1 IntersectedObjects

```
int Afterwarp.ObjectModelView.IntersectedObjects [get]
```

Returns number of objects that passed intersection during last selection attempt.

7.58.3.2 IntersectedRays

```
int Afterwarp.ObjectModelView.IntersectedRays [get]
```

Returns number of ray vs AABB intersections performed last time selection was performed.

7.58.3.3 Layers

```
ulong Afterwarp.ObjectModelView.Layers [get], [set]
```

Returns or changes layers that determine object visibility in the view.

7.58.3.4 ObjectCount

```
int Afterwarp.ObjectModelView.ObjectCount [get]
```

Returns number of existing objects that are visible in the view.

7.58.3.5 ObjectsNotCulled

```
int Afterwarp.ObjectModelView.ObjectsNotCulled [get]
```

Returns number of objects that were not performed cull test, either because they were determined to be fully visible, or because they were marked as hierarchyless.

7.58.3.6 Owner

```
ObjectModels Afterwarp.ObjectModelView.Owner [get]
```

Returns parent of the object model's view.

7.58.3.7 Projection

```
Matrix4x4 Afterwarp.ObjectModelView.Projection [get], [set]
```

Returns or changes "Projection" transformation matrix of the container.

7.58.3.8 View

```
Matrix4x4 Afterwarp.ObjectModelView.View [get], [set]
```

Returns or changes "View" transformation matrix of the container.

7.58.3.9 ViewProjection

```
Matrix4x4 Afterwarp.ObjectModelView.ViewProjection [get]
```

Returns combined "View/Projection" matrix of the container.

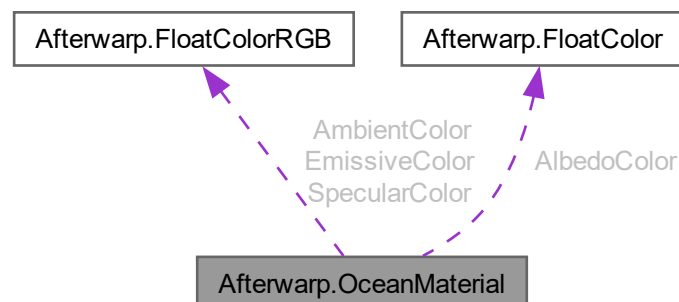
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.59 Afterwarp.OceanMaterial Struct Reference

Material that describes ocean water surface in a 3D simulation.

Collaboration diagram for Afterwarp.OceanMaterial:



Public Member Functions

- [OceanMaterial](#) ([FloatColorRGB](#) ambientColor, float fresnel, [FloatColor](#) albedoColor, [FloatColorRGB](#) specularColor, float specularExponent, [FloatColorRGB](#) emissiveColor, float extinction, float shadows, uint bitmask)

Creates new ocean material with the given values.

Public Attributes

- [FloatColorRGB AmbientColor](#)
Ambient color.
- float [Fresnel](#)
Fresnel Coefficient (reflectance at normal incidence).
- [FloatColor AlbedoColor](#)
Ambient color that is always received by the ocean regardless the position of light.
- [FloatColorRGB SpecularColor](#)
Specular color, which is added depending on reflected vector from ocean's surface.
- float [SpecularExponent](#)
Specular color, which is added depending on reflected vector from ocean's surface.
- [FloatColorRGB EmissiveColor](#)
Emissive color, which is always added before any lights.
- float [Extinction](#)
Extinction of the ocean's surface, which determines its transparency.
- float [Shadows](#)
- uint [Bitmask](#)

Properties

- static [OceanMaterial Default](#) [get]
Returns default ocean material.

7.59.1 Detailed Description

Material that describes ocean water surface in a 3D simulation.

7.59.2 Constructor & Destructor Documentation

7.59.2.1 OceanMaterial()

```
Afterwarp.OceanMaterial.OceanMaterial (
    FloatColorRGB ambientColor,
    float fresnel,
    FloatColor albedoColor,
    FloatColorRGB specularColor,
    float specularExponent,
    FloatColorRGB emissiveColor,
    float extinction,
    float shadows,
    uint bitmask ) [inline]
```

Creates new ocean material with the given values.

7.59.3 Member Data Documentation

7.59.3.1 AlbedoColor

`FloatColor` Afterwarp.OceanMaterial.AlbedoColor

Ambient color that is always received by the ocean regardless the position of light.

7.59.3.2 AmbientColor

`FloatColorRGB` Afterwarp.OceanMaterial.AmbientColor

Ambient color.

7.59.3.3 Bitmask

`uint` Afterwarp.OceanMaterial.Bitmask

A bitmask that defines what lights affect this object. A value of zero means that this object is affected by all lights.

7.59.3.4 EmissiveColor

`FloatColorRGB` Afterwarp.OceanMaterial.EmissiveColor

Emissive color, which is always added before any lights.

7.59.3.5 Extinction

`float` Afterwarp.OceanMaterial.Extinction

Extinction of the ocean's surface, which determines its transparency.

7.59.3.6 Fresnel

`float` Afterwarp.OceanMaterial.Fresnel

Fresnel Coefficient (reflectance at normal incidence).

7.59.3.7 Shadows

`float` Afterwarp.OceanMaterial.Shadows

Strength of a shadows when projected to the object's surface. A value of zero means shadows are not rendered and/or ignored.

7.59.3.8 SpecularColor

`FloatColorRGB Afterwarp.OceanMaterial.SpecularColor`

Specular color, which is added depending on reflected vector from ocean's surface.

7.59.3.9 SpecularExponent

`float Afterwarp.OceanMaterial.SpecularExponent`

Specular color, which is added depending on reflected vector from ocean's surface.

7.59.4 Property Documentation

7.59.4.1 Default

`OceanMaterial Afterwarp.OceanMaterial.Default [static], [get]`

Returns default ocean material.

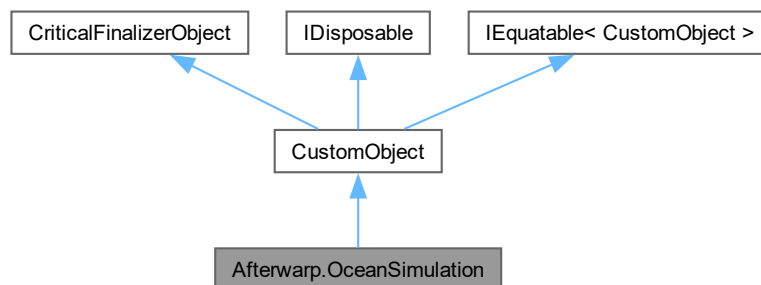
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

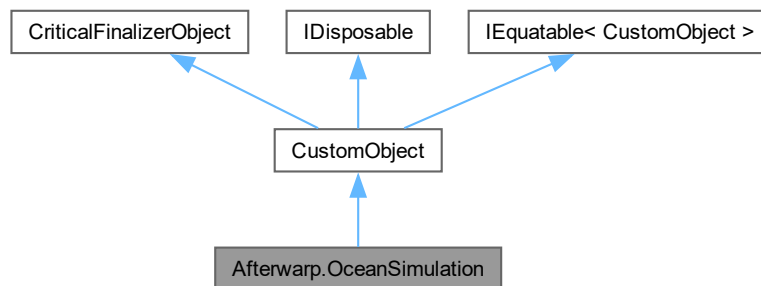
7.60 Afterwarp.OceanSimulation Class Reference

3D ocean semi-infinite wave field rendering module.

Inheritance diagram for `Afterwarp.OceanSimulation`:



Collaboration diagram for Afterwarp.OceanSimulation:



Public Member Functions

- [OceanSimulation](#) ([Device](#) device)
Creates new instance of 3D ocean rendering module.
- [Texture GetWavesTexture](#) (int index)
Returns an existing ocean waves simulation texture. This would be mostly useful for debug purposes.
- void [Update](#) (double latency)
Executes the simulation of ocean waves and updates internal textures.
- void [Render](#) ([Texture](#) linearDepths, [SceneLights](#) lights, [ShadowCastingAtlas](#) atlas=null)
Renders the simulation to an existing active rendering surface.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the module.
- [OceanWavesParameters Parameters](#) [get, set]
Returns or changes ocean waves simulation parameters.
- [Point Sections](#) [get, set]
Returns or changes the number of grid sub-divisions in screen-space.
- Matrix4x4 [View](#) [get, set]
Returns or changes "View" transformation matrix.
- Matrix4x4 [Projection](#) [get, set]
Returns or changes "Projection" transformation matrix.
- Vector3 [Scale](#) [get, set]
Returns or changes scale of the ocean's displacement map.
- Vector4 [Plane](#) [get, set]

- *Returns or changes grid's plane equation.*
- float [ViewDistance](#) [get, set]
Returns or changes maximum height-map viewing distance.
- uint [Attributes](#) [get, set]
Returns or changes attributes that define the rendering behavior of the module.
- [OceanMaterial](#) [Material](#) [get, set]
Returns or changes ocean's material.
- [Sampler](#) [SamplerShadow](#) [get]
Returns sampler for reading from shadow map.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.60.1 Detailed Description

3D ocean semi-infinite wave field rendering module.

7.60.2 Constructor & Destructor Documentation

7.60.2.1 OceanSimulation()

```
Afterwarp.OceanSimulation.OceanSimulation (
    Device device ) [inline]
```

Creates new instance of 3D ocean rendering module.

7.60.3 Member Function Documentation

7.60.3.1 GetWavesTexture()

```
Texture Afterwarp.OceanSimulation.GetWavesTexture (
    int index ) [inline]
```

Returns an existing ocean waves simulation texture. This would be mostly useful for debug purposes.

7.60.3.2 Render()

```
void Afterwarp.OceanSimulation.Render (
    Texture linearDepths,
    SceneLights lights,
    ShadowCastingAtlas atlas = null ) [inline]
```

Renders the simulation to an existing active rendering surface.

7.60.3.3 Update()

```
void Afterwarp.OceanSimulation.Update (
    double latency ) [inline]
```

Executes the simulation of ocean waves and updates internal textures.

7.60.4 Property Documentation

7.60.4.1 Attributes

```
uint Afterwarp.OceanSimulation.Attributes [get], [set]
```

Returns or changes attributes that define the rendering behavior of the module.

7.60.4.2 Device

```
Device Afterwarp.OceanSimulation.Device [get]
```

Returns device associated with the module.

7.60.4.3 Material

```
OceanMaterial Afterwarp.OceanSimulation.Material [get], [set]
```

Returns or changes ocean's material.

7.60.4.4 Parameters

```
OceanWavesParameters Afterwarp.OceanSimulation.Parameters [get], [set]
```

Returns or changes ocean waves simulation parameters.

7.60.4.5 Plane

```
Vector4 Afterwarp.OceanSimulation.Plane [get], [set]
```

Returns or changes grid's plane equation.

7.60.4.6 Projection

`Matrix4x4 Afterwarp.OceanSimulation.Projection [get], [set]`

Returns or changes "Projection" transformation matrix.

7.60.4.7 SamplerShadow

`Sampler Afterwarp.OceanSimulation.SamplerShadow [get]`

Returns sampler for reading from shadow map.

7.60.4.8 Scale

`Vector3 Afterwarp.OceanSimulation.Scale [get], [set]`

Returns or changes scale of the ocean's displacement map.

7.60.4.9 Sections

`Point Afterwarp.OceanSimulation.Sections [get], [set]`

Returns or changes the number of grid sub-divisions in screen-space.

7.60.4.10 View

`Matrix4x4 Afterwarp.OceanSimulation.View [get], [set]`

Returns or changes "View" transformation matrix.

7.60.4.11 ViewDistance

`float Afterwarp.OceanSimulation.ViewDistance [get], [set]`

Returns or changes maximum height-map viewing distance.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.61 Afterwarp.OceanWavesParameters Struct Reference

Parameters that define the appearance of waves simulation.

Public Member Functions

- [OceanWavesParameters](#) (int resolution, Vector2 wind, float waveSize, float choppiness)
Creates new ocean wave simulation parameters with the give nvalues.

Public Attributes

- int [Resolution](#)
Resolution of the simulation textures.
- Vector2 [Wind](#)
General direction in which the waves propagate.
- float [WaveSize](#)
Size of the generated waves.
- float [Choppiness](#)
Choppiness of the generated waves.

Properties

- static [OceanWavesParameters Default](#) [get]
Returns default ocean wave simulation parameters.

7.61.1 Detailed Description

Parameters that define the appearance of waves simulation.

7.61.2 Constructor & Destructor Documentation

7.61.2.1 OceanWavesParameters()

```
Afterwarp.OceanWavesParameters.OceanWavesParameters (
    int resolution,
    Vector2 wind,
    float waveSize,
    float choppiness ) [inline]
```

Creates new ocean wave simulation parameters with the give nvalues.

7.61.3 Member Data Documentation

7.61.3.1 Choppiness

```
float Afterwarp.OceanWavesParameters.Choppiness
```

Choppiness of the generated waves.

7.61.3.2 Resolution

```
int Afterwarp.OceanWavesParameters.Resolution
```

Resolution of the simulation textures.

7.61.3.3 WaveSize

```
float Afterwarp.OceanWavesParameters.WaveSize
```

Size of the generated waves.

7.61.3.4 Wind

```
Vector2 Afterwarp.OceanWavesParameters.Wind
```

General direction in which the waves propagate.

7.61.4 Property Documentation

7.61.4.1 Default

```
OceanWavesParameters Afterwarp.OceanWavesParameters.Default [static], [get]
```

Returns default ocean wave simulation parameters.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.62 Afterwarp.ParallaxMappingParameters Struct Reference

Parameters that define how parallax mapping is performed.

Public Member Functions

- [ParallaxMappingParameters](#) (float scale, int samplesMin, int samplesMax, float occlusion)
Creates new parallax mapping parameters with the given values.

Public Attributes

- float [Scale](#)
Scale that defines how height value from parallax map translates into 3D depth.
- int [SamplesMin](#)
Minimum number of samples to read per pixel.
- int [SamplesMax](#)
Maximum number of samples to read per pixel.
- float [Occlusion](#)
Occlusion strength applied from Occlusion Map (alpha-channel of Normal Map).

Properties

- static [ParallaxMappingParameters Default](#) [get]
Returns default parallax mapping parameters.

7.62.1 Detailed Description

Parameters that define how parallax mapping is performed.

7.62.2 Constructor & Destructor Documentation

7.62.2.1 ParallaxMappingParameters()

```
Afterwarp.ParallaxMappingParameters.ParallaxMappingParameters (
    float scale,
    int samplesMin,
    int samplesMax,
    float occlusion ) [inline]
```

Creates new parallax mapping parameters with the given values.

7.62.3 Member Data Documentation

7.62.3.1 Occlusion

```
float Afterwarp.ParallaxMappingParameters.Occlusion
```

Occlusion strength applied from Occlusion Map (alpha-channel of Normal Map).

7.62.3.2 SamplesMax

```
int Afterwarp.ParallaxMappingParameters.SamplesMax
```

Maximum number of samples to read per pixel.

7.62.3.3 SamplesMin

```
int Afterwarp.ParallaxMappingParameters.SamplesMin
```

Minimum number of samples to read per pixel.

7.62.3.4 Scale

```
float Afterwarp.ParallaxMappingParameters.Scale
```

Scale that defines how height value from parallax map translates into 3D depth.

7.62.4 Property Documentation

7.62.4.1 Default

`ParallaxMappingParameters` `Afterwarp.ParallaxMappingParameters.Default` [static], [get]

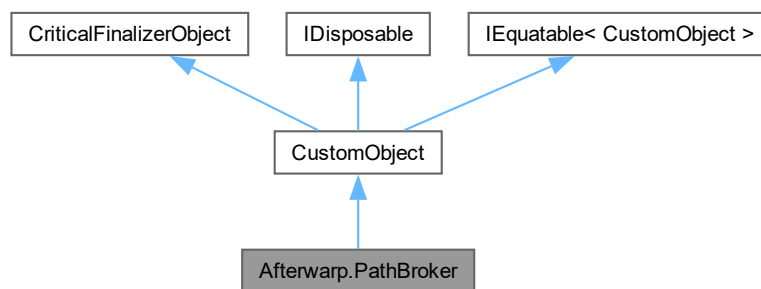
Returns default parallax mapping parameters.

The documentation for this struct was generated from the following file:

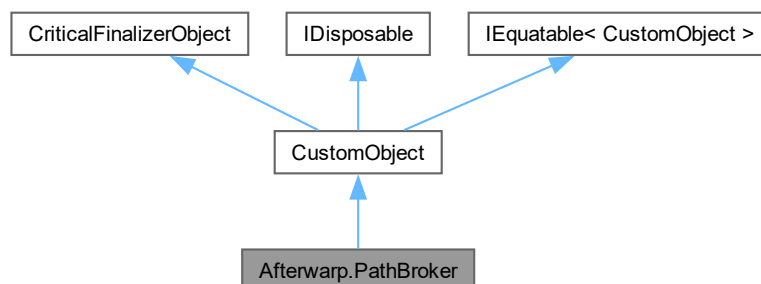
- [Afterwarp.Types.cs](#)

7.63 Afterwarp.PathBroker Class Reference

Inheritance diagram for `Afterwarp.PathBroker`:



Collaboration diagram for `Afterwarp.PathBroker`:



Public Member Functions

- [PathBroker](#) ()
Creates new instance of canvas buffer.
- void [Fill](#) ([CanvasBuffer](#) buffer, [PathElement](#)[] path, [ColorRect](#) colors, bool quality=true)
- void [Stroke](#) ([CanvasBuffer](#) buffer, [PathElement](#)[] path, float thickness, uint color, [PathJoint](#) joints=[PathJoint.None](#), [LineCaps](#) caps=[LineCaps.Butt](#), float miterLimit=1.0f, float smoothStep=1.0f)
Pre-renders a stroke for the given path.
- void [Reset](#) ()
Resets internal cache and releases any allocated memory.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Additional Inherited Members**Protected Member Functions inherited from [Afterwarp.CustomObject](#)**

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

7.63.1 Detailed Description

A container for vertices and indices defining one or more triangular meshes that can be suitable for rendering with a 2D canvas.

7.63.2 Constructor & Destructor Documentation**7.63.2.1 PathBroker()**

```
Afterwarp.PathBroker.PathBroker ( ) [inline]
```

Creates new instance of canvas buffer.

7.63.3 Member Function Documentation

7.63.3.1 Fill()

```
void Afterwarp.PathBroker.Fill (
    CanvasBuffer buffer,
    PathElement[] path,
    ColorRect colors,
    bool quality = true ) [inline]
```

Pre-renders a fill for the given path. "quality" parameter determines how the resulting contours are triangulated.

7.63.3.2 Reset()

```
void Afterwarp.PathBroker.Reset ( )
```

Resets internal cache and releases any allocated memory.

7.63.3.3 Stroke()

```
void Afterwarp.PathBroker.Stroke (
    CanvasBuffer buffer,
    PathElement[] path,
    float thickness,
    uint color,
    PathJoint joints = PathJoint::None,
    LineCaps caps = LineCaps::Butt,
    float miterLimit = 1::0f,
    float smoothStep = 1::0f ) [inline]
```

Pre-renders a stroke for the given path.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.64 Afterwarp.PathBuilder Class Reference

A container and helper module to facilitate creation of path elements.

Public Member Functions

- [PathBuilder](#) ()
Creates an empty container.
- void [Clear](#) ()
Removes all existing elements.
- void [ClosePath](#) ()
Closes current path.
- void [MoveTo](#) (Vector2 position, bool relative=false)
Moves position to the given coordinates. This basically means creating a new sub-path.
- void [LineTo](#) (Vector2 position, bool relative=false)
- void [CurveTo](#) (Vector2 controlPoint1, Vector2 controlPoint2, Vector2 position, bool relative=false)
- void [SmoothCurveTo](#) (Vector2 controlPoint2, Vector2 position, bool relative=false)
- void [QuadCurveTo](#) (Vector2 controlPoint, Vector2 position, bool relative=false)
- void [SmoothQuadCurveTo](#) (Vector2 position, bool relative=false)
- void [Rectangle](#) (RectF rectangle, bool clockwise=false, bool relative=false)
Produces a new rectangle path at the given coordinates.
- void [Circle](#) (Vector2 center, float radius, bool clockwise=false, bool relative=false)
Produces a new circle path at the given coordinates.
- void [Diamond](#) (Vector2 center, float radius, bool clockwise=false, bool relative=false)
Produces a new rotated quadrilateral (diamond-shaped) path at the given coordinates.
- void [Gear](#) (Vector2 center, float radius, float amplitude, int steps, int teeth, bool clockwise=false)
Produces a new gear-looking shape at the given coordinates.
- void [Flatness](#) (float flatness)

Properties

- [PathElement](#)[] [Elements](#) [get]
Returns the resulting array of path elements.

7.64.1 Detailed Description

A container and helper module to facilitate creation of path elements.

7.64.2 Constructor & Destructor Documentation**7.64.2.1 PathBuilder()**

```
Afterwarp.PathBuilder.PathBuilder ( ) [inline]
```

Creates an empty container.

7.64.3 Member Function Documentation**7.64.3.1 Circle()**

```
void Afterwarp.PathBuilder.Circle (
    Vector2 center,
    float radius,
    bool clockwise = false,
    bool relative = false ) [inline]
```

Produces a new circle path at the given coordinates.

7.64.3.2 Clear()

```
void Afterwarp.PathBuilder.Clear ( ) [inline]
```

Removes all existing elements.

7.64.3.3 ClosePath()

```
void Afterwarp.PathBuilder.ClosePath ( ) [inline]
```

Closes current path.

7.64.3.4 CurveTo()

```
void Afterwarp.PathBuilder.CurveTo (
    Vector2 controlPoint1,
    Vector2 controlPoint2,
    Vector2 position,
    bool relative = false ) [inline]
```

Produces a cubic Bezier curve between previous position, mirrored previous control point and the given control point and coordinates, adjusting current position accordingly.

7.64.3.5 Diamond()

```
void Afterwarp.PathBuilder.Diamond (
    Vector2 center,
    float radius,
    bool clockwise = false,
    bool relative = false ) [inline]
```

Produces a new rotated quadrilateral (diamond-shaped) path at the given coordinates.

7.64.3.6 Flatness()

```
void Afterwarp.PathBuilder.Flatness (
    float flatness ) [inline]
```

Changes flatness (distance tolerance for Bezier curves) for the path. This will take effect for the following commands.

7.64.3.7 Gear()

```
void Afterwarp.PathBuilder.Gear (
    Vector2 center,
    float radius,
    float amplitude,
    int steps,
    int teeth,
    bool clockwise = false ) [inline]
```

Produces a new gear-looking shape at the given coordinates.

7.64.3.8 LineTo()

```
void Afterwarp.PathBuilder.LineTo (
    Vector2 position,
    bool relative = false ) [inline]
```

Produces a line between previous position and the given coordinates, adjusting current position accordingly.

7.64.3.9 MoveTo()

```
void Afterwarp.PathBuilder.MoveTo (
    Vector2 position,
    bool relative = false ) [inline]
```

Moves position to the given coordinates. This basically means creating a new sub-path.

7.64.3.10 QuadCurveTo()

```
void Afterwarp.PathBuilder.QuadCurveTo (
    Vector2 controlPoint,
    Vector2 position,
    bool relative = false ) [inline]
```

Produces a quadratic Bezier curve between previous position, two control points and the given coordinates, adjusting current position accordingly.

7.64.3.11 Rectangle()

```
void Afterwarp.PathBuilder.Rectangle (
    RectF rectangle,
    bool clockwise = false,
    bool relative = false ) [inline]
```

Produces a new rectangle path at the given coordinates.

7.64.3.12 SmoothCurveTo()

```
void Afterwarp.PathBuilder.SmoothCurveTo (
    Vector2 controlPoint2,
    Vector2 position,
    bool relative = false ) [inline]
```

Produces a cubic Bezier curve between previous position, mirrored previous control point and the given control point and coordinates, adjusting current position accordingly.

7.64.3.13 SmoothQuadCurveTo()

```
void Afterwarp.PathBuilder.SmoothQuadCurveTo (
    Vector2 position,
    bool relative = false ) [inline]
```

Produces a quadratic Bezier curve between previous position, mirrored previous control point and the given coordinates, adjusting current position accordingly.

7.64.4 Property Documentation

7.64.4.1 Elements

`PathElement [] Afterwarp.PathBuilder.Elements [get]`

Returns the resulting array of path elements.

The documentation for this class was generated from the following file:

- [Afterwarp.Types.cs](#)

7.65 Afterwarp.PathCommand Struct Reference

Path commands and attribute bits.

Static Public Attributes

- const byte `Close` = 0x00
Close path, connecting previous position to the first position in the path.
- const byte `MoveTo` = 0x01
- const byte `LineTo` = 0x02
- const byte `CurveTo` = 0x03
- const byte `QuadCurveTo` = 0x04
- const byte `Flatness` = 0x1A
- const byte `ToleranceAngle` = 0x1B
- const byte `LimitCusp` = 0x1C
- const byte `Smooth` = 0x40
- const byte `Relative` = 0x80

7.65.1 Detailed Description

Path commands and attribute bits.

7.65.2 Member Data Documentation

7.65.2.1 Close

`const byte Afterwarp.PathCommand.Close = 0x00 [static]`

Close path, connecting previous position to the first position in the path.

7.65.2.2 CurveTo

`const byte Afterwarp.PathCommand.CurveTo = 0x03 [static]`

Draw a cubic Bezier curve from current position to the given coordinates and with the given control points. This is followed by three pairs of (x, y) coordinates, the first two being control points and last one being the destination position.

7.65.2.3 Flatness

```
const byte Afterwarp.PathCommand.Flatness = 0x1A [static]
```

Change "flatness" value used for converting Bezier curves into line segments. This is followed by a single value. Default value is "0.25". Note: if "Relative" flag is set, then "length" field is interpreted as a 16-bit floating-point (float16 or "half float"), and it is assumed that no additional values are following this command.

7.65.2.4 LimitCusp

```
const byte Afterwarp.PathCommand.LimitCusp = 0x1C [static]
```

Change cusp limit angle for Bezier curves to the given value. This is followed by a single value. A value of zero or less means cusp limit angle is not checked. Default value is zero. Note: if "Relative" flag is set, then "length" field is interpreted as a 16-bit floating-point (float16 or "half float"), and it is assumed that no additional values are following this command.

7.65.2.5 LineTo

```
const byte Afterwarp.PathCommand.LineTo = 0x02 [static]
```

Draw a line from current position to the given coordinates. This is followed by a pair of (x, y) coordinates.

7.65.2.6 MoveTo

```
const byte Afterwarp.PathCommand.MoveTo = 0x01 [static]
```

Move position to the given coordinates. This is followed by a pair of (x, y) coordinates.

7.65.2.7 QuadCurveTo

```
const byte Afterwarp.PathCommand.QuadCurveTo = 0x04 [static]
```

Draw a quadratic Bezier curve from current position to the given coordinates and with the given control point. This is followed by two pairs of (x, y) coordinates, the first one being a control point and last one being the destination position.

7.65.2.8 Relative

```
const byte Afterwarp.PathCommand.Relative = 0x80 [static]
```

A bit flag that indicates that command is relative to previous position. If this is the first command in the path, then it would be considered as absolute (or relative to zero, which would be the same). This flag has also special meaning when changing tolerances and angles (see above).

7.65.2.9 Smooth

```
const byte Afterwarp.PathCommand.Smooth = 0x40 [static]
```

A bit flag that indicates that command should take previous control point mirrored around previous position and use it as the first control point for the command that requires it. This only applies to Bezier curves. If this flag is specified, then there will be one less control point following the command declaration (so for "CurveTo", it'll be just one control point, and for "QuadCurveTo" there will be no control points).

7.65.2.10 ToleranceAngle

```
const byte Afterwarp.PathCommand.ToleranceAngle = 0x1B [static]
```

Change angle tolerance for Bezier curves to the given value. This is followed by a single value. A value of zero or less means angle tolerance is not checked. Default value is zero. A typical value for this would be something like $\pi / 7$. Note: if "Relative" flag is set, then "length" field is interpreted as a 16-bit floating-point (float16 or "half float"), and it is assumed that no additional values are following this command.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.66 Afterwarp.PathElement Struct Reference

A single element in path declaration.

Public Member Functions

- [PathElement](#) (float value=0.0f)
Creates path element equal to a given floating-point value.
- [PathElement](#) (byte command, ushort length=0)
Creates path element being a specific command with an optional length parameter.

Public Attributes

- float [Value](#)
A single 32-bit floating-point value of the path coordinates.

Properties

- byte [Command](#) [get, set]
Path command enumeration combined with one of its attribute bits.
- ushort [Length](#) [get, set]
Number of "Value" elements that follows this declaration.

7.66.1 Detailed Description

A single element in path declaration.

7.66.2 Constructor & Destructor Documentation

7.66.2.1 PathElement() [1/2]

```
Afterwarp.PathElement.PathElement (
    float value = 0::0f ) [inline]
```

Creates path element equal to a given floating-point value.

7.66.2.2 PathElement() [2/2]

```
Afterwarp.PathElement.PathElement (
    byte command,
    ushort length = 0 ) [inline]
```

Creates path element being a specific command with an optional length parameter.

7.66.3 Member Data Documentation

7.66.3.1 Value

```
float Afterwarp.PathElement.Value
```

A single 32-bit floating-point value of the path coordinates.

7.66.4 Property Documentation

7.66.4.1 Command

```
byte Afterwarp.PathElement.Command [get], [set]
```

Path command enumeration combined with one of its attribute bits.

7.66.4.2 Length

```
ushort Afterwarp.PathElement.Length [get], [set]
```

Number of "Value" elements that follows this declaration.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.67 Afterwarp.Point Struct Reference

2D integer point.

Public Member Functions

- [Point](#) (int value)
Creates 2D integer point with both coordinates set to the given value.
- [Point](#) (int x, int y)
Creates 2D integer point with the given values.
- [Point](#) (Vector2 vector)
Creates 2D integer point with the given values.
- override readonly bool [Equals](#) (object obj)
Tests whether current 2D integer vector is the same as a possibly another vector.
- override readonly int [GetHashCode](#) ()
Returns hash code for the 2D integer vector.
- readonly int [Dot](#) ([Point](#) point)
- readonly int [Cross](#) ([Point](#) point)
- readonly float [Distance](#) ([Point](#) point)
Calculates distance between current and given 2D integer vectors.
- readonly [Point Average](#) ([Point](#) point)
Calculates average between current and given 2D integer vectors.

Static Public Member Functions

- static [Point operator+](#) ([Point](#) point1, [Point](#) point2)
Adds two 2D integer vectors.
- static [Point operator-](#) ([Point](#) point1, [Point](#) point2)
Subtracts one 2D integer vector from another.
- static [Point operator*](#) ([Point](#) point1, [Point](#) point2)
Multiplies one 2D integer vector by another.
- static [Point operator/](#) ([Point](#) point1, [Point](#) point2)
Divides one 2D integer vector by another.
- static [Point operator*](#) ([Point](#) point, int theta)
Multiplies 2D integer vector by a coefficient.
- static [Point operator/](#) ([Point](#) point, int theta)
Divides 2D integer vector by a coefficient.
- static bool [operator==](#) ([Point](#) point1, [Point](#) point2)
Tests whether two points are equal.
- static bool [operator!=](#) ([Point](#) point1, [Point](#) point2)
Tests whether two points are not equal.

Public Attributes

- int [X](#)
Horizontal coordinate in 2D space.
- int [Y](#)
Vertical coordinate in 2D space.

Properties

- readonly [Point Transpose](#) [get]
Returns 2D integer vector with X and Y swapped.
- readonly bool [Empty](#) [get]
Tests whether 2D integer vector has both X and Y equal to zero.
- readonly float [Length](#) [get]
Returns length of current 2D integer vector.
- readonly float [Angle](#) [get]
Calculates angle (in radians) at which current vector is pointing at, in range of $[-\pi, \pi]$.
- static [Point Zero](#) [get]
Returns a 2D integer vector, where both coordinates are zero.
- static [Point One](#) [get]
Returns a 2D integer vector, where both coordinates are one.
- static [Point UnitX](#) [get]
Returns a 2D integer vector with values (1, 0).
- static [Point UnitY](#) [get]
Returns a 2D integer vector with values (0, 1).

7.67.1 Detailed Description

2D integer point.

7.67.2 Constructor & Destructor Documentation

7.67.2.1 Point() [1/3]

```
Afterwarp.Point.Point (
    int value ) [inline]
```

Creates 2D integer point with both coordinates set to the given value.

7.67.2.2 Point() [2/3]

```
Afterwarp.Point.Point (
    int x,
    int y ) [inline]
```

Creates 2D integer point with the given values.

7.67.2.3 Point() [3/3]

```
Afterwarp.Point.Point (
    Vector2 vector ) [inline]
```

Creates 2D integer point with the given values.

7.67.3 Member Function Documentation

7.67.3.1 Average()

```
readonly Point Afterwarp.Point.Average (
    Point point )
```

Calculates average between current and given 2D integer vectors.

7.67.3.2 Cross()

```
readonly int Afterwarp.Point.Cross (
    Point point )
```

Calculates a so-called "cross-product" between current and the given 2D integer vectors, or a geometric analog of thereof.

7.67.3.3 Distance()

```
readonly float Afterwarp.Point.Distance (
    Point point )
```

Calculates distance between current and given 2D integer vectors.

7.67.3.4 Dot()

```
readonly int Afterwarp.Point.Dot (
    Point point )
```

Calculates dot-product between current and the given 2D integer vectors. The dot-product is an indirect measure of angle between two vectors.

7.67.3.5 Equals()

```
override readonly bool Afterwarp.Point.Equals (
    object obj ) [inline]
```

Tests whether current 2D integer vector is the same as a possibly another vector.

7.67.3.6 GetHashCode()

```
override readonly int Afterwarp.Point.GetHashCode ( )
```

Returns hash code for the 2D integer vector.

7.67.3.7 operator!=(())

```
static bool Afterwarp.Point.operator!=(  
    Point point1,  
    Point point2 ) [inline], [static]
```

Tests whether two points are not equal.

7.67.3.8 operator*() [1/2]

```
static Point Afterwarp.Point.operator*(  
    Point point,  
    int theta ) [inline], [static]
```

Multiplies 2D integer vector by a coefficient.

7.67.3.9 operator*() [2/2]

```
static Point Afterwarp.Point.operator*(  
    Point point1,  
    Point point2 ) [inline], [static]
```

Multiplies one 2D integer vector by another.

7.67.3.10 operator+()

```
static Point Afterwarp.Point.operator+ (  
    Point point1,  
    Point point2 ) [inline], [static]
```

Adds two 2D integer vectors.

7.67.3.11 operator-()

```
static Point Afterwarp.Point.operator- (  
    Point point1,  
    Point point2 ) [inline], [static]
```

Subtracts one 2D integer vector from another.

7.67.3.12 operator/() [1/2]

```
static Point Afterwarp.Point.operator/ (  
    Point point,  
    int theta ) [inline], [static]
```

Divides 2D integer vector by a coefficient.

7.67.3.13 operator/() [2/2]

```
static Point Afterwarp.Point.operator/ (
    Point point1,
    Point point2 ) [inline], [static]
```

Divides one 2D integer vector by another.

7.67.3.14 operator==()

```
static bool Afterwarp.Point.operator== (
    Point point1,
    Point point2 ) [inline], [static]
```

Tests whether two points are equal.

7.67.4 Member Data Documentation

7.67.4.1 X

```
int Afterwarp.Point.X
```

Horizontal coordinate in 2D space.

7.67.4.2 Y

```
int Afterwarp.Point.Y
```

Vertical coordinate in 2D space.

7.67.5 Property Documentation

7.67.5.1 Angle

```
readonly float Afterwarp.Point.Angle [get]
```

Calculates angle (in radians) at which current vector is pointing at, in range of $[-\pi, \pi]$.

7.67.5.2 Empty

```
readonly bool Afterwarp.Point.Empty [get]
```

Tests whether 2D integer vector has both X and Y equal to zero.

7.67.5.3 Length

```
readonly float Afterwarp.Point.Length [get]
```

Returns length of current 2D integer vector.

7.67.5.4 One

```
Point Afterwarp.Point.One [static], [get]
```

Returns a 2D integer vector, where both coordinates are one.

7.67.5.5 Transpose

```
readonly Point Afterwarp.Point.Transpose [get]
```

Returns 2D integer vector with X and Y swapped.

7.67.5.6 UnitX

```
Point Afterwarp.Point.UnitX [static], [get]
```

Returns a 2D integer vector with values (1, 0).

7.67.5.7 UnitY

```
Point Afterwarp.Point.UnitY [static], [get]
```

Returns a 2D integer vector with values (0, 1).

7.67.5.8 Zero

```
Point Afterwarp.Point.Zero [static], [get]
```

Returns a 2D integer vector, where both coordinates are zero.

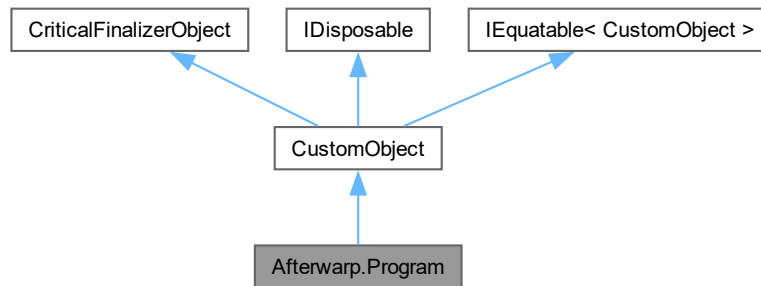
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

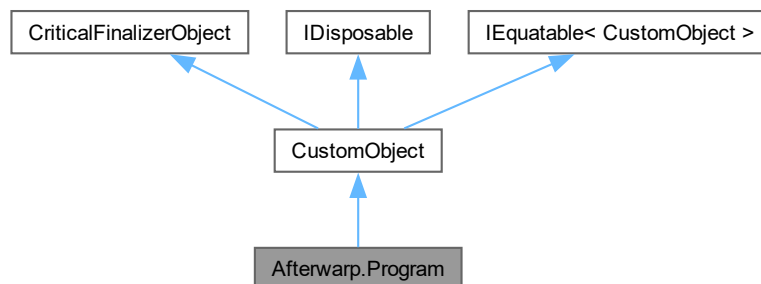
7.68 Afterwarp.Program Class Reference

Shader program that is typically executed on graphics hardware.

Inheritance diagram for Afterwarp.Program:



Collaboration diagram for Afterwarp.Program:



Public Member Functions

- `Program` (`Device` device, `VertexElement[]` vertexElements, `ProgramElement[]` programElements, `ProgramVariable[]` programVariables, `byte[]` shaderVertex, `byte[]` shaderHull, `byte[]` shaderDomain, `byte[]` shaderGeometry, `byte[]` shaderPixel)
Creates a new instance of shader program.
- void `Update` (int variableIndex, Array variableData)
- void `Update` (string variableName, Array variableData)
- void `Bind` (`Buffer` buffer, uint channel=0u, uint offset=0u)
- void `Unbind` (`Buffer` buffer, uint channel=0)
- *Removes association that was previously made with `Bind()` between the buffer and the given channel.*
- void `ResetBindings` ()
- *Resets any bindings that were previously made as if `Unbind()` would be called for each of them.*
- void `ResetCache` ()

- void [Commit](#) ()
- void [Begin](#) ()
Activates the shader program and prepares for rendering.
- void [End](#) ()
Deactivates the shader program, resets previously active input streams and bindings.
- void [Draw](#) ([PrimitiveTopology](#) topology, int vertexCount, int baseVertex=0)
Renders primitives with the given number of vertices.
- void [DrawIndexed](#) ([PrimitiveTopology](#) topology, int indexCount, int firstIndex=0, int baseVertex=0)
Renders indexed primitives with the given number of indices.
- void [DrawInstances](#) ([PrimitiveTopology](#) topology, int vertexCount, int instanceCount, int baseVertex=0)
Renders multiple number of instances of the given primitives.
- void [DrawInstancesIndexed](#) ([PrimitiveTopology](#) topology, int indexCount, int instanceCount, int firstIndex=0, int baseVertex=0)
Renders multiple number of instances of the given indexed primitives.
- void [SetPatchVertices](#) (int patchVertices)
Specifies number of vertices in each patch. This only applies when rendering patch topology.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with this program.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.68.1 Detailed Description

Shader program that is typically executed on graphics hardware.

7.68.2 Constructor & Destructor Documentation

7.68.2.1 Program()

```
Afterwarp.Program.Program (
    Device device,
    VertexElement[] vertexElements,
    ProgramElement[] programElements,
    ProgramVariable[] programVariables,
    byte[] shaderVertex,
    byte[] shaderHull,
    byte[] shaderDomain,
    byte[] shaderGeometry,
    byte[] shaderPixel ) [inline]
```

Creates a new instance of shader program.

7.68.3 Member Function Documentation

7.68.3.1 Begin()

```
void Afterwarp.Program.Begin ( ) [inline]
```

Activates the shader program and prepares for rendering.

7.68.3.2 Bind()

```
void Afterwarp.Program.Bind (
    Buffer buffer,
    uint channel = 0u,
    uint offset = 0u ) [inline]
```

Assigns a generic buffer (or a portion of starting from the given offset in bytes) to a particular channel. Vertex and index buffers are associated with input stream channels, whereas constant buffers are associated with shader program channels - each of these being unrelated and dependent on shader vertex and program element declarations. The current bindings can be considered valid until End(), ResetBindings() or ResetCache() is called.

7.68.3.3 Commit()

```
void Afterwarp.Program.Commit ( ) [inline]
```

Applies any binding changes to constant and/or vertex buffers, which usually occur when a draw call is issued, to occur immediately.

7.68.3.4 Draw()

```
void Afterwarp.Program.Draw (
    PrimitiveTopology topology,
    int vertexCount,
    int baseVertex = 0 ) [inline]
```

Renders primitives with the given number of vertices.

7.68.3.5 DrawIndexed()

```
void Afterwarp.Program.DrawIndexed (
    PrimitiveTopology topology,
    int indexCount,
    int firstIndex = 0,
    int baseVertex = 0 ) [inline]
```

Renders indexed primitives with the given number of indices.

7.68.3.6 DrawInstances()

```
void Afterwarp.Program.DrawInstances (
    PrimitiveTopology topology,
    int vertexCount,
    int instanceCount,
    int baseVertex = 0 ) [inline]
```

Renders multiple number of instances of the given primitives.

7.68.3.7 DrawInstancesIndexed()

```
void Afterwarp.Program.DrawInstancesIndexed (
    PrimitiveTopology topology,
    int indexCount,
    int instanceCount,
    int firstIndex = 0,
    int baseVertex = 0 ) [inline]
```

Renders multiple number of instances of the given indexed primitives.

7.68.3.8 End()

```
void Afterwarp.Program.End ( )
```

Deactivates the shader program, resets previously active input streams and bindings.

7.68.3.9 ResetBindings()

```
void Afterwarp.Program.ResetBindings ( )
```

Resets any bindings that were previously made as if Unbind() would be called for each of them.

7.68.3.10 ResetCache()

```
void Afterwarp.Program.ResetCache ( )
```

Resets any cache associated with vertex buffers. In case of OpenGL, this resets internal Vertex Array Object (VAO) cache.

7.68.3.11 SetPatchVertices()

```
void Afterwarp.Program.SetPatchVertices (
    int patchVertices ) [inline]
```

Specifies number of vertices in each patch. This only applies when rendering patch topology.

7.68.3.12 Unbind()

```
void Afterwarp.Program.Unbind (
    Buffer buffer,
    uint channel = 0 ) [inline]
```

Removes association that was previously made with Bind() between the buffer and the given channel.

7.68.3.13 Update() [1/2]

```
void Afterwarp.Program.Update (
    int variableIndex,
    Array variableData ) [inline]
```

Updates a portion of a specific variable that corresponds to the given index. This function should only be used inside Begin() and End() call block.

7.68.3.14 Update() [2/2]

```
void Afterwarp.Program.Update (
    string variableName,
    Array variableData ) [inline]
```

Updates a portion of a specific variable identified by its name (case-sensitive). This function should only be used inside Begin() and End() call block.

7.68.4 Property Documentation

7.68.4.1 Device

```
Device Afterwarp.Program.Device [get]
```

Returns device associated with this program.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.69 Afterwarp.ProgramElement Struct Reference

Structure that describes a single physical element of shader program.

Public Member Functions

- [ProgramElement](#) (string name, [ShaderElement](#) element, [ShaderType](#) shader, uint channel, int index, uint size)
Creates a program element structure with the given values.

Public Attributes

- byte[] [NameBytes](#)
UTF-8 encoded name of the element.
- [ShaderElement](#) Element
Type of shader element.
- [ShaderType](#) Shader
Shader to which this element belongs to.
- uint [Channel](#)
A virtual channel index, which will be associated with the appropriate buffer attached to that channel.
- int [Index](#)
- uint [Size](#)
Size of element in bytes.

Properties

- string [Name](#) [get, set]
Name of the element (e.g. variable, buffer or texture name).

7.69.1 Detailed Description

Structure that describes a single physical element of shader program.

7.69.2 Constructor & Destructor Documentation

7.69.2.1 ProgramElement()

```
Afterwarp.ProgramElement.ProgramElement (  
    string name,  
    ShaderElement element,  
    ShaderType shader,  
    uint channel,  
    int index,  
    uint size ) [inline]
```

Creates a program element structure with the given values.

7.69.3 Member Data Documentation

7.69.3.1 Channel

```
uint Afterwarp.ProgramElement.Channel
```

A virtual channel index, which will be associated with the appropriate buffer attached to that channel.

7.69.3.2 Element

```
ShaderElement Afterwarp.ProgramElement.Element
```

Type of shader element.

7.69.3.3 Index

```
int Afterwarp.ProgramElement.Index
```

Index of physical slot to which the element should be bound to. This can be either an actual binding point or a register number.

7.69.3.4 NameBytes

```
byte [] Afterwarp.ProgramElement.NameBytes
```

UTF-8 encoded name of the element.

7.69.3.5 Shader

```
ShaderType Afterwarp.ProgramElement.Shader
```

Shader to which this element belongs to.

7.69.3.6 Size

```
uint Afterwarp.ProgramElement.Size
```

Size of element in bytes.

7.69.4 Property Documentation

7.69.4.1 Name

```
string Afterwarp.ProgramElement.Name [get], [set]
```

Name of the element (e.g. variable, buffer or texture name).

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.70 Afterwarp.ProgramVariable Struct Reference

Structure that describes a single (uniform) variable in a shader program.

Public Member Functions

- [ProgramVariable](#) (string name, int index, [ShaderType](#) shader, [ElementFormat](#) format, int count, uint size, uint offset)

Creates a new program variable structure with the given values.

Public Attributes

- byte[] [NameBytes](#)
UTF-8 encoded name of the element.
- int [Index](#)
Index that uniquely identifies this variable.
- [ShaderType](#) [Shader](#)
Shader to which this element belongs to.
- [ElementFormat](#) [Format](#)
Data format of each value in this variable.
- int [Count](#)
- uint [Size](#)
Size of variable in bytes.
- uint [Offset](#)
Offset in a buffer where variable starts.

Properties

- string [Name](#) [get, set]
Name of the element (e.g. variable, buffer or texture name).

7.70.1 Detailed Description

Structure that describes a single (uniform) variable in a shader program.

7.70.2 Constructor & Destructor Documentation

7.70.2.1 ProgramVariable()

```
Afterwarp.ProgramVariable.ProgramVariable (
    string name,
    int index,
    ShaderType shader,
    ElementFormat format,
    int count,
    uint size,
    uint offset ) [inline]
```

Creates a new program variable structure with the given values.

7.70.3 Member Data Documentation

7.70.3.1 Count

```
int Afterwarp.ProgramVariable.Count
```

Number of values used by this variable. If "format" is "Undefined", this indicates number of bytes the element occupies.

7.70.3.2 Format

```
ElementFormat Afterwarp.ProgramVariable.Format
```

Data format of each value in this variable.

7.70.3.3 Index

```
int Afterwarp.ProgramVariable.Index
```

Index that uniquely identifies this variable.

7.70.3.4 NameBytes

```
byte [ ] Afterwarp.ProgramVariable.NameBytes
```

UTF-8 encoded name of the element.

7.70.3.5 Offset

```
uint Afterwarp.ProgramVariable.Offset
```

Offset in a buffer where variable starts.

7.70.3.6 Shader

```
ShaderType Afterwarp.ProgramVariable.Shader
```

Shader to which this element belongs to.

7.70.3.7 Size

```
uint Afterwarp.ProgramVariable.Size
```

Size of variable in bytes.

7.70.4 Property Documentation

7.70.4.1 Name

```
string Afterwarp.ProgramVariable.Name [get], [set]
```

Name of the element (e.g. variable, buffer or texture name).

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.71 Afterwarp.Widget.PropertyValue Struct Reference

A widget iteration value, consisting of a property name, type, behavior and its value.

Public Member Functions

- [PropertyValue](#) (string name, [WidgetPropertyType](#) type, [WidgetPropertyBehavior](#) behavior, object value)
Constructs a widget property iteration value.

Properties

- string [Name](#) [get]
Name of the property (case-insensitive).
- [WidgetPropertyType](#) [Type](#) [get]
Type of the property.
- [WidgetPropertyBehavior](#) [Behavior](#) [get]
Behavior of the property.
- object [Value](#) [get]
Value of the property.

7.71.1 Detailed Description

A widget iteration value, consisting of a property name, type, behavior and its value.

7.71.2 Constructor & Destructor Documentation

7.71.2.1 PropertyValue()

```
Afterwarp.Widget.PropertyValue.PropertyValue (  
    string name,  
    WidgetPropertyType type,  
    WidgetPropertyBehavior behavior,  
    object value ) [inline]
```

Constructs a widget property iteration value.

7.71.3 Property Documentation

7.71.3.1 Behavior

`WidgetPropertyBehavior` `Afterwarp.Widget.PropertyValue.Behavior` [get]

Behavior of the property.

7.71.3.2 Name

`string` `Afterwarp.Widget.PropertyValue.Name` [get]

Name of the property (case-insensitive).

7.71.3.3 Type

`WidgetPropertyType` `Afterwarp.Widget.PropertyValue.Type` [get]

Type of the property.

7.71.3.4 Value

`object` `Afterwarp.Widget.PropertyValue.Value` [get]

Value of the property.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.72 Afterwarp.Quad Struct Reference

Floating-point quadrilateral defined by four vertices in clockwise order starting from top/left.

Public Member Functions

- [Quad](#) (Vector2 topLeft, Vector2 topRight, Vector2 bottomRight, Vector2 bottomLeft)
Creates quadrilateral with individually specified vertices.
- [Quad](#) (float left, float top, float width, float height)
Creates quadrilateral with top/left position, width and height.
- [Quad](#) ([RectF](#) rect)
Creates quadrilateral from a given floating-point rectangle.
- [Quad](#) ([Rect](#) rect)
Creates quadrilateral from a given integer rectangle.
- readonly [Quad Offset](#) (Vector2 delta)
Displaces vertices of current quadrilateral by the specified offset.
- readonly [Quad Offset](#) (float deltaX, float deltaY)
Displaces vertices of current quadrilateral by the specified offset.
- readonly [Quad Scale](#) (float scale, bool centered=true)
- readonly [Quad Transform](#) (Matrix3x2 matrix)
Transforms (multiplies) vertices of current quadrilateral by the specified matrix.

Static Public Member Functions

- static [Quad Scaled](#) (float left, float top, float width, float height, float scale, bool centered=true)
- static [Quad Rotated](#) (Vector2 origin, Vector2 size, Vector2 center, float angle, float scale=1.0f)
- static [Quad Rotated](#) (Vector2 origin, Vector2 size, float angle, float scale=1.0f)
- static [Quad RotatedTopLeft](#) (Vector2 topLeft, Vector2 size, Vector2 center, float angle, float scale=1.0f)
- static [Quad SkewedHoriz](#) (Vector2 origin, Vector2 size, float angle)
- static [Quad SkewedVert](#) (Vector2 origin, Vector2 size, float angle)

Public Attributes

- Vector2 [TopLeft](#)
Top/left vertex position.
- Vector2 [TopRight](#)
Top/right vertex position.
- Vector2 [BottomRight](#)
Bottom/right vertex position.
- Vector2 [BottomLeft](#)
Bottom/left vertex position.

Properties

- readonly [Quad Mirror](#) [get]
- readonly [Quad Flip](#) [get]
- static [Quad Zero](#) [get]
Quadrilateral with all vertices set to zero.
- static [Quad Unity](#) [get]
Quadrilateral with vertices set at four corners set between (0, 0) and (1, 1).

7.72.1 Detailed Description

Floating-point quadrilateral defined by four vertices in clockwise order starting from top/left.

7.72.2 Constructor & Destructor Documentation

7.72.2.1 Quad() [1/4]

```
Afterwarp.Quad.Quad (
    Vector2 topLeft,
    Vector2 topRight,
    Vector2 bottomRight,
    Vector2 bottomLeft ) [inline]
```

Creates quadrilateral with individually specified vertices.

7.72.2.2 Quad() [2/4]

```
Afterwarp.Quad.Quad (
    float left,
    float top,
    float width,
    float height ) [inline]
```

Creates quadrilateral with top/left position, width and height.

7.72.2.3 Quad() [3/4]

```
Afterwarp.Quad.Quad (
    RectF rect ) [inline]
```

Creates quadrilateral from a given floating-point rectangle.

7.72.2.4 Quad() [4/4]

```
Afterwarp.Quad.Quad (
    Rect rect ) [inline]
```

Creates quadrilateral from a given integer rectangle.

7.72.3 Member Function Documentation**7.72.3.1 Offset()** [1/2]

```
readonly Quad Afterwarp.Quad.Offset (
    float deltaX,
    float deltaY )
```

Displaces vertices of current quadrilateral by the specified offset.

7.72.3.2 Offset() [2/2]

```
readonly Quad Afterwarp.Quad.Offset (
    Vector2 delta )
```

Displaces vertices of current quadrilateral by the specified offset.

7.72.3.3 Rotated() [1/2]

```
static Quad Afterwarp.Quad.Rotated (
    Vector2 origin,
    Vector2 size,
    float angle,
    float scale = 1.0f ) [inline], [static]
```

Creates quadrilateral specified by its dimensions. The rectangle is then rotated and scaled around its center and placed at the specified origin.

7.72.3.4 Rotated() [2/2]

```
static Quad Afterwarp.Quad.Rotated (
    Vector2 origin,
    Vector2 size,
    Vector2 center,
    float angle,
    float scale = 1::Of ) [inline], [static]
```

Creates quadrilateral specified by its dimensions. The rectangle is then rotated and scaled around the specified middle point (assumed to be inside rectangle's dimensions) and placed in center of the specified origin.

7.72.3.5 RotatedTopLeft()

```
static Quad Afterwarp.Quad.RotatedTopLeft (
    Vector2 topLeft,
    Vector2 size,
    Vector2 center,
    float angle,
    float scale = 1::Of ) [inline], [static]
```

Creates quadrilateral specified by top-left corner and size. The rectangle is then rotated and scaled around the specified middle point (assumed to be inside rectangle's dimensions) and placed in the center of the specified origin. The difference between this method and Rotated() is that the rotation does not preserve centering of the rectangle in case where middle point is not actually located in the middle.

7.72.3.6 Scale()

```
readonly Quad Afterwarp.Quad.Scale (
    float scale,
    bool centered = true ) [inline]
```

Rescales vertices of current quadrilateral by the provided coefficient, optionally centering them around zero origin.

7.72.3.7 Scaled()

```
static Quad Afterwarp.Quad.Scaled (
    float left,
    float top,
    float width,
    float height,
    float scale,
    bool centered = true ) [inline], [static]
```

Creates quadrilateral with the specified top left corner and the given dimensions, which are scaled by the provided coefficient.

7.72.3.8 SkewedHoriz()

```
static Quad Afterwarp.Quad.SkewedHoriz (
    Vector2 origin,
    Vector2 size,
    float angle ) [inline], [static]
```

Creates a horizontally skewed quadrilateral, oriented around its origin and with the given dimensions. A skew angle defines quadrilateral's top/left corner.

7.72.3.9 SkewedVert()

```
static Quad Afterwarp.Quad.SkewedVert (
    Vector2 origin,
    Vector2 size,
    float angle ) [inline], [static]
```

Creates a vertically skewed quadrilateral, oriented around its origin and with the given dimensions. A skew angle defines quadrilateral's top/left corner.

7.72.3.10 Transform()

```
readonly Quad Afterwarp.Quad.Transform (
    Matrix3x2 matrix )
```

Transforms (multiplies) vertices of current quadrilateral by the specified matrix.

7.72.4 Member Data Documentation

7.72.4.1 BottomLeft

```
Vector2 Afterwarp.Quad.BottomLeft
```

Bottom/left vertex position.

7.72.4.2 BottomRight

```
Vector2 Afterwarp.Quad.BottomRight
```

Bottom/right vertex position.

7.72.4.3 TopLeft

```
Vector2 Afterwarp.Quad.TopLeft
```

Top/left vertex position.

7.72.4.4 TopRight

`Vector2 Afterwarp.Quad.TopRight`

Top/right vertex position.

7.72.5 Property Documentation

7.72.5.1 Flip

readonly `Quad` `Afterwarp.Quad.Flip` [get]

Creates quadrilateral from current one but having top vertices exchanged with the bottom ones, effectively flipping it vertically.

7.72.5.2 Mirror

readonly `Quad` `Afterwarp.Quad.Mirror` [get]

Creates quadrilateral from current one but having left vertices exchanged with the right ones, effectively mirroring it horizontally.

7.72.5.3 Unity

`Quad` `Afterwarp.Quad.Unity` [static], [get]

Quadrilateral with vertices set at four corners set between (0, 0) and (1, 1).

7.72.5.4 Zero

`Quad` `Afterwarp.Quad.Zero` [static], [get]

Quadrilateral with all vertices set to zero.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.73 Afterwarp.RandomSequence Struct Reference

Pseudo-random number generator (PRNG) utility module.

Public Member Functions

- [RandomSequence](#) ()
Initializes random number generator state using local time.
- [RandomSequence](#) (ulong seed)
- int [Ranged](#) (int range)
Generates a random value in range [0, Range) using relatively linear distribution.

Properties

- uint [Raw](#) [get]
Generates next random number in the sequence from the specified context.
- ulong [Raw64](#) [get]
Returns 64-bit random number by generating two 32-bit numbers.
- float [Value](#) [get]
- double [ValueDouble](#) [get]
- float [Gaussian](#) [get]
Generates a random value using Gaussian distribution with mean 0 and standard deviation of 1.
- readonly ulong [State](#) [get]
Returns current PRNG state.

7.73.1 Detailed Description

Pseudo-random number generator (PRNG) utility module.

7.73.2 Constructor & Destructor Documentation

7.73.2.1 RandomSequence() [1/2]

```
Afterwarp.RandomSequence.RandomSequence ( ) [inline]
```

Initializes random number generator state using local time.

7.73.2.2 RandomSequence() [2/2]

```
Afterwarp.RandomSequence.RandomSequence (
    ulong seed )
```

Initializes random number generator state with a given seed to generate a reproducible sequence of random numbers.

7.73.3 Member Function Documentation

7.73.3.1 Ranged()

```
int Afterwarp.RandomSequence.Ranged (
    int range )
```

Generates a random value in range [0, Range) using relatively linear distribution.

7.73.4 Property Documentation

7.73.4.1 Gaussian

```
float Afterwarp.RandomSequence.Gaussian [get]
```

Generates a random value using Gaussian distribution with mean 0 and standard deviation of 1.

7.73.4.2 Raw

```
uint Afterwarp.RandomSequence.Raw [get]
```

Generates next random number in the sequence from the specified context.

7.73.4.3 Raw64

```
ulong Afterwarp.RandomSequence.Raw64 [get]
```

Returns 64-bit random number by generating two 32-bit numbers.

7.73.4.4 State

```
readonly ulong Afterwarp.RandomSequence.State [get]
```

Returns current PRNG state.

7.73.4.5 Value

```
float Afterwarp.RandomSequence.Value [get]
```

Generates a random value in range [0, 1) with 23 bits of precision, using relatively linear distribution.

7.73.4.6 ValueDouble

```
double Afterwarp.RandomSequence.ValueDouble [get]
```

Generates a random value in range [0, 1) with 53 bits of precision, using relatively linear distribution.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.74 Afterwarp.Ray Struct Reference

A structure that defines a ray in 3D space.

Public Member Functions

- [Ray](#) (Vector3 origin, Vector3 direction)
Creates a new ray with the given values.
- [Ray](#) (Vector2 position, Vector2 surfaceSize, Matrix4x4 viewInverse, Matrix4x4 projection)
Creates a new ray based on 2D position, screen size, inverse view and projection matrices.
- bool [IntersectTriangle](#) (Vector3 vertex1, Vector3 vertex2, Vector3 vertex3, out Vector2 intersection, out float distance, bool backFacing=true)
- bool [IntersectCubeVolume](#) (Matrix4x4 world, out float distance)
- bool [IntersectPlane](#) (Vector3 planePoint, Vector3 planeNormal, out Vector3 intersection, out float distance)

Public Attributes

- Vector3 [Origin](#)
Origin of the ray.
- Vector3 [Direction](#)
Direction of the ray.

7.74.1 Detailed Description

A structure that defines a ray in 3D space.

7.74.2 Constructor & Destructor Documentation

7.74.2.1 Ray() [1/2]

```
Afterwarp.Ray.Ray (
    Vector3 origin,
    Vector3 direction ) [inline]
```

Creates a new ray with the given values.

7.74.2.2 Ray() [2/2]

```
Afterwarp.Ray.Ray (
    Vector2 position,
    Vector2 surfaceSize,
    Matrix4x4 viewInverse,
    Matrix4x4 projection ) [inline]
```

Creates a new ray based on 2D position, screen size, inverse view and projection matrices.

7.74.3 Member Function Documentation

7.74.3.1 IntersectCubeVolume()

```
bool Afterwarp.Ray.IntersectCubeVolume (
    Matrix4x4 world,
    out float distance ) [inline]
```

Tests whether the current ray intersects with one of triangles of minimalistic cube volume, whose vertices are first transformed by the given world matrix, returning distance to intersection point.

7.74.3.2 IntersectPlane()

```
bool Afterwarp.Ray.IntersectPlane (
    Vector3 planePoint,
    Vector3 planeNormal,
    out Vector3 intersection,
    out float distance ) [inline]
```

Tests whether the current ray intersects with a plane specified by a point on the plane and its surface normal, returning the actual point of intersection and distance from ray's origin.

7.74.3.3 IntersectTriangle()

```
bool Afterwarp.Ray.IntersectTriangle (
    Vector3 vertex1,
    Vector3 vertex2,
    Vector3 vertex3,
    out Vector2 intersection,
    out float distance,
    bool backFacing = true ) [inline]
```

Tests whether the current ray intersects with the given front or back facing triangle and if so, calculates the resulting barycentric coordinates and distance from origin to triangle.

7.74.4 Member Data Documentation

7.74.4.1 Direction

```
Vector3 Afterwarp.Ray.Direction
```

Direction of the ray.

7.74.4.2 Origin

```
Vector3 Afterwarp.Ray.Origin
```

Origin of the ray.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.75 Afterwarp.Rect Struct Reference

Rectangle defined by integer top and left margins, width and height.

Public Member Functions

- [Rect](#) (int left, int top, int width, int height)
Creates rectangle by specifying its individual boundaries.
- readonly bool [Contains](#) (int x, int y)
Tests whether the specified point is contained within current rectangle.
- readonly bool [Contains](#) ([Rect](#) rect)
Tests whether a given rectangle is contained within current one.
- readonly bool [Overlaps](#) ([Rect](#) rect)
Tests whether a given rectangle overlaps with the current one.
- readonly [Rect Intersect](#) ([Rect](#) rect)
Calculates rectangle that results from intersection between current and the specified one.
- readonly [Rect Join](#) ([Rect](#) rect)
Calculates rectangle that results from union between current and the specified one.
- readonly [Rect Offset](#) (int deltaX, int deltaY)
Calculates displaced rectangle by certain offset.
- readonly [Rect Inflate](#) (int deltaX, int deltaY)
- override readonly bool [Equals](#) (object obj)
Tests whether current rectangle is the same as another rectangle.
- override readonly int [GetHashCode](#) ()
Returns hash code for the rectangle.

Static Public Member Functions

- static [Rect Bounds](#) (int left, int top, int right, int bottom)
Creates rectangle by specifying its individual boundaries.
- static bool [operator==](#) ([Rect](#) rect1, [Rect](#) rect2)
Tests whether two rectangles are equal.
- static bool [operator!=](#) ([Rect](#) rect1, [Rect](#) rect2)
Tests whether two rectangles are not equal.

Public Attributes

- int [Left](#)
Left position of the rectangle.
- int [Top](#)
Top position of the rectangle.
- int [Width](#)
Width of the rectangle.
- int [Height](#)
Height of the rectangle.

Properties

- readonly bool [Empty](#) [get]
Indicates whether the rectangle is empty, that is, having width or height of zero or less.
- int [Right](#) [get, set]
Right edge of the rectangle.
- int [Bottom](#) [get, set]
Bottom edge of the rectangle.
- static [Rect Zero](#) [get]
Returns an empty rectangle, where all elements are set to zero.

7.75.1 Detailed Description

Rectangle defined by integer top and left margins, width and height.

7.75.2 Constructor & Destructor Documentation

7.75.2.1 Rect()

```
Afterwarp.Rect.Rect (
    int left,
    int top,
    int width,
    int height ) [inline]
```

Creates rectangle by specifying its individual boundaries.

7.75.3 Member Function Documentation

7.75.3.1 Bounds()

```
static Rect Afterwarp.Rect.Bounds (
    int left,
    int top,
    int right,
    int bottom ) [inline], [static]
```

Creates rectangle by specifying its individual boundaries.

7.75.3.2 Contains() [1/2]

```
readonly bool Afterwarp.Rect.Contains (
    int x,
    int y )
```

Tests whether the specified point is contained within current rectangle.

7.75.3.3 Contains() [2/2]

```
readonly bool Afterwarp.Rect.Contains (
    Rect rect ) [inline]
```

Tests whether a given rectangle is contained within current one.

7.75.3.4 Equals()

```
override readonly bool Afterwarp.Rect.Equals (
    object obj ) [inline]
```

Tests whether current rectangle is the same as another rectangle.

7.75.3.5 GetHashCode()

```
override readonly int Afterwarp.Rect.GetHashCode ( )
```

Returns hash code for the rectangle.

7.75.3.6 Inflate()

```
readonly Rect Afterwarp.Rect.Inflate (
    int deltaX,
    int deltaY )
```

Returns rectangle with left and top decremented, while right and bottom incremented by a given offset.

7.75.3.7 Intersect()

```
readonly Rect Afterwarp.Rect.Intersect (
    Rect rect ) [inline]
```

Calculates rectangle that results from intersection between current and the specified one.

7.75.3.8 Join()

```
readonly Rect Afterwarp.Rect.Join (
    Rect rect ) [inline]
```

Calculates rectangle that results from union between current and the specified one.

7.75.3.9 Offset()

```
readonly Rect Afterwarp.Rect.Offset (
    int deltaX,
    int deltaY )
```

Calculates displaced rectangle by certain offset.

7.75.3.10 operator"!=()"

```
static bool Afterwarp.Rect.operator!= (
    Rect rect1,
    Rect rect2 ) [inline], [static]
```

Tests whether two rectangles are not equal.

7.75.3.11 operator==()

```
static bool Afterwarp.Rect.operator== (
    Rect rect1,
    Rect rect2 ) [inline], [static]
```

Tests whether two rectangles are equal.

7.75.3.12 Overlaps()

```
readonly bool Afterwarp.Rect.Overlaps (
    Rect rect ) [inline]
```

Tests whether a given rectangle overlaps with the current one.

7.75.4 Member Data Documentation

7.75.4.1 Height

```
int Afterwarp.Rect.Height
```

Height of the rectangle.

7.75.4.2 Left

```
int Afterwarp.Rect.Left
```

Left position of the rectangle.

7.75.4.3 Top

```
int Afterwarp.Rect.Top
```

Top position of the rectangle.

7.75.4.4 Width

```
int Afterwarp.Rect.Width
```

Width of the rectangle.

7.75.5 Property Documentation

7.75.5.1 Bottom

```
int Afterwarp.Rect.Bottom [get], [set]
```

Bottom edge of the rectangle.

7.75.5.2 Empty

`readonly bool Afterwarp.Rect.Empty [get]`

Indicates whether the rectangle is empty, that is, having width or height of zero or less.

7.75.5.3 Right

`int Afterwarp.Rect.Right [get], [set]`

Right edge of the rectangle.

7.75.5.4 Zero

`Rect Afterwarp.Rect.Zero [static], [get]`

Returns an empty rectangle, where all elements are set to zero.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.76 Afterwarp.RectF Struct Reference

Rectangle defined by floating-point top and left margins, width and height.

Public Member Functions

- [RectF](#) (float left, float top, float width, float height)
Creates rectangle by specifying its individual boundaries.
- [RectF](#) (Vector2 origin, Vector2 size)
Creates rectangle by specifying its origin and size.
- [RectF](#) ([Rect](#) rect)
Creates rectangle with floating-point coordinates from an integer one.
- `readonly bool` [Contains](#) (float x, float y)
Tests whether the specified point is contained within current rectangle.
- `readonly bool` [Contains](#) (Vector2 point)
Tests whether the specified point is contained within current rectangle.
- `readonly bool` [Contains](#) ([RectF](#) rect)
Tests whether a given rectangle is contained within current one.
- `readonly bool` [Overlaps](#) ([RectF](#) rect)
Tests whether a given rectangle overlaps with the current one.
- `readonly` [RectF Intersect](#) ([RectF](#) rect)
Calculates rectangle that results from intersection between current and the specified one.
- `readonly` [RectF Join](#) ([RectF](#) rect)
Calculates rectangle that results from union between current and the specified one.
- `readonly` [RectF Offset](#) (float deltaX, float deltaY)
Calculates displaced rectangle by certain offset.
- `readonly` [RectF Offset](#) (Vector2 delta)
Calculates displaced rectangle by certain offset.
- `readonly` [RectF Inflate](#) (float deltaX, float deltaY)
- `readonly` [RectF Inflate](#) (Vector2 delta)

Static Public Member Functions

- static [RectF Bounds](#) (float left, float top, float right, float bottom)
Creates rectangle by specifying its individual boundaries.
- static [RectF Bounds](#) (Vector2 topLeft, Vector2 bottomRight)
Creates rectangle by specifying its top/left and bottom/right boundaries.

Public Attributes

- float [Left](#)
Left position of the rectangle.
- float [Top](#)
Top position of the rectangle.
- float [Width](#)
Width of the rectangle.
- float [Height](#)
Height of the rectangle.

Properties

- readonly bool [Empty](#) [get]
Indicates whether the rectangle is empty, that is, having width or height of zero or less.
- float [Right](#) [get, set]
Right edge of the rectangle.
- float [Bottom](#) [get, set]
Bottom edge of the rectangle.
- readonly Vector2 [Size](#) [get]
Returns size of the rectangle.
- Vector2 [TopLeft](#) [get, set]
Top and left corner of the rectangle.
- Vector2 [TopRight](#) [get, set]
Top and right corner of the rectangle.
- Vector2 [BottomRight](#) [get, set]
Bottom and right corner of the rectangle.
- Vector2 [BottomLeft](#) [get, set]
Bottom and left corner of the rectangle.
- static [RectF Zero](#) [get]
Returns an empty rectangle, where all elements are set to zero.
- static [RectF Unity](#) [get]
Returns a rectangle positioned at zero origin with width and height set to one.

7.76.1 Detailed Description

Rectangle defined by floating-point top and left margins, width and height.

7.76.2 Constructor & Destructor Documentation

7.76.2.1 RectF() [1/3]

```
Afterwarp.RectF.RectF (
    float left,
    float top,
    float width,
    float height ) [inline]
```

Creates rectangle by specifying its individual boundaries.

7.76.2.2 RectF() [2/3]

```
Afterwarp.RectF.RectF (
    Vector2 origin,
    Vector2 size ) [inline]
```

Creates rectangle by specifying its origin and size.

7.76.2.3 RectF() [3/3]

```
Afterwarp.RectF.RectF (
    Rect rect ) [inline]
```

Creates rectangle with floating-point coordinates from an integer one.

7.76.3 Member Function Documentation

7.76.3.1 Bounds() [1/2]

```
static RectF Afterwarp.RectF.Bounds (
    float left,
    float top,
    float right,
    float bottom ) [inline], [static]
```

Creates rectangle by specifying its individual boundaries.

7.76.3.2 Bounds() [2/2]

```
static RectF Afterwarp.RectF.Bounds (
    Vector2 topLeft,
    Vector2 bottomRight ) [inline], [static]
```

Creates rectangle by specifying its top/left and bottom/right boundaries.

7.76.3.3 Contains() [1/3]

```
readonly bool Afterwarp.RectF.Contains (
    float x,
    float y )
```

Tests whether the specified point is contained within current rectangle.

7.76.3.4 Contains() [2/3]

```
readonly bool Afterwarp.RectF.Contains (
    RectF rect ) [inline]
```

Tests whether a given rectangle is contained within current one.

7.76.3.5 Contains() [3/3]

```
readonly bool Afterwarp.RectF.Contains (
    Vector2 point )
```

Tests whether the specified point is contained within current rectangle.

7.76.3.6 Inflate() [1/2]

```
readonly RectF Afterwarp.RectF.Inflate (
    float deltaX,
    float deltaY ) [inline]
```

Returns rectangle with left and top decremented, while right and bottom incremented by a given offset.

7.76.3.7 Inflate() [2/2]

```
readonly RectF Afterwarp.RectF.Inflate (
    Vector2 delta )
```

Returns rectangle with left and top decremented, while right and bottom incremented by a given offset.

7.76.3.8 Intersect()

```
readonly RectF Afterwarp.RectF.Intersect (
    RectF rect ) [inline]
```

Calculates rectangle that results from intersection between current and the specified one.

7.76.3.9 Join()

```
readonly RectF Afterwarp.RectF.Join (
    RectF rect ) [inline]
```

Calculates rectangle that results from union between current and the specified one.

7.76.3.10 Offset() [1/2]

```
readonly RectF Afterwarp.RectF.Offset (
    float deltaX,
    float deltaY ) [inline]
```

Calculates displaced rectangle by certain offset.

7.76.3.11 Offset() [2/2]

```
readonly RectF Afterwarp.RectF.Offset (
    Vector2 delta )
```

Calculates displaced rectangle by certain offset.

7.76.3.12 Overlaps()

```
readonly bool Afterwarp.RectF.Overlaps (
    RectF rect ) [inline]
```

Tests whether a given rectangle overlaps with the current one.

7.76.4 Member Data Documentation

7.76.4.1 Height

```
float Afterwarp.RectF.Height
```

Height of the rectangle.

7.76.4.2 Left

```
float Afterwarp.RectF.Left
```

Left position of the rectangle.

7.76.4.3 Top

```
float Afterwarp.RectF.Top
```

Top position of the rectangle.

7.76.4.4 Width

```
float Afterwarp.RectF.Width
```

Width of the rectangle.

7.76.5 Property Documentation

7.76.5.1 Bottom

```
float Afterwarp.RectF.Bottom [get], [set]
```

Bottom edge of the rectangle.

7.76.5.2 BottomLeft

```
Vector2 Afterwarp.RectF.BottomLeft [get], [set]
```

Bottom and left corner of the rectangle.

7.76.5.3 BottomRight

```
Vector2 Afterwarp.RectF.BottomRight [get], [set]
```

Bottom and right corner of the rectangle.

7.76.5.4 Empty

```
readonly bool Afterwarp.RectF.Empty [get]
```

Indicates whether the rectangle is empty, that is, having width or height of zero or less.

7.76.5.5 Right

```
float Afterwarp.RectF.Right [get], [set]
```

Right edge of the rectangle.

7.76.5.6 Size

```
readonly Vector2 Afterwarp.RectF.Size [get]
```

Returns size of the rectangle.

7.76.5.7 TopLeft

```
Vector2 Afterwarp.RectF.TopLeft [get], [set]
```

Top and left corner of the rectangle.

7.76.5.8 TopRight

`Vector2 Afterwarp.RectF.TopRight [get], [set]`

Top and right corner of the rectangle.

7.76.5.9 Unity

`RectF Afterwarp.RectF.Unity [static], [get]`

Returns a rectangle positioned at zero origin with width and height set to one.

7.76.5.10 Zero

`RectF Afterwarp.RectF.Zero [static], [get]`

Returns an empty rectangle, where all elements are set to zero.

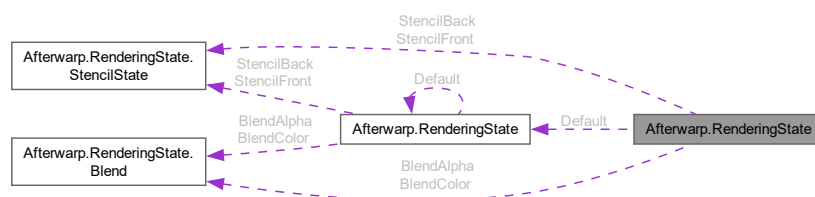
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.77 Afterwarp.RenderingState Struct Reference

Parameters that define depth, stencil, rasterizer and blending operations.

Collaboration diagram for Afterwarp.RenderingState:



Classes

- struct [Blend](#)
Blending parameters that are specific to a certain component, or a portion of color.
- struct [State](#)
State flags that can be combined together to form a rendering operation state.
- struct [StencilState](#)
Stencil state that is independent for each front and back facing.

Public Member Functions

- [RenderingState](#) (uint states, [TriangleFace](#) cullFace, [ComparisonFunc](#) depthFunc, float depthBias, float slopeDepthBias, float clampDepthBias, byte stencilRefValue, byte stencilRefMask, byte stencilWriteMask, Vector4 blendConstant, [StencilState](#) stencilFront, [StencilState](#) stencilBack, [Blend](#) blendColor, [Blend](#) blendAlpha)

Creates a rendering state with the given values.

Public Attributes

- uint [States](#)
Combination of state flags that define rendering operation.
- [TriangleFace](#) [CullFace](#)
How triangle faces should be culled.
- [ComparisonFunc](#) [DepthFunc](#)
Depth buffer comparison function.
- float [DepthBias](#)
Constant offset added to depth to alleviate z-fighting issues.
- float [SlopeDepthBias](#)
Slope coefficient used for relative depth when calculating depth bias.
- float [ClampDepthBias](#)
Value to clamp depth bias, preventing artifacts on very high slopes.
- byte [StencilRefValue](#)
Reference value for the stencil test.
- byte [StencilRefMask](#)
Mask that is applied to reference value and stored value after the test.
- byte [StencilWriteMask](#)
Mask that is used to enable or disable writing of individual bits to stencil buffer.
- Vector4 [BlendConstant](#)
Custom alpha-blending constant.
- [StencilState](#) [StencilFront](#)
Stencil function and operation that are performed for front-facing triangles.
- [StencilState](#) [StencilBack](#)
Stencil function and operation that are performed for back-facing triangles.
- [Blend](#) [BlendColor](#)
Blending parameters that should be applied to color portion of components.
- [Blend](#) [BlendAlpha](#)
Blending parameters that should be applied to alpha portion of components.

Static Public Attributes

- static readonly [RenderingState](#) [Default](#) = _default
Default rendering state.

7.77.1 Detailed Description

Parameters that define depth, stencil, rasterizer and blending operations.

7.77.2 Constructor & Destructor Documentation

7.77.2.1 RenderingState()

```
Afterwarp.RenderingState.RenderingState (
    uint states,
    TriangleFace cullFace,
    ComparisonFunc depthFunc,
    float depthBias,
    float slopeDepthBias,
    float clampDepthBias,
    byte stencilRefValue,
    byte stencilRefMask,
    byte stencilWriteMask,
    Vector4 blendConstant,
    StencilState stencilFront,
    StencilState stencilBack,
    Blend blendColor,
    Blend blendAlpha ) [inline]
```

Creates a rendering state with the given values.

7.77.3 Member Data Documentation

7.77.3.1 BlendAlpha

`Blend` Afterwarp.RenderingState.BlendAlpha

Blending parameters that should be applied to alpha portion of components.

7.77.3.2 BlendColor

`Blend` Afterwarp.RenderingState.BlendColor

Blending parameters that should be applied to color portion of components.

7.77.3.3 BlendConstant

`Vector4` Afterwarp.RenderingState.BlendConstant

Custom alpha-blending constant.

7.77.3.4 ClampDepthBias

`float` Afterwarp.RenderingState.ClampDepthBias

Value to clamp depth bias, preventing artifacts on very high slopes.

7.77.3.5 CullFace

`TriangleFace` Afterwarp.RenderingState.CullFace

How triangle faces should be culled.

7.77.3.6 Default

readonly `RenderingState` Afterwarp.RenderingState.Default = `_default` [static]

Default rendering state.

7.77.3.7 DepthBias

`float` Afterwarp.RenderingState.DepthBias

Constant offset added to depth to alleviate z-fighting issues.

7.77.3.8 DepthFunc

`ComparisonFunc` Afterwarp.RenderingState.DepthFunc

Depth buffer comparison function.

7.77.3.9 SlopeDepthBias

`float` Afterwarp.RenderingState.SlopeDepthBias

Slope coefficient used for relative depth when calculating depth bias.

7.77.3.10 States

`uint` Afterwarp.RenderingState.States

Combination of state flags that define rendering operation.

7.77.3.11 StencilBack

`StencilState` Afterwarp.RenderingState.StencilBack

Stencil function and operation that are performed for back-facing triangles.

7.77.3.12 StencilFront

`StencilState` Afterwarp.RenderingState.StencilFront

Stencil function and operation that are performed for front-facing triangles.

7.77.3.13 StencilRefMask

```
byte Afterwarp.RenderingState.StencilRefMask
```

Mask that is applied to reference value and stored value after the test.

7.77.3.14 StencilRefValue

```
byte Afterwarp.RenderingState.StencilRefValue
```

Reference value for the stencil test.

7.77.3.15 StencilWriteMask

```
byte Afterwarp.RenderingState.StencilWriteMask
```

Mask that is used to enable or disable writing of individual bits to stencil buffer.

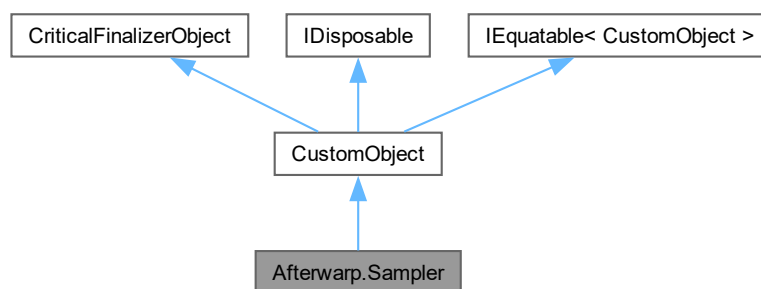
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

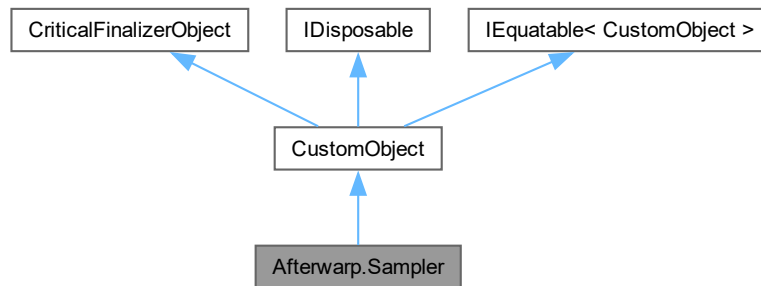
7.78 Afterwarp.Sampler Class Reference

Sampler object that defines how texture is read by the shaders.

Inheritance diagram for Afterwarp.Sampler:



Collaboration diagram for Afterwarp.Sampler:



Public Member Functions

- [Sampler](#) ([Device](#) device, [SamplerState?](#) samplerState=null)
Creates a new instance of sampler object.
- void [Bind](#) (uint channel)
Binds sampler object to a particular channel.
- void [Unbind](#) (uint channel)
Unbinds sampler object from the particular channel.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device associated with this sampler object.
- [SamplerState](#) [State](#) [get, set]
Current sampler object parameters.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.78.1 Detailed Description

Sampler object that defines how texture is read by the shaders.

7.78.2 Constructor & Destructor Documentation

7.78.2.1 Sampler()

```
Afterwarp.Sampler.Sampler (
    Device device,
    SamplerState? samplerState = null ) [inline]
```

Creates a new instance of sampler object.

7.78.3 Member Function Documentation

7.78.3.1 Bind()

```
void Afterwarp.Sampler.Bind (
    uint channel ) [inline]
```

Binds sampler object to a particular channel.

7.78.3.2 Unbind()

```
void Afterwarp.Sampler.Unbind (
    uint channel )
```

Unbinds sampler object from the particular channel.

7.78.4 Property Documentation

7.78.4.1 Device

```
Device Afterwarp.Sampler.Device [get]
```

Returns device associated with this sampler object.

7.78.4.2 State

`SamplerState` `Afterwarp.Sampler.State` `[get]`, `[set]`

Current sampler object parameters.

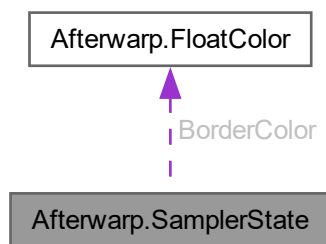
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.79 Afterwarp.SamplerState Struct Reference

Sampler parameters that are associated with a particular texture unit.

Collaboration diagram for `Afterwarp.SamplerState`:



Public Member Functions

- `SamplerState` (`TextureFilter` filterMin, `TextureFilter` filterMag, `TextureFilter` filterMip, `TextureAddress` addressU, `TextureAddress` addressV, `TextureAddress` addressW, `FloatColor` borderColor, int anisotropy, float minLOD, float maxLOD, float biasLOD, `ComparisonFunc` compareFunc, bool compareToRef)

Creates new sampler state with the given values.

Public Attributes

- `TextureFilter` FilterMin
Minification texture filtering.
- `TextureFilter` FilterMag
Magnification texture filtering.
- `TextureFilter` FilterMip
Mipmapping texture filtering.
- `TextureAddress` AddressU
Texture coordinate addressing for U coordinate.
- `TextureAddress` AddressV
Texture coordinate addressing for V coordinate.

- [TextureAddress AddressW](#)
Texture coordinate addressing for W coordinate.
- [FloatColor BorderColor](#)
Special color constant to be used for TextureAddress::Border addressing mode.
- [int Anisotropy](#)
Total number of samples to be used with anisotropic filtering.
- [float MinLOD](#)
Minimum level of detail for texture mipmap range.
- [float MaxLOD](#)
Maximum level of detail for texture mipmap range.
- [float BiasLOD](#)
Offset from the calculated mipmap level.
- [ComparisonFunc CompareFunc](#)
Comparison function for texture values.
- [bool CompareToRef](#)
Indicates whether comparison should be made against a particular reference value or simply fetched.

Properties

- [static SamplerState Default](#) [get]
Returns sampler state with default values.

7.79.1 Detailed Description

Sampler parameters that are associated with a particular texture unit.

7.79.2 Constructor & Destructor Documentation

7.79.2.1 SamplerState()

```
Afterwarp.SamplerState.SamplerState (
    TextureFilter filterMin,
    TextureFilter filterMag,
    TextureFilter filterMip,
    TextureAddress addressU,
    TextureAddress addressV,
    TextureAddress addressW,
    FloatColor borderColor,
    int anisotropy,
    float minLOD,
    float maxLOD,
    float biasLOD,
    ComparisonFunc compareFunc,
    bool compareToRef ) [inline]
```

Creates new sampler state with the given values.

7.79.3 Member Data Documentation

7.79.3.1 AddressU

`TextureAddress` Afterwarp.SamplerState.AddressU

Texture coordinate addressing for U coordinate.

7.79.3.2 AddressV

`TextureAddress` Afterwarp.SamplerState.AddressV

Texture coordinate addressing for V coordinate.

7.79.3.3 AddressW

`TextureAddress` Afterwarp.SamplerState.AddressW

Texture coordinate addressing for W coordinate.

7.79.3.4 Anisotropy

`int` Afterwarp.SamplerState.Anisotropy

Total number of samples to be used with anisotropic filtering.

7.79.3.5 BiasLOD

`float` Afterwarp.SamplerState.BiasLOD

Offset from the calculated mipmap level.

7.79.3.6 BorderColor

`FloatColor` Afterwarp.SamplerState.BorderColor

Special color constant to be used for `TextureAddress::Border` addressing mode.

7.79.3.7 CompareFunc

`ComparisonFunc` Afterwarp.SamplerState.CompareFunc

Comparison function for texture values.

7.79.3.8 CompareToRef

```
bool Afterwarp.SamplerState.CompareToRef
```

Indicates whether comparison should be made against a particular reference value or simply fetched.

7.79.3.9 FilterMag

```
TextureFilter Afterwarp.SamplerState.FilterMag
```

Magnification texture filtering.

7.79.3.10 FilterMin

```
TextureFilter Afterwarp.SamplerState.FilterMin
```

Minification texture filtering.

7.79.3.11 FilterMip

```
TextureFilter Afterwarp.SamplerState.FilterMip
```

Mipmapping texture filtering.

7.79.3.12 MaxLOD

```
float Afterwarp.SamplerState.MaxLOD
```

Maximum level of detail for texture mipmap range.

7.79.3.13 MinLOD

```
float Afterwarp.SamplerState.MinLOD
```

Minimum level of detail for texture mipmap range.

7.79.4 Property Documentation

7.79.4.1 Default

```
SamplerState Afterwarp.SamplerState.Default [static], [get]
```

Returns sampler state with default values.

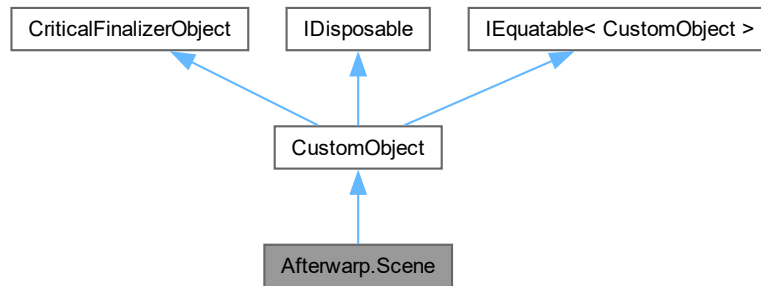
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

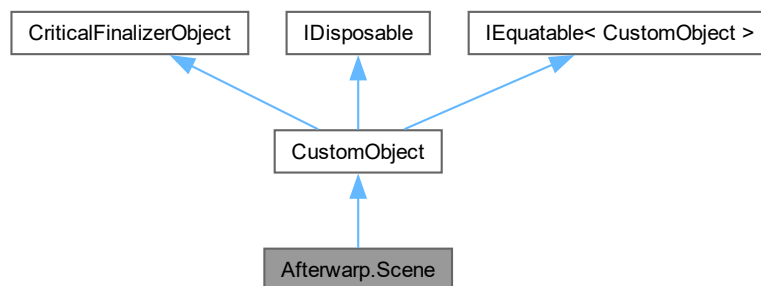
7.80 Afterwarp.Scene Class Reference

A module for rendering 3D scenes.

Inheritance diagram for Afterwarp.Scene:



Collaboration diagram for Afterwarp.Scene:



Classes

- struct [Attribute](#)
Cumulative attributes that define rendering behavior of the scene.

Public Member Functions

- [VertexElement\[\] GetVertexElements](#) (int index)
Returns items in vertex element declaration associated with the given index.
- void [SetVertexElements](#) (int index, [VertexElement\[\]](#) elements)
Changes vertex element declaration associated with the given index.
- void [SetVertexElementsFromTextModeller](#) (int index)
- void [Instances](#) ([Matrix4x4\[\]](#) transforms, [FloatColor\[\]](#) colors, int count)

- void **Begin** ()
Activates the appropriate shader program and begins rendering the scene.
- void **End** ()
Finishes rendering the scene and deactivates previously activated shader program.
- void **ResetCache** ()
Clears cached buffers and resources in the scene.
- **Sampler GetSampler** (**SceneSamplerType** type)
Returns sampler of the given type associated with the scene.
- **Texture GetTexture** (**SceneTextureType** type)
Returns texture of the given type associated with the scene.
- void **SetTexture** (**SceneTextureType** type, **Texture** texture)
Changes texture of the given type associated with the scene.
- bool **TryPrepare** ()
- void **Prepare** ()

Public Member Functions inherited from **Afterwarp.CustomObject**

- void **Dispose** ()
- bool **Equals** (**CustomObject** other)
- override bool **Equals** (object obj)
- override int **GetHashCode** ()

Static Public Member Functions

- static **Scene CreateDepthNormals** (**Device** device)
Creates a new instance of 3D depths/normals rendering scene.
- static **Scene CreateModeling** (**Device** device)
Creates a new instance of 3D modeling rendering scene.

Properties

- **Device Device** [get]
Returns device associated with the scene.
- int **ActiveVertexElements** [get, set]
Returns or changes currently active index of vertex elements declaration.
- uint **Attributes** [get, set]
Returns or changes attributes that define the rendering behavior of the scene.
- Matrix4x4 **World** [get, set]
Returns or changes "World" transformation matrix.
- Matrix4x4 **View** [get, set]
Returns or changes "View" transformation matrix.
- Matrix4x4 **Projection** [get, set]
Returns or changes "Projection" transformation matrix.
- bool **Rendering** [get]
Indicates that the rendering is currently taking place (inside "Begin" / "End" block).
- int **InstancesCount** [get]
Returns maximum number of supported instances.
- **ObjectMaterial Material** [get, set]
Returns or changes scene object's material properties.

- Vector4 [ToneMappingCoefficients](#) [get, set]
- [ParallaxMappingParameters](#) [ParallaxMapping](#) [get, set]
Returns or changes parameters that define Parallax Mapping technique is performed.
- [SceneLights](#) [Lights](#) [get, set]
- [ShadowCastingAtlas](#) [Atlas](#) [get, set]
- [TextureCabinet](#) [TextureCabinet](#) [get, set]
- [Program](#) [Program](#) [get]
Returns currently active shader program handle.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.80.1 Detailed Description

A module for rendering 3D scenes.

7.80.2 Member Function Documentation

7.80.2.1 Begin()

```
void Afterwarp.Scene.Begin ( ) [inline]
```

Activates the appropriate shader program and begins rendering the scene.

7.80.2.2 CreateDepthNormals()

```
static Scene Afterwarp.Scene.CreateDepthNormals (
    Device device ) [inline], [static]
```

Creates a new instance of 3D depths/normals rendering scene.

7.80.2.3 CreateModeling()

```
static Scene Afterwarp.Scene.CreateModeling (
    Device device ) [inline], [static]
```

Creates a new instance of 3D modeling rendering scene.

7.80.2.4 End()

```
void Afterwarp.Scene.End ( )
```

Finishes rendering the scene and deactivates previously activated shader program.

7.80.2.5 GetSampler()

```
Sampler Afterwarp.Scene.GetSampler (
    SceneSamplerType type ) [inline]
```

Returns sampler of the given type associated with the scene.

7.80.2.6 GetTexture()

```
Texture Afterwarp.Scene.GetTexture (
    SceneTextureType type ) [inline]
```

Returns texture of the given type associated with the scene.

7.80.2.7 GetVertexElements()

```
VertexElement[] Afterwarp.Scene.GetVertexElements (
    int index ) [inline]
```

Returns items in vertex element declaration associated with the given index.

7.80.2.8 Instances()

```
void Afterwarp.Scene.Instances (
    Matrix4x4[] transforms,
    FloatColor[] colors,
    int count ) [inline]
```

Supplies instance information to the pipeline. The maximum number of elements must not exceed the value returned by "InstancesCount".

7.80.2.9 Prepare()

```
void Afterwarp.Scene.Prepare ( ) [inline]
```

Configures rendering parameters for the scene, retrieves light indices from light container module and updates internal buffers required for rendering the scene. This must be called outside of "Begin" / "End" block, but before starting any rendering. Ambient occlusion texture, shadow casting atlas, view and projection matrices have been assigned prior this call.

7.80.2.10 ResetCache()

```
void Afterwarp.Scene.ResetCache ( )
```

Clears cached buffers and resources in the scene.

7.80.2.11 SetTexture()

```
void Afterwarp.Scene.SetTexture (
    SceneTextureType type,
    Texture texture ) [inline]
```

Changes texture of the given type associated with the scene.

7.80.2.12 SetVertexElements()

```
void Afterwarp.Scene.SetVertexElements (
    int index,
    VertexElement[] elements ) [inline]
```

Changes vertex element declaration associated with the given index.

7.80.2.13 SetVertexElementsFromTextModeller()

```
void Afterwarp.Scene.SetVertexElementsFromTextModeller (
    int index ) [inline]
```

Changes vertex element declaration to match the one of 2D and 3D text rendering module, associated with the given index.

7.80.2.14 TryPrepare()

```
bool Afterwarp.Scene.TryPrepare ( )
```

Configures rendering parameters for the scene, retrieves light indices from light container module and updates internal buffers required for rendering the scene. This must be called outside of "Begin" / "End" block, but before starting any rendering. Ambient occlusion texture, shadow casting atlas, texture cabinet, view and projection matrices have been assigned prior this call. Returns true on success. For depth/normals rendering, this step is not required.

7.80.3 Property Documentation

7.80.3.1 ActiveVertexElements

```
int Afterwarp.Scene.ActiveVertexElements [get], [set]
```

Returns or changes currently active index of vertex elements declaration.

7.80.3.2 Atlas

```
ShadowCastingAtlas Afterwarp.Scene.Atlas [get], [set]
```

Returns or changes shadow casting atlas associated with the scene. Note: this must be called outside of "Begin" / "End" block and before calling "Prepare" function.

7.80.3.3 Attributes

```
uint Afterwarp.Scene.Attributes [get], [set]
```

Returns or changes attributes that define the rendering behavior of the scene.

7.80.3.4 Device

```
Device Afterwarp.Scene.Device [get]
```

Returns device associated with the scene.

7.80.3.5 InstancesCount

```
int Afterwarp.Scene.InstancesCount [get]
```

Returns maximum number of supported instances.

7.80.3.6 Lights

```
SceneLights Afterwarp.Scene.Lights [get], [set]
```

Returns or changes lights container module associated with the scene. Note: this must be called outside of "Begin" / "End" block and before calling "Prepare" function.

7.80.3.7 Material

```
ObjectMaterial Afterwarp.Scene.Material [get], [set]
```

Returns or changes scene object's material properties.

7.80.3.8 ParallaxMapping

`ParallaxMappingParameters` Afterwarp.Scene.ParallaxMapping [get], [set]

Returns or changes parameters that define Parallax Mapping technique is performed.

7.80.3.9 Program

`Program` Afterwarp.Scene.Program [get]

Returns currently active shader program handle.

7.80.3.10 Projection

`Matrix4x4` Afterwarp.Scene.Projection [get], [set]

Returns or changes "Projection" transformation matrix.

7.80.3.11 Rendering

`bool` Afterwarp.Scene.Rendering [get]

Indicates that the rendering is currently taking place (inside "Begin" / "End" block).

7.80.3.12 TextureCabinet

`TextureCabinet` Afterwarp.Scene.TextureCabinet [get], [set]

Returns or changes texture cabinetc associated with the scene. Note: this must be called outside of "Begin" / "End" block and before calling "Prepare" function.

7.80.3.13 ToneMappingCoefficients

`Vector4` Afterwarp.Scene.ToneMappingCoefficients [get], [set]

Returns or changes tone-mapping RGB luma coefficients and maximum allowed level of white. These parameters are used only when "glassy" (OIT) technique is being performed.

7.80.3.14 View

`Matrix4x4` Afterwarp.Scene.View [get], [set]

Returns or changes "View" transformation matrix.

7.80.3.15 World

Matrix4x4 Afterwarp.Scene.World [get], [set]

Returns or changes "World" transformation matrix.

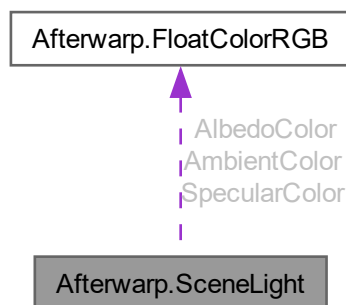
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.81 Afterwarp.SceneLight Struct Reference

Light source parameters in 3D scene.

Collaboration diagram for Afterwarp.SceneLight:



Public Member Functions

- [SceneLight](#) ([FloatColorRGB](#) ambientColor, [FloatColorRGB](#) albedoColor, [FloatColorRGB](#) specularColor, Vector3 position, Vector3 direction, float specularExponent=1.0f, float intensity=1.0f, float attenuationStart=float.MaxValue, float attenuationEnd=float.MaxValue, float angle=(float) Math.PI, float angleCutoff=(float) Math.PI, int shadowCaster=-1, uint bitmask=0, IntPtr payload=new IntPtr())

Creates a new light structure with the given values.

Public Attributes

- [FloatColorRGB](#) [AmbientColor](#)
Ambient color.
- float [AttenuationStart](#)
Range at which light attenuation starts (FLT_MAX means no attenuation should be applied).
- [FloatColorRGB](#) [AlbedoColor](#)
Albedo color.
- float [AttenuationEnd](#)

- Range at which light attenuation ends. This must be equal to or higher than `attenuationStart`.*

 - [FloatColorRGB SpecularColor](#)
Specular color.
 - float [SpecularExponent](#)
Exponent for specular light reflections.
 - Vector3 [Position](#)
Position of the light source.
 - float [Intensity](#)
Intensity of the light.
 - Vector3 [Direction](#)
Direction of the light source (non-zero value indicates a spotlight, zero values means omni-light).
 - float [Angle](#)
Angle at which spotlight starts to fade in range of $[0, \pi]$.
 - float [AngleCutoff](#)
Angle after which spotlight is completely dark. This should be equal to or higher than `angle`.
 - int [ShadowCaster](#)
 - uint [Bitmask](#)
 - uint [Reserved](#)
Reserved, must be set to zero.
 - IntPtr [Payload](#)
Custom payload value.

Properties

- static [SceneLight Default](#) [get]
Returns default light parameters.

7.81.1 Detailed Description

Light source parameters in 3D scene.

7.81.2 Constructor & Destructor Documentation

7.81.2.1 SceneLight()

```
Afterwarp.SceneLight.SceneLight (
    FloatColorRGB ambientColor,
    FloatColorRGB albedoColor,
    FloatColorRGB specularColor,
    Vector3 position,
    Vector3 direction,
    float specularExponent = 1::0f,
    float intensity = 1::0f,
    float attenuationStart = float::MaxValue,
    float attenuationEnd = float::MaxValue,
    float angle = (float)Math::PI,
    float angleCutoff = (float)Math::PI,
    int shadowCaster = -1,
    uint bitmask = 0,
    IntPtr payload = new IntPtr() ) [inline]
```

Creates a new light structure with the given values.

7.81.3 Member Data Documentation

7.81.3.1 AlbedoColor

`FloatColorRGB Afterwarp.SceneLight.AlbedoColor`

Albedo color.

7.81.3.2 AmbientColor

`FloatColorRGB Afterwarp.SceneLight.AmbientColor`

Ambient color.

7.81.3.3 Angle

`float Afterwarp.SceneLight.Angle`

Angle at which spotlight starts to fade in range of [0, PI).

7.81.3.4 AngleCutoff

`float Afterwarp.SceneLight.AngleCutoff`

Angle after which spotlight is completely dark. This should be equal to or higher than `angle`.

7.81.3.5 AttenuationEnd

`float Afterwarp.SceneLight.AttenuationEnd`

Range at which light attenuation ends. This must be equal to or higher than `attenuationStart`.

7.81.3.6 AttenuationStart

`float Afterwarp.SceneLight.AttenuationStart`

Range at which light attenuation starts (FLT_MAX means no attenuation should be applied).

7.81.3.7 Bitmask

`uint Afterwarp.SceneLight.Bitmask`

Bitmask that defines what objects are affected by this light. A value of zero means that light affects all objects.

7.81.3.8 Direction

```
Vector3 Afterwarp.SceneLight.Direction
```

Direction of the light source (non-zero value indicates a spotlight, zero values means omni-light).

7.81.3.9 Intensity

```
float Afterwarp.SceneLight.Intensity
```

Intensity of the light.

7.81.3.10 Payload

```
IntPtr Afterwarp.SceneLight.Payload
```

Custom payload value.

7.81.3.11 Position

```
Vector3 Afterwarp.SceneLight.Position
```

Position of the light source.

7.81.3.12 Reserved

```
uint Afterwarp.SceneLight.Reserved
```

Reserved, must be set to zero.

7.81.3.13 ShadowCaster

```
int Afterwarp.SceneLight.ShadowCaster
```

Index of the shadow caster associated with the light source. A value of -1 indicates light source does not cast shadows.

7.81.3.14 SpecularColor

```
FloatColorRGB Afterwarp.SceneLight.SpecularColor
```

Specular color.

7.81.3.15 SpecularExponent

`float Afterwarp.SceneLight.SpecularExponent`

Exponent for specular light reflections.

7.81.4 Property Documentation

7.81.4.1 Default

`SceneLight Afterwarp.SceneLight.Default [static], [get]`

Returns default light parameters.

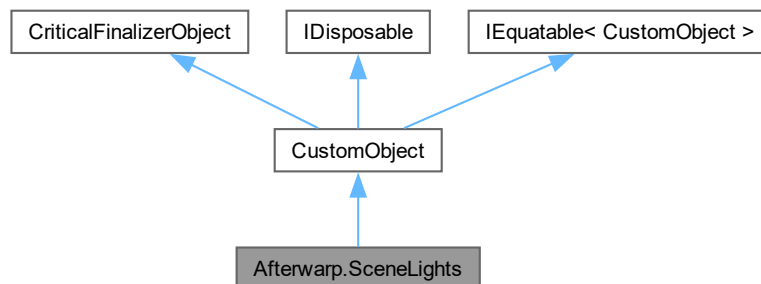
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

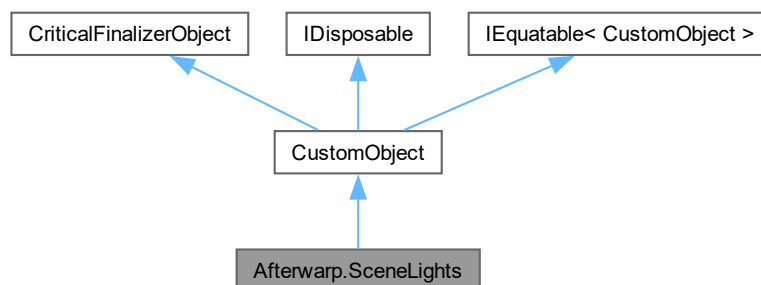
7.82 Afterwarp.SceneLights Class Reference

Module for storing and working with lights in a 3D scene.

Inheritance diagram for Afterwarp.SceneLights:



Collaboration diagram for Afterwarp.SceneLights:



Public Member Functions

- [SceneLights](#) ([Device](#) device)
Creates new instance of 3D lights management module.
- void [Add](#) ([SceneLight?](#) light=null)
Adds and returns a new light with default parameters.
- [SceneLight](#) [GetLight](#) (int index)
Retrieves an existing 3D light parameters associated with the given index.
- void [SetLight](#) (int index, [SceneLight](#) light)
Changes an existing 3D light parameters associated with the given index.
- void [Erase](#) (int index)
Erases light with the given index.
- void [Clear](#) ()
Erases light with the given index.
- void [Execute](#) (Matrix4x4 view, Matrix4x4 projection)
- void [RenderDebug](#) (bool intensity)
Renders light culling debug information. A call to "Execute" must be performed before this.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device associated with the module.
- int [Count](#) [get]
Returns total number of light sources.
- [Point](#) [Size](#) [get, set]
Returns or updates size of viewing area, which should match the rendereable surface size in pixels.
- int [ClusterSize](#) [get, set]
- int [DepthSlices](#) [get, set]
- [ClustersCullingMode](#) [CullingMode](#) [get, set]
- [Point](#) [Clusters](#) [get]
Returns number of calculated clusters that are accomodated horizontally and vertically accordingly.
- [Buffer](#) [Indices](#) [get]

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- [IntPtr _handle](#)
Wrapped object's handle.

7.82.1 Detailed Description

Module for storing and working with lights in a 3D scene.

7.82.2 Constructor & Destructor Documentation

7.82.2.1 SceneLights()

```
Afterwarp.SceneLights.SceneLights (
    Device device ) [inline]
```

Creates new instance of 3D lights management module.

7.82.3 Member Function Documentation

7.82.3.1 Add()

```
void Afterwarp.SceneLights.Add (
    SceneLight? light = null ) [inline]
```

Adds and returns a new light with default parameters.

7.82.3.2 Clear()

```
void Afterwarp.SceneLights.Clear ( )
```

Erases light with the given index.

7.82.3.3 Erase()

```
void Afterwarp.SceneLights.Erase (
    int index ) [inline]
```

Erases light with the given index.

7.82.3.4 Execute()

```
void Afterwarp.SceneLights.Execute (
    Matrix4x4 view,
    Matrix4x4 projection ) [inline]
```

Calculates cluster frustums, then performs light culling for each of the clusters. This populates an internal GPU buffer containing light indices for each of the clusters.

7.82.3.5 GetLight()

```
SceneLight Afterwarp.SceneLights.GetLight (
    int index ) [inline]
```

Retrieves an existing 3D light parameters associated with the given index.

7.82.3.6 RenderDebug()

```
void Afterwarp.SceneLights.RenderDebug (
    bool intensity ) [inline]
```

Renders light culling debug information. A call to "Execute" must be performed before this.

7.82.3.7 SetLight()

```
void Afterwarp.SceneLights.SetLight (
    int index,
    SceneLight light ) [inline]
```

Changes an existing 3D light parameters associated with the given index.

7.82.4 Property Documentation

7.82.4.1 Clusters

```
Point Afterwarp.SceneLights.Clusters [get]
```

Returns number of calculated clusters that are accomodated horizontally and vertically accordingly.

7.82.4.2 ClusterSize

```
int Afterwarp.SceneLights.ClusterSize [get], [set]
```

Size of each individual light cluster on the screen. Visible light indices will be calculated for each of the clusters, so scenes with more tightly packed light sources would benefit from smaller cluster size. A scene with either few light sources or light sources that occupy very large visible areas would benefit from bigger cluster sizes.

7.82.4.3 Count

```
int Afterwarp.SceneLights.Count [get]
```

Returns total number of light sources.

7.82.4.4 CullingMode

`ClustersCullingMode` `Afterwarp.SceneLights.CullingMode` [get], [set]

Returns or changes a mode used for packing calculated light indices for each of the clusters. High quality mode enables more light sources to be used simultaneously, whereas performance mode improves rendering performance on weaker GPUs assuming that few light sources are present.

7.82.4.5 DepthSlices

`int` `Afterwarp.SceneLights.DepthSlices` [get], [set]

Number of cluster slices per viewable depth on the screen. Visible light indices will be calculated for each of the clusters, so scenes with more tightly packed light sources would benefit from a bigger number of depth slices. A scene with either few light sources or light sources that occupy very large visible areas would benefit from a smaller number of depth slices.

7.82.4.6 Device

`Device` `Afterwarp.SceneLights.Device` [get]

Returns device associated with the module.

7.82.4.7 Indices

`Buffer` `Afterwarp.SceneLights.Indices` [get]

Returns light indices that have been calculated for each of the clusters. This is useful for a custom rendering engine that wants to use clustered light information.

7.82.4.8 Size

`Point` `Afterwarp.SceneLights.Size` [get], [set]

Returns or updates size of viewing area, which should match the renderable surface size in pixels.

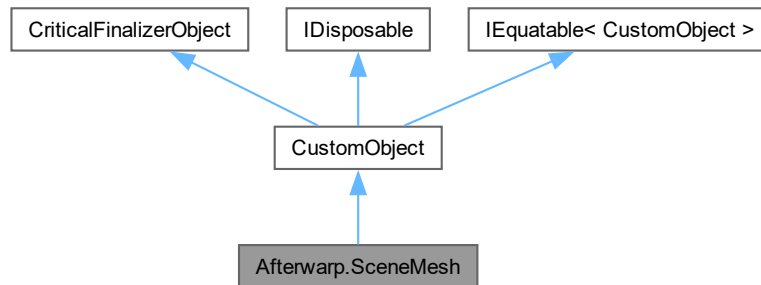
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

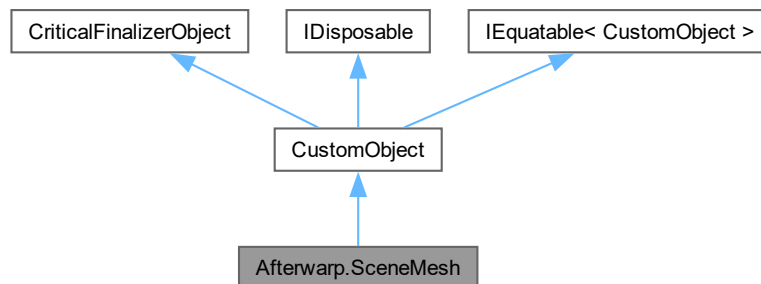
7.83 Afterwarp.SceneMesh Class Reference

A container for a 3D scene mesh model, its optional voxel representation and a bounding volume.

Inheritance diagram for Afterwarp.SceneMesh:



Collaboration diagram for Afterwarp.SceneMesh:



Public Member Functions

- void **Bounds** (out Vector3 boundsMin, out Vector3 boundsMax)
Returns minimum and maximum mesh boundaries.
- void **SetSlice** (SceneMesh parent, MeshMetaTag tag)
Specify the parent mesh and the tag to which this slice would refer to.
- void **GetSlice** (out SceneMesh parent, out MeshMetaTag parentTag)
- int **AutoDraw** (Scene scene, ObjectMaterial? material=null, FloatColor? color=null, PrimitiveTopology topology=PrimitiveTopology.Triangles, int instanceCount=0, int elementCount=0, int firstIndex=0, int baseVertex=0, uint options=0)
- int **AutoDrawSliced** (Scene scene, ObjectMaterial? material=null, FloatColor? color=null, PrimitiveTopology topology=PrimitiveTopology.Triangles, int instanceCount=0, uint options=0)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Static Public Attributes

- const byte [VertexElementsIndexUndefined](#) = 0xFF
A special "undefined" value for a vertex elements index.

Properties

- string [Name](#) [get]
Returns name of the mesh (case-insensitive).
- IntPtr [Payload](#) [get]
Returns payload associated with the mesh.
- [MeshModel](#) [Model](#) [get]
Returns a rendereable mesh model, if such is present.
- [MeshVoxel](#) [Voxel](#) [get]
A voxel representation of the mesh.
- [SceneMeshMaterials](#) [Materials](#) [get]
Returns mesh materials associated with the mesh.
- [MeshMetaTags](#) [Tags](#) [get]
Returns meta tags that represent relevant regions in a 3D mesh, if such are available.
- [SceneMeshLatches](#) [Latches](#) [get]
Returns an integrated collection of latches associated with the 3D mesh.
- float [Scale](#) [get]
Returns scale of the mesh.
- Vector3 [Size](#) [get]
Returns size of the mesh.
- byte [VertexElementsIndex](#) [get, set]

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- [IntPtr _handle](#)

Wrapped object's handle.

7.83.1 Detailed Description

A container for a 3D scene mesh model, its optional voxel representation and a bounding volume.

7.83.2 Member Function Documentation

7.83.2.1 AutoDraw()

```
int Afterwarp.SceneMesh.AutoDraw (
    Scene scene,
    ObjectMaterial? material = null,
    FloatColor? color = null,
    PrimitiveTopology topology = PrimitiveTopology::Triangles,
    int instanceCount = 0,
    int elementCount = 0,
    int firstIndex = 0,
    int baseVertex = 0,
    uint options = 0 ) [inline]
```

Issues one or more draw calls for the scene mesh, assigning materials and textures accordingly. If *material* is `null`, default material will be used. If *instanceCount* is zero, then non-instancing draw calls will be issued. If *scene* is `null`, this does not issue any draw calls, instead counting any tentative draw calls that would be issued; this can be used to determine if any portions of the object would be rendered to decide whether to perform certain pass or not.

7.83.2.2 AutoDrawSliced()

```
int Afterwarp.SceneMesh.AutoDrawSliced (
    Scene scene,
    ObjectMaterial? material = null,
    FloatColor? color = null,
    PrimitiveTopology topology = PrimitiveTopology::Triangles,
    int instanceCount = 0,
    uint options = 0 ) [inline]
```

Issues one or more draw calls for the scene mesh, assigning materials and textures accordingly. If scene mesh is a slice, then renders the portions of the slice accordingly. If *material* is `null`, default material will be used. If *instanceCount* is zero, then non-instancing draw call is issued. If *scene* is `null`, this does not issue any draw calls, instead counting any tentative draw calls that would be issued; this can be used to determine if any portions of the object would be rendered to decide whether to perform certain pass or not.

7.83.2.3 Bounds()

```
void Afterwarp.SceneMesh.Bounds (
    out Vector3 boundsMin,
    out Vector3 boundsMax ) [inline]
```

Returns minimum and maximum mesh boundaries.

7.83.2.4 GetSlice()

```
void Afterwarp.SceneMesh.GetSlice (
    out SceneMesh parent,
    out MeshMetaTag parentTag ) [inline]
```

Returns parent mesh that contains the actual 3D model to be rendered and tag that contains information about what portion of parent mesh should be rendered

7.83.2.5 SetSlice()

```
void Afterwarp.SceneMesh.SetSlice (
    SceneMesh parent,
    MeshMetaTag tag ) [inline]
```

Specify the parent mesh and the tag to which this slice would refer to.

7.83.3 Member Data Documentation

7.83.3.1 VertexElementsIndexUndefined

```
const byte Afterwarp.SceneMesh.VertexElementsIndexUndefined = 0xFF [static]
```

A special "undefined" value for an vertex elements index.

7.83.4 Property Documentation

7.83.4.1 Latches

```
SceneMeshLatches Afterwarp.SceneMesh.Latches [get]
```

Returns an integrated collection of latches associated with the 3D mesh.

7.83.4.2 Materials

```
SceneMeshMaterials Afterwarp.SceneMesh.Materials [get]
```

Returns mesh materials associated with the mesh.

7.83.4.3 Model

```
MeshModel Afterwarp.SceneMesh.Model [get]
```

Returns a rendereable mesh model, if such is present.

7.83.4.4 Name

```
string Afterwarp.SceneMesh.Name [get]
```

Returns name of the mesh (case-insensitive).

7.83.4.5 Payload

```
IntPtr Afterwarp.SceneMesh.Payload [get]
```

Returns payload associated with the mesh.

7.83.4.6 Scale

```
float Afterwarp.SceneMesh.Scale [get]
```

Returns scale of the mesh.

7.83.4.7 Size

```
Vector3 Afterwarp.SceneMesh.Size [get]
```

Returns size of the mesh.

7.83.4.8 Tags

```
MeshMetaTags Afterwarp.SceneMesh.Tags [get]
```

Returns meta tags that represent relevant regions in a 3D mesh, if such are available.

7.83.4.9 VertexElementsIndex

```
byte Afterwarp.SceneMesh.VertexElementsIndex [get], [set]
```

Returns or changes index of vertex elements that describe the format of mesh vertex buffer. A value of 0xFF means index is undefined.

7.83.4.10 Voxel

```
MeshVoxel Afterwarp.SceneMesh.Voxel [get]
```

A voxel representation of the mesh.

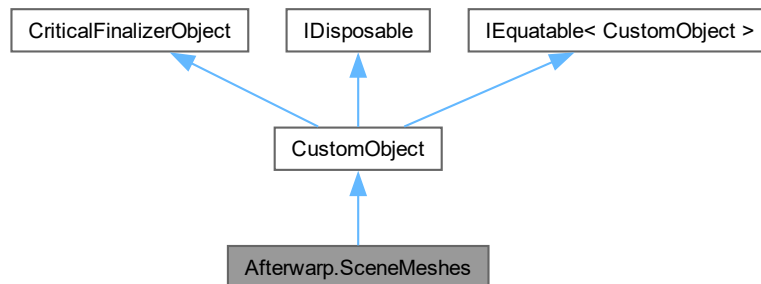
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

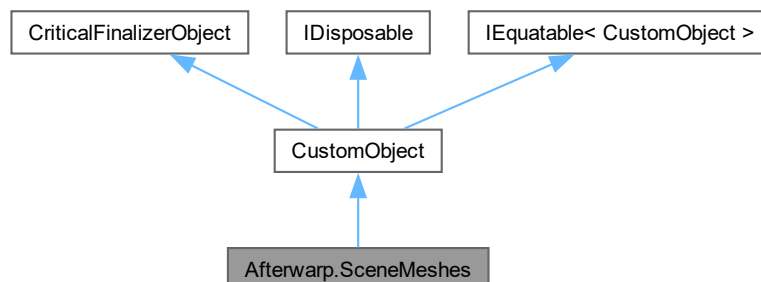
7.84 Afterwarp.SceneMeshes Class Reference

A collection of 3D scene mesh containers.

Inheritance diagram for Afterwarp.SceneMeshes:



Collaboration diagram for Afterwarp.SceneMeshes:



Public Member Functions

- [SceneMeshes](#) ([Device](#) device)
Creates new instance of object container.
- [SceneMesh Add](#) (string name, [Vector3](#) boundsMin, [Vector3](#) boundsMax, float scale=1.0f, [IntPtr](#) payload=default)
Creates a new mesh with the given parameters.
- [SceneMesh AddFromBuffer](#) (string name, [MeshBuffer](#) buffer, [VertexElement](#)[] vertexElements, uint channel=0, uint semanticIndex=0, [Vector3?](#) boundsMin=null, [Vector3?](#) boundsMax=null, float scale=1.0f, [IntPtr](#) payload=default)
Creates a new mesh, loading contents from a given mesh buffer.
- [SceneMesh TryAddFromFile](#) (string name, string fileName, [VertexElement](#)[] vertexElements, [MeshBuffer.LoadSceneFeedback](#) feedback, out string debug, ref uint options, uint channel=0, uint semanticIndex=0, float scale=1.0f, [IntPtr](#) payload=default)

- [SceneMesh AddFromFile](#) (string name, string fileName, [VertexElement](#)[] vertexElements, ref uint options, [MeshBuffer.LoadSaveFeedback](#) feedback=null, uint channel=0, uint semanticIndex=0, float scale=1.0f, IntPtr payload=default)
- [SceneMesh Mesh](#) (int index)
Returns mesh with the given index.
- [SceneMesh TryMesh](#) (string name)
Attempts to retrieve an existing mesh with the given name (case-insensitive).
- [SceneMesh Mesh](#) (string name)
Returns mesh with the given name (case-insensitive).
- [SceneMesh TryPayload](#) (IntPtr payload)
Returns a mesh that corresponds to the given payload, if such exists.
- [SceneMesh Payload](#) (IntPtr payload)
Returns a mesh that corresponds to the given payload.
- bool [Exists](#) (string name)
Tests whether a mesh with the given name exists.
- bool [PayloadExists](#) (IntPtr payload)
Tests whether a mesh with the given payload exists.
- void [Erase](#) ([SceneMesh](#) mesh)
Erases the given mesh.
- void [Clear](#) ()
Removes all existing meshes.
- void [Slice](#) ([SceneMesh](#) parent, byte tag=[MeshMetaTagType.Object](#), string prefix=null)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the collection.
- int [Count](#) [get]
Returns number of existing meshes.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)

Wrapped object's handle.

7.84.1 Detailed Description

A collection of 3D scene mesh containers.

7.84.2 Constructor & Destructor Documentation

7.84.2.1 SceneMeshes()

```
Afterwarp.SceneMeshes.SceneMeshes (
    Device device ) [inline]
```

Creates new instance of object container.

7.84.3 Member Function Documentation

7.84.3.1 Add()

```
SceneMesh Afterwarp.SceneMeshes.Add (
    string name,
    Vector3 boundsMin,
    Vector3 boundsMax,
    float scale = 1::0f,
    IntPtr payload = default ) [inline]
```

Creates a new mesh with the given parameters.

7.84.3.2 AddFromBuffer()

```
SceneMesh Afterwarp.SceneMeshes.AddFromBuffer (
    string name,
    MeshBuffer buffer,
    VertexElement[] vertexElements,
    uint channel = 0,
    uint semanticIndex = 0,
    Vector3? boundsMin = null,
    Vector3? boundsMax = null,
    float scale = 1::0f,
    IntPtr payload = default ) [inline]
```

Creates a new mesh, loading contents from a given mesh buffer.

7.84.3.3 AddFromFile()

```
SceneMesh Afterwarp.SceneMeshes.AddFromFile (
    string name,
    string fileName,
    VertexElement[] vertexElements,
    ref uint options,
    MeshBuffer::LoadSaveFeedback feedback = null,
    uint channel = 0,
    uint semanticIndex = 0,
    float scale = 1::0f,
    IntPtr payload = default ) [inline]
```

Creates a new mesh, loading contents from a file on disk, which can be either a Wavefront OBJ file (with ".obj" extension), or a proprietary binary file (with ".mesh" extension). If the source file is in OBJ format, this function will also try to automatically load an accompanying voxel file (same file name, but with ".voxel" extension) and/or latches (".latch" extension), if such is present at the same path.

7.84.3.4 Clear()

```
void Afterwarp.SceneMeshes.Clear ( )
```

Removes all existing meshes.

7.84.3.5 Erase()

```
void Afterwarp.SceneMeshes.Erase (
    SceneMesh mesh )
```

Erases the given mesh.

7.84.3.6 Exists()

```
bool Afterwarp.SceneMeshes.Exists (
    string name )
```

Tests whether a mesh with the given name exists.

7.84.3.7 Mesh() [1/2]

```
SceneMesh Afterwarp.SceneMeshes.Mesh (
    int index ) [inline]
```

Returns mesh with the given index.

7.84.3.8 Mesh() [2/2]

```
SceneMesh Afterwarp.SceneMeshes.Mesh (
    string name ) [inline]
```

Returns mesh with the given name (case-insensitive).

7.84.3.9 Payload()

```
SceneMesh Afterwarp.SceneMeshes.Payload (
    IntPtr payload ) [inline]
```

Returns a mesh that corresponds to the given payload.

7.84.3.10 PayloadExists()

```
bool Afterwarp.SceneMeshes.PayloadExists (
    IntPtr payload )
```

Tests whether a mesh with the given payload exists.

7.84.3.11 Slice()

```
void Afterwarp.SceneMeshes.Slice (
    SceneMesh parent,
    byte tag = MeshMetaTagType::Object,
    string prefix = null ) [inline]
```

Splits the given parent mesh into "dummy" sub-meshes by specific meta-tag type. Each sub-mesh splitted this way will refer to parent and one of its tag accordingly. This enables sharing the same 3D model for rendering, while having customized voxel representation and/or latches for each splitted segment.

7.84.3.12 TryAddFromFile()

```
SceneMesh Afterwarp.SceneMeshes.TryAddFromFile (
    string name,
    string fileName,
    VertexElement[] vertexElements,
    MeshBuffer::LoadSaveFeedback feedback,
    out string debug,
    ref uint options,
    uint channel = 0,
    uint semanticIndex = 0,
    float scale = 1.0f,
    IntPtr payload = default ) [inline]
```

Creates a new mesh, loading contents from a file on disk, which can be either a Wavefront OBJ file (with ".obj" extension), or a proprietary binary file (with ".mesh" extension). If the source file is in OBJ format, this function will also try to automatically load an accompanying voxel file (same file name, but with ".voxel" extension) and/or latches (".latch" extension), if such is present at the same path. If mesh could not be loaded, a null-handle object will be returned and debug string will contain error information.

7.84.3.13 TryMesh()

```
SceneMesh Afterwarp.SceneMeshes.TryMesh (
    string name ) [inline]
```

Attempts to retrieve an existing mesh with the given name (case-insensitive).

7.84.3.14 TryPayload()

```
SceneMesh Afterwarp.SceneMeshes.TryPayload (
    IntPtr payload ) [inline]
```

Returns a mesh that corresponds to the given payload, if such exists.

7.84.4 Property Documentation

7.84.4.1 Count

```
int Afterwarp.SceneMeshes.Count [get]
```

Returns number of existing meshes.

7.84.4.2 Device

```
Device Afterwarp.SceneMeshes.Device [get]
```

Returns device associated with the collection.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.85 Afterwarp.SceneMeshLatch Struct Reference

A latch in 3D scene mesh, which describes a bone joint connection or a movement waypoint.

Public Attributes

- byte[] [NameBytes](#)
Name of the latch described as an array of UTF-8 encoded bytes.
- Vector3 [Position](#)
Size of the font.
- ushort [Type](#)
Type of the latch.
- ushort [Group](#)
Group of the latch.
- Quaternion [Orientation](#)
Orientation of the latch.

Properties

- string [Name](#) [get, set]
Name of the latch.

7.85.1 Detailed Description

A latch in 3D scene mesh, which describes a bone joint connection or a movement waypoint.

7.85.2 Member Data Documentation

7.85.2.1 Group

```
ushort Afterwarp.SceneMeshLatch.Group
```

Group of the latch.

7.85.2.2 NameBytes

```
byte [ ] Afterwarp.SceneMeshLatch.NameBytes
```

Name of the latch described as an array of UTF-8 encoded bytes.

7.85.2.3 Orientation

```
Quaternion Afterwarp.SceneMeshLatch.Orientation
```

Orientation of the latch.

7.85.2.4 Position

```
Vector3 Afterwarp.SceneMeshLatch.Position
```

Size of the font.

7.85.2.5 Type

```
ushort Afterwarp.SceneMeshLatch.Type
```

Type of the latch.

7.85.3 Property Documentation

7.85.3.1 Name

```
string Afterwarp.SceneMeshLatch.Name [get], [set]
```

Name of the latch.

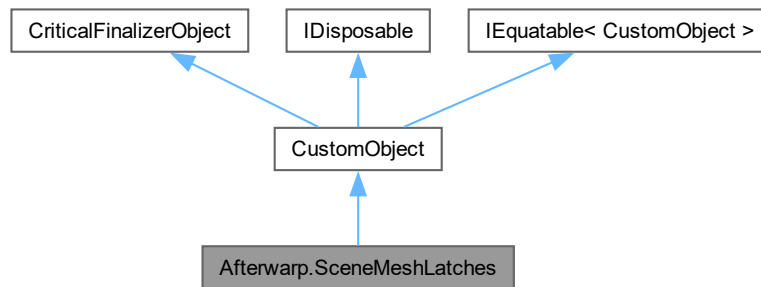
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

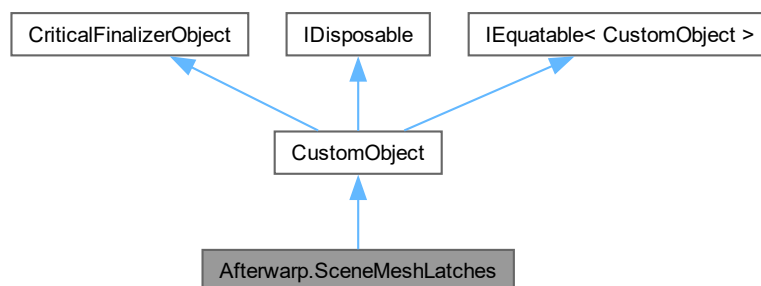
7.86 Afterwarp.SceneMeshLatches Class Reference

Container for scene mesh latches that define how meshes connect with each other.

Inheritance diagram for Afterwarp.SceneMeshLatches:



Collaboration diagram for Afterwarp.SceneMeshLatches:



Public Member Functions

- [SceneMeshLatches](#) ()
Creates new instance of scene mesh material container.
- [SceneMeshLatch GetLatch](#) (int index)
Returns an existing latch with the given index.
- void [SetLatch](#) (int index, [SceneMeshLatch](#) latch)
Updates an existing latch with the given index.
- int [GetLatchIndex](#) (string name)
Returns an index of a latch with the given name (case-insensitive) index or -1 if it doesn't exist.
- int [Add](#) (string name, int type=[SceneMeshLatchType.Waypoint](#), int group=0, Vector3? position=null, Quaternion? orientation=null)
Adds a new latch to the container and returns its index.
- void [Erase](#) (int index)

- *Removes an existing latch with the given index.*
- void [Clear](#) ()
- *Removes and releases all existing latches.*
- void [Copy](#) ([SceneMeshLatches](#) other)
- *Copies latches from another collection.*
- void [TakeAway](#) ([SceneMeshLatches](#) other)
- *Takes away the internal contents from another collection without doing any copying.*
- bool [TryLoadFromFile](#) (string fileName)
- *Loads a collection of latches from file on disk.*
- void [LoadFromFile](#) (string fileName)
- *Loads a collection of latches from file on disk.*
- bool [TryLoadFromFileInMemory](#) (Array buffer)
- *Loads a collection of latches from file in memory.*
- void [LoadFromFileInMemory](#) (Array buffer)
- *Loads a collection of latches from file in memory.*
- void [SaveToFile](#) (string fileName)
- *Saves a collection of latches to file on disk.*
- float [GetWaypointDistance](#) (int group=0)
- *Returns calculated waypoint traveling distance for the given latch group.*
- void [InvalidateWaypoints](#) ()

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- int [Count](#) [get]
- *Returns total number of existing latches.*

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
- *Wrapped object's handle.*

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
- *Releases the object's resources.*

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
- *Wrapped object's handle.*

7.86.1 Detailed Description

Container for scene mesh latches that define how meshes connect with each other.

7.86.2 Constructor & Destructor Documentation

7.86.2.1 SceneMeshLatches()

```
Afterwarp.SceneMeshLatches.SceneMeshLatches ( ) [inline]
```

Creates new instance of scene mesh material container.

7.86.3 Member Function Documentation

7.86.3.1 Add()

```
int Afterwarp.SceneMeshLatches.Add (
    string name,
    int type = SceneMeshLatchType::Waypoint,
    int group = 0,
    Vector3? position = null,
    Quaternion? orientation = null ) [inline]
```

Adds a new latch to the container and returns its index.

7.86.3.2 Clear()

```
void Afterwarp.SceneMeshLatches.Clear ( )
```

Removes and releases all existing latches.

7.86.3.3 Copy()

```
void Afterwarp.SceneMeshLatches.Copy (
    SceneMeshLatches other ) [inline]
```

Copies latches from another collection.

7.86.3.4 Erase()

```
void Afterwarp.SceneMeshLatches.Erase (
    int index )
```

Removes an existing latch with the given index.

7.86.3.5 GetLatch()

```
SceneMeshLatch Afterwarp.SceneMeshLatches.GetLatch (
    int index ) [inline]
```

Returns an existing latch with the given index.

7.86.3.6 GetLatchIndex()

```
int Afterwarp.SceneMeshLatches.GetLatchIndex (
    string name ) [inline]
```

Returns an index of a latch with the given name (case-insensitive) index or -1 if it doesn't exist.

7.86.3.7 GetWaypointDistance()

```
float Afterwarp.SceneMeshLatches.GetWaypointDistance (
    int group = 0 ) [inline]
```

Returns calculated waypoint traveling distance for the given latch group.

7.86.3.8 InvalidateWaypoints()

```
void Afterwarp.SceneMeshLatches.InvalidateWaypoints ( )
```

Triggers recalculation of the interpolated waypoint positions. This should be done when one or more latches have been modified.

7.86.3.9 LoadFromFile()

```
void Afterwarp.SceneMeshLatches.LoadFromFile (
    string fileName ) [inline]
```

Loads a collection of latches from file on disk.

7.86.3.10 LoadFromFileInMemory()

```
void Afterwarp.SceneMeshLatches.LoadFromFileInMemory (
    Array buffer ) [inline]
```

Loads a collection of latches from file in memory.

7.86.3.11 SaveToFile()

```
void Afterwarp.SceneMeshLatches.SaveToFile (
    string fileName ) [inline]
```

Saves a collection of latches to file on disk.

7.86.3.12 SetLatch()

```
void Afterwarp.SceneMeshLatches.SetLatch (
    int index,
    SceneMeshLatch latch ) [inline]
```

Updates an existing latch with the given index.

7.86.3.13 TakeAway()

```
void Afterwarp.SceneMeshLatches.TakeAway (
    SceneMeshLatches other ) [inline]
```

Takes away the internal contents from another collection without doing any copying.

7.86.3.14 TryLoadFromFile()

```
bool Afterwarp.SceneMeshLatches.TryLoadFromFile (
    string fileName ) [inline]
```

Loads a collection of latches from file on disk.

7.86.3.15 TryLoadFromFileInMemory()

```
bool Afterwarp.SceneMeshLatches.TryLoadFromFileInMemory (
    Array buffer ) [inline]
```

Loads a collection of latches from file in memory.

7.86.4 Property Documentation

7.86.4.1 Count

```
int Afterwarp.SceneMeshLatches.Count [get]
```

Returns total number of existing latches.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.87 Afterwarp.SceneMeshLatchType Struct Reference

Type of latch used in the 3D scene mesh.

Static Public Attributes

- const byte [Joint](#) = 0x01
Bone joint connection.
- const byte [Waypoint](#) = 0x02
Movement waypoint.

7.87.1 Detailed Description

Type of latch used in the 3D scene mesh.

7.87.2 Member Data Documentation

7.87.2.1 Joint

```
const byte Afterwarp.SceneMeshLatchType.Joint = 0x01 [static]
```

Bone joint connection.

7.87.2.2 Waypoint

```
const byte Afterwarp.SceneMeshLatchType.Waypoint = 0x02 [static]
```

Movement waypoint.

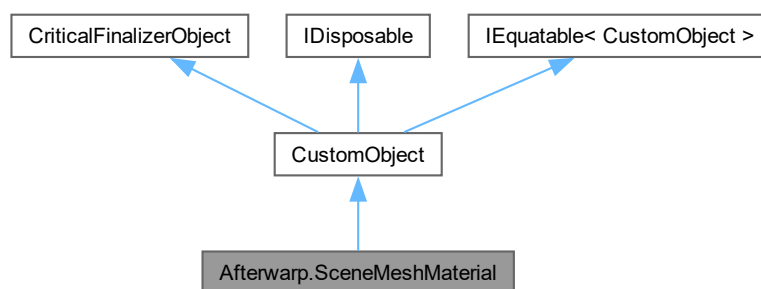
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

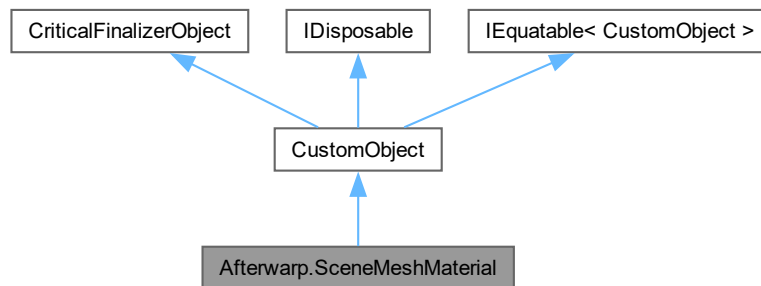
7.88 Afterwarp.SceneMeshMaterial Class Reference

Material that describes how a portion of a scene mesh geometry should look like.

Inheritance diagram for Afterwarp.SceneMeshMaterial:



Collaboration diagram for Afterwarp.SceneMeshMaterial:



Public Member Functions

- [Texture GetTexture](#) ([SceneMeshTexture](#) type)
Returns a texture associated with the given type or `null` if such does not exist.
- void [SetTexture](#) ([Texture](#) texture, [SceneMeshTexture](#) type)
- void [ReleaseTextures](#) ()
Releases existing textures.
- [SceneMeshMaterialRange GetRange](#) (int index)
Returns a range with the given index.
- void [SetRange](#) (int index, [SceneMeshMaterialRange](#) range)
Updates a range with the given index.
- int [AddRange](#) ([SceneMeshMaterialRange](#) range)
Adds a given range to the list of existing ranges and returns its index.
- void [ClearRanges](#) ()
Removes all existing ranges.
- void [Commit](#) ([Device](#) device, uint attributes=0)
Converts existing "scratch" textures into hardware-based textures.
- void [Copy](#) ([SceneMeshMaterial](#) material)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- string [Name](#) [get]
Material name.
- [SceneMeshMaterialShading Shading](#) [get, set]
Shading parameters of the material.
- int [RangeCount](#) [get]
Returns number of existing ranges.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.88.1 Detailed Description

Material that describes how a portion of a scene mesh geometry should look like.

7.88.2 Member Function Documentation

7.88.2.1 AddRange()

```
int Afterwarp.SceneMeshMaterial.AddRange (
    SceneMeshMaterialRange range ) [inline]
```

Adds a given range to the list of existing ranges and returns its index.

7.88.2.2 ClearRanges()

```
void Afterwarp.SceneMeshMaterial.ClearRanges ( )
```

Removes all existing ranges.

7.88.2.3 Commit()

```
void Afterwarp.SceneMeshMaterial.Commit (
    Device device,
    uint attributes = 0 ) [inline]
```

Converts existing "scratch" textures into hardware-based textures.

7.88.2.4 Copy()

```
void Afterwarp.SceneMeshMaterial.Copy (
    SceneMeshMaterial material ) [inline]
```

Copies contents from another material. In case of textures, creates new instances of textures and copies the contents of textures from the source.

7.88.2.5 GetRange()

```
SceneMeshMaterialRange Afterwarp.SceneMeshMaterial.GetRange (
    int index ) [inline]
```

Returns a range with the given index.

7.88.2.6 GetTexture()

```
Texture Afterwarp.SceneMeshMaterial.GetTexture (
    SceneMeshTexture type ) [inline]
```

Returns a texture associated with the given type or `null` if such does not exist.

7.88.2.7 ReleaseTextures()

```
void Afterwarp.SceneMeshMaterial.ReleaseTextures ( )
```

Releases existing textures.

7.88.2.8 SetRange()

```
void Afterwarp.SceneMeshMaterial.SetRange (
    int index,
    SceneMeshMaterialRange range ) [inline]
```

Updates a range with the given index.

7.88.2.9 SetTexture()

```
void Afterwarp.SceneMeshMaterial.SetTexture (
    Texture texture,
    SceneMeshTexture type ) [inline]
```

Changes texture of the given type. Note: material will take ownership of the given texture instance and relinquish ownership of instance that was previously assigned (if it is not `null`).

7.88.3 Property Documentation

7.88.3.1 Name

```
string Afterwarp.SceneMeshMaterial.Name [get]
```

Material name.

7.88.3.2 RangeCount

```
int Afterwarp.SceneMeshMaterial.RangeCount [get]
```

Returns number of existing ranges.

7.88.3.3 Shading

```
SceneMeshMaterialShading Afterwarp.SceneMeshMaterial.Shading [get], [set]
```

Shading parameters of the material.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.89 Afterwarp.SceneMeshMaterialRange Struct Reference

A range of indices in the mesh that a material applies to.

Public Member Functions

- [SceneMeshMaterialRange](#) (int indexStart, int indexCount)
Creates a structure with the given values.

Public Attributes

- int [IndexStart](#)
Starting index.
- int [IndexCount](#)
Number of indices.

7.89.1 Detailed Description

A range of indices in the mesh that a material applies to.

7.89.2 Constructor & Destructor Documentation

7.89.2.1 SceneMeshMaterialRange()

```
Afterwarp.SceneMeshMaterialRange.SceneMeshMaterialRange (
    int indexStart,
    int indexCount ) [inline]
```

Creates a structure with the given values.

7.89.3 Member Data Documentation

7.89.3.1 IndexCount

```
int Afterwarp.SceneMeshMaterialRange.IndexCount
```

Number of indices.

7.89.3.2 IndexStart

```
int Afterwarp.SceneMeshMaterialRange.IndexStart
```

Starting index.

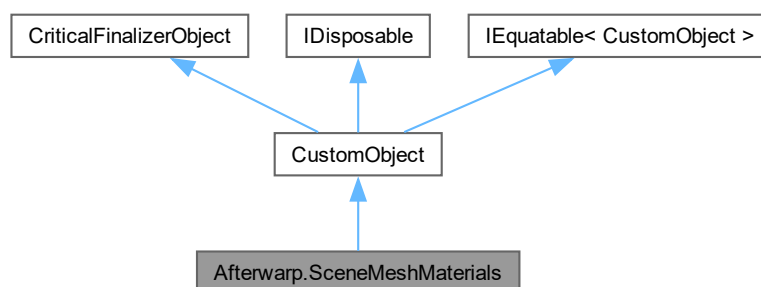
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

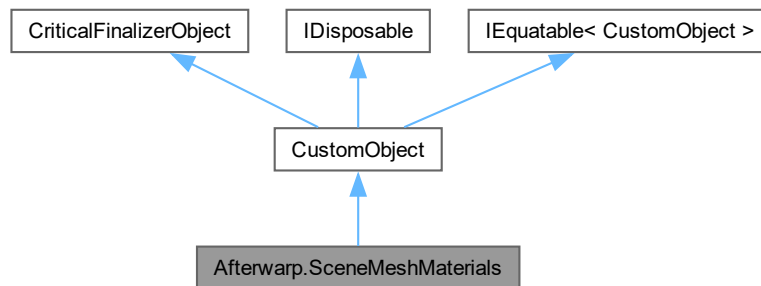
7.90 Afterwarp.SceneMeshMaterials Class Reference

Container for scene mesh materials that describe the appearance of scene mesh geometry.

Inheritance diagram for Afterwarp.SceneMeshMaterials:



Collaboration diagram for Afterwarp.SceneMeshMaterials:



Public Member Functions

- [SceneMeshMaterials](#) ()
Creates new instance of scene mesh material container.
- [SceneMeshMaterial GetMaterial](#) (int index)
Returns an existing material with the given index.
- int [Add](#) (string name)
Adds a new material to the scene mesh and returns its index.
- void [Erase](#) (int index)
Removes a material with the given index.
- void [Clear](#) ()
Removes and releases all existing materials.
- void [Commit](#) ([Device](#) device, uint attributes=0)
Converts existing "scratch" textures into hardware-based textures.
- void [Copy](#) ([SceneMeshMaterials](#) other)
Copies materials into the collection from another container.
- void [TakeAway](#) ([SceneMeshMaterials](#) other)
Takes away the internal contents from another material container without doing any copying.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- string [Name](#) [get, set]
Returns or changes the name of the library.
- int [Count](#) [get]
Returns total number of existing materials.
- bool [Texturing](#) [get]
Reviews existing materials to determine whether any of them use texturing.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.90.1 Detailed Description

Container for scene mesh materials that describe the appearance of scene mesh geometry.

7.90.2 Constructor & Destructor Documentation

7.90.2.1 SceneMeshMaterials()

```
Afterwarp.SceneMeshMaterials.SceneMeshMaterials ( ) [inline]
```

Creates new instance of scene mesh material container.

7.90.3 Member Function Documentation

7.90.3.1 Add()

```
int Afterwarp.SceneMeshMaterials.Add (
    string name ) [inline]
```

Adds a new material to the scene mesh and returns its index.

7.90.3.2 Clear()

```
void Afterwarp.SceneMeshMaterials.Clear ( )
```

Removes and releases all existing materials.

7.90.3.3 Commit()

```
void Afterwarp.SceneMeshMaterials.Commit (
    Device device,
    uint attributes = 0 ) [inline]
```

Converts existing "scratch" textures into hardware-based textures.

7.90.3.4 Copy()

```
void Afterwarp.SceneMeshMaterials.Copy (
    SceneMeshMaterials other ) [inline]
```

Copies materials into the collection from another container.

7.90.3.5 Erase()

```
void Afterwarp.SceneMeshMaterials.Erase (
    int index )
```

Removes a material with the given index.

7.90.3.6 GetMaterial()

```
SceneMeshMaterial Afterwarp.SceneMeshMaterials.GetMaterial (
    int index ) [inline]
```

Returns an existing material with the given index.

7.90.3.7 TakeAway()

```
void Afterwarp.SceneMeshMaterials.TakeAway (
    SceneMeshMaterials other ) [inline]
```

Takes away the internal contents from another material container without doing any copying.

7.90.4 Property Documentation

7.90.4.1 Count

```
int Afterwarp.SceneMeshMaterials.Count [get]
```

Returns total number of existing materials.

7.90.4.2 Name

```
string Afterwarp.SceneMeshMaterials.Name [get], [set]
```

Returns or changes the name of the library.

7.90.4.3 Texturing

```
bool Afterwarp.SceneMeshMaterials.Texturing [get]
```

Reviews existing materials to determine whether any of them use texturing.

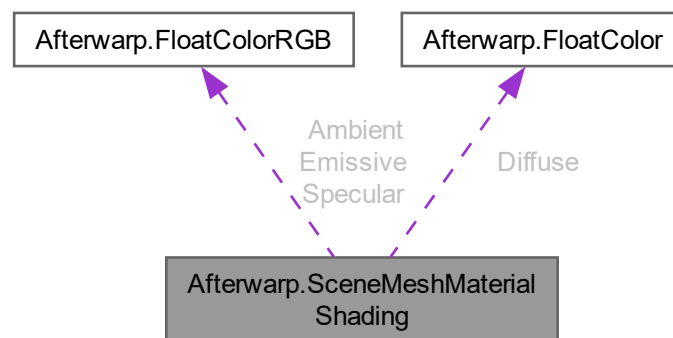
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.91 Afterwarp.SceneMeshMaterialShading Struct Reference

Shading parameters for scene mesh material.

Collaboration diagram for Afterwarp.SceneMeshMaterialShading:



Public Member Functions

- `SceneMeshMaterialShading` (`FloatColorRGB` ambient, float bloom, `FloatColor` diffuse, `FloatColorRGB` specular, float specularExponent, `FloatColorRGB` emissive, int lighting, `Vector3` roughness)
Creates a structure with the given values.

Public Attributes

- [FloatColorRGB Ambient](#)
- float [Bloom](#)
- [FloatColor Diffuse](#)
Diffuse color (alpha-channel denotes transparency).
- [FloatColorRGB Specular](#)
Specular color.
- float [SpecularExponent](#)
Specular exponent.
- [FloatColorRGB Emissive](#)
Emissive color.
- int [Lighting](#)
- Vector3 [Roughness](#)
- float [Reserved](#)
Reserved value (should be set to zero).

7.91.1 Detailed Description

Shading parameters for scene mesh material.

7.91.2 Constructor & Destructor Documentation

7.91.2.1 SceneMeshMaterialShading()

```
Afterwarp.SceneMeshMaterialShading.SceneMeshMaterialShading (
    FloatColorRGB ambient,
    float bloom,
    FloatColor diffuse,
    FloatColorRGB specular,
    float specularExponent,
    FloatColorRGB emissive,
    int lighting,
    Vector3 roughness ) [inline]
```

Creates a structure with the given values.

7.91.3 Member Data Documentation

7.91.3.1 Ambient

[FloatColorRGB](#) Afterwarp.SceneMeshMaterialShading.Ambient

Ambient color. If one of the values is NaN or Infinity, then ambient color should be deduced from diffuse color.

7.91.3.2 Bloom

float Afterwarp.SceneMeshMaterialShading.Bloom

Bloom (glare) factor. If this value is NaN or Infinity, then bloom factor is deduced from object material.

7.91.3.3 Diffuse

`FloatColor` Afterwarp.SceneMeshMaterialShading.Diffuse

Diffuse color (alpha-channel denotes transparency).

7.91.3.4 Emissive

`FloatColorRGB` Afterwarp.SceneMeshMaterialShading.Emissive

Emissive color.

7.91.3.5 Lighting

`int` Afterwarp.SceneMeshMaterialShading.Lighting

Lighting (0: Compatibility, 1: Default, 2: Phong, 3: Minnaert, 4: Cook-Torrance, 5: Oren-Nayar) Compatibility mode ignores ambient color, specular color and specular exponent: ambient color is deduced from diffuse color, whereas specular color and exponent are taken from object material. Default mode considers all colors and factors, but multiplies them with the corresponding values from object material. Starting at value of 2 (Phong) and higher, all colors and factors are considered and they are considered exclusively, overriding corresponding parameters from object material.

7.91.3.6 Reserved

`float` Afterwarp.SceneMeshMaterialShading.Reserved

Reserved value (should be set to zero).

7.91.3.7 Roughness

`Vector3` Afterwarp.SceneMeshMaterialShading.Roughness

Roughness values for non-Phong lighting models. If a certain value is NaN or Infinity, then that value is deduced from object material.

7.91.3.8 Specular

`FloatColorRGB` Afterwarp.SceneMeshMaterialShading.Specular

Specular color.

7.91.3.9 SpecularExponent

`float` Afterwarp.SceneMeshMaterialShading.SpecularExponent

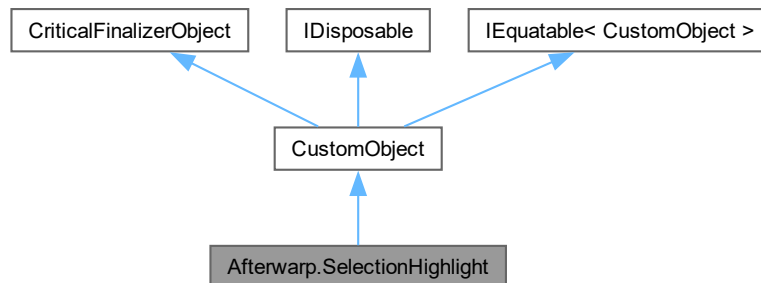
Specular exponent.

The documentation for this struct was generated from the following file:

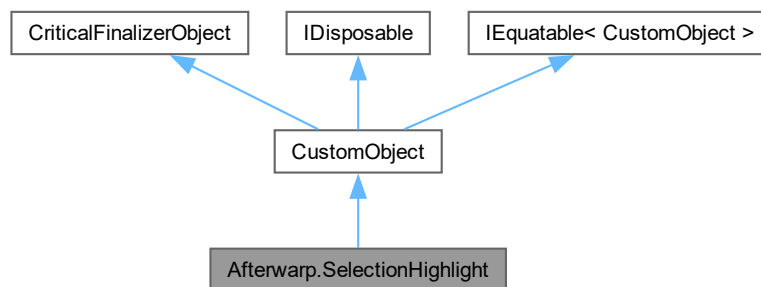
- [Afterwarp.Types.cs](#)

7.92 Afterwarp.SelectionHighlight Class Reference

Inheritance diagram for Afterwarp.SelectionHighlight:



Collaboration diagram for Afterwarp.SelectionHighlight:



Public Member Functions

- `SelectionHighlight` (`Device` device, `Point` size, `PixelFormat` depthStencil=`PixelFormat.D32F`, int samples=8, bool grayscale=false)
Creates new instance of Selection Highlight module.
- void `Clear` ()
Clears the primary rendering textures.
- void `Begin` ()
Begins rendering with the module.
- void `End` ()
Finishes rendering with the module.
- void `Filter` ()
Performs final post-filtering of the textures to produce the highlight selection effect.
- `Texture Texture` (`SelectionHighlightTextureType` type=`SelectionHighlightTextureType.Highlight`)
Returns texture of the given type.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the module.
- [Point Size](#) [get, set]
Returns or updates size of the rendering surfaces.
- [PixelFormat DepthStencil](#) [get]
Pixel format used for depth/stencil buffer.
- int [Samples](#) [get]
Returns number of samples used for multisampling.
- bool [Grayscale](#) [get]
Whether the technique is performed in a more optimal, grayscale mode, instead of full color.
- [SelectionHighlightParameters Parameters](#) [get, set]
Returns or changes parameters that define the behavior and characteristics of the technique.
- bool [Rendering](#) [get]
Indicates that the rendering is currently performed to the primary textures.
- [Quad TextureCoordinates](#) [get]
Returns texture coordinates that can be passed to canvas for rendering the highlight.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.92.1 Detailed Description

A module that shows a selection highlight around one or more 2D and/or 3D objects in multiple colors with an optional outline.

7.92.2 Constructor & Destructor Documentation

7.92.2.1 SelectionHighlight()

```
Afterwarp.SelectionHighlight.SelectionHighlight (
    Device device,
    Point size,
    PixelFormat depthStencil = PixelFormat::D32F,
    int samples = 8,
    bool grayscale = false ) [inline]
```

Creates new instance of Selection Highlight module.

7.92.3 Member Function Documentation

7.92.3.1 Begin()

```
void Afterwarp.SelectionHighlight.Begin ( ) [inline]
```

Begins rendering with the module.

7.92.3.2 Clear()

```
void Afterwarp.SelectionHighlight.Clear ( ) [inline]
```

Clears the primary rendering textures.

7.92.3.3 End()

```
void Afterwarp.SelectionHighlight.End ( )
```

Finishes rendering with the module.

7.92.3.4 Filter()

```
void Afterwarp.SelectionHighlight.Filter ( ) [inline]
```

Performs final post-filtering of the textures to produce the highlight selection effect.

7.92.3.5 Texture()

```
Texture Afterwarp.SelectionHighlight.Texture (
    SelectionHighlightTextureType type = SelectionHighlightTextureType::Highlight )
[inline]
```

Returns texture of the given type.

7.92.4 Property Documentation

7.92.4.1 DepthStencil

`PixelFormat` Afterwarp.SelectionHighlight.DepthStencil [get]

Pixel format used for depth/stencil buffer.

7.92.4.2 Device

`Device` Afterwarp.SelectionHighlight.Device [get]

Returns device associated with the module.

7.92.4.3 Grayscale

`bool` Afterwarp.SelectionHighlight.Grayscale [get]

Whether the technique is performed in a more optimal, grayscale mode, instead of full color.

7.92.4.4 Parameters

`SelectionHighlightParameters` Afterwarp.SelectionHighlight.Parameters [get], [set]

Returns or changes parameters that define the behavior and characteristics of the technique.

7.92.4.5 Rendering

`bool` Afterwarp.SelectionHighlight.Rendering [get]

Indicates that the rendering is currently performed to the primary textures.

7.92.4.6 Samples

`int` Afterwarp.SelectionHighlight.Samples [get]

Returns number of samples used for multisampling.

7.92.4.7 Size

`Point` Afterwarp.SelectionHighlight.Size [get], [set]

Returns or updates size of the rendering surfaces.

7.92.4.8 TextureCoordinates

`Quad Afterwarp.SelectionHighlight.TextureCoordinates [get]`

Returns texture coordinates that can be passed to canvas for rendering the highlight.

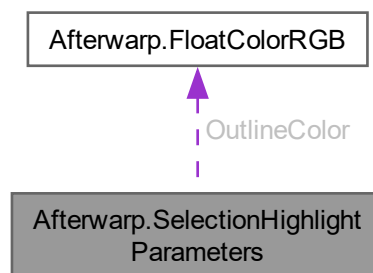
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.93 Afterwarp.SelectionHighlightParameters Struct Reference

Parameters that define how selection highlight technique is performed.

Collaboration diagram for Afterwarp.SelectionHighlightParameters:



Public Member Functions

- [SelectionHighlightParameters](#) ([FloatColorRGB](#) outlineColor, float outlineStart, float glowIntensity, int blur↔ Passes, Vector2 blurOffset)
Creates new selection highlight parameters with the given values.

Public Attributes

- [FloatColorRGB](#) [OutlineColor](#)
Color of the selection glow's outline.
- float [OutlineStart](#)
At which position outline begins (values of 1 or higher mean no outline).
- float [GlowIntensity](#)
How intense should the glow appear.
- int [BlurPasses](#)
Number of passes for the blur effect.
- Vector2 [BlurOffset](#)
Offset for the blur effect.

Properties

- static [SelectionHighlightParameters Default](#) [get]
Returns default selection highlight parameters.

7.93.1 Detailed Description

Parameters that define how selection highlight technique is performed.

7.93.2 Constructor & Destructor Documentation

7.93.2.1 SelectionHighlightParameters()

```
Afterwarp.SelectionHighlightParameters.SelectionHighlightParameters (
    FloatColorRGB outlineColor,
    float outlineStart,
    float glowIntensity,
    int blurPasses,
    Vector2 blurOffset ) [inline]
```

Creates new selection highlight parameters with the given values.

7.93.3 Member Data Documentation

7.93.3.1 BlurOffset

```
Vector2 Afterwarp.SelectionHighlightParameters.BlurOffset
```

Offset for the blur effect.

7.93.3.2 BlurPasses

```
int Afterwarp.SelectionHighlightParameters.BlurPasses
```

Number of passes for the blur effect.

7.93.3.3 GlowIntensity

```
float Afterwarp.SelectionHighlightParameters.GlowIntensity
```

How intense should the glow appear.

7.93.3.4 OutlineColor

```
FloatColorRGB Afterwarp.SelectionHighlightParameters.OutlineColor
```

Color of the selection glow's outline.

7.93.3.5 OutlineStart

```
float Afterwarp.SelectionHighlightParameters.OutlineStart
```

At which position outline begins (values of 1 or higher mean no outline).

7.93.4 Property Documentation

7.93.4.1 Default

```
SelectionHighlightParameters Afterwarp.SelectionHighlightParameters.Default [static], [get]
```

Returns default selection highlight parameters.

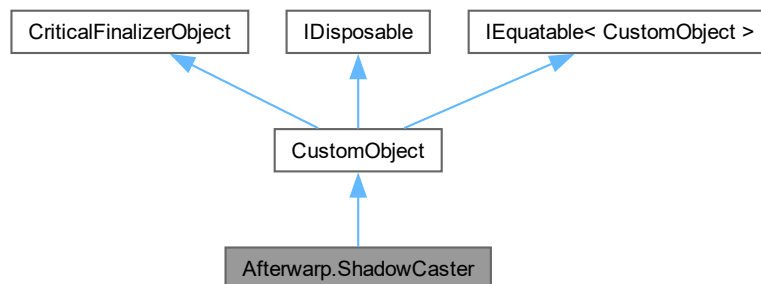
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

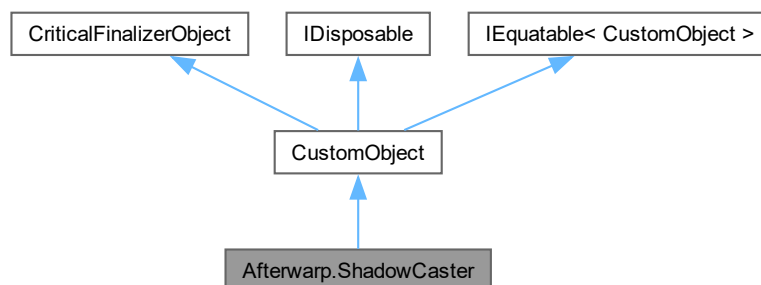
7.94 Afterwarp.ShadowCaster Class Reference

A source that casts shadows that can be shared among multiple light sources.

Inheritance diagram for Afterwarp.ShadowCaster:



Collaboration diagram for Afterwarp.ShadowCaster:



Public Member Functions

- void [Clear](#) ()
Clears the integrated textures.
- void [Begin](#) ()
Starts rendering to shadow caster's texture to build a shadow map.
- void [End](#) ()
Finishes rendering to shadow caster's texture.
- void [Filter](#) ()
Performs filtering on the shadow map and accomodates it on the atlas.
- [Texture GetTexture](#) (int index)
Returns one of the textures currently used by the shadow caster.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the shadow caster.
- [Point Position](#) [get]
Returns position in pixels of the shadow map in parent atlas.
- [Point Size](#) [get]
Returns size of the shadow map in pixels.
- Matrix4x4 [ViewProjection](#) [get, set]
Returns or updates a combined view and projection matrices of the shadow caster.
- bool [Rendering](#) [get]
Indicates whether rendering is taking place.
- [ShadowCastingAtlas Atlas](#) [get]
Returns parent texture atlas associated with the shadow caster.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- [IntPtr _handle](#)

Wrapped object's handle.

7.94.1 Detailed Description

A source that casts shadows that can be shared among multiple light sources.

7.94.2 Member Function Documentation

7.94.2.1 Begin()

```
void Afterwarp.ShadowCaster.Begin ( ) [inline]
```

Starts rendering to shadow caster's texture to build a shadow map.

7.94.2.2 Clear()

```
void Afterwarp.ShadowCaster.Clear ( ) [inline]
```

Clears the integrated textures.

7.94.2.3 End()

```
void Afterwarp.ShadowCaster.End ( )
```

Finishes rendering to shadow caster's texture.

7.94.2.4 Filter()

```
void Afterwarp.ShadowCaster.Filter ( ) [inline]
```

Performs filtering on the shadow map and accomodates it on the atlas.

7.94.2.5 GetTexture()

```
Texture Afterwarp.ShadowCaster.GetTexture (
    int index ) [inline]
```

Returns one of the textures currently used by the shadow caster.

7.94.3 Property Documentation

7.94.3.1 Atlas

[ShadowCastingAtlas](#) Afterwarp.ShadowCaster.Atlas [get]

Returns parent texture atlas associated with the shadow caster.

7.94.3.2 Device

[Device](#) Afterwarp.ShadowCaster.Device [get]

Returns device associated with the shadow caster.

7.94.3.3 Position

[Point](#) Afterwarp.ShadowCaster.Position [get]

Returns position in pixels of the shadow map in parent atlas.

7.94.3.4 Rendering

`bool` Afterwarp.ShadowCaster.Rendering [get]

Indicates whether rendering is taking place.

7.94.3.5 Size

[Point](#) Afterwarp.ShadowCaster.Size [get]

Returns size of the shadow map in pixels.

7.94.3.6 ViewProjection

`Matrix4x4` Afterwarp.ShadowCaster.ViewProjection [get], [set]

Returns or updates a combined view and projection matrices of the shadow caster.

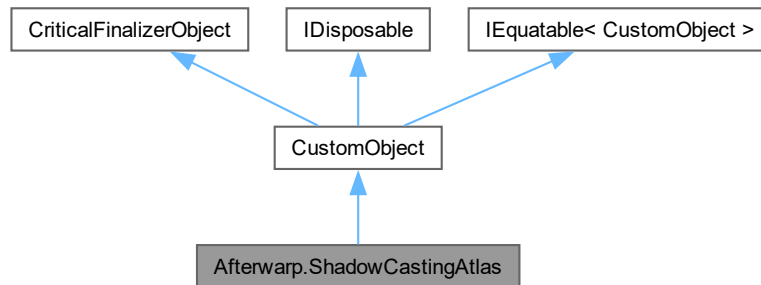
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

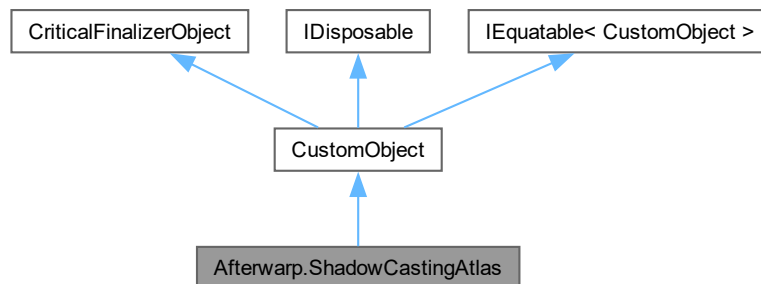
7.95 Afterwarp.ShadowCastingAtlas Class Reference

Module that manages shadow maps produced by one or more shadow casters.

Inheritance diagram for Afterwarp.ShadowCastingAtlas:



Collaboration diagram for Afterwarp.ShadowCastingAtlas:



Public Member Functions

- [ShadowCastingAtlas](#) ([Device](#) device, [Point](#) size, [TechniqueShadows](#) technique=[TechniqueShadows.EVSM](#), int samples=8, int padding=1)
Creates new instance of shadow casting atlas module.
- [ShadowCaster Add](#) ([Point](#) size, bool shared=true)
- void [Erase](#) ([ShadowCaster](#) caster)
Removes an existing shadow caster.
- void [Clear](#) ()
Removes all existing shadow casters.
- [ShadowCaster Caster](#) (int index)
Returns shadow caster for the given index.
- void [BorderFill](#) ()

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device associated with the atlas.
- [TechniqueShadows](#) [Technique](#) [get]
Returns shadow rendering technique.
- [ShadowParameters](#) [Parameters](#) [get, set]
Parameters that define how shadow maps are rendered and their characteristics.
- [Point Size](#) [get]
Returns size of the atlas in pixels.
- int [Samples](#) [get]
Returns number of samples used for shadow map multisampling.
- int [Padding](#) [get]
Returns padding used to accomodate shadow maps.
- [Texture](#) [Texture](#) [get]
Returns shadow mapping atlas texture containing existing shadow maps.
- int [Count](#) [get]
Returns number of existing shadow casters.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.95.1 Detailed Description

Module that manages shadow maps produced by one or more shadow casters.

7.95.2 Constructor & Destructor Documentation

7.95.2.1 ShadowCastingAtlas()

```
Afterwarp.ShadowCastingAtlas.ShadowCastingAtlas (
    Device device,
    Point size,
    TechniqueShadows technique = TechniqueShadows::EVSM,
    int samples = 8,
    int padding = 1 ) [inline]
```

Creates new instance of shadow casting atlas module.

7.95.3 Member Function Documentation

7.95.3.1 Add()

```
ShadowCaster Afterwarp.ShadowCastingAtlas.Add (
    Point size,
    bool shared = true ) [inline]
```

Creates a new shadow caster with the given size and associated with the given view. Shared parameter determines whether the caster should share its shadow map textures with others to reduce memory footprint. In a partial rendering, where more portions are added to shadow map in next frames, the shadow caster must own its shadow map.

7.95.3.2 BorderFill()

```
void Afterwarp.ShadowCastingAtlas.BorderFill ( ) [inline]
```

Clears the surface of shadow mapping atlas texture using border color to avoid edge artifacts. This is typically required after shadow parameters have been changed.

7.95.3.3 Caster()

```
ShadowCaster Afterwarp.ShadowCastingAtlas.Caster (
    int index ) [inline]
```

Returns shadow caster for the given index.

7.95.3.4 Clear()

```
void Afterwarp.ShadowCastingAtlas.Clear ( )
```

Removes all existing shadow casters.

7.95.3.5 Erase()

```
void Afterwarp.ShadowCastingAtlas.Erase (
    ShadowCaster caster )
```

Removes an existing shadow caster.

7.95.4 Property Documentation

7.95.4.1 Count

```
int Afterwarp.ShadowCastingAtlas.Count [get]
```

Returns number of existing shadow casters.

7.95.4.2 Device

```
Device Afterwarp.ShadowCastingAtlas.Device [get]
```

Returns device associated with the atlas.

7.95.4.3 Padding

```
int Afterwarp.ShadowCastingAtlas.Padding [get]
```

Returns padding used to accomodate shadow maps.

7.95.4.4 Parameters

```
ShadowParameters Afterwarp.ShadowCastingAtlas.Parameters [get], [set]
```

Parameters that define how shadow maps are rendered and their characteristics.

7.95.4.5 Samples

```
int Afterwarp.ShadowCastingAtlas.Samples [get]
```

Returns number of samples used for shadow map multisampling.

7.95.4.6 Size

```
Point Afterwarp.ShadowCastingAtlas.Size [get]
```

Returns size of the atlas in pixels.

7.95.4.7 Technique

`TechniqueShadows` `Afterwarp.ShadowCastingAtlas.Technique` [get]

Returns shadow rendering technique.

7.95.4.8 Texture

`Texture` `Afterwarp.ShadowCastingAtlas.Texture` [get]

Returns shadow mapping atlas texture containing existing shadow maps.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.96 Afterwarp.ShadowParameters Struct Reference

Parameters that define how shadows are rendered.

Public Member Functions

- [ShadowParameters](#) (float exponent1, float exponent2, float variance, float bleedSigma, float bias, float blurSigma, int blurSamples)
Creates new shadow parameters with the given values.

Public Attributes

- float [Exponent1](#)
Primary exponent for ESM and EVSM techniques.
- float [Exponent2](#)
Secondary exponent for ESM and EVSM techniques.
- float [Variance](#)
Maximum variance delimiter used with VSM and EVSM techniques.
- float [BleedSigma](#)
Light bleeding sigma for reducing light bleeding artifacts with VSM and EVSM techniques.
- float [Bias](#)
Bias parameter that is applied to depth derivatives in VSM technique.
- float [BlurSigma](#)
Sigma value for shadow map's gaussian blur.
- int [BlurSamples](#)
Number of samples for shadow map's gaussian blur.

Properties

- static [ShadowParameters Default](#) [get]
Returns default shadow parameters.

7.96.1 Detailed Description

Parameters that define how shadows are rendered.

7.96.2 Constructor & Destructor Documentation

7.96.2.1 ShadowParameters()

```
Afterwarp.ShadowParameters.ShadowParameters (
    float exponent1,
    float exponent2,
    float variance,
    float bleedSigma,
    float bias,
    float blurSigma,
    int blurSamples ) [inline]
```

Creates new shadow parameters with the given values.

7.96.3 Member Data Documentation

7.96.3.1 Bias

```
float Afterwarp.ShadowParameters.Bias
```

Bias parameter that is applied to depth derivatives in VSM technique.

7.96.3.2 BleedSigma

```
float Afterwarp.ShadowParameters.BleedSigma
```

Light bleeding sigma for reducing light bleeding artifacts with VSM and EVSM techniques.

7.96.3.3 BlurSamples

```
int Afterwarp.ShadowParameters.BlurSamples
```

Number of samples for shadow map's gaussian blur.

7.96.3.4 BlurSigma

```
float Afterwarp.ShadowParameters.BlurSigma
```

Sigma value for shadow map's gaussian blur.

7.96.3.5 Exponent1

```
float Afterwarp.ShadowParameters.Exponent1
```

Primary exponent for ESM and EVSM techniques.

7.96.3.6 Exponent2

```
float Afterwarp.ShadowParameters.Exponent2
```

Secondary exponent for ESM and EVSM techniques.

7.96.3.7 Variance

```
float Afterwarp.ShadowParameters.Variance
```

Maximum variance delimiter used with VSM and EVSM techniques.

7.96.4 Property Documentation

7.96.4.1 Default

```
ShadowParameters Afterwarp.ShadowParameters.Default [static], [get]
```

Returns default shadow parameters.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.97 Afterwarp.SignedDistanceField Struct Reference

Signed Distance Field (SDF) parameters.

Public Member Functions

- [SignedDistanceField](#) ([SuperSampleSDF](#) superSampleSDF, float signedFieldDistance, Vector3 outlineOffsetSDF, float outlineDistanceMinSDF, float outlineDistanceMaxSDF)
Creates signed distance field parameters with the given values.

Public Attributes

- [SuperSampleSDF](#) [SuperSampleSDF](#)
Type of super-sampling for the field's rendering.
- float [SignedFieldDistance](#)
Distance that was used for generating the field.
- Vector3 [OutlineOffsetSDF](#)
Color offset in YCH color space for rendering the shape's outline.
- float [OutlineDistanceMinSDF](#)
Minimum field distance for the shape's outline.
- float [OutlineDistanceMaxSDF](#)
Maximum field distance for the shape's outline.

Properties

- static [SignedDistanceField Default](#) [get]
Returns default signed distance field (SDF) parameters.

7.97.1 Detailed Description

Signed Distance Field (SDF) parameters.

7.97.2 Constructor & Destructor Documentation

7.97.2.1 SignedDistanceField()

```
Afterwarp.SignedDistanceField.SignedDistanceField (
    SuperSampleSDF superSampleSDF,
    float signedFieldDistance,
    Vector3 outlineOffsetSDF,
    float outlineDistanceMinSDF,
    float outlineDistanceMaxSDF ) [inline]
```

Creates signed distance field parameters with the given values.

7.97.3 Member Data Documentation

7.97.3.1 OutlineDistanceMaxSDF

```
float Afterwarp.SignedDistanceField.OutlineDistanceMaxSDF
```

Maximum field distance for the shape's outline.

7.97.3.2 OutlineDistanceMinSDF

```
float Afterwarp.SignedDistanceField.OutlineDistanceMinSDF
```

Minimum field distance for the shape's outline.

7.97.3.3 OutlineOffsetSDF

`Vector3 Afterwarp.SignedDistanceField.OutlineOffsetSDF`

Color offset in YCH color space for rendering the shape's outline.

7.97.3.4 SignedFieldDistance

`float Afterwarp.SignedDistanceField.SignedFieldDistance`

Distance that was used for generating the field.

7.97.3.5 SuperSampleSDF

`SuperSampleSDF Afterwarp.SignedDistanceField.SuperSampleSDF`

Type of super-sampling for the field's rendering.

7.97.4 Property Documentation

7.97.4.1 Default

`SignedDistanceField Afterwarp.SignedDistanceField.Default [static], [get]`

Returns default signed distance field (SDF) parameters.

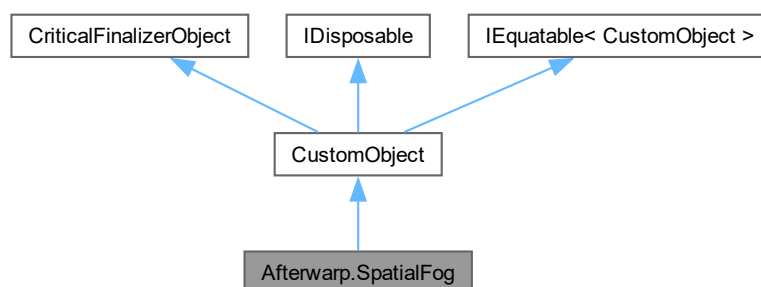
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

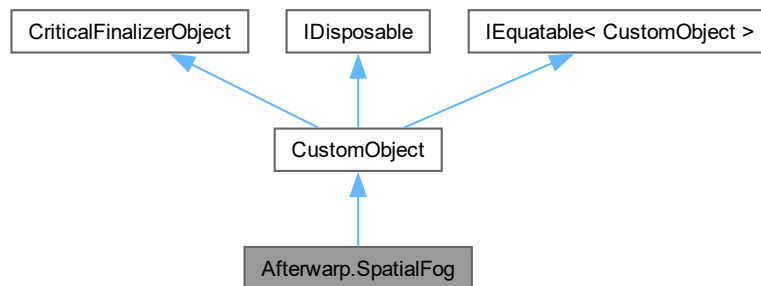
7.98 Afterwarp.SpatialFog Class Reference

Module that performs both ground fog and distance-based fog simultaneously.

Inheritance diagram for `Afterwarp.SpatialFog`:



Collaboration diagram for Afterwarp.SpatialFog:



Public Member Functions

- [SpatialFog](#) ([Device](#) device)
Creates new instance of spatial fog module.
- void [Execute](#) ([Texture](#) destination, [Texture](#) linearDepths)
- void [ExecuteGlassy](#) ([Texture](#) destination, [Texture](#) linearDepths)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the module.
- [FogParameters Parameters](#) [get, set]
Returns or changes parameters that define spatial fog characteristics.
- [Matrix4x4 View](#) [get, set]
Returns or changes 3D view transformation matrix.
- [Matrix4x4 Projection](#) [get, set]
Returns or changes 3D projection transformation matrix.
- [FogFormula Formula](#) [get, set]
Returns or changes spatial fog calculation formula.
- [DepthFogDistance Distance](#) [get, set]
Returns or changes distance calculation formula for the spatial fog.

Properties inherited from [Afterwarp.CustomObject](#)

- [IntPtr Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.98.1 Detailed Description

Module that performs both ground fog and distance-based fog simultaneously.

7.98.2 Constructor & Destructor Documentation

7.98.2.1 SpatialFog()

```
Afterwarp.SpatialFog.SpatialFog (
    Device device ) [inline]
```

Creates new instance of spatial fog module.

7.98.3 Member Function Documentation

7.98.3.1 Execute()

```
void Afterwarp.SpatialFog.Execute (
    Texture destination,
    Texture linearDepths ) [inline]
```

Computes fog and applies it to the destination high-dynamic range (HDR) color texture. The module requires an existing linear depths texture (optionally multisampled) to be provided.

7.98.3.2 ExecuteGlassy()

```
void Afterwarp.SpatialFog.ExecuteGlassy (
    Texture destination,
    Texture linearDepths ) [inline]
```

Computes fog and applies it to the destination "glassy" composition texture. The module requires an existing linear depths texture (optionally multisampled) to be provided.

7.98.4 Property Documentation

7.98.4.1 Device

`Device` Afterwarp.SpatialFog.Device [get]

Returns device associated with the module.

7.98.4.2 Distance

`DepthFogDistance` Afterwarp.SpatialFog.Distance [get], [set]

Returns or changes distance calculation formula for the spatial fog.

7.98.4.3 Formula

`FogFormula` Afterwarp.SpatialFog.Formula [get], [set]

Returns or changes spatial fog calculation formula.

7.98.4.4 Parameters

`FogParameters` Afterwarp.SpatialFog.Parameters [get], [set]

Returns or changes parameters that define spatial fog characteristics.

7.98.4.5 Projection

`Matrix4x4` Afterwarp.SpatialFog.Projection [get], [set]

Returns or changes 3D projection transformation matrix.

7.98.4.6 View

`Matrix4x4` Afterwarp.SpatialFog.View [get], [set]

Returns or changes 3D view transformation matrix.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.99 Afterwarp.RenderingState.State Struct Reference

State flags that can be combined together to form a rendering operation state.

Static Public Attributes

- const uint [DepthTest](#) = 0x0001u
Depth testing should be performed. If this state is not present, depth writing is also disabled.
- const uint [DepthWrite](#) = 0x0002u
Depth values should be written to Depth Buffer.
- const uint [StencilTest](#) = 0x0004u
- const uint [DepthClip](#) = 0x0008u
Geometry should be clipped by the corresponding depth limits.
- const uint [ScissorClip](#) = 0x0010u
Geometry should be clipped by the corresponding scissor rectangles.
- const uint [Wireframe](#) = 0x0020u
Primitives should be rendered in a wireframe mode.
- const uint [CullClockwise](#) = 0x0040u
- const uint [Multisampling](#) = 0x0080u
Geometry should be rendered with multisampling.
- const uint [LineAntialias](#) = 0x0100u
Standard 3D lines should be rendered with antialiasing.
- const uint [ColorWrite](#) = 0x0200u
Writing to color buffer should be performed.
- const uint [BlendEnable](#) = 0x0400u
Blending operation should be performed.
- const uint [AlphaToCoverage](#) = 0x0800u
Alpha channel should be converted to coverage samples.
- const uint [CubeMapSeamless](#) = 0x1000u
Cube Maps should be sampled from neighbor faces at seams.
- const uint [PerSampleShading](#) = 0x2000u

7.99.1 Detailed Description

State flags that can be combined together to form a rendering operation state.

7.99.2 Member Data Documentation

7.99.2.1 AlphaToCoverage

```
const uint Afterwarp.RenderingState.State.AlphaToCoverage = 0x0800u [static]
```

Alpha channel should be converted to coverage samples.

7.99.2.2 BlendEnable

```
const uint Afterwarp.RenderingState.State.BlendEnable = 0x0400u [static]
```

Blending operation should be performed.

7.99.2.3 ColorWrite

```
const uint Afterwarp.RenderingState.State.ColorWrite = 0x0200u [static]
```

Writing to color buffer should be performed.

7.99.2.4 CubeMapSeamless

```
const uint Afterwarp.RenderingState.State.CubeMapSeamless = 0x1000u [static]
```

Cube Maps should be sampled from neighbor faces at seams.

7.99.2.5 CullClockwise

```
const uint Afterwarp.RenderingState.State.CullClockwise = 0x0040u [static]
```

Triangles having vertices in clockwise order would be culled. If this flag is not set, then triangles having vertices in counter-clockwise order would be culled.

7.99.2.6 DepthClip

```
const uint Afterwarp.RenderingState.State.DepthClip = 0x0008u [static]
```

Geometry should be clipped by the corresponding depth limits.

7.99.2.7 DepthTest

```
const uint Afterwarp.RenderingState.State.DepthTest = 0x0001u [static]
```

Depth testing should be performed. If this state is not present, depth writing is also disabled.

7.99.2.8 DepthWrite

```
const uint Afterwarp.RenderingState.State.DepthWrite = 0x0002u [static]
```

Depth values should be written to Depth Buffer.

7.99.2.9 LineAntialias

```
const uint Afterwarp.RenderingState.State.LineAntialias = 0x0100u [static]
```

Standard 3D lines should be rendered with antialiasing.

7.99.2.10 Multisampling

```
const uint Afterwarp.RenderingState.State.Multisampling = 0x0080u [static]
```

Geometry should be rendered with multisampling.

7.99.2.11 PerSampleShading

```
const uint Afterwarp.RenderingState.State.PerSampleShading = 0x2000u [static]
```

Enables fragment shader execution for each individual MSAA sample. This has only effect on OpenGL (ES) based implementations.

7.99.2.12 ScissorClip

```
const uint Afterwarp.RenderingState.State.ScissorClip = 0x0010u [static]
```

Geometry should be clipped by the corresponding scissor rectangles.

7.99.2.13 StencilTest

```
const uint Afterwarp.RenderingState.State.StencilTest = 0x0004u [static]
```

Stencil testing should be performed. If this state is not present, no other stencil operation will occur.

7.99.2.14 Wireframe

```
const uint Afterwarp.RenderingState.State.Wireframe = 0x0020u [static]
```

Primitives should be rendered in a wireframe mode.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.100 Afterwarp.RenderingState.StencilState Struct Reference

Stencil state that is independent for each front and back facing.

Public Member Functions

- [StencilState](#) ([ComparisonFunc](#) func, [StencilOp](#) failOp, [StencilOp](#) depthFailOp, [StencilOp](#) depthPassOp)
Creates stencil state structure with the given values.

Public Attributes

- [ComparisonFunc Func](#)
Stencil test function.
- [StencilOp FailOp](#)
Action to take when stencil test fails.
- [StencilOp DepthFailOp](#)
Action to take when stencil test passes but depth test fails.
- [StencilOp DepthPassOp](#)

Properties

- static [StencilState Default](#) [get]
Returns default stencil state.

7.100.1 Detailed Description

Stencil state that is independent for each front and back facing.

7.100.2 Constructor & Destructor Documentation

7.100.2.1 StencilState()

```
Afterwarp.RenderingState.StencilState.StencilState (  
    ComparisonFunc func,  
    StencilOp failOp,  
    StencilOp depthFailOp,  
    StencilOp depthPassOp ) [inline]
```

Creates stencil state structure with the given values.

7.100.3 Member Data Documentation

7.100.3.1 DepthFailOp

[StencilOp](#) Afterwarp.RenderingState.StencilState.DepthFailOp

Action to take when stencil test passes but depth test fails.

7.100.3.2 DepthPassOp

[StencilOp](#) Afterwarp.RenderingState.StencilState.DepthPassOp

Action to take when both stencil and depth tests pass, or when stencil test passes and either there is no depth buffer or depth testing is disabled.

7.100.3.3 FailOp

`StencilOp Afterwarp.RenderingState.StencilState.FailOp`

Action to take when stencil test fails.

7.100.3.4 Func

`ComparisonFunc Afterwarp.RenderingState.StencilState.Func`

Stencil test function.

7.100.4 Property Documentation

7.100.4.1 Default

`StencilState Afterwarp.RenderingState.StencilState.Default [static], [get]`

Returns default stencil state.

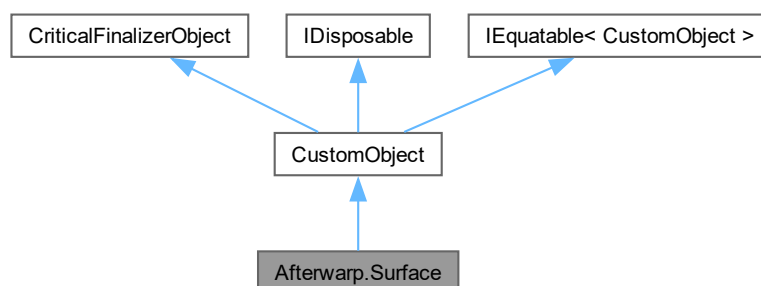
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

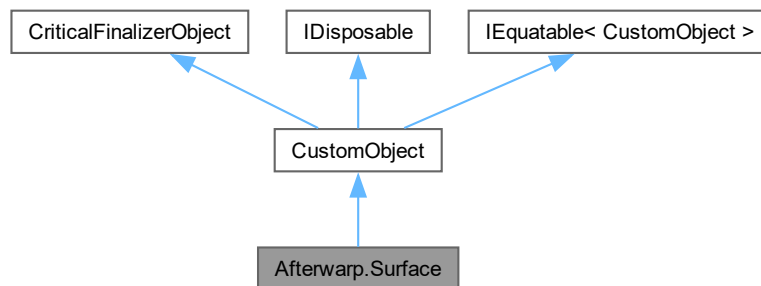
7.101 Afterwarp.Surface Class Reference

Surface that contains image in memory for pixel manipulation.

Inheritance diagram for Afterwarp.Surface:



Collaboration diagram for Afterwarp.Surface:



Public Member Functions

- [Surface](#) (int width, int height, [PixelFormat](#) format=[PixelFormat.Unknown](#))
- [Surface](#) (string fileName, [AlphaFormatRequest](#) formatRequest=[AlphaFormatRequest.DontCare](#))
- [Surface](#) (Array fileContents, string extension, [AlphaFormatRequest](#) formatRequest=[AlphaFormatRequest.DontCare](#))
- IntPtr [GetScanline](#) (int index)

Provides unmanaged pointer to a first pixel at the given scanline index (that is, row number).
- IntPtr [GetPixelPtr](#) (int x, int y)

Provides unmanaged pointer to the pixel data at the given coordinates.
- void [Resize](#) (int width, int height, [PixelFormat](#) format=[PixelFormat.Unknown](#))

Redefines surface size to the specified width, height and pixel format, discarding previous contents.
- void [ConvertFormat](#) ([PixelFormat](#) format)

Converts surface from its current format to another one.
- uint [GetPixel](#) (int x, int y)

Reads a single pixel from the surface.
- void [SetPixel](#) (int x, int y, uint color)

Writes a single pixel to the surface.
- uint [GetPixelBilinear](#) (float x, float y)
- void [Copy](#) ([Surface](#) source, [Rect?](#) sourceRect=null, [Point?](#) destPos=null)
- void [Copy](#) ([Surface](#) source)
- void [Stretch](#) ([Surface](#) source, [RectF?](#) destRect=null, [RectF?](#) sourceRect=null)
- void [StretchBilinear](#) ([Surface](#) source, [RectF?](#) destRect=null, [RectF?](#) sourceRect=null)
- void [Clear](#) ()

Clears the entire surface with zeros.
- void [Clear](#) (uint color)

Clears the entire surface with a given color. This does pixel format conversion when appropriate.
- void [ResetAlpha](#) (bool opaque=true)

Processes surface pixels, setting alpha-channel to either fully translucent or fully opaque.
- void [PremultiplyAlpha](#) ()
- void [UnpremultiplyAlpha](#) ()
- void [Mirror](#) ()

Mirrors the visible image on surface horizontally.
- void [Flip](#) ()

Flips the visible image on surface vertically.
- void [Invert](#) ()

Inverts colors in the surface.

- void [ShrinkFrom](#) ([Surface](#) source)
- void [SaveToFile](#) (string fileName, int quality=50)
- uint [SaveToFileInMemory](#) (byte[] fileContents, string extension, int quality=50)
- void [MakeSignedDistanceField](#) ([Surface](#) source, float spread=1.0f, [Point?](#) destPos=null, [Rect?](#) sourceRect=null)

Calculates signed distance field from alpha values of the source.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- IntPtr [Bits](#) [get]
- uint [Pitch](#) [get]
- int [BytesPerPixel](#) [get]
Returns number of bytes each pixel occupies.
- int [Width](#) [get]
Returns surface width in pixels.
- int [Height](#) [get]
Returns surface height in pixels.
- uint [ByteSize](#) [get]
Returns surface size in bytes.
- [PixelFormat](#) [Format](#) [get]
Returns pixel format in which pixels are stored.
- bool [PremultipliedAlpha](#) [get, set]
- bool [Empty](#) [get]
- bool [HasAlphaChannel](#) [get]

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.101.1 Detailed Description

Surface that contains image in memory for pixel manipulation.

7.101.2 Constructor & Destructor Documentation

7.101.2.1 Surface() [1/3]

```
Afterwarp.Surface.Surface (
    int width,
    int height,
    PixelFormat format = PixelFormat::Unknown ) [inline]
```

Creates surface with the given width, height and pixel format. If PixelFormat.Unknown is specified, then default pixel format would be picked that best represents the current platform.

7.101.2.2 Surface() [2/3]

```
Afterwarp.Surface.Surface (
    string fileName,
    AlphaFormatRequest formatRequest = AlphaFormatRequest::DontCare ) [inline]
```

Creates a new instance of raster surface containing image loaded from external file. The preference for premultiplied or non-premultiplied alpha-channel is merely a suggestion, which can influence how the image is loaded (depending on underlying implementation). This function typically supports most common file formats such as BMP, PNG, JPEG, but may also support other formats like GIF and TIFF.

7.101.2.3 Surface() [3/3]

```
Afterwarp.Surface.Surface (
    Array fileContents,
    string extension,
    AlphaFormatRequest formatRequest = AlphaFormatRequest::DontCare ) [inline]
```

Creates a new instance of raster surface containing image loaded from file that has been preloaded into memory. The internal file format is determined by "extension" parameter, which should contain a valid file extension such as ".png" (case-insensitive). The preference for premultiplied or non-premultiplied alpha-channel is merely a suggestion, which can influence how the image is loaded (depending on underlying implementation). This function typically supports most common file formats such as BMP, PNG, JPEG, but may also support other formats like GIF and TIFF.

7.101.3 Member Function Documentation

7.101.3.1 Clear() [1/2]

```
void Afterwarp.Surface.Clear ( )
```

Clears the entire surface with zeros.

7.101.3.2 Clear() [2/2]

```
void Afterwarp.Surface.Clear (
    uint color ) [inline]
```

Clears the entire surface with a given color. This does pixel format conversion when appropriate.

7.101.3.3 ConvertFormat()

```
void Afterwarp.Surface.ConvertFormat (
    PixelFormat format ) [inline]
```

Converts surface from its current format to another one.

7.101.3.4 Copy() [1/2]

```
void Afterwarp.Surface.Copy (
    Surface source ) [inline]
```

Copies entire contents from a source surface to the current one. If the current surface has size and/or pixel format not specified, these will be copied from the source surface as well. If current surface is not empty, then its pixel format will not be modified - in this case, pixel format conversion may occur. This function will try to ensure that current surface size matches the source surface and if this cannot be achieved, it will fail.

7.101.3.5 Copy() [2/2]

```
void Afterwarp.Surface.Copy (
    Surface source,
    Rect? sourceRect = null,
    Point? destPos = null ) [inline]
```

Copies a portion of source surface to the current one according to specified source rectangle and destination position. If source rectangle is empty (or null), then the entire source surface will be copied. This function does the appropriate clipping and pixel format conversion. It does not change current surface size or pixel format. Note that "premultiplied alpha" flag of either current or source surfaces is completely ignored.

7.101.3.6 Flip()

```
void Afterwarp.Surface.Flip ( ) [inline]
```

Flips the visible image on surface vertically.

7.101.3.7 GetPixel()

```
uint Afterwarp.Surface.GetPixel (
    int x,
    int y )
```

Reads a single pixel from the surface.

7.101.3.8 GetPixelBilinear()

```
uint Afterwarp.Surface.GetPixelBilinear (
    float x,
    float y )
```

Retrieves pixel from floating-point coordinates, interpolating linearly between four neighbor pixels as necessary to get an accurate match. This can be used for limitless stretching, to get color values that lie between individual pixels and slowly change from one pixel to another.

7.101.3.9 GetPixelPtr()

```
IntPtr Afterwarp.Surface.GetPixelPtr (
    int x,
    int y )
```

Provides unmanaged pointer to the pixel data at the given coordinates.

7.101.3.10 GetScanline()

```
IntPtr Afterwarp.Surface.GetScanline (
    int index )
```

Provides unmanaged pointer to a first pixel at the given scanline index (that is, row number).

7.101.3.11 Invert()

```
void Afterwarp.Surface.Invert ( ) [inline]
```

Inverts colors in the surface.

7.101.3.12 MakeSignedDistanceField()

```
void Afterwarp.Surface.MakeSignedDistanceField (
    Surface source,
    float spread = 1f,
    Point? destPos = null,
    Rect? sourceRect = null ) [inline]
```

Calculates signed distance field from alpha values of the source.

7.101.3.13 Mirror()

```
void Afterwarp.Surface.Mirror ( ) [inline]
```

Mirrors the visible image on surface horizontally.

7.101.3.14 PremultiplyAlpha()

```
void Afterwarp.Surface.PremultiplyAlpha ( ) [inline]
```

Processes the whole surface, premultiplying each pixel's red, green and blue values by the corresponding alpha-channel value, resulting in image with premultiplied alpha. Note that this is an irreversible process, during which some color information is lost permanently (smaller alpha values contribute to bigger information loss). This is generally useful to prepare the image for generating mipmaps and/or alpha-blending, to get more accurate visual results.

7.101.3.15 ResetAlpha()

```
void Afterwarp.Surface.ResetAlpha (
    bool opaque = true ) [inline]
```

Processes surface pixels, setting alpha-channel to either fully translucent or fully opaque.

7.101.3.16 Resize()

```
void Afterwarp.Surface.Resize (
    int width,
    int height,
    PixelFormat format = PixelFormat::Unknown ) [inline]
```

Redefines surface size to the specified width, height and pixel format, discarding previous contents.

7.101.3.17 SaveToFile()

```
void Afterwarp.Surface.SaveToFile (
    string fileName,
    int quality = 50 ) [inline]
```

Saves the contents of the surface to external file. This function typically supports most common file formats such as BMP, PNG, JPEG, but may also support other formats like GIF and TIFF. Quality parameter, in case of formats such as JPEG, determines the compression ratio (between 0 and 100).

7.101.3.18 SaveToFileInMemory()

```
uint Afterwarp.Surface.SaveToFileInMemory (
    byte[] fileContents,
    string extension,
    int quality = 50 ) [inline]
```

Saves the contents of surface to a file in memory and returns the size of resulting file. This function will save up to the size in bytes of given array, even if it means that not a complete file will be saved. If "fileContents" is null, then this function will provide an estimate size required to store the file in memory. This function typically supports most common file formats such as BMP, PNG, JPEG, but may also support other formats like GIF and TIFF. Quality parameter, in case of formats such as JPEG, determines the compression ratio (between 0 and 100).

7.101.3.19 SetPixel()

```
void Afterwarp.Surface.SetPixel (
    int x,
    int y,
    uint color ) [inline]
```

Writes a single pixel to the surface.

7.101.3.20 ShrinkFrom()

```
void Afterwarp.Surface.ShrinkFrom (
    Surface source ) [inline]
```

This function works similarly to Copy(), except that it produces image with half of size, averaging each four pixels to one. This is specifically useful to generate mipmaps.

7.101.3.21 Stretch()

```
void Afterwarp.Surface.Stretch (
    Surface source,
    RectF? destRect = null,
    RectF? sourceRect = null ) [inline]
```

This function works similarly to Copy(), but provides stretching and/or shrinking. That is, it copies source surface rectangle onto destination rectangle with point filtering. Clipping and pixel format conversion is done as necessary.

7.101.3.22 StretchBilinear()

```
void Afterwarp.Surface.StretchBilinear (
    Surface source,
    RectF? destRect = null,
    RectF? sourceRect = null ) [inline]
```

This function works similarly to Copy(), but provides bilinear stretching. That is, it copies source surface rectangle onto destination rectangle with linear filtering. Clipping and pixel format conversion is done as necessary. Note that this function is meant for stretching only; shrinking although will also work, but result in inaccurate results as shrinking requires calculating average of variable number of pixels depending on shrink ratio.

7.101.3.23 UnpremultiplyAlpha()

```
void Afterwarp.Surface.UnpremultiplyAlpha ( ) [inline]
```

Processes the whole surface, dividing each pixel by its alpha-value, resulting in image with non-premultiplied alpha. During this process, some color information may be lost due to precision issues. This can be useful to obtain original pixel information from image that has been previously premultiplied; however, this does not recover lost information during premultiplication process. For instance, pixels that had alpha value of zero and were premultiplied, lose all information and cannot be recovered; pixels with alpha value of 128 (that is, 50% opaque) lose half of their precision and after "unpremultiply" process will have values multiples of 2.

7.101.4 Property Documentation

7.101.4.1 Bits

```
IntPtr Afterwarp.Surface.Bits [get]
```

Returns unmanaged pointer to top/left corner of pixel data contained by this surface, with horizontal rows arranged linearly from top to bottom.

7.101.4.2 ByteSize

```
nuint Afterwarp.Surface.ByteSize [get]
```

Returns surface size in bytes.

7.101.4.3 BytesPerPixel

```
int Afterwarp.Surface.BytesPerPixel [get]
```

Returns number of bytes each pixel occupies.

7.101.4.4 Empty

```
bool Afterwarp.Surface.Empty [get]
```

Checks whether the surface has width or height set to zero, has undefined pixel format, unknown pitch, unknown bytes per pixel, or access bits set to null. If this returns false, then the surface can be considered valid.

7.101.4.5 Format

```
PixelFormat Afterwarp.Surface.Format [get]
```

Returns pixel format in which pixels are stored.

7.101.4.6 HasAlphaChannel

```
bool Afterwarp.Surface.HasAlphaChannel [get]
```

Processes the whole surface to determine whether it has meaningful alpha-channel. A surface that has all its pixels with alpha-channel set to fully translucent or fully opaque (but not mixed) is considered lacking alpha-channel. On the other hand, a surface that has at least one pixel with alpha-channel value different than any other pixel, is considered to have alpha-channel. This is useful to determine whether the surface can be stored in one of pixel formats lacking alpha-channel, to avoid losing any transparency information.

7.101.4.7 Height

```
int Afterwarp.Surface.Height [get]
```

Returns surface height in pixels.

7.101.4.8 Pitch

```
uint Afterwarp.Surface.Pitch [get]
```

Returns number of bytes each horizontal row of pixels occupies. This may differ from an actual calculated number and may include unused or even protected memory locations, which should be considered read-only and be skipped.

7.101.4.9 PremultipliedAlpha

```
bool Afterwarp.Surface.PremultipliedAlpha [get], [set]
```

Indicates whether the pixels in this surface have their alpha premultiplied or not. This is just an informative parameter; to actually convert pixels from one mode to another, use `PremultiplyAlpha()` and `UnpremultiplyAlpha()` methods.

7.101.4.10 Width

```
int Afterwarp.Surface.Width [get]
```

Returns surface width in pixels.

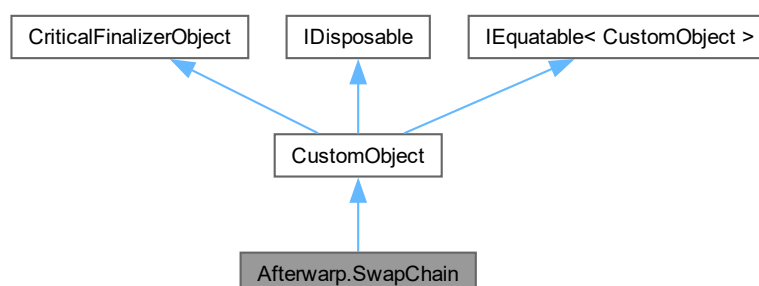
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

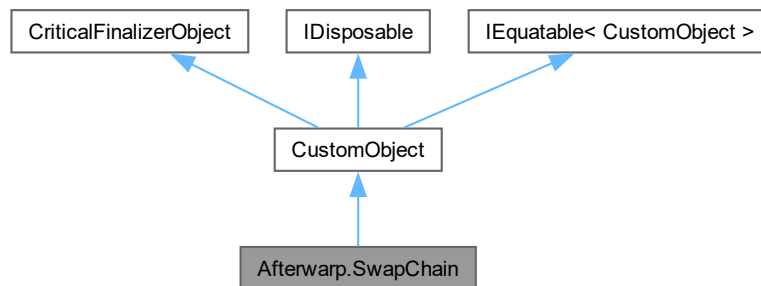
7.102 Afterwarp.SwapChain Class Reference

Swap-chain that enables double-buffered rendering to a particular window.

Inheritance diagram for `Afterwarp.SwapChain`:



Collaboration diagram for Afterwarp.SwapChain:



Public Member Functions

- [SwapChain](#) ([Device](#) device, [IntPtr](#) windowHandle, [Point?](#) size=null, [PixelFormat](#) format=[PixelFormat.Unknown](#), [PixelFormat](#) depthStencil=[PixelFormat.Unknown](#), int multisamples=0, bool vsync=false)
- void [Resize](#) ([Point](#) size)
Resizes the swap-chain.
- void [Begin](#) ()
Begins rendering on the swap-chain.
- void [End](#) ()
Finishes rendering on the swap-chain and flips the buffers.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device associated with this swap-chain.
- [IntPtr](#) [WindowHandle](#) [get]
Returns handle of the associated window.
- int [Width](#) [get]
Returns swap-chain width.
- int [Height](#) [get]
Returns swap-chain height.
- [PixelFormat](#) [Format](#) [get]
Returns pixel format of swap-chain rendering surface.
- [PixelFormat](#) [DepthStencil](#) [get]
Returns pixel format of swap-chain depth/stencil surface.
- int [Multisamples](#) [get]
Returns number of samples of swap-chain surfaces.
- bool [VSync](#) [get]
Returns whether to wait for a vertical retrace.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.102.1 Detailed Description

Swap-chain that enables double-buffered rendering to a particular window.

7.102.2 Constructor & Destructor Documentation

7.102.2.1 SwapChain()

```
Afterwarp.SwapChain.SwapChain (
    Device device,
    IntPtr windowHandle,
    Point? size = null,
    PixelFormat format = PixelFormat::Unknown,
    PixelFormat depthStencil = PixelFormat::Unknown,
    int multisamples = 0,
    bool vsync = false ) [inline]
```

Creates a new rendering swap-chain associated with the given device and window handle. Once the swap-chain has been created for a particular window under OpenGL-based backends, it would not be possible to remove this association without re-creating the window.

7.102.3 Member Function Documentation

7.102.3.1 Begin()

```
void Afterwarp.SwapChain.Begin ( ) [inline]
```

Begins rendering on the swap-chain.

7.102.3.2 End()

```
void Afterwarp.SwapChain.End ( )
```

Finishes rendering on the swap-chain and flips the buffers.

7.102.3.3 Resize()

```
void Afterwarp.SwapChain.Resize (
    Point size ) [inline]
```

Resizes the swap-chain.

7.102.4 Property Documentation

7.102.4.1 DepthStencil

```
PixelFormat Afterwarp.SwapChain.DepthStencil [get]
```

Returns pixel format of swap-chain depth/stencil surface.

7.102.4.2 Device

```
Device Afterwarp.SwapChain.Device [get]
```

Returns device associated with this swap-chain.

7.102.4.3 Format

```
PixelFormat Afterwarp.SwapChain.Format [get]
```

Returns pixel format of swap-chain rendering surface.

7.102.4.4 Height

```
int Afterwarp.SwapChain.Height [get]
```

Returns swap-chain height.

7.102.4.5 Multisamples

```
int Afterwarp.SwapChain.Multisamples [get]
```

Returns number of samples of swap-chain surfaces.

7.102.4.6 VSync

```
bool Afterwarp.SwapChain.VSync [get]
```

Returns whether to wait for a vertical retrace.

7.102.4.7 Width

```
int Afterwarp.SwapChain.Width [get]
```

Returns swap-chain width.

7.102.4.8 WindowHandle

```
IntPtr Afterwarp.SwapChain.WindowHandle [get]
```

Returns handle of the associated window.

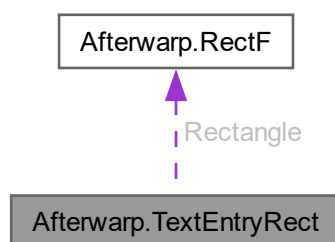
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.103 Afterwarp.TextEntryRect Struct Reference

Individual text entry that will appear as rendered.

Collaboration diagram for Afterwarp.TextEntryRect:



Public Attributes

- nint [Position](#)
Starting character position inside the text (for Unicode, first byte).
- [RectF Rectangle](#)
Character's rectangle.

7.103.1 Detailed Description

Individual text entry that will appear as rendered.

7.103.2 Member Data Documentation

7.103.2.1 Position

```
nint Afterwarp.TextEntryRect.Position
```

Starting character position inside the text (for Unicode, first byte).

7.103.2.2 Rectangle

```
RectF Afterwarp.TextEntryRect.Rectangle
```

Character's rectangle.

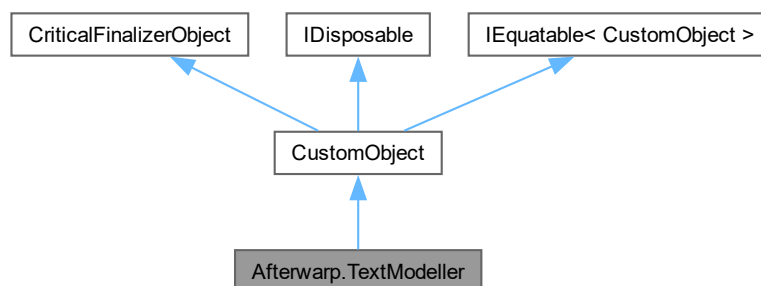
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

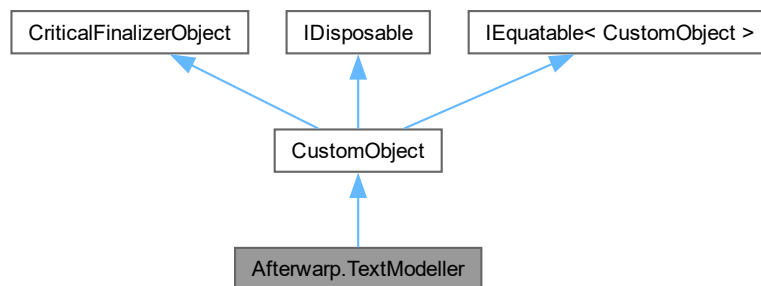
7.104 Afterwarp.TextModeller Class Reference

2D and 3D text rendering module.

Inheritance diagram for Afterwarp.TextModeller:



Collaboration diagram for Afterwarp.TextModeller:



Classes

- class [FontProvider](#)
A provider of font geometry for the modeller.

Public Member Functions

- [TextModeller](#) (Device device)
Creates new instance of text renderer associated with a particular canvas.
- [FontProvider SetFontParameters](#) ([FontParameters](#) parameters)
- [Vector2 Extent](#) (string text, [TextRenderModifiers?](#) modifiers=null)
Returns the logical dimensions that the given text will occupy when rendered.
- [RectF ExtentByShape](#) (string text, [TextRenderModifiers?](#) modifiers=null)
Calculates a physical shape area that a given text occupies when rendered at zero position.
- [Vector2 TextRects](#) (string text, out [TextEntryRect\[\]](#) rects, [TextRenderModifiers?](#) modifiers=null)
- void [Draw](#) ([Canvas](#) canvas, [Vector2](#) position, string text, [ColorPair](#) colors, [TextAlignment](#) align↔
Horiz=[TextAlignment.Middle](#), [TextAlignment](#) alignVert=[TextAlignment.Middle](#), bool alignByShape=false, [TextRenderModifiers?](#) modifiers=null, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
Draws 2D text at the given position with specific alignment directly to canvas.
- void [Draw](#) ([CanvasBuffer](#) buffer, [Vector2](#) position, string text, [ColorPair](#) colors, [TextAlignment](#) align↔
Horiz=[TextAlignment.Middle](#), [TextAlignment](#) alignVert=[TextAlignment.Middle](#), bool alignByShape=false, [TextRenderModifiers?](#) modifiers=null)
Draws 2D text at the given position with specific alignment to a canvas buffer.
- void [DrawCurved](#) ([Canvas](#) canvas, [Vector2](#) position, string text, float radius, float angle, [ColorPair](#) colors, [TextRenderModifiers?](#) modifiers=null, [BlendingEffect](#) effect=[BlendingEffect.Normal](#))
- void [DrawCurved](#) ([CanvasBuffer](#) buffer, [Vector2](#) position, string text, float radius, float angle, [ColorPair](#) colors, [TextRenderModifiers?](#) modifiers=null)
- void [Draw](#) ([Vector3](#) position, string text, float depth, [ColorPair](#) colors, [TextAlignment](#) alignHoriz=[TextAlignment.Middle](#), [TextAlignment](#) alignVert=[TextAlignment.Middle](#), [TextAlignment](#) alignDepth=[TextAlignment.Start](#), bool align↔
ByShape=false, [TextRenderModifiers?](#) modifiers=null)
- void [DrawCurved](#) ([Vector3](#) position, string text, float radius, float angle, float depth, [ColorPair](#) colors, [TextAlignment](#) alignDepth=[TextAlignment.Start](#), [TextRenderModifiers?](#) modifiers=null)
- void [DrawDepthCurved](#) ([Vector3](#) position, string text, float radius, float angle, float depth, [ColorPair](#) colors, [TextRenderModifiers?](#) modifiers=null)
- void [Clear](#) ()

- *Clears GPU buffers for rendering 3D text.*
- void [Prepare](#) ()
Prepares and fills GPU buffers with an existing 3D text rendering data.
- void [Render](#) ([Program](#) program, uint channel=0)
Issues draw call with existing GPU buffers attached.
- void [CopyToMeshBuffer](#) ([MeshBuffer](#) buffer)
Issues draw call with existing GPU buffers attached.
- void [Reset](#) ()
Releases existing resources, character records and buffers.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device](#) [Device](#) [get]
Returns device class associated with the text modeller.
- [FontProvider](#) [Provider](#) [get, set]
- Matrix4x4 [Transform](#) [get, set]
Returns or changes an existing 3D transformation matrix.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.104.1 Detailed Description

2D and 3D text rendering module.

7.104.2 Constructor & Destructor Documentation

7.104.2.1 TextModeller()

```
Afterwarp.TextModeller.TextModeller (
    Device device ) [inline]
```

Creates new instance of text renderer associated with a particular canvas.

7.104.3 Member Function Documentation

7.104.3.1 Clear()

```
void Afterwarp.TextModeller.Clear ( )
```

Clears GPU buffers for rendering 3D text.

7.104.3.2 CopyToMeshBuffer()

```
void Afterwarp.TextModeller.CopyToMeshBuffer (
    MeshBuffer buffer ) [inline]
```

Issues draw call with existing GPU buffers attached.

7.104.3.3 Draw() [1/3]

```
void Afterwarp.TextModeller.Draw (
    Canvas canvas,
    Vector2 position,
    string text,
    ColorPair colors,
    TextAlignment alignHoriz = TextAlignment::Middle,
    TextAlignment alignVert = TextAlignment::Middle,
    bool alignByShape = false,
    TextRenderModifiers? modifiers = null,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws 2D text at the given position with specific alignment directly to canvas.

7.104.3.4 Draw() [2/3]

```
void Afterwarp.TextModeller.Draw (
    CanvasBuffer buffer,
    Vector2 position,
    string text,
    ColorPair colors,
    TextAlignment alignHoriz = TextAlignment::Middle,
    TextAlignment alignVert = TextAlignment::Middle,
    bool alignByShape = false,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws 2D text at the given position with specific alignment to a canvas buffer.

7.104.3.5 Draw() [3/3]

```
void Afterwarp.TextModeller.Draw (
    Vector3 position,
    string text,
    float depth,
    ColorPair colors,
    TextAlignment alignHoriz = TextAlignment::Middle,
    TextAlignment alignVert = TextAlignment::Middle,
    TextAlignment alignDepth = TextAlignment::Start,
    bool alignByShape = false,
    TextRenderModifiers? modifiers = null ) [inline]
```

Renders a volumetric 3D text at the given position, orientation and alignment. The text is oriented in XZ plane with front side oriented towards positive Y axis.

7.104.3.6 DrawCurved() [1/3]

```
void Afterwarp.TextModeller.DrawCurved (
    Canvas canvas,
    Vector2 position,
    string text,
    float radius,
    float angle,
    ColorPair colors,
    TextRenderModifiers? modifiers = null,
    BlendingEffect effect = BlendingEffect::Normal ) [inline]
```

Draws a curved 2D text at the given position with specific alignment directly to canvas. Note: if angle is "NaN", then the text is centered vertically according to the curvature; that is, if the radius would converge to infinity, the text would converge to look like if it wasn't curved.

7.104.3.7 DrawCurved() [2/3]

```
void Afterwarp.TextModeller.DrawCurved (
    CanvasBuffer buffer,
    Vector2 position,
    string text,
    float radius,
    float angle,
    ColorPair colors,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws a curved 2D text at the given position with specific alignment to a canvas buffer. Note: if angle is "NaN", then the text is centered vertically according to the curvature; that is, if the radius would converge to infinity, the text would converge to look like if it wasn't curved.

7.104.3.8 DrawCurved() [3/3]

```
void Afterwarp.TextModeller.DrawCurved (
    Vector3 position,
    string text,
    float radius,
    float angle,
    float depth,
    ColorPair colors,
    TextAlignment alignDepth = TextAlignment::Start,
    TextRenderModifiers? modifiers = null ) [inline]
```

Renders a curved volumetric 3D text at the given position, orientation and alignment. The text is oriented in XZ plane with front side oriented towards positive Y axis, and the rotation is performed around Y axis. Note: if angle is "NaN", then the text is centered vertically according to the curvature; that is, if the radius would converge to infinity, the text would converge to look like if it wasn't curved.

7.104.3.9 DrawDepthCurved()

```
void Afterwarp.TextModeller.DrawDepthCurved (
    Vector3 position,
    string text,
    float radius,
    float angle,
    float depth,
    ColorPair colors,
    TextRenderModifiers? modifiers = null ) [inline]
```

Renders a curved volumetric 3D text at the given position, orientation and alignment. The text is oriented in XZ plane with front side oriented towards positive Y axis, and the rotation is performed around Z axis. Note: if angle is "NaN", then the text is centered vertically according to the curvature; that is, if the radius would converge to infinity, the text would converge to look like if it wasn't curved.

7.104.3.10 Extent()

```
Vector2 Afterwarp.TextModeller.Extent (
    string text,
    TextRenderModifiers? modifiers = null ) [inline]
```

Returns the logical dimensions that the given text will occupy when rendered.

7.104.3.11 ExtentByShape()

```
RectF Afterwarp.TextModeller.ExtentByShape (
    string text,
    TextRenderModifiers? modifiers = null ) [inline]
```

Calculates a physical shape area that a given text occupies when rendered at zero position.

7.104.3.12 Prepare()

```
void Afterwarp.TextModeller.Prepare ( ) [inline]
```

Prepares and fills GPU buffers with an existing 3D text rendering data.

7.104.3.13 Render()

```
void Afterwarp.TextModeller.Render (
    Program program,
    uint channel = 0 ) [inline]
```

Issues draw call with existing GPU buffers attached.

7.104.3.14 Reset()

```
void Afterwarp.TextModeller.Reset ( )
```

Releases existing resources, character records and buffers.

7.104.3.15 SetFontParameters()

```
FontProvider Afterwarp.TextModeller.SetFontParameters (
    FontParameters parameters ) [inline]
```

Changes currently specified font parameters used by the text modeller. Internally, this searches for an existing font provider and if such is not found, creates a new one with the given parameters. Therefore, when working with multiple fonts, it is much more efficient to save provider returned by this function and then assign it directly, rather than setting the same font parameters again.

7.104.3.16 TextRects()

```
Vector2 Afterwarp.TextModeller.TextRects (
    string text,
    out TextEntryRect[] rects,
    TextRenderModifiers? modifiers = null ) [inline]
```

Provides information regarding individual character position and sizes for the given text string when rendered. This can be useful for components such as text edit box, for highlighting and selecting different characters. Returns the actual width and height of the text.

7.104.4 Property Documentation

7.104.4.1 Device

```
Device Afterwarp.TextModeller.Device [get]
```

Returns device class associated with the text modeller.

7.104.4.2 Provider

`FontProvider` `Afterwarp.TextModeller.Provider` `[get]`, `[set]`

Returns or changes an existing font provider. When working with multiple fonts, changing provider is significantly faster than setting font attributes.

7.104.4.3 Transform

`Matrix4x4` `Afterwarp.TextModeller.Transform` `[get]`, `[set]`

Returns or changes an existing 3D transformation matrix.

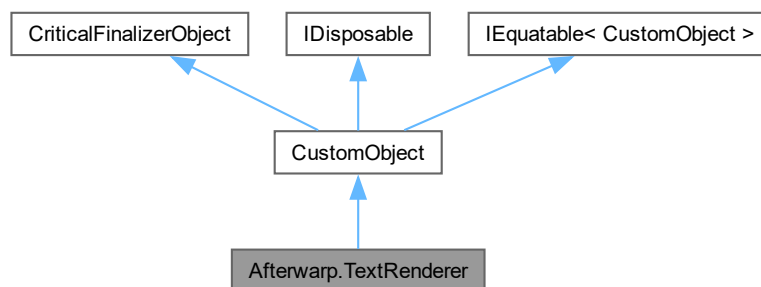
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

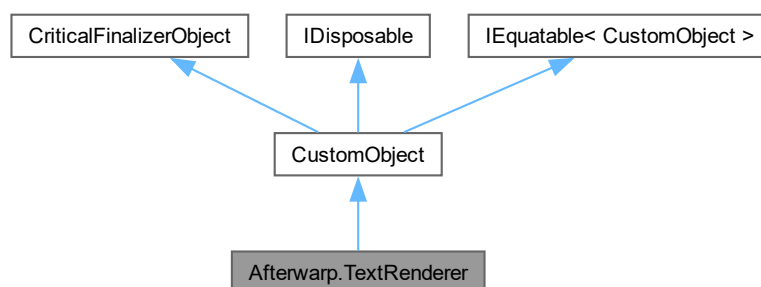
7.105 Afterwarp.TextRenderer Class Reference

Text renderer utility that can draw text on the canvas.

Inheritance diagram for `Afterwarp.TextRenderer`:



Collaboration diagram for `Afterwarp.TextRenderer`:



Public Member Functions

- [TextRenderer](#) ([Canvas](#) canvas, [Point](#) textureSize, [PixelFormat](#) pixelFormat=[PixelFormat.Unknown](#), bool mip↵ Mapping=false)
Creates new instance of text renderer associated with a particular canvas.
- [Vector2 Extent](#) (string text, [TextRenderModifiers?](#) modifiers=null)
Returns the logical dimensions that the given text will occupy when rendered.
- [RectF ExtentByPixels](#) (string text, [TextRenderModifiers?](#) modifiers=null)
Calculates the actual rectangle in pixels that the given text will occupy when rendered.
- [Vector2 TextRects](#) (string text, out [TextEntryRect](#)[] rects, [TextRenderModifiers?](#) modifiers=null)
- void [Draw](#) ([Vector2](#) position, string text, [ColorPair](#) colors, float alpha=1.0f, [TextRenderModifiers?](#) modifiers=null)
Draws text at the given position.
- void [DrawAligned](#) ([Vector2](#) position, string text, [ColorPair](#) colors, [TextAlignment](#) horizAlign=[TextAlignment.Middle](#), [TextAlignment](#) vertAlign=[TextAlignment.Middle](#), float alpha=1.0f, bool alignToPixels=true, [TextRenderModifiers?](#) modifiers=null)
Draws text at the given position with specific alignment.
- void [DrawCentered](#) ([Vector2](#) position, string text, [ColorPair](#) colors, float alpha=1.0f, bool alignToPixels=true, [TextRenderModifiers?](#) modifiers=null)
Draws text centered around the specified position.
- void [DrawAlignedByPixels](#) ([Vector2](#) position, string text, [ColorPair](#) colors, [TextAlignment](#) horiz↵ Align=[TextAlignment.Middle](#), [TextAlignment](#) vertAlign=[TextAlignment.Middle](#), float alpha=1.0f, bool alignTo↵ Pixels=true, [TextRenderModifiers?](#) modifiers=null)
Draws text at the given position with specific alignment.
- void [DrawCenteredByPixels](#) ([Vector2](#) position, string text, [ColorPair](#) colors, float alpha=1.0f, bool alignTo↵ Pixels=true, [TextRenderModifiers?](#) modifiers=null)
Draws text centered around the specified position.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Canvas Canvas](#) [get]
Returns canvas class associated with the text renderer.
- [FontParameters Parameters](#) [get, set]
Font settings and characteristics that define how text will appear.

Properties inherited from [Afterwarp.CustomObject](#)

- [IntPtr Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.105.1 Detailed Description

Text renderer utility that can draw text on the canvas.

7.105.2 Constructor & Destructor Documentation

7.105.2.1 TextRenderer()

```
Afterwarp.TextRenderer.TextRenderer (
    Canvas canvas,
    Point textureSize,
    PixelFormat pixelFormat = PixelFormat::Unknown,
    bool mipMapping = false ) [inline]
```

Creates new instance of text renderer associated with a particular canvas.

7.105.3 Member Function Documentation

7.105.3.1 Draw()

```
void Afterwarp.TextRenderer.Draw (
    Vector2 position,
    string text,
    ColorPair colors,
    float alpha = 1::0f,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws text at the given position.

7.105.3.2 DrawAligned()

```
void Afterwarp.TextRenderer.DrawAligned (
    Vector2 position,
    string text,
    ColorPair colors,
    TextAlignment horizAlign = TextAlignment::Middle,
    TextAlignment vertAlign = TextAlignment::Middle,
    float alpha = 1::0f,
    bool alignToPixels = true,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws text at the given position with specific alignment.

7.105.3.3 DrawAlignedByPixels()

```
void Afterwarp.TextRenderer.DrawAlignedByPixels (
    Vector2 position,
    string text,
    ColorPair colors,
    TextAlignment horizAlign = TextAlignment::Middle,
    TextAlignment vertAlign = TextAlignment::Middle,
    float alpha = 1::0f,
    bool alignToPixels = true,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws text at the given position with specific alignment.

7.105.3.4 DrawCentered()

```
void Afterwarp.TextRenderer.DrawCentered (
    Vector2 position,
    string text,
    ColorPair colors,
    float alpha = 1::0f,
    bool alignToPixels = true,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws text centered around the specified position.

7.105.3.5 DrawCenteredByPixels()

```
void Afterwarp.TextRenderer.DrawCenteredByPixels (
    Vector2 position,
    string text,
    ColorPair colors,
    float alpha = 1::0f,
    bool alignToPixels = true,
    TextRenderModifiers? modifiers = null ) [inline]
```

Draws text centered around the specified position.

7.105.3.6 Extent()

```
Vector2 Afterwarp.TextRenderer.Extent (
    string text,
    TextRenderModifiers? modifiers = null ) [inline]
```

Returns the logical dimensions that the given text will occupy when rendered.

7.105.3.7 ExtentByPixels()

```
RectF Afterwarp.TextRenderer.ExtentByPixels (
    string text,
    TextRenderModifiers? modifiers = null ) [inline]
```

Calculates the actual rectangle in pixels that the given text will occupy when rendered.

7.105.3.8 TextRects()

```
Vector2 Afterwarp.TextRenderer.TextRects (
    string text,
    out TextEntryRect[] rects,
    TextRenderModifiers? modifiers = null ) [inline]
```

Provides information regarding individual character position and sizes for the given text string when rendered. This can be useful for components such as text edit box, for highlighting and selecting different characters. The actual width and height of the text is, optionally, provided as well. Returns an actual rectangle dimensions that would encompass all rectangles.

7.105.4 Property Documentation

7.105.4.1 Canvas

```
Canvas Afterwarp.TextRenderer.Canvas [get]
```

Returns canvas class associated with the text renderer.

7.105.4.2 Parameters

```
FontParameters Afterwarp.TextRenderer.Parameters [get], [set]
```

Font settings and characteristics that define how text will appear.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.106 Afterwarp.TextRenderModifiers Struct Reference

Modifier attributes that can be applied to the rendered text.

Public Member Functions

- [TextRenderModifiers](#) (float scale, float interleave=0.0f, float verticalSpace=0.0f, [BlendingEffect](#) effect=[BlendingEffect.Undefined](#))
Creates new text rendering modifier parameters with the given values.

Public Attributes

- float [Scale](#)
- float [Interleave](#)
- float [VerticalSpace](#)
Global spacing that will be added vertically when drawing multiple lines of text.
- [BlendingEffect](#) [Effect](#)
Blending effect to use (if left undefined, normal blending effect will be used).

7.106.1 Detailed Description

Modifier attributes that can be applied to the rendered text.

7.106.2 Constructor & Destructor Documentation

7.106.2.1 TextRenderModifiers()

```
Afterwarp.TextRenderModifiers.TextRenderModifiers (
    float scale,
    float interleave = 0::0f,
    float verticalSpace = 0::0f,
    BlendingEffect effect = BlendingEffect::Undefined ) [inline]
```

Creates new text rendering modifier parameters with the given values.

7.106.3 Member Data Documentation

7.106.3.1 Effect

[BlendingEffect](#) Afterwarp.TextRenderModifiers.Effect

Blending effect to use (if left undefined, normal blending effect will be used).

7.106.3.2 Interleave

float Afterwarp.TextRenderModifiers.Interleave

Global spacing that will be added horizontally between text letters. This can be used to expand or shrink the text.

7.106.3.3 Scale

```
float Afterwarp.TextRenderModifiers.Scale
```

Global font scale that is applied to the whole rendered text. A default value of zero means that font scale will not be modified. Using values other than zero or one would likely result in not pixel-perfect text rendering, appearing blurry. However, this can be used for real-time text animations.

7.106.3.4 VerticalSpace

```
float Afterwarp.TextRenderModifiers.VerticalSpace
```

Global spacing that will be added vertically when drawing multiple lines of text.

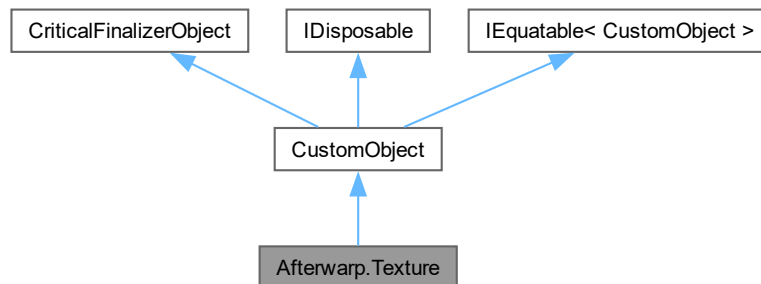
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

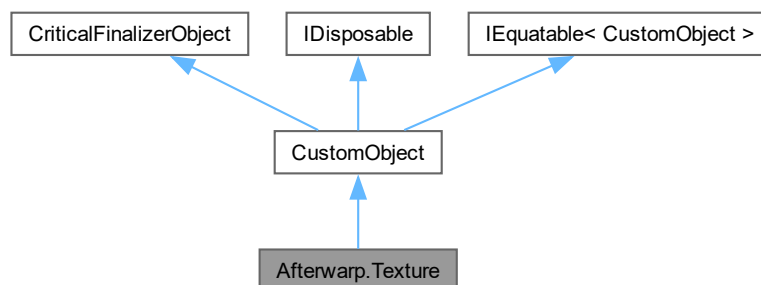
7.107 Afterwarp.Texture Class Reference

Texture that contains image data typically stored in GPU memory.

Inheritance diagram for Afterwarp.Texture:



Collaboration diagram for Afterwarp.Texture:



Classes

- struct [Attribute](#)

Attribute that defines special behavior and/or unique characteristics of the texture.

Public Member Functions

- [Texture](#) ([Device](#) device, [TextureParameters](#) parameters)
Creates a new instance of texture.
- [Texture](#) ([Device](#) device, string fileName, [PixelFormat](#) format=[PixelFormat.Unknown](#), uint attributes=0)
- [Texture](#) ([Device](#) device, Array fileContents, string extension, [PixelFormat](#) format=[PixelFormat.Unknown](#), uint attributes=0)
- [Texture](#) ([Device](#) device, [Surface](#) surface, uint attributes=0)
Creates a new instance of texture object loading its contents from an existing surface.
- void [Update](#) (Array content, uint pitch=0, int layer=0, [Rect?](#) rect=null, int mipLevel=0)
- void [Retrieve](#) (Array content, uint pitch=0, int layer=0, [Rect?](#) rect=null, int mipLevel=0)
- void [Retrieve](#) ([Surface](#) surface, int layer=0, [Point?](#) destPos=null, [Rect?](#) sourceRect=null, int mipLevel=0)
- void [Copy](#) ([Texture](#) source, int destLayer=0, [Point?](#) destPos=null, int srcLayer=0, [Rect?](#) sourceRect=null, int destMipLevel=0, int srcMipLevel=0)
- void [Copy](#) ([Surface](#) source, int layer=0, [Point?](#) destPos=null, [Rect?](#) sourceRect=null, int mipLevel=0)
- void [LoadFromFile](#) (string fileName, int layer=0, [Point?](#) destPos=null, [Rect?](#) sourceRect=null, int mipLevel=0)
Loads image from disk and copies it to the specified location and parameters.
- void [SaveToFile](#) (string fileName, int quality=50, int layer=0, [Rect?](#) sourceRect=null, int mipLevel=0)
Saves the appropriate portion of texture to external file.
- void [Clear](#) ()
Clears entire texture surface and fills pixels with zeros.
- void [Clear](#) ([FloatColor](#) color, int layer=0, int mipLevel=0)
- void [Bind](#) (uint channel=0)
Binds texture to the specified rendering channel.
- void [Unbind](#) (uint channel=0)
Unbinds texture from the specified rendering channel.
- void [Attach](#) ([Texture](#) attachment, int layer=0, int mipLevel=0)
- void [Detach](#) ()
Removes all previous drawable texture attachments.
- void [Begin](#) (int layer=0, int mipLevel=0)
Starts rendering on the drawable texture.
- void [End](#) ()
Finishes rendering on the drawable texture.
- void [GenerateMipMaps](#) ()
Generates texture's mipmaps from its base (level = 0) image.
- void [ResetCache](#) ()

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Static Public Member Functions

- static [Texture CreateParallax](#) ([Device](#) device, string fileName, [PixelFormat](#) format=[PixelFormat.Unknown](#), uint attributes=0)
- static [Texture CreateNormalsAndOcclusion](#) ([Device](#) device, string fileNameNormalMap, string fileName↔OcclusionMap, uint attributes=0)

Properties

- [Device Device](#) [get]
Returns device associated with this texture.
- [IntPtr PlatformHandle](#) [get]
Returns platform-specific texture handle.
- [TextureParameters Parameters](#) [get]
Retrieves current texture parameters.

Properties inherited from [Afterwarp.CustomObject](#)

- [IntPtr Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- [IntPtr _handle](#)
Wrapped object's handle.

7.107.1 Detailed Description

Texture that contains image data typically stored in GPU memory.

7.107.2 Constructor & Destructor Documentation

7.107.2.1 [Texture\(\)](#) [1/4]

```
Afterwarp.Texture.Texture (
    Device device,
    TextureParameters parameters ) [inline]
```

Creates a new instance of texture.

7.107.2.2 Texture() [2/4]

```
Afterwarp.Texture.Texture (
    Device device,
    string fileName,
    PixelFormat format = PixelFormat::Unknown,
    uint attributes = 0 ) [inline]
```

Creates a new instance of texture object loading its contents from an external file. The preference for premultiplied or non-premultiplied alpha-channel is merely a suggestion, which can influence how the image is loaded (depending on underlying implementation). This function typically supports most common file formats such as BMP, PNG, JPEG, but may also support other formats like GIF and TIFF.

7.107.2.3 Texture() [3/4]

```
Afterwarp.Texture.Texture (
    Device device,
    Array fileContents,
    string extension,
    PixelFormat format = PixelFormat::Unknown,
    uint attributes = 0 ) [inline]
```

Creates a new instance of texture object loading its contents from an image file that has been preloaded into memory. The internal file format is determined by extension parameter, which should contain a valid file extension such as ".png" (case-insensitive). This function typically supports most common file formats such as BMP, PNG, JPEG, but may also support other formats like GIF and TIFF.

7.107.2.4 Texture() [4/4]

```
Afterwarp.Texture.Texture (
    Device device,
    Surface surface,
    uint attributes = 0 ) [inline]
```

Creates a new instance of texture object loading its contents from an existing surface.

7.107.3 Member Function Documentation

7.107.3.1 Attach()

```
void Afterwarp.Texture.Attach (
    Texture attachment,
    int layer = 0,
    int mipLevel = 0 ) [inline]
```

Attaches another drawable texture with the current texture into a common MRT rendering stack. Drawable textures will be available to shaders in the same order of attachment.

7.107.3.2 Begin()

```
void Afterwarp.Texture.Begin (
    int layer = 0,
    int mipLevel = 0 ) [inline]
```

Starts rendering on the drawable texture.

7.107.3.3 Bind()

```
void Afterwarp.Texture.Bind (
    uint channel = 0 ) [inline]
```

Binds texture to the specified rendering channel.

7.107.3.4 Clear() [1/2]

```
void Afterwarp.Texture.Clear ( ) [inline]
```

Clears entire texture surface and fills pixels with zeros.

7.107.3.5 Clear() [2/2]

```
void Afterwarp.Texture.Clear (
    FloatColor color,
    int layer = 0,
    int mipLevel = 0 ) [inline]
```

Clears texture surface with the given color. This must be called outside of device and texture Begin() / End() blocks.

7.107.3.6 Copy() [1/2]

```
void Afterwarp.Texture.Copy (
    Surface source,
    int layer = 0,
    Point? destPos = null,
    Rect? sourceRect = null,
    int mipLevel = 0 ) [inline]
```

Copies a portion of source surface to the current texture according to specified source rectangle and destination position. If source rectangle is empty, then the entire source surface will be copied. This function does the appropriate clipping and pixel format conversion. Note that `premultipliedAlpha` attribute of either current or source surface is completely ignored.

7.107.3.7 Copy() [2/2]

```
void Afterwarp.Texture.Copy (
    Texture source,
    int destLayer = 0,
    Point? destPos = null,
    int srcLayer = 0,
    Rect? sourceRect = null,
    int destMipLevel = 0,
    int srcMipLevel = 0 ) [inline]
```

Copies a portion of source texture to the current one according to specified source rectangle and destination position. If source rectangle is empty, then the entire source texture will be copied. This function does the appropriate clipping and pixel format conversion (albeit at significant performance cost). Note that "premultiplied alpha" attribute of either current or source texture is completely ignored.

7.107.3.8 CreateNormalsAndOcclusion()

```
static Texture Afterwarp.Texture.CreateNormalsAndOcclusion (
    Device device,
    string fileNameNormalMap,
    string fileNameOcclusionMap,
    uint attributes = 0 ) [inline], [static]
```

Creates texture containing normal map and occlusion map from external files on disk. Both normal and occlusion map must have equal sizes.

7.107.3.9 CreateParallax()

```
static Texture Afterwarp.Texture.CreateParallax (
    Device device,
    string fileName,
    PixelFormat format = PixelFormat::Unknown,
    uint attributes = 0 ) [inline], [static]
```

Creates texture containing displacement map for Parallax Mapping technique from an external file on disk.

7.107.3.10 Detach()

```
void Afterwarp.Texture.Detach ( )
```

Removes all previous drawable texture attachments.

7.107.3.11 End()

```
void Afterwarp.Texture.End ( ) [inline]
```

Finishes rendering on the drawable texture.

7.107.3.12 GenerateMipMaps()

```
void Afterwarp.Texture.GenerateMipMaps ( ) [inline]
```

Generates texture's mipmaps from its base (level = 0) image.

7.107.3.13 LoadFromFile()

```
void Afterwarp.Texture.LoadFromFile (
    string fileName,
    int layer = 0,
    Point? destPos = null,
    Rect? sourceRect = null,
    int mipLevel = 0 ) [inline]
```

Loads image from disk and copies it to the specified location and parameters.

7.107.3.14 ResetCache()

```
void Afterwarp.Texture.ResetCache ( )
```

Resets any cache associated with the texture, which means purging any temporary resources. This also removes all existing texture attachments.

7.107.3.15 Retrieve() [1/2]

```
void Afterwarp.Texture.Retrieve (
    Array content,
    uint pitch = 0,
    int layer = 0,
    Rect? rect = null,
    int mipLevel = 0 ) [inline]
```

Retrieves content from the given location of texture's owned memory to the user pointer. The data is copied in its original format and source rectangle, if specified, must be within valid bounds (no clipping is performed).

7.107.3.16 Retrieve() [2/2]

```
void Afterwarp.Texture.Retrieve (
    Surface surface,
    int layer = 0,
    Point? destPos = null,
    Rect? sourceRect = null,
    int mipLevel = 0 ) [inline]
```

Copies a portion of current texture to the destination surface according to specified source rectangle and destination position. If source rectangle is empty, then the entire texture surface will be copied. This function does the appropriate clipping and pixel format conversion. Note that "premultiplied alpha" attribute of either current or source surface is completely ignored.

7.107.3.17 SaveToFile()

```
void Afterwarp.Texture.SaveToFile (
    string fileName,
    int quality = 50,
    int layer = 0,
    Rect? sourceRect = null,
    int mipLevel = 0 ) [inline]
```

Saves the appropriate portion of texture to external file.

7.107.3.18 Unbind()

```
void Afterwarp.Texture.Unbind (
    uint channel = 0 ) [inline]
```

Unbinds texture from the specified rendering channel.

7.107.3.19 Update()

```
void Afterwarp.Texture.Update (
    Array content,
    uint pitch = 0,
    int layer = 0,
    Rect? rect = null,
    int mipLevel = 0 ) [inline]
```

Copies content from user pointer to texture's owned memory at the given location. The data is assumed to match texture's pixel format and destination rectangle, if specified, must be within valid bounds (no clipping is performed).

7.107.4 Property Documentation

7.107.4.1 Device

```
Device Afterwarp.Texture.Device [get]
```

Returns device associated with this texture.

7.107.4.2 Parameters

```
TextureParameters Afterwarp.Texture.Parameters [get]
```

Retrieves current texture parameters.

7.107.4.3 PlatformHandle

```
IntPtr Afterwarp.Texture.PlatformHandle [get]
```

Returns platform-specific texture handle.

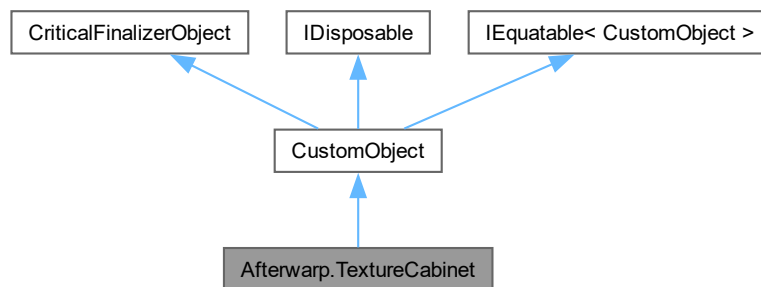
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

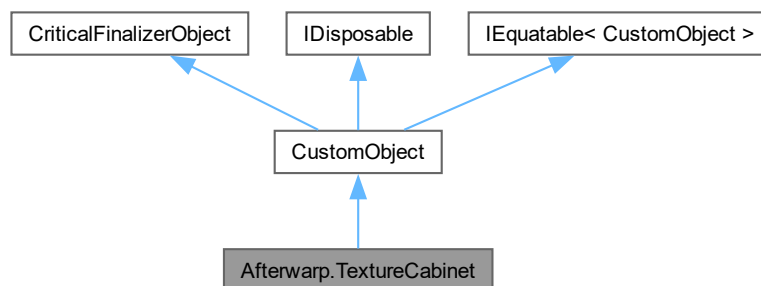
7.108 Afterwarp.TextureCabinet Class Reference

A container and management module for drawable textures for rendering and processing 3D scene.

Inheritance diagram for Afterwarp.TextureCabinet:



Collaboration diagram for Afterwarp.TextureCabinet:



Classes

- struct [Attribute](#)

Cumulative attributes that define rendering characteristics of the scene.

Public Member Functions

- [TextureCabinet](#) ([Device](#) device, [Point](#) size, [TextureFidelity](#) fidelity=[TextureFidelity.High](#), int samples=8, [PixelFormat](#) depthStencil=[PixelFormat.Unknown](#))
Creates new instance of texture container and management module.
- [Texture GetTexture](#) ([TextureCabinetType](#) type)
Returns texture from the container of the given type.
- void [Clear](#) ([TextureCabinetPass](#) pass, [FloatColor?](#) color=null)
- void [Begin](#) ([TextureCabinetPass](#) pass)
Begins rendering the given pass.
- void [End](#) ()
Ends rendering the pass that was previously started.
- void [Filter](#) ([TextureCabinetFilterType](#) filter, [Matrix4x4?](#) projection=null)
- void [Resolve](#) ()
Renders a multisample HDR color texture into a final color texture.
- void [Present](#) ()
Renders the final color texture on the current rendering surface.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Device Device](#) [get]
Returns device associated with the container.
- [Point Size](#) [get, set]
Returns or updates the size of textures in pixels.
- int [Samples](#) [get]
Returns number of samples used in the textures.
- [PixelFormat DepthStencil](#) [get]
Returns format used for depth/stencil buffers.
- [TextureFidelity Fidelity](#) [get]
Returns level of fidelity used for choosing integrated pixel formats.
- uint [Attributes](#) [get, set]
Returns or changes attributes that define the rendering characteristics of the scene.
- [Texture Texture](#) [get]
Returns final color texture that can contains the scene.
- bool [Rendering](#) [get]
Indicates that one of the passes is currently being rendered.
- [ToneMappingBloom ToneMappingBloom](#) [get, set]
Returns or changes tone-mapping and bloom rendering characteristics.
- [AmbientOcclusionParameters AmbientOcclusionParameters](#) [get, set]
Returns or changes ambient occlusion characteristics.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)
Wrapped object's handle.

7.108.1 Detailed Description

A container and management module for drawable textures for rendering and processing 3D scene.

7.108.2 Constructor & Destructor Documentation

7.108.2.1 TextureCabinet()

```
Afterwarp.TextureCabinet.TextureCabinet (
    Device device,
    Point size,
    TextureFidelity fidelity = TextureFidelity::High,
    int samples = 8,
    PixelFormat depthStencil = PixelFormat::Unknown ) [inline]
```

Creates new instance of texture container and management module.

7.108.3 Member Function Documentation

7.108.3.1 Begin()

```
void Afterwarp.TextureCabinet.Begin (
    TextureCabinetPass pass ) [inline]
```

Begins rendering the given pass.

7.108.3.2 Clear()

```
void Afterwarp.TextureCabinet.Clear (
    TextureCabinetPass pass,
    FloatColor? color = null ) [inline]
```

Clears the content of textures related to the given pass. Note: the given color is used only in main/color pass.

7.108.3.3 End()

```
void Afterwarp.TextureCabinet.End ( )
```

Ends rendering the pass that was previously started.

7.108.3.4 Filter()

```
void Afterwarp.TextureCabinet.Filter (
    TextureCabinetFilterType filter,
    Matrix4x4? projection = null ) [inline]
```

Performs processing and filtering of the textures according to the filter type. For occlusion, performs depth linearization and computes ambient occlusion. For bloom, downscales the color texture, applies blur to compute glare, combines the effects and then performs HDR tone-mapping. For order-independent transparency (OIT or "glassy") pass, performs compositioning of the textures. This step must be performed outside of "Begin" and "End" block after finishing each corresponding rendering pass but before starting next one. For performance reasons, the filtering should be performed as late as possible after finishing previous pass (e.g. after doing some CPU calculations) to enable GPU to do work in parallel. Note: OIT rendering pass does not require projection matrix.

7.108.3.5 GetTexture()

```
Texture Afterwarp.TextureCabinet.GetTexture (
    TextureCabinetType type ) [inline]
```

Returns texture from the container of the given type.

7.108.3.6 Present()

```
void Afterwarp.TextureCabinet.Present ( ) [inline]
```

Renders the final color texture on the current rendering surface.

7.108.3.7 Resolve()

```
void Afterwarp.TextureCabinet.Resolve ( ) [inline]
```

Renders a multisample HDR color texture into a final color texture.

7.108.4 Property Documentation

7.108.4.1 AmbientOcclusionParameters

`AmbientOcclusionParameters` Afterwarp.TextureCabinet.AmbientOcclusionParameters [get], [set]

Returns or changes ambient occlusion characteristics.

7.108.4.2 Attributes

`uint` Afterwarp.TextureCabinet.Attributes [get], [set]

Returns or changes attributes that define the rendering characteristics of the scene.

7.108.4.3 DepthStencil

`PixelFormat` Afterwarp.TextureCabinet.DepthStencil [get]

Returns format used for depth/stencil buffers.

7.108.4.4 Device

`Device` Afterwarp.TextureCabinet.Device [get]

Returns device associated with the container.

7.108.4.5 Fidelity

`TextureFidelity` Afterwarp.TextureCabinet.Fidelity [get]

Returns level of fidelity used for choosing integrated pixel formats.

7.108.4.6 Rendering

`bool` Afterwarp.TextureCabinet.Rendering [get]

Indicates that one of the passes is currently being rendered.

7.108.4.7 Samples

`int` Afterwarp.TextureCabinet.Samples [get]

Returns number of samples used in the textures.

7.108.4.8 Size

`Point` `Afterwarp.TextureCabinet.Size` [get], [set]

Returns or updates the size of textures in pixels.

7.108.4.9 Texture

`Texture` `Afterwarp.TextureCabinet.Texture` [get]

Returns final color texture that can contains the scene.

7.108.4.10 ToneMappingBloom

`ToneMappingBloom` `Afterwarp.TextureCabinet.ToneMappingBloom` [get], [set]

Returns or changes tone-mapping and bloom rendering characteristics.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.109 Afterwarp.TextureParameters Struct Reference

Texture parameters and characteristics.

Public Member Functions

- [TextureParameters](#) (int width, int height, int layers, [PixelFormat](#) format, [TextureType](#) type=[TextureType.Surface](#), uint attributes=0, int multisamples=0, [PixelFormat](#) depthStencil=[PixelFormat.Unknown](#))

Creates a new set of texture parameters with the given values.

Public Attributes

- int [Width](#)
Texture width (in pixels).
- int [Height](#)
Texture height (in pixels).
- int [Layers](#)
Number of texture layers in case of Texture Array or depth in case of 3D textures.
- [PixelFormat](#) [Format](#)
Format of the texture's pixels.
- [TextureType](#) [Type](#)
Texture type that defines how it is composed.
- uint [Attributes](#)
Attributes that define the characteristics of the texture.
- int [Multisamples](#)
Number of multisamples used in the render target.
- [PixelFormat](#) [DepthStencil](#)
Depth/stencil format that will be used to create an associated buffer.

7.109.1 Detailed Description

Texture parameters and characteristics.

7.109.2 Constructor & Destructor Documentation

7.109.2.1 TextureParameters()

```
Afterwarp.TextureParameters.TextureParameters (
    int width,
    int height,
    int layers,
    PixelFormat format,
    TextureType type = TextureType::Surface,
    uint attributes = 0,
    int multisamples = 0,
    PixelFormat depthStencil = PixelFormat::Unknown ) [inline]
```

Creates a new set of texture parameters with the given values.

7.109.3 Member Data Documentation

7.109.3.1 Attributes

```
uint Afterwarp.TextureParameters.Attributes
```

Attributes that define the characteristics of the texture.

7.109.3.2 DepthStencil

```
PixelFormat Afterwarp.TextureParameters.DepthStencil
```

Depth/stencil format that will be used to create an associated buffer.

7.109.3.3 Format

```
PixelFormat Afterwarp.TextureParameters.Format
```

Format of the texture's pixels.

7.109.3.4 Height

```
int Afterwarp.TextureParameters.Height
```

Texture height (in pixels).

7.109.3.5 Layers

```
int Afterwarp.TextureParameters.Layers
```

Number of texture layers in case of Texture Array or depth in case of 3D textures.

7.109.3.6 Multisamples

```
int Afterwarp.TextureParameters.Multisamples
```

Number of multisamples used in the render target.

7.109.3.7 Type

```
TextureType Afterwarp.TextureParameters.Type
```

Texture type that defines how it is composed.

7.109.3.8 Width

```
int Afterwarp.TextureParameters.Width
```

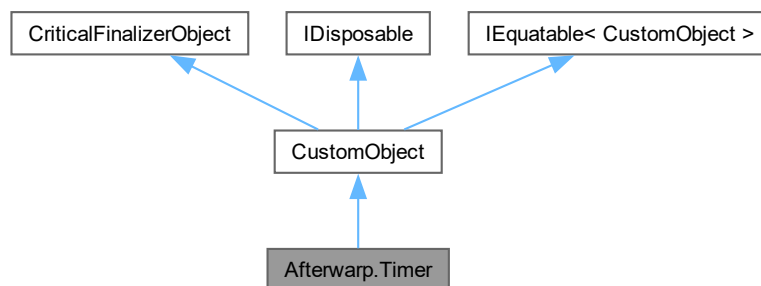
Texture width (in pixels).

The documentation for this struct was generated from the following file:

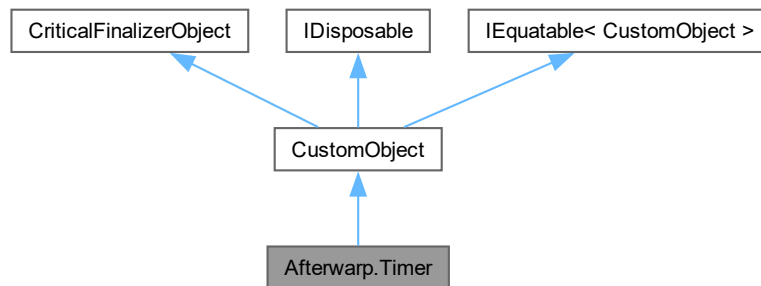
- [Afterwarp.Types.cs](#)

7.110 Afterwarp.Timer Class Reference

Inheritance diagram for Afterwarp.Timer:



Collaboration diagram for Afterwarp.Timer:



Public Member Functions

- [Timer](#) ()
Creates new instance of timer module.
- bool [NextSlice](#) ()
- bool [UpdateNextSlice](#) ()
- void [Update](#) ()
Updates latency, calculates average frame rate and updates fixed-rate token counter.
- void [Reset](#) ()
- void [TrimSkippedTimeSlices](#) (long slicesToTrim)

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- double [Speed](#) [get, set]
- float [FrameRate](#) [get]
- ulong [Latency](#) [get]
- long [TimeSlice](#) [get]
Returns current time slice.
- long [SkippedTimeSlices](#) [get]
Returns total number of skipped time slices.
- int [ExtractTokens](#) [get]
Extracts existing fixed-rate token counter from the timer.

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)

Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)

Wrapped object's handle.

7.110.1 Detailed Description

Timer that can be used in multimedia interactive applications to ensure fixed processing rate and/or variable processing rate (e.g. as fast as possible) with a fixed-rate counter.

7.110.2 Constructor & Destructor Documentation

7.110.2.1 Timer()

```
Afterwarp.Timer.Timer ( ) [inline]
```

Creates new instance of timer module.

7.110.3 Member Function Documentation

7.110.3.1 NextSlice()

```
bool Afterwarp.Timer.NextSlice ( )
```

Ensures that next processing time slice is reached. Returns whether the method was executed within the correct time slice (in other words, no timer slices have been lost, which could result in stuttering).

7.110.3.2 Reset()

```
void Afterwarp.Timer.Reset ( )
```

Resets internal structures of the timer, restarts the timing calculations and sets current time slice to zero. This can be useful when an unscheduled and/or a time-consuming task was executed.

7.110.3.3 TrimSkippedTimeSlices()

```
void Afterwarp.Timer.TrimSkippedTimeSlices (
    long slicesToTrim ) [inline]
```

Reduces number of skipped time slices by the given quantity. If the quantity is larger than the actual number of skipped time slices, resets that number to zero.

7.110.3.4 Update()

```
void Afterwarp.Timer.Update ( )
```

Updates latency, calculates average frame rate and updates fixed-rate token counter.

7.110.3.5 UpdateNextSlice()

```
bool Afterwarp.Timer.UpdateNextSlice ( )
```

Ensures that next processing time slice is reached, updates latency and calculates average frame rate. Returns whether the method was executed within the correct time slice (in other words, no timer slices have been lost, which could result in stuttering). Updates fixed-rate token counter.

7.110.4 Property Documentation

7.110.4.1 ExtractTokens

```
int Afterwarp.Timer.ExtractTokens [get]
```

Extracts existing fixed-rate token counter from the timer.

7.110.4.2 FrameRate

```
float Afterwarp.Timer.FrameRate [get]
```

Returns calculated average frame rate in frames per second. This value is updated roughly two times per second and can only be used for informative purposes (e.g. displaying frame rate in the application). For precise real-time indications it is recommended to use "Latency".

7.110.4.3 Latency

```
ulong Afterwarp.Timer.Latency [get]
```

Returns time (in microseconds) calculated between previous and current frames. This can be a direct indication of rendering performance as it tells how much time it took to render (and possibly, process) the frame.

7.110.4.4 SkippedTimeSlices

```
long Afterwarp.Timer.SkippedTimeSlices [get]
```

Returns total number of skipped time slices.

7.110.4.5 Speed

```
double Afterwarp.Timer.Speed [get], [set]
```

Returns or changes multimedia timer processing speed (in terms of frames per second). Typical (and default) speed is 60 frames per second.

7.110.4.6 TimeSlice

```
long Afterwarp.Timer.TimeSlice [get]
```

Returns current time slice.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.111 Afterwarp.ToneMappingBloom Struct Reference

Parameters that define behavior of tone-mapping and bloom effects.

Public Member Functions

- [ToneMappingBloom](#) (float bloomThreshold, float bloomGamma, Vector3 bloomCoefficients, float bloomColorShift, float bloomBlurSigma, int bloomBlurSamples, float toneWhite, Vector3 toneFactors, float frostedPower, float glassyBuckets)
Creates tone-mapping and bloom parameters with the given values.

Public Attributes

- float [BloomThreshold](#)
Amount to be subtracted from initial HDR colors to determine bloom.
- float [BloomGamma](#)
Inverse gamma power to adjust the bloom color after threshold.
- Vector3 [BloomCoefficients](#)
Grayscale coefficients for red, green and blue components.
- float [BloomColorShift](#)
Amount of color to shift into grayscale.
- float [BloomBlurSigma](#)
Sigma value of Gaussian Blur for bloom pixels.
- int [BloomBlurSamples](#)
Number of bloom pixels to sample during Gaussian blur.
- float [ToneWhite](#)
Maximum allowed level of white for tone-mapping.
- Vector3 [ToneFactors](#)
Conversion coefficients to calculate Luma from RGB during tone-mapping.
- float [FrostedPower](#)
Power used to adjust the curve of frosted glass equation.
- float [GlassyBuckets](#)
Number of buckets per pixel for OIT technique.

Properties

- static [ToneMappingBloom Default](#) [get]
Returns default tone-mapping and bloom parameters.

7.111.1 Detailed Description

Parameters that define behavior of tone-mapping and bloom effects.

7.111.2 Constructor & Destructor Documentation

7.111.2.1 ToneMappingBloom()

```
Afterwarp.ToneMappingBloom.ToneMappingBloom (  
    float bloomThreshold,  
    float bloomGamma,  
    Vector3 bloomCoefficients,  
    float bloomColorShift,  
    float bloomBlurSigma,  
    int bloomBlurSamples,  
    float toneWhite,  
    Vector3 toneFactors,  
    float frostedPower,  
    float glassyBuckets ) [inline]
```

Creates tone-mapping and bloom parameters with the given values.

7.111.3 Member Data Documentation

7.111.3.1 BloomBlurSamples

```
int Afterwarp.ToneMappingBloom.BloomBlurSamples
```

Number of bloom pixels to sample during Gaussian blur.

7.111.3.2 BloomBlurSigma

```
float Afterwarp.ToneMappingBloom.BloomBlurSigma
```

Sigma value of Gaussian Blur for bloom pixels.

7.111.3.3 BloomCoefficients

```
Vector3 Afterwarp.ToneMappingBloom.BloomCoefficients
```

Grayscale coefficients for red, green and blue components.

7.111.3.4 BloomColorShift

```
float Afterwarp.ToneMappingBloom.BloomColorShift
```

Amount of color to shift into grayscale.

7.111.3.5 BloomGamma

```
float Afterwarp.ToneMappingBloom.BloomGamma
```

Inverse gamma power to adjust the bloom color after threshold.

7.111.3.6 BloomThreshold

```
float Afterwarp.ToneMappingBloom.BloomThreshold
```

Amount to be subtracted from initial HDR colors to determine bloom.

7.111.3.7 FrostedPower

```
float Afterwarp.ToneMappingBloom.FrostedPower
```

Power used to adjust the curve of frosted glass equation.

7.111.3.8 GlassyBuckets

```
float Afterwarp.ToneMappingBloom.GlassyBuckets
```

Number of buckets per pixel for OIT technique.

7.111.3.9 ToneFactors

```
Vector3 Afterwarp.ToneMappingBloom.ToneFactors
```

Conversion coefficients to calculate Luma from RGB during tone-mapping.

7.111.3.10 ToneWhite

```
float Afterwarp.ToneMappingBloom.ToneWhite
```

Maximum allowed level of white for tone-mapping.

7.111.4 Property Documentation

7.111.4.1 Default

```
ToneMappingBloom Afterwarp.ToneMappingBloom.Default [static], [get]
```

Returns default tone-mapping and bloom parameters.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.112 Afterwarp.Utility Struct Reference

Static Public Member Functions

- static uint [MakeColor](#) (uint color, int alpha)
Creates 32-bit RGBA color with the specified color value, having its alpha-channel multiplied by the specified coefficient and divided by 255.
- static uint [MakeColor](#) (uint color, float alpha)
Creates 32-bit RGBA color with the specified color value, having its alpha-channel multiplied by the specified floating-point coefficient.
- static uint [MakeColorWithGray](#) (uint color, int gray, int alpha)
Creates 32-bit RGBA color where the original color value has its components multiplied by the given grayscale value and alpha-channel multiplied by the specified coefficient, and all components divided by 255.
- static uint [MakeColorWithGray](#) (uint color, float gray, float alpha)
Creates 32-bit RGBA color where the original color value has its components multiplied by the given grayscale value and alpha-channel multiplied by the specified coefficient.
- static uint [MakeColorRGB](#) (int red, int green, int blue, int alpha)
Creates 32-bit RGBA color using specified individual components.
- static uint [MakeColorRGB](#) (float red, float green, float blue, float alpha)
Creates 32-bit RGBA color using specified individual components.
- static uint [MakeColorGray](#) (int gray, int alpha)
Creates 32-bit RGBA color using specified grayscale and alpha values.
- static uint [MakeColorGray](#) (float gray, float alpha)
Creates 32-bit RGBA color using specified grayscale and alpha values.
- static uint [MakeColorAlpha](#) (int alpha)
Creates 32-bit RGBA white color with the specified alpha-channel value.
- static uint [MakeColorAlpha](#) (float alpha)
Creates 32-bit RGBA white color with the specified alpha-channel value.
- static int [GetColorAlpha](#) (uint color)
Returns alpha-channel from the given 32-bit RGBA color.
- static float [GetColorAlphaF](#) (uint color)
Returns alpha-channel from the given 32-bit RGBA color.
- static uint [PremultiplyAlpha](#) (uint color)
- static uint [UnpremultiplyAlpha](#) (uint color)
- static uint [DisplaceRB](#) (uint color)
Switches red and blue channels in 32-bit RGBA color value.
- static uint [InvertColor](#) (uint color)
Inverts each of the components in the color, including alpha-channel.
- static uint [AddColors](#) (uint color1, uint color2)
Adds two 32-bit RGBA color values together clamping the resulting values.
- static uint [SubtractColors](#) (uint color1, uint color2)
Subtracts two 32-bit RGBA color values clamping the resulting values.
- static uint [MultiplyColors](#) (uint color1, uint color2)
Multiplies two 32-bit RGBA color values together.
- static uint [AverageColors](#) (uint color1, uint color2)
Computes average of two given 32-bit RGBA color values.
- static uint [AverageFourColors](#) (uint color1, uint color2, uint color3, uint color4)
Computes average of four given 32-bit RGBA color values.
- static uint [AverageSixColors](#) (uint color1, uint color2, uint color3, uint color4, uint color5, uint color6)
Computes average of six given 32-bit RGBA color values.
- static uint [BlendColors](#) (uint color1, uint color2, int alpha)

Computes alpha-blending for a pair of 32-bit RGBA colors values.

- static uint [BlendFourColors](#) (uint topLeft, uint topRight, uint bottomRight, uint bottomLeft, int alphaX, int alphaY, bool horizontal=true)

Computes resulting alpha-blended value between four 32-bit RGBA colors using linear interpolation.

- static uint [ComposeColors](#) (uint color1, uint color2)
- static int [ColorToGray](#) (uint color)
- static float [ColorToGrayF](#) (uint color)
- static float [SineTransform](#) (float value)

Transforms value in range of [0, 1] by using sine wave (kind of "accelerate then decelerate").

- static float [SineAccelerate](#) (float value)
- static float [SineDecelerate](#) (float value)
- static float [SineCycle](#) (float value)
- static float [SineTwoCycle](#) (float value)

Transforms value in range of [0, 1] to go full cycle through sine function (0 -> 1 -> 0 -> -1 -> 0).

- static float [BiasTransform](#) (float value, float bias)
- static float [GainTransform](#) (float value, float bias)
- static uint [VertexElementsEstimatePitch](#) ([VertexElement](#)[] elements, uint channel=0)

Makes an estimation of pitch for the given channel based on existing element offsets.

- static IntPtr [AssignPayload](#)< T > (T obj)

Assigns an object as payload and returns an IntPtr handle.

- static T [RetrievePayload](#)< T > (IntPtr payload)

Retrieves an object from a payload pointer.

- static void [FreePayload](#) (IntPtr payload)

Releases a GC handle associated with the payload pointer.

- static [MouseButton ToUI](#) (MouseButtons button)

Converts Windows Forms button to UI button type.

- static [Key ToUI](#) (Keys key)

Converts Windows Forms key code to UI key enumeration.

- static Cursor [FromUI](#) ([AppCursor](#) cursor)

Converts UI application cursor type to Windows Forms cursor.

- static Matrix4x4 [MatrixPerspective4x4](#) (float fieldOfView, float aspectRatio, float nearPlane, float farPlane)
- static Matrix4x4 [MatrixOrthographic4x4](#) (float width, float height, float rangeMin, float rangeMax)

Creates orthogonal projection matrix defined by the viewing volume in 3D space.

- static Matrix4x4 [MatrixLookAt4x4](#) (Vector3 origin, Vector3 target, Vector3 ceiling)

Creates a left-handed view matrix.

Properties

- static ulong [SystemTicks](#) [get]

Retrieves number of microseconds that passed since application startup.

7.112.1 Member Function Documentation

7.112.1.1 AddColors()

```
static uint Afterwarp.Utility.AddColors (
    uint color1,
    uint color2 ) [static]
```

Adds two 32-bit RGBA color values together clamping the resulting values.

7.112.1.2 AssignPayload< T >()

```
static IntPtr Afterwarp.Utility.AssignPayload< T > (
    T obj ) [inline], [static]
```

Assigns an object as payload and returns an IntPtr handle.

Type Constraints

***T* : class**

7.112.1.3 AverageColors()

```
static uint Afterwarp.Utility.AverageColors (
    uint color1,
    uint color2 ) [static]
```

Computes average of two given 32-bit RGBA color values.

7.112.1.4 AverageFourColors()

```
static uint Afterwarp.Utility.AverageFourColors (
    uint color1,
    uint color2,
    uint color3,
    uint color4 ) [static]
```

Computes average of four given 32-bit RGBA color values.

7.112.1.5 AverageSixColors()

```
static uint Afterwarp.Utility.AverageSixColors (
    uint color1,
    uint color2,
    uint color3,
    uint color4,
    uint color5,
    uint color6 ) [static]
```

Computes average of six given 32-bit RGBA color values.

7.112.1.6 BiasTransform()

```
static float Afterwarp.Utility.BiasTransform (
    float value,
    float bias ) [static]
```

Transforms value in range of [0, 1] with a power function, adding bias to either start of the curve or its end. Bias value of 0.25 would be similar to "SineAccelerate", whereas value of 0.75 would be similar to that of "SineDecelerate".

7.112.1.7 BlendColors()

```
static uint Afterwarp.Utility.BlendColors (
    uint color1,
    uint color2,
    int alpha ) [static]
```

Computes alpha-blending for a pair of 32-bit RGBA colors values.

Parameters

<i>color1</i>	Source color for blending operation.
<i>color2</i>	Destination color for blending operation.
<i>alpha</i>	Blending coefficient in [0, 255] range.

Returns

The resulting color.

7.112.1.8 BlendFourColors()

```
static uint Afterwarp.Utility.BlendFourColors (
    uint topLeft,
    uint topRight,
    uint bottomRight,
    uint bottomLeft,
    int alphaX,
    int alphaY,
    bool horizontal = true ) [static]
```

Computes resulting alpha-blended value between four 32-bit RGBA colors using linear interpolation.

7.112.1.9 ColorToGray()

```
static int Afterwarp.Utility.ColorToGray (
    uint color ) [static]
```

Returns grayscale value in range of [0..255] from the given 32-bit RGBA color value. The resulting value can be considered the color's "Luma". The alpha-channel is ignored.

7.112.1.10 ColorToGrayF()

```
static float Afterwarp.Utility.ColorToGrayF (
    uint color ) [static]
```

Returns grayscale value in range of [0, 1] from the given 32-bit RGBA color value. The resulting value can be considered the color's "Luma". The alpha-channel is ignored.

7.112.1.11 ComposeColors()

```
static uint Afterwarp.Utility.ComposeColors (
    uint color1,
    uint color2 ) [static]
```

Composes source color onto destination color using source and destination color alphas. This essentially means alpha-blending source color onto destination using source's alpha, but with appropriate alpha composition.

7.112.1.12 DisplaceRB()

```
static uint Afterwarp.Utility.DisplaceRB (
    uint color ) [static]
```

Switches red and blue channels in 32-bit RGBA color value.

7.112.1.13 FreePayload()

```
static void Afterwarp.Utility.FreePayload (
    IntPtr payload ) [inline], [static]
```

Releases a GC handle associated with the payload pointer.

7.112.1.14 FromUI()

```
static Cursor Afterwarp.Utility.FromUI (
    AppCursor cursor ) [static]
```

Converts UI application cursor type to Windows Forms cursor.

7.112.1.15 GainTransform()

```
static float Afterwarp.Utility.GainTransform (
    float value,
    float bias ) [static]
```

Transforms value in range of [0, 1] with a power function, adding gain at start and end of the curve, similar to "↔ SineTransform". Values of "gain" closer to zero produce stronger distortions, whereas values close to zero produce almost linear curves.

7.112.1.16 GetColorAlpha()

```
static int Afterwarp.Utility.GetColorAlpha (
    uint color ) [static]
```

Returns alpha-channel from the given 32-bit RGBA color.

Parameters

<i>color</i>	The source 32-bit RGBA color.
--------------	-------------------------------

Returns

Alpha-channel of the source color in [0, 255] range.

7.112.1.17 GetColorAlphaF()

```
static float Afterwarp.Utility.GetColorAlphaF (
    uint color ) [static]
```

Returns alpha-channel from the given 32-bit RGBA color.

Parameters

<i>color</i>	The source 32-bit RGBA color.
--------------	-------------------------------

Returns

Alpha-channel of the source color in [0, 1] range.

7.112.1.18 InvertColor()

```
static uint Afterwarp.Utility.InvertColor (
    uint color ) [static]
```

Inverts each of the components in the color, including alpha-channel.

7.112.1.19 MakeColor() [1/2]

```
static uint Afterwarp.Utility.MakeColor (
    uint color,
    float alpha ) [static]
```

Creates 32-bit RGBA color with the specified color value, having its alpha-channel multiplied by the specified floating-point coefficient.

Parameters

<i>color</i>	32-bit RGBA color.
<i>alpha</i>	Alpha value in [0, 1] range.

Returns

The resulting color with its alpha-channel multiplied by the given alpha value.

7.112.1.20 MakeColor() [2/2]

```
static uint Afterwarp.Utility.MakeColor (
    uint color,
    int alpha ) [static]
```

Creates 32-bit RGBA color with the specified color value, having its alpha-channel multiplied by the specified coefficient and divided by 255.

Parameters

<i>color</i>	32-bit RGBA color.
<i>alpha</i>	Alpha value in [0, 255] range.

Returns

The resulting color with its alpha-channel multiplied by the given alpha value and divided by 255.

7.112.1.21 MakeColorAlpha() [1/2]

```
static uint Afterwarp.Utility.MakeColorAlpha (  
    float alpha ) [static]
```

Creates 32-bit RGBA white color with the specified alpha-channel value.

Parameters

<i>alpha</i>	Alpha-channel value in [0, 1] range.
--------------	--------------------------------------

Returns

The resulting 32-bit RGBA color.

7.112.1.22 MakeColorAlpha() [2/2]

```
static uint Afterwarp.Utility.MakeColorAlpha (  
    int alpha ) [static]
```

Creates 32-bit RGBA white color with the specified alpha-channel value.

Parameters

<i>alpha</i>	Alpha-channel value in [0, 255] range.
--------------	--

Returns

The resulting 32-bit RGBA color.

7.112.1.23 MakeColorGray() [1/2]

```
static uint Afterwarp.Utility.MakeColorGray (  
    float gray,  
    float alpha ) [static]
```

Creates 32-bit RGBA color using specified grayscale and alpha values.

Parameters

<i>gray</i>	Grayscale value in [0, 1] range.
<i>alpha</i>	Alpha-channel value in [0, 1] range.

Returns

The resulting 32-bit RGBA color.

7.112.1.24 MakeColorGray() [2/2]

```
static uint Afterwarp.Utility.MakeColorGray (  
    int gray,  
    int alpha ) [static]
```

Creates 32-bit RGBA color using specified grayscale and alpha values.

Parameters

<i>gray</i>	Grayscale value in [0, 255] range.
<i>alpha</i>	Alpha-channel value in [0, 255] range.

Returns

The resulting 32-bit RGBA color.

7.112.1.25 MakeColorRGB() [1/2]

```
static uint Afterwarp.Utility.MakeColorRGB (  
    float red,  
    float green,  
    float blue,  
    float alpha ) [static]
```

Creates 32-bit RGBA color using specified individual components.

Parameters

<i>red</i>	Red component value in [0, 1] range.
<i>green</i>	Green component value in [0, 1] range.
<i>blue</i>	Blue component value in [0, 1] range.
<i>alpha</i>	Alpha-channel value in [0, 1] range.

Returns

The resulting 32-bit RGBA color.

7.112.1.26 MakeColorRGB() [2/2]

```
static uint Afterwarp.Utility.MakeColorRGB (  
    int red,  
    int green,  
    int blue,  
    int alpha ) [static]
```

Creates 32-bit RGBA color using specified individual components.

Parameters

<i>red</i>	Red component value in [0, 255] range.
<i>green</i>	Green component value in [0, 255] range.
<i>blue</i>	Blue component value in [0, 255] range.
<i>alpha</i>	Alpha-channel value in [0, 255] range.

Returns

The resulting 32-bit RGBA color.

7.112.1.27 MakeColorWithGray() [1/2]

```
static uint Afterwarp.Utility.MakeColorWithGray (  
    uint color,  
    float gray,  
    float alpha ) [static]
```

Creates 32-bit RGBA color where the original color value has its components multiplied by the given grayscale value and alpha-channel multiplied by the specified coefficient.

Parameters

<i>color</i>	32-bit RGBA color.
<i>gray</i>	Grayscale value in [0, 1] range.
<i>alpha</i>	Alpha value in [0, 1] range.

Returns

The resulting color with its red, green and blue channels multiplied by the given grayscale value and its alpha-channel multiplied by the given alpha value.

7.112.1.28 MakeColorWithGray() [2/2]

```
static uint Afterwarp.Utility.MakeColorWithGray (  
    uint color,  
    int gray,  
    int alpha ) [static]
```

Creates 32-bit RGBA color where the original color value has its components multiplied by the given grayscale value and alpha-channel multiplied by the specified coefficient, and all components divided by 255.

Parameters

<i>color</i>	32-bit RGBA color.
<i>gray</i>	Grayscale value in [0, 255] range.
<i>alpha</i>	Alpha value in [0, 255] range.

Returns

The resulting color with its red, green and blue channels multiplied by the given grayscale value and its alpha-channel multiplied by the given alpha value, then all components divided by 255.

7.112.1.29 MatrixLookAt4x4()

```
static Matrix4x4 Afterwarp.Utility.MatrixLookAt4x4 (
    Vector3 origin,
    Vector3 target,
    Vector3 ceiling ) [inline], [static]
```

Creates a left-handed view matrix.

7.112.1.30 MatrixOrthographic4x4()

```
static Matrix4x4 Afterwarp.Utility.MatrixOrthographic4x4 (
    float width,
    float height,
    float rangeMin,
    float rangeMax ) [inline], [static]
```

Creates orthogonal projection matrix defined by the viewing volume in 3D space.

7.112.1.31 MatrixPerspective4x4()

```
static Matrix4x4 Afterwarp.Utility.MatrixPerspective4x4 (
    float fieldOfView,
    float aspectRatio,
    float nearPlane,
    float farPlane ) [inline], [static]
```

Creates a left-handed perspective projection matrix based on a field of view, aspect ratio, and near and far view plane distances using reversed depth technique.

7.112.1.32 MultiplyColors()

```
static uint Afterwarp.Utility.MultiplyColors (
    uint color1,
    uint color2 ) [static]
```

Multiplies two 32-bit RGBA color values together.

7.112.1.33 PremultiplyAlpha()

```
static uint Afterwarp.Utility.PremultiplyAlpha (
    uint color ) [static]
```

Takes 32-bit RGBA color with unpremultiplied alpha and multiplies each of red, green, and blue components by its alpha-channel, resulting in premultiplied alpha color.

7.112.1.34 RetrievePayload< T >()

```
static T Afterwarp.Utility.RetrievePayload< T > (
    IntPtr payload ) [inline], [static]
```

Retrieves an object from a payload pointer.

Type Constraints

T: *class*

7.112.1.35 SineAccelerate()

```
static float Afterwarp.Utility.SineAccelerate (
    float value ) [static]
```

Transforms value in range of [0, 1] using "accelerating" sine wave. The curve starts at zero with almost no "velocity", while in the end incrementing almost linearly.

7.112.1.36 SineCycle()

```
static float Afterwarp.Utility.SineCycle (
    float value ) [static]
```

Transforms value in range of [0, 1] to go through full sine curve in range [0 -> 1 -> 0]. Note that the curve goes from zero to one and then back to zero.

7.112.1.37 SineDecelerate()

```
static float Afterwarp.Utility.SineDecelerate (
    float value ) [static]
```

Transforms value in range of [0, 1] using "decelerating" sine wave. The curve starts at zero, incrementing almost linearly, while in the end going to almost a complete stop.

7.112.1.38 SineTransform()

```
static float Afterwarp.Utility.SineTransform (
    float value ) [static]
```

Transforms value in range of [0, 1] by using sine wave (kind of "accelerate then decelerate").

7.112.1.39 SineTwoCycle()

```
static float Afterwarp.Utility.SineTwoCycle (
    float value ) [static]
```

Transforms value in range of [0, 1] to go full cycle through sine function (0 -> 1 -> 0 -> -1 -> 0).

7.112.1.40 SubtractColors()

```
static uint Afterwarp.Utility.SubtractColors (
    uint color1,
    uint color2 ) [static]
```

Subtracts two 32-bit RGBA color values clamping the resulting values.

7.112.1.41 ToUI() [1/2]

```
static Key Afterwarp.Utility.ToUI (
    Keys key ) [static]
```

Converts Windows Forms key code to UI key enumeration.

7.112.1.42 ToUI() [2/2]

```
static MouseButton Afterwarp.Utility.ToUI (
    MouseButton button ) [static]
```

Converts Windows Forms button to UI button type.

7.112.1.43 UnpremultiplyAlpha()

```
static uint Afterwarp.Utility.UnpremultiplyAlpha (
    uint color ) [static]
```

Takes 32-bit RGBA color with premultiplied alpha channel and divides each of its red, green, and blue components by alpha, resulting in unpremultiplied alpha color.

7.112.1.44 VertexElementsEstimatePitch()

```
static uint Afterwarp.Utility.VertexElementsEstimatePitch (
    VertexElement[] elements,
    uint channel = 0 ) [inline], [static]
```

Makes an estimation of pitch for the given channel based on existing element offsets.

7.112.2 Property Documentation

7.112.2.1 SystemTicks

```
ulong Afterwarp.Utility.SystemTicks [static], [get]
```

Retrieves number of microseconds that passed since application startup.

The documentation for this struct was generated from the following files:

- [Afterwarp.Graphics.cs](#)
- [Afterwarp.Types.cs](#)

7.113 Afterwarp.VertexElement Struct Reference

Structure that describes the layout of a single buffer element.

Public Member Functions

- [VertexElement](#) (string name, [ElementFormat](#) format, int count, uint channel=0, uint offset=0)
Creates a vertex element structure with the given values.

Public Attributes

- byte[] [NameBytes](#)
UTF-8 encoded name of the component.
- [ElementFormat](#) [Format](#)
Data format of each value in this element.
- int [Count](#)
- uint [Channel](#)
- uint [Offset](#)
Offset in bytes relative to origin of the first buffer element to the current one.

Static Public Attributes

- const uint [ChannelNormalized](#) = 0x80000000u
Special bit added to vertex element channel indicating that the integer values are normalized.
- const uint [ChannelIndexBuffer](#) = 0x7FFFFFFFu
Special channel number that corresponds to index buffer.

Properties

- string [Name](#) [get, set]
Name of the component (variable name in GLSL or semantic name in HLSL).

7.113.1 Detailed Description

Structure that describes the layout of a single buffer element.

7.113.2 Constructor & Destructor Documentation

7.113.2.1 VertexElement()

```
Afterwarp.VertexElement.VertexElement (
    string name,
    ElementFormat format,
    int count,
    uint channel = 0,
    uint offset = 0 ) [inline]
```

Creates a vertex element structure with the given values.

7.113.3 Member Data Documentation

7.113.3.1 Channel

```
uint Afterwarp.VertexElement.Channel
```

Channel to which the values of this element will be bound to. The most significant bit of this field has a special purpose, meaning that the data values are normalized (applies only to integers), so the number of channel is stored in first 31 bits.

7.113.3.2 ChannelIndexBuffer

```
const uint Afterwarp.VertexElement.ChannelIndexBuffer = 0x7FFFFFFFu [static]
```

Special channel number that corresponds to index buffer.

7.113.3.3 ChannelNormalized

```
const uint Afterwarp.VertexElement.ChannelNormalized = 0x80000000u [static]
```

Special bit added to vertex element channel indicating that the integer values are normalized.

7.113.3.4 Count

```
int Afterwarp.VertexElement.Count
```

Number of values used by this element. If "format" is "Undefined", this indicates number of bytes the element occupies.

7.113.3.5 Format

`ElementFormat` Afterwarp.VertexElement.Format

Data format of each value in this element.

7.113.3.6 NameBytes

`byte []` Afterwarp.VertexElement.NameBytes

UTF-8 encoded name of the component.

7.113.3.7 Offset

`uint` Afterwarp.VertexElement.Offset

Offset in bytes relative to origin of the first buffer element to the current one.

7.113.4 Property Documentation

7.113.4.1 Name

`string` Afterwarp.VertexElement.Name `[get]`, `[set]`

Name of the component (variable name in GLSL or semantic name in HLSL).

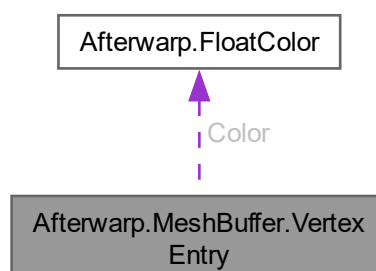
The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.114 Afterwarp.MeshBuffer.VertexEntry Struct Reference

A vertex element of mesh buffer with interleaved data.

Collaboration diagram for Afterwarp.MeshBuffer.VertexEntry:



Public Member Functions

- [VertexEntry](#) (Vector3 position, Vector3 normal, Vector3 tangent, Vector2 texCoord, [FloatColor](#) color)
Creates vertex element structure with the given values.

Public Attributes

- Vector3 [Position](#)
Vertex 3D position.
- Vector3 [Normal](#)
Vertex 3D normal (normalized).
- Vector3 [Tangent](#)
Vertex 3D tangent (normalized).
- Vector2 [TexCoord](#)
Vertex 2D texture coordinates.
- [FloatColor](#) [Color](#)
Vertex RGBA color.

7.114.1 Detailed Description

A vertex element of mesh buffer with interleaved data.

7.114.2 Constructor & Destructor Documentation

7.114.2.1 VertexEntry()

```
Afterwarp.MeshBuffer.VertexEntry.VertexEntry (
    Vector3 position,
    Vector3 normal,
    Vector3 tangent,
    Vector2 texCoord,
    FloatColor color ) [inline]
```

Creates vertex element structure with the given values.

7.114.3 Member Data Documentation

7.114.3.1 Color

```
FloatColor Afterwarp.MeshBuffer.VertexEntry.Color
```

Vertex RGBA color.

7.114.3.2 Normal

```
Vector3 Afterwarp.MeshBuffer.VertexEntry.Normal
```

Vertex 3D normal (normalized).

7.114.3.3 Position

```
Vector3 Afterwarp.MeshBuffer.VertexEntry.Position
```

Vertex 3D position.

7.114.3.4 Tangent

```
Vector3 Afterwarp.MeshBuffer.VertexEntry.Tangent
```

Vertex 3D tangent (normalized).

7.114.3.5 TexCoord

```
Vector2 Afterwarp.MeshBuffer.VertexEntry.TexCoord
```

Vertex 2D texture coordinates.

The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.115 Afterwarp.Volume Struct Reference

Static Public Member Functions

- static void [CalculateNearFarPlanes](#) (Matrix4x4 worldView, out float nearPlane, out float farPlane)
Calculates near and far planes for a bounding box transformed by the given world/view matrix.
- static [RectF CalculateVisibleFrame](#) (Matrix4x4 worldViewProjection, Vector2 surfaceSize)

7.115.1 Member Function Documentation

7.115.1.1 CalculateNearFarPlanes()

```
static void Afterwarp.Volume.CalculateNearFarPlanes (  
    Matrix4x4 worldView,  
    out float nearPlane,  
    out float farPlane ) [inline], [static]
```

Calculates near and far planes for a bounding box transformed by the given world/view matrix.

7.115.1.2 CalculateVisibleFrame()

```
static RectF Afterwarp.Volume.CalculateVisibleFrame (
    Matrix4x4 worldViewProjection,
    Vector2 surfaceSize ) [inline], [static]
```

Calculates visible surface frame for a bounding box transformed by the given world/view/projection matrix.

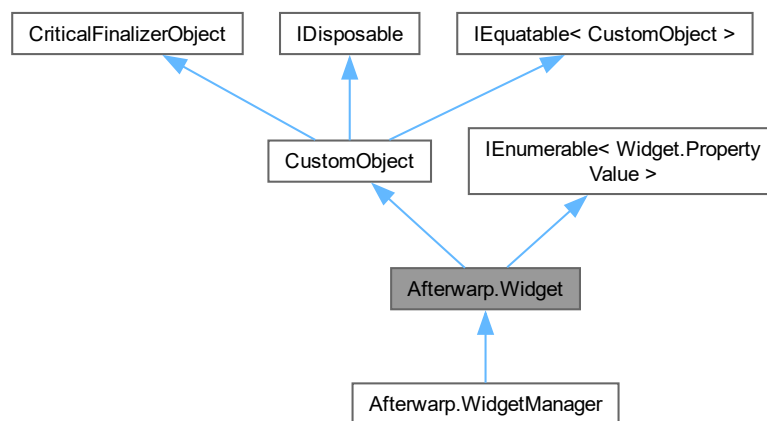
The documentation for this struct was generated from the following file:

- [Afterwarp.Graphics.cs](#)

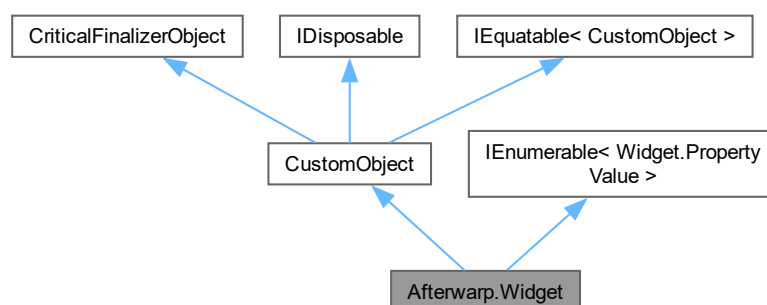
7.116 Afterwarp.Widget Class Reference

A widget module serving as a base element of user interface.

Inheritance diagram for Afterwarp.Widget:



Collaboration diagram for Afterwarp.Widget:



Classes

- class [Iterator](#)
Iterator for accessing properties in the widget.
- struct [PropertyValue](#)
A widget iteration value, consisting of a property name, type, behavior and its value.

Public Member Functions

- delegate bool [ExternalEventFunc](#) ([Widget](#) sender, string eventName)
External callback event type.
- [Widget](#) ([Widget](#) manager, string className, string instanceName=null, IntPtr payload=new IntPtr())
Creates a new instance of widget.
- Vector2 [LocalToScreen](#) (Vector2 localPos)
Converts local widget position to screen coordinates.
- Vector2 [ScreenToLocal](#) (Vector2 screenPos)
Converts screen coordinates to local widget position.
- void [BringToFront](#) ()
Puts the widget on front of other widgets.
- void [SendToBack](#) ()
Puts the widget behind all other widgets.
- void [Invalidate](#) ()
Schedules widget repaint during next update cycle.
- void [Accomodate](#) ()
Schedules widget repositioning during next update cycle.
- bool [Update](#) (double latency)
- void [AcceptMouse](#) ([MouseEvent](#) _event, [MouseButton](#) button, Vector2 position)
Makes the widget receive and process mouse input.
- void [AcceptKey](#) ([KeyEvent](#) _event, [Key](#) key, uint charCode)
Makes the widget receive and process keyboard input.
- [Widget](#) [GetChild](#) (int index)
Returns an existing widget's child with the given index.
- int [GetChildIndex](#) ([Widget](#) widget)
- [Widget](#) [FindAt](#) (Vector2 position)
Returns an existing widget's child with the given index.
- bool [InvokeEvent](#) (string eventName)
Returns an existing widget's child with the given index.
- object [Get](#) (string propertyName)
- void [Set](#) (string propertyName, object value, bool invokeChanged=true)
- T [Get< T >](#) (string propertyName)
Returns value of an existing property with the given name (case-insensitive) as a specific type.
- void [Set< T >](#) (string propertyName, T value, bool invokeChanged=true)
- [Margins](#) [GetPadding](#) (int zoneIndex)
Returns widget's padding for a particular zone (spacing inside of widget).
- void [SetPadding](#) (int zoneIndex, [Margins](#) margins)
Changes widget's padding for a particular zone (spacing inside of widget).
- [RectF](#) [GetClientRect](#) (int zoneIndex)
- IEnumerator< [PropertyValue](#) > [GetEnumerator](#) ()
Creates enumerator for iterating through widget's properties.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [WidgetManager Manager](#) [get]
Manager associated with the widget.
- IntPtr [Payload](#) [get]
Returns widget's payload.
- [Widget Parent](#) [get, set]
Retrieves or changes widget's parent.
- int [ChildCount](#) [get]
Number of existing widget's children.
- int [PropertyCount](#) [get]
Number of existing properties.
- [ExternalEventFunc ExternalEvent](#) [get, set]
Widget's external event callback.
- string [Name](#) [get, set]
Widget's name (case-insensitive).
- string [ClassName](#) [get]
Widget's class name (case-insensitive).
- string [Style](#) [get, set]
Widget's visual style.
- [WidgetAlignment Alignment](#) [get, set]
Widget's alignment.
- bool [Visible](#) [get, set]
Widget's visibility.
- bool [Enabled](#) [get, set]
Indicates whether the widget is able to receive keyboard input.
- bool [Focused](#) [get, set]
Indicates whether the widget is focused or not.
- int [ZoneCount](#) [get]
- int [ParentZone](#) [get, set]
Child's identifier that corresponds to certain parent's client area.
- [Margins Margins](#) [get, set]
Widget's margins (spacing outside of the widget).
- [RectF Rect](#) [get, set]
Widget's rectangle (in logical units).

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)

Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)

Wrapped object's handle.

7.116.1 Detailed Description

A widget module serving as a base element of user interface.

7.116.2 Constructor & Destructor Documentation

7.116.2.1 Widget()

```
Afterwarp.Widget.Widget (
    Widget manager,
    string className,
    string instanceName = null,
    IntPtr payload = new IntPtr() ) [inline]
```

Creates a new instance of widget.

7.116.3 Member Function Documentation

7.116.3.1 AcceptKey()

```
void Afterwarp.Widget.AcceptKey (
    KeyEvent _event,
    Key key,
    uint charCode ) [inline]
```

Makes the widget receive and process keyboard input.

7.116.3.2 AcceptMouse()

```
void Afterwarp.Widget.AcceptMouse (
    MouseEvent _event,
    MouseButton button,
    Vector2 position ) [inline]
```

Makes the widget receive and process mouse input.

7.116.3.3 Accomodate()

```
void Afterwarp.Widget.Accomodate ( )
```

Schedules widget repositioning during next update cycle.

7.116.3.4 BringToFront()

```
void Afterwarp.Widget.BringToFront ( )
```

Puts the widget on front of other widgets.

7.116.3.5 ExternalEventFunc()

```
delegate bool Afterwarp.Widget.ExternalEventFunc (
    Widget sender,
    string eventName )
```

External callback event type.

7.116.3.6 FindAt()

```
Widget Afterwarp.Widget.FindAt (
    Vector2 position ) [inline]
```

Returns an existing widget's child with the given index.

7.116.3.7 Get()

```
object Afterwarp.Widget.Get (
    string propertyName ) [inline]
```

Returns value of an existing property with the given name (case-insensitive). If such property is not found, returns null.

7.116.3.8 Get< T >()

```
T Afterwarp.Widget.Get< T > (
    string propertyName ) [inline]
```

Returns value of an existing property with the given name (case-insensitive) as a specific type.

7.116.3.9 GetChild()

```
Widget Afterwarp.Widget.GetChild (
    int index ) [inline]
```

Returns an existing widget's child with the given index.

7.116.3.10 GetChildIndex()

```
int Afterwarp.Widget.GetChildIndex (
    Widget widget ) [inline]
```

Attempts to find a widget among given parent's widget children and returns its index. If the given child is not found among parent's widget children, returns -1.

7.116.3.11 GetClientRect()

```
RectF Afterwarp.Widget.GetClientRect (
    int zoneIndex ) [inline]
```

Returns widget client area's rectangle. If widget does not support placement of child controls, then client rectangle should be its entire area.

7.116.3.12 GetEnumerator()

```
IEnumerator< PropertyValue > Afterwarp.Widget.GetEnumerator ( )
```

Creates enumerator for iterating through widget's properties.

7.116.3.13 GetPadding()

```
Margins Afterwarp.Widget.GetPadding (
    int zoneIndex ) [inline]
```

Returns widget's padding for a particular zone (spacing inside of widget).

7.116.3.14 Invalidate()

```
void Afterwarp.Widget.Invalidate ( )
```

Schedules widget repaint during next update cycle.

7.116.3.15 InvokeEvent()

```
bool Afterwarp.Widget.InvokeEvent (
    string eventName ) [inline]
```

Returns an existing widget's child with the given index.

7.116.3.16 LocalToScreen()

```
Vector2 Afterwarp.Widget.LocalToScreen (
    Vector2 localPos ) [inline]
```

Converts local widget position to screen coordinates.

7.116.3.17 ScreenToLocal()

```
Vector2 Afterwarp.Widget.ScreenToLocal (
    Vector2 screenPos ) [inline]
```

Converts screen coordinates to local widget position.

7.116.3.18 SendToBack()

```
void Afterwarp.Widget.SendToBack ( )
```

Puts the widget behind all other widgets.

7.116.3.19 Set()

```
void Afterwarp.Widget.Set (
    string propertyName,
    object value,
    bool invokeChanged = true ) [inline]
```

Changes contents of an existing property with the given name (case-insensitive) or creates a new property with the given value.

7.116.3.20 Set< T >()

```
void Afterwarp.Widget.Set< T > (
    string propertyName,
    T value,
    bool invokeChanged = true ) [inline]
```

Changes contents of an existing property with the given name (case-insensitive) or creates a new one. Raises an exception if the property cannot be updated.

7.116.3.21 SetPadding()

```
void Afterwarp.Widget.SetPadding (
    int zoneIndex,
    Margins margins ) [inline]
```

Changes widget's padding for a particular zone (spacing inside of widget).

7.116.3.22 Update()

```
bool Afterwarp.Widget.Update (
    double latency )
```

Updates widget and its children. `latency` must be a number of milliseconds elapsed since previous call to update; this is necessary to ensure proper animations in the widgets. Returns `true` if the application needs to be repainted or `false` otherwise.

7.116.4 Property Documentation

7.116.4.1 Alignment

`WidgetAlignment` Afterwarp.Widget.Alignment [get], [set]

Widget's alignment.

7.116.4.2 ChildCount

`int` Afterwarp.Widget.ChildCount [get]

Number of existing widget's children.

7.116.4.3 ClassName

`string` Afterwarp.Widget.ClassName [get]

Widget's class name (case-insensitive).

7.116.4.4 Enabled

`bool` Afterwarp.Widget.Enabled [get], [set]

Indicates whether the widget is able to receive keyboard input.

7.116.4.5 ExternalEvent

`ExternalEventFunc` Afterwarp.Widget.ExternalEvent [get], [set]

Widget's external event callback.

7.116.4.6 Focused

`bool` Afterwarp.Widget.Focused [get], [set]

Indicates whether the widget is focused or not.

7.116.4.7 Manager

`WidgetManager` Afterwarp.Widget.Manager [get]

Manager associated with the widget.

7.116.4.8 Margins

`Margins` Afterwarp.Widget.Margins [get], [set]

Widget's margins (spacing outside of the widget).

7.116.4.9 Name

`string` Afterwarp.Widget.Name [get], [set]

Widget's name (case-insensitive).

7.116.4.10 Parent

`Widget` Afterwarp.Widget.Parent [get], [set]

Retrieves or changes widget's parent.

7.116.4.11 ParentZone

`int` Afterwarp.Widget.ParentZone [get], [set]

Child's identifier that corresponds to certain parent's client area.

7.116.4.12 Payload

`IntPtr` Afterwarp.Widget.Payload [get]

Returns widget's payload.

7.116.4.13 PropertyCount

`int` Afterwarp.Widget.PropertyCount [get]

Number of existing properties.

7.116.4.14 Rect

`RectF` Afterwarp.Widget.Rect [get], [set]

Widget's rectangle (in logical units).

7.116.4.15 Style

`string` Afterwarp.Widget.Style [get], [set]

Widget's visual style.

7.116.4.16 Visible

```
bool Afterwarp.Widget.Visible [get], [set]
```

Widget's visibility.

7.116.4.17 ZoneCount

```
int Afterwarp.Widget.ZoneCount [get]
```

Returns number of supported client zones. If widget does not support placement of child controls, this function returns -1.

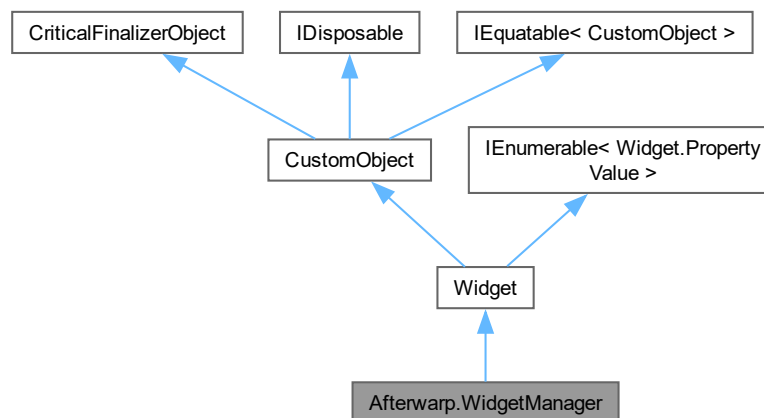
The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

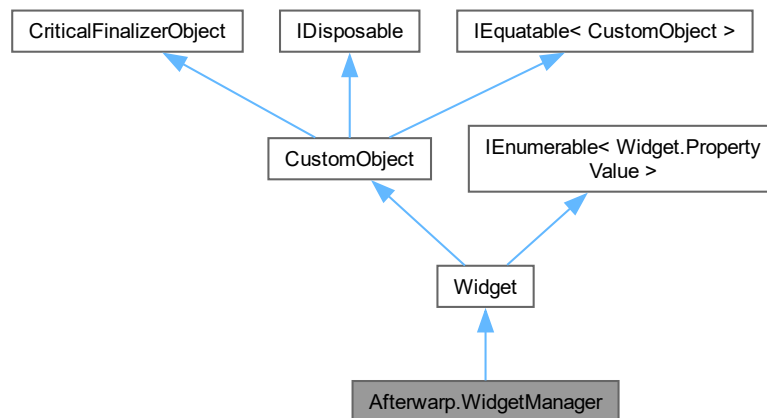
7.117 Afterwarp.WidgetManager Class Reference

A service module that contains and manages all other widgets.

Inheritance diagram for Afterwarp.WidgetManager:



Collaboration diagram for Afterwarp.WidgetManager:



Public Member Functions

- **WidgetManager** ([Application](#) application, [TextRenderer](#) textRenderer, string instanceName=null, IntPtr payload=new IntPtr(), [PixelFormat](#) format=[PixelFormat.Unknown](#), int multisamples=8, uint attributes=[WidgetManagerAttribute.Composition](#))
Creates a new instance of widget manager.
- **Widget GetWidget** (string instanceName)
Returns an existing widget with the given name (case-insensitive).
- **Widget TryGetWidget** (string instanceName)
- **Widget TryGetWidget** (IntPtr payload)
- void **Present** ()
Renders widget manager's area and all controls within.
- **Texture GetTexture** ([WidgetManagerTextureType](#) textureType)
Returns one of textures used for the composition.

Public Member Functions inherited from [Afterwarp.Widget](#)

- delegate bool **ExternalEventFunc** ([Widget](#) sender, string eventName)
External callback event type.
- **Widget** ([Widget](#) manager, string className, string instanceName=null, IntPtr payload=new IntPtr())
Creates a new instance of widget.
- Vector2 **LocalToScreen** (Vector2 localPos)
Converts local widget position to screen coordinates.
- Vector2 **ScreenToLocal** (Vector2 screenPos)
Converts screen coordinates to local widget position.
- void **BringToFront** ()
Puts the widget on front of other widgets.
- void **SendToBack** ()
Puts the widget behind all other widgets.
- void **Invalidate** ()
Schedules widget repaint during next update cycle.

- void [Accomodate](#) ()
Schedules widget repositioning during next update cycle.
- bool [Update](#) (double latency)
- void [AcceptMouse](#) ([MouseEvent](#) _event, [MouseButton](#) button, [Vector2](#) position)
Makes the widget receive and process mouse input.
- void [AcceptKey](#) ([KeyEvent](#) _event, [Key](#) key, uint charCode)
Makes the widget receive and process keyboard input.
- [Widget](#) [GetChild](#) (int index)
Returns an existing widget's child with the given index.
- int [GetChildIndex](#) ([Widget](#) widget)
- [Widget](#) [FindAt](#) ([Vector2](#) position)
Returns an existing widget's child with the given index.
- bool [InvokeEvent](#) (string eventName)
Returns an existing widget's child with the given index.
- object [Get](#) (string propertyName)
- void [Set](#) (string propertyName, object value, bool invokeChanged=true)
- T [Get< T >](#) (string propertyName)
Returns value of an existing property with the given name (case-insensitive) as a specific type.
- void [Set< T >](#) (string propertyName, T value, bool invokeChanged=true)
- [Margins](#) [GetPadding](#) (int zoneIndex)
Returns widget's padding for a particular zone (spacing inside of widget).
- void [SetPadding](#) (int zoneIndex, [Margins](#) margins)
Changes widget's padding for a particular zone (spacing inside of widget).
- [RectF](#) [GetClientRect](#) (int zoneIndex)
- [IEnumerator< PropertyValue >](#) [GetEnumerator](#) ()
Creates enumerator for iterating through widget's properties.

Public Member Functions inherited from [Afterwarp.CustomObject](#)

- void [Dispose](#) ()
- bool [Equals](#) ([CustomObject](#) other)
- override bool [Equals](#) (object obj)
- override int [GetHashCode](#) ()

Properties

- [Application](#) [Application](#) [get]
Returns application associated with the widget manager.
- [TextRenderer](#) [TextRenderer](#) [get]
Returns application associated with the widget manager.
- int [BatchCount](#) [get]
Returns total number of canvas batches during UI rendering.
- [PixelFormat](#) [Format](#) [get]
Returns pixel format of composition textures.
- int [Multisamples](#) [get]
Returns multisamples of rendering composition texture.
- uint [Attributes](#) [get]
Returns attributes used when the manager was created.
- double [Scale](#) [get, set]
General scale of widget manager and all widgets parented to it.
- [AppCursor](#) [Cursor](#) [get]
Currently active application cursor.

Properties inherited from [Afterwarp.Widget](#)

- [WidgetManager Manager](#) [get]
Manager associated with the widget.
- IntPtr [Payload](#) [get]
Returns widget's payload.
- [Widget Parent](#) [get, set]
Retrieves or changes widget's parent.
- int [ChildCount](#) [get]
Number of existing widget's children.
- int [PropertyCount](#) [get]
Number of existing properties.
- [ExternalEventFunc ExternalEvent](#) [get, set]
Widget's external event callback.
- string [Name](#) [get, set]
Widget's name (case-insensitive).
- string [ClassName](#) [get]
Widget's class name (case-insensitive).
- string [Style](#) [get, set]
Widget's visual style.
- [WidgetAlignment Alignment](#) [get, set]
Widget's alignment.
- bool [Visible](#) [get, set]
Widget's visibility.
- bool [Enabled](#) [get, set]
Indicates whether the widget is able to receive keyboard input.
- bool [Focused](#) [get, set]
Indicates whether the widget is focused or not.
- int [ZoneCount](#) [get]
- int [ParentZone](#) [get, set]
Child's identifier that corresponds to certain parent's client area.
- [Margins Margins](#) [get, set]
Widget's margins (spacing outside of the widget).
- [RectF Rect](#) [get, set]
Widget's rectangle (in logical units).

Properties inherited from [Afterwarp.CustomObject](#)

- IntPtr [Handle](#) [get]
Wrapped object's handle.

Additional Inherited Members

Protected Member Functions inherited from [Afterwarp.CustomObject](#)

- virtual void [Dispose](#) (bool disposing)
Releases the object's resources.

Protected Attributes inherited from [Afterwarp.CustomObject](#)

- IntPtr [_handle](#)

Wrapped object's handle.

7.117.1 Detailed Description

A service module that contains and manages all other widgets.

7.117.2 Constructor & Destructor Documentation

7.117.2.1 WidgetManager()

```
Afterwarp.WidgetManager.WidgetManager (
    Application application,
    TextRenderer textRenderer,
    string instanceName = null,
    IntPtr payload = new IntPtr(),
    PixelFormat format = PixelFormat::Unknown,
    int multisamples = 8,
    uint attributes = WidgetManagerAttribute::Composition ) [inline]
```

Creates a new instance of widget manager.

7.117.3 Member Function Documentation

7.117.3.1 GetTexture()

```
Texture Afterwarp.WidgetManager.GetTexture (
    WidgetManagerTextureType textureType ) [inline]
```

Returns one of textures used for the composition.

7.117.3.2 GetWidget()

```
Widget Afterwarp.WidgetManager.GetWidget (
    string instanceName ) [inline]
```

Returns an existing widget with the given name (case-insensitive).

7.117.3.3 Present()

```
void Afterwarp.WidgetManager.Present ( ) [inline]
```

Renders widget manager's area and all controls within.

7.117.3.4 TryGetWidget() [1/2]

```
Widget Afterwarp.WidgetManager.TryGetWidget (
    IntPtr payload ) [inline]
```

Returns an existing widget with the given payload. If widget with the given payload is not found, returns `null` instead of raising exception.

7.117.3.5 TryGetWidget() [2/2]

```
Widget Afterwarp.WidgetManager.TryGetWidget (
    string instanceName ) [inline]
```

Returns an existing widget with the given name (case-insensitive). If widget with the given name is not found, returns `null` instead of raising exception.

7.117.4 Property Documentation

7.117.4.1 Application

```
Application Afterwarp.WidgetManager.Application [get]
```

Returns application associated with the widget manager.

7.117.4.2 Attributes

```
uint Afterwarp.WidgetManager.Attributes [get]
```

Returns attributes used when the manager was created.

7.117.4.3 BatchCount

```
int Afterwarp.WidgetManager.BatchCount [get]
```

Returns total number of canvas batches during UI rendering.

7.117.4.4 Cursor

```
AppCursor Afterwarp.WidgetManager.Cursor [get]
```

Currently active application cursor.

7.117.4.5 Format

```
PixelFormat Afterwarp.WidgetManager.Format [get]
```

Returns pixel format of composition textures.

7.117.4.6 Multisamples

```
int Afterwarp.WidgetManager.Multisamples [get]
```

Returns multisamples of rendering composition texture.

7.117.4.7 Scale

```
double Afterwarp.WidgetManager.Scale [get], [set]
```

General scale of widget manager and all widgets parented to it.

7.117.4.8 TextRenderer

```
TextRenderer Afterwarp.WidgetManager.TextRenderer [get]
```

Returns application associated with the widget manager.

The documentation for this class was generated from the following file:

- [Afterwarp.Graphics.cs](#)

7.118 Afterwarp.WidgetManagerAttribute Struct Reference

Widget manager cumulative attribute flags that define its behavior.

Static Public Attributes

- const uint [Composition](#) = 0x01
Widgets support of composition effects such as glass and blur.
- const uint [Design](#) = 0x80
Design mode bit that enables additional functionality for creating interfaces on the fly.

7.118.1 Detailed Description

Widget manager cumulative attribute flags that define its behavior.

7.118.2 Member Data Documentation

7.118.2.1 Composition

```
const uint Afterwarp.WidgetManagerAttribute.Composition = 0x01 [static]
```

Widgets support of composition effects such as glass and blur.

7.118.2.2 Design

```
const uint Afterwarp.WidgetManagerAttribute.Design = 0x80 [static]
```

Design mode bit that enables additional functionality for creating interfaces on the fly.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

7.119 Afterwarp.WidgetProperty Struct Reference

Information about a widget property.

Public Attributes

- `byte[]` [NameBytes](#)
Name of the property described as an array of UTF-8 encoded bytes.
- [WidgetPropertyType](#) `Type`
Data type of the property.
- [WidgetPropertyBehavior](#) `Behavior`
Behavior of the property.
- `ushort` [Attributes](#)
Additional attributes of the property.
- `ushort` [Location](#)
- `ushort` [Reserved](#)
Reserved bits, must be set to zero.
- `byte[]` [DataBytes](#)
Contents of the property.

Properties

- `string` [Name](#) [get, set]
Name of the property.

7.119.1 Detailed Description

Information about a widget property.

7.119.2 Member Data Documentation

7.119.2.1 Attributes

```
ushort Afterwarp.WidgetProperty.Attributes
```

Additional attributes of the property.

7.119.2.2 Behavior

`WidgetPropertyBehavior` Afterwarp.WidgetProperty.Behavior

Behavior of the property.

7.119.2.3 DataBytes

`byte []` Afterwarp.WidgetProperty.DataBytes

Contents of the property.

7.119.2.4 Location

`ushort` Afterwarp.WidgetProperty.Location

Location of the property in the list of existing properties. This is an opaque value that loosely determines next property to be iterated. The actual value is temporary and valid only during continuous iteration. Any change to properties invalidates all previous iteration values. A value of zero indicates an invalid or "undefined" location.

7.119.2.5 NameBytes

`byte []` Afterwarp.WidgetProperty.NameBytes

Name of the property described as an array of UTF-8 encoded bytes.

7.119.2.6 Reserved

`ushort` Afterwarp.WidgetProperty.Reserved

Reserved bits, must be set to zero.

7.119.2.7 Type

`WidgetPropertyType` Afterwarp.WidgetProperty.Type

Data type of the property.

7.119.3 Property Documentation

7.119.3.1 Name

`string` Afterwarp.WidgetProperty.Name [get], [set]

Name of the property.

The documentation for this struct was generated from the following file:

- [Afterwarp.Types.cs](#)

Chapter 8

File Documentation

8.1 Afterwarp.API.cs File Reference

Classes

- struct [Afterwarp.API](#)

Namespaces

- namespace [Afterwarp](#)

8.2 Afterwarp.Graphics.cs File Reference

Classes

- struct [Afterwarp.API](#)
- class [Afterwarp.API.Exception](#)
Exception type raised by [Afterwarp](#) classes.
- class [Afterwarp.CustomObject](#)
A high-level class that wraps one of [Afterwarp](#)'s objects.
- class [Afterwarp.Device](#)
Device class that handles initialization, rendering, state management and other common tasks.
- struct [Afterwarp.Device.Attribute](#)
Device attributes that define its behavior.
- class [Afterwarp.SwapChain](#)
Swap-chain that enables double-buffered rendering to a particular window.
- class [Afterwarp.Buffer](#)
Generic device buffer object that can contain vertices, indices or shader data.
- class [Afterwarp.Program](#)
Shader program that is typically executed on graphics hardware.
- class [Afterwarp.ComputeProgram](#)
Compute shader program for general processing on GPU (GPGPU).
- class [Afterwarp.Sampler](#)
Sampler object that defines how texture is read by the shaders.

- class [Afterwarp.Surface](#)
Surface that contains image in memory for pixel manipulation.
- class [Afterwarp.Texture](#)
Texture that contains image data typically stored in GPU memory.
- struct [Afterwarp.Texture.Attribute](#)
Attribute that defines special behavior and/or unique characteristics of the texture.
- class [Afterwarp.CanvasBuffer](#)
- class [Afterwarp.PathBroker](#)
- class [Afterwarp.Canvas](#)
- struct [Afterwarp.Canvas.Attribute](#)
Canvas attribute flags that can be combined together.
- class [Afterwarp.ImageAtlas](#)
Image atlas, sub-images of which can be rendered with canvas.
- class [Afterwarp.TextRenderer](#)
Text renderer utility that can draw text on the canvas.
- class [Afterwarp.TextModeller](#)
2D and 3D text rendering module.
- class [Afterwarp.TextModeller.FontProvider](#)
A provider of font geometry for the modeller.
- class [Afterwarp.Grapher](#)
3D graph plotting module.
- class [Afterwarp.MeshModel](#)
3D mesh model that contains both vertex and index buffers.
- class [Afterwarp.MeshMetaTag](#)
Meta-tag, which describes a certain important axis-aligned bounding area inside a mesh.
- class [Afterwarp.MeshMetaTags](#)
List of meta-tags that describe important axis-aligned bounding areas inside a mesh.
- class [Afterwarp.MeshBuffer](#)
Buffer that contains 3D mesh information.
- struct [Afterwarp.MeshBuffer.VertexEntry](#)
A vertex element of mesh buffer with interleaved data.
- class [Afterwarp.GaussianBlur](#)
Gaussian Blur module.
- class [Afterwarp.KawaseBlur](#)
Kawase Blur module.
- class [Afterwarp.ColorDithering](#)
- class [Afterwarp.SelectionHighlight](#)
- class [Afterwarp.SpatialFog](#)
Module that performs both ground fog and distance-based fog simultaneously.
- class [Afterwarp.SceneLights](#)
Module for storing and working with lights in a 3D scene.
- class [Afterwarp.ObjectMaterials](#)
Container for storing 3D object materials.
- class [Afterwarp.TextureCabinet](#)
A container and management module for drawable textures for rendering and processing 3D scene.
- struct [Afterwarp.TextureCabinet.Attribute](#)
Cumulative attributes that define rendering characteristics of the scene.
- class [Afterwarp.ShadowCaster](#)
A source that casts shadows that can be shared among multiple light sources.
- class [Afterwarp.ShadowCastingAtlas](#)
Module that manages shadow maps produced by one or more shadow casters.

- class [Afterwarp.Scene](#)
A module for rendering 3D scenes.
- struct [Afterwarp.Scene.Attribute](#)
Cumulative attributes that define rendering behavior of the scene.
- class [Afterwarp.OceanSimulation](#)
3D ocean semi-infinite wave field rendering module.
- class [Afterwarp.ObjectModel](#)
Object that represents a 3D model in the scene.
- struct [Afterwarp.ObjectModel.Attribute](#)
Attribute bit flags for an object that define its status.
- class [Afterwarp.ObjectModels](#)
A container of objects and their volume representation in 3D world.
- class [Afterwarp.ObjectModels.Iterator](#)
Iterator for accessing individual objects in the container.
- class [Afterwarp.ObjectModelView](#)
A container that represents a particular view that watches objects in the world.
- class [Afterwarp.MeshVoxel](#)
3D mesh voxel representation object.
- class [Afterwarp.SceneMeshMaterial](#)
Material that describes how a portion of a scene mesh geometry should look like.
- class [Afterwarp.SceneMeshMaterials](#)
Container for scene mesh materials that describe the appearance of scene mesh geometry.
- class [Afterwarp.SceneMeshLatches](#)
Container for scene mesh latches that define how meshes connect with each other.
- class [Afterwarp.SceneMesh](#)
A container for a 3D scene mesh model, its optional voxel representation and a bounding volume.
- class [Afterwarp.SceneMeshes](#)
A collection of 3D scene mesh containers.
- class [Afterwarp.ActorCamera](#)
- class [Afterwarp.Timer](#)
- class [Afterwarp.Application](#)
Application execution manager that handles OS events and controls application's window.
- class [Afterwarp.Widget](#)
A widget module serving as a base element of user interface.
- struct [Afterwarp.Widget.PropertyValue](#)
A widget iteration value, consisting of a property name, type, behavior and its value.
- class [Afterwarp.Widget.Iterator](#)
Iterator for accessing properties in the widget.
- class [Afterwarp.WidgetManager](#)
A service module that contains and manages all other widgets.
- struct [Afterwarp.RandomSequence](#)
Pseudo-random number generator (PRNG) utility module.
- struct [Afterwarp.Library](#)
Shared library service functions.
- struct [Afterwarp.Ray](#)
A structure that defines a ray in 3D space.
- struct [Afterwarp.Volume](#)
- struct [Afterwarp.Utility](#)

Namespaces

- namespace [Afterwarp](#)

8.3 Afterwarp.Types.cs File Reference

Classes

- struct [Afterwarp.API](#)
- struct [Afterwarp.DeviceClear](#)
Type of surface layer that should be cleared.
- struct [Afterwarp.DeviceBehavior](#)
Device behavior attributes that define the rendering code path.
- struct [Afterwarp.RenderingState](#)
Parameters that define depth, stencil, rasterizer and blending operations.
- struct [Afterwarp.RenderingState.State](#)
State flags that can be combined together to form a rendering operation state.
- struct [Afterwarp.RenderingState.StencilState](#)
Stencil state that is independent for each front and back facing.
- struct [Afterwarp.RenderingState.Blend](#)
Blending parameters that are specific to a certain component, or a portion of color.
- struct [Afterwarp.VertexElement](#)
Structure that describes the layout of a single buffer element.
- struct [Afterwarp.ProgramElement](#)
Structure that describes a single physical element of shader program.
- struct [Afterwarp.ProgramVariable](#)
Structure that describes a single (uniform) variable in a shader program.
- struct [Afterwarp.ComputeBindTextureFormat](#)
Compute texture binding parameters.
- struct [Afterwarp.PathCommand](#)
Path commands and attribute bits.
- struct [Afterwarp.PathElement](#)
A single element in path declaration.
- class [Afterwarp.PathBuilder](#)
A container and helper module to facilitate creation of path elements.
- struct [Afterwarp.SamplerState](#)
Sampler parameters that are associated with a particular texture unit.
- struct [Afterwarp.TextureParameters](#)
Texture parameters and characteristics.
- struct [Afterwarp.SignedDistanceField](#)
Signed Distance Field (SDF) parameters.
- struct [Afterwarp.CanvasSamplerState](#)
Sampler parameters that can be easily configured by the canvas.
- struct [Afterwarp.ImageRegion](#)
An image region defined by its texture number and bounding rectangle on that texture.
- struct [Afterwarp.FontAttribute](#)
Font attribute flags that define some visual characteristics.
- struct [Afterwarp.TextEntryRect](#)
Individual text entry that will appear as rendered.

- struct [Afterwarp.TextRenderModifiers](#)
Modifier attributes that can be applied to the rendered text.
- struct [Afterwarp.FontEffect](#)
Attributes and characteristics that define how font glyphs are rasterized.
- struct [Afterwarp.FontParameters](#)
Parameters that define the appearance and important characteristics of the font.
- struct [Afterwarp.FogParameters](#)
Structure containing spatial fog characteristics.
- struct [Afterwarp.ShadowParameters](#)
Parameters that define how shadows are rendered.
- struct [Afterwarp.ToneMappingBloom](#)
Parameters that define behavior of tone-mapping and bloom effects.
- struct [Afterwarp.AmbientOcclusionParameters](#)
Parameters that define how ambient occlusion is performed.
- struct [Afterwarp.ParallaxMappingParameters](#)
Parameters that define how parallax mapping is performed.
- struct [Afterwarp.SceneLight](#)
Light source parameters in 3D scene.
- struct [Afterwarp.ObjectMaterial](#)
Material that describes a particular object in 3D world.
- struct [Afterwarp.OceanWavesParameters](#)
Parameters that define the appearance of waves simulation.
- struct [Afterwarp.OceanMaterial](#)
Material that describes ocean water surface in a 3D simulation.
- struct [Afterwarp.SelectionHighlightParameters](#)
Parameters that define how selection highlight technique is performed.
- struct [Afterwarp.MeshMetaTagType](#)
Type of the mesh meta-tag.
- struct [Afterwarp.MeshMetaTagPortion](#)
A portion of the mesh corresponding to a particular tag.
- struct [Afterwarp.SceneMeshMaterialShading](#)
Shading parameters for scene mesh material.
- struct [Afterwarp.SceneMeshMaterialRange](#)
A range of indices in the mesh that a material applies to.
- struct [Afterwarp.SceneMeshLatchType](#)
Type of latch used in the 3D scene mesh.
- struct [Afterwarp.SceneMeshLatch](#)
A latch in 3D scene mesh, which describes a bone joint connection or a movement waypoint.
- struct [Afterwarp.MeshLoadingOption](#)
Non-exclusive options passed and/or returned from native mesh loader.
- struct [Afterwarp.AutoDrawOption](#)
Cumulative options that can be passed to one of scene mesh "AutoDraw" helper functions.
- struct [Afterwarp.MeshAligns](#)
Alignment for all three axes that determine the placement of mesh around its model.
- struct [Afterwarp.CameraConstraints](#)
- struct [Afterwarp.ApplicationConfiguration](#)
Structure that contains all the parameters necessary to create a new application and its window.
- struct **Afterwarp.ApplicationEvents**
- struct [Afterwarp.WidgetProperty](#)
Information about a widget property.
- struct [Afterwarp.WidgetManagerAttribute](#)

Widget manager cumulative attribute flags that define its behavior.

- struct [Afterwarp.Utility](#)
- struct [Afterwarp.ColorPair](#)
- struct [Afterwarp.ColorRect](#)
- struct [Afterwarp.FloatColorRGB](#)
- struct [Afterwarp.FloatColor](#)
- struct [Afterwarp.Point](#)

2D integer point.

- struct [Afterwarp.Rect](#)

Rectangle defined by integer top and left margins, width and height.

- struct [Afterwarp.RectF](#)

Rectangle defined by floating-point top and left margins, width and height.

- struct [Afterwarp.Margins](#)

Margins defined by top, left, right and bottom edge paddings.

- struct [Afterwarp.Quad](#)

Floating-point quadrilateral defined by four vertices in clockwise order starting from top/left.

Namespaces

- namespace [Afterwarp](#)

Enumerations

- enum [Afterwarp.DeviceTechnology](#) : byte {
[Afterwarp.Unknown](#) , [Afterwarp.Direct3D](#) , [Afterwarp.OpenGL](#) , [Afterwarp.OpenGL_ES](#) ,
[Afterwarp.Vulkan](#) , [Afterwarp.Metal](#) , [Afterwarp.WebGL](#) , [Afterwarp.Software](#) ,
[Afterwarp.Proprietary](#) }

Type of graphics technology used in the application.

- enum [Afterwarp.DevicePlatform](#) : byte {
[Afterwarp.Unknown](#) , [Afterwarp.Windows](#) , [Afterwarp.Linux](#) , [Afterwarp.Unix](#) ,
[Afterwarp.OSX](#) , [Afterwarp.iOS](#) , [Afterwarp.Android](#) }

Type of OS platform the application is running on.

- enum [Afterwarp.PixelFormat](#) : byte {
[Afterwarp.Unknown](#) , [Afterwarp.R8](#) , [Afterwarp.RG8](#) , [Afterwarp.RGBA8](#) ,
[Afterwarp.R16](#) , [Afterwarp.RG16](#) , [Afterwarp.RGBA16](#) , [Afterwarp.R8S](#) ,
[Afterwarp.RG8S](#) , [Afterwarp.RGBA8S](#) , [Afterwarp.R16S](#) , [Afterwarp.RG16S](#) ,
[Afterwarp.RGBA16S](#) , [Afterwarp.R8U](#) , [Afterwarp.RG8U](#) , [Afterwarp.RGBA8U](#) ,
[Afterwarp.R16U](#) , [Afterwarp.RG16U](#) , [Afterwarp.RGBA16U](#) , [Afterwarp.R32U](#) ,
[Afterwarp.RG32U](#) , [Afterwarp.RGBA32U](#) , [Afterwarp.R8I](#) , [Afterwarp.RG8I](#) ,
[Afterwarp.RGBA8I](#) , [Afterwarp.R16I](#) , [Afterwarp.RG16I](#) , [Afterwarp.RGBA16I](#) ,
[Afterwarp.R32I](#) , [Afterwarp.RG32I](#) , [Afterwarp.RGBA32I](#) , [Afterwarp.R16F](#) ,
[Afterwarp.RG16F](#) , [Afterwarp.RGBA16F](#) , [Afterwarp.R32F](#) , [Afterwarp.RG32F](#) ,
[Afterwarp.RGBA32F](#) , [Afterwarp.RG11B10F](#) , [Afterwarp.RGB9E5F](#) , [Afterwarp.RGB10A2](#) ,
[Afterwarp.RGB10A2U](#) , [Afterwarp.RGBX8](#) , [Afterwarp.RGBA8_SRGB](#) , [Afterwarp.BGRA8](#) ,
[Afterwarp.BGRX8](#) , [Afterwarp.BGRA8_SRGB](#) , [Afterwarp.BGR10A2](#) , [Afterwarp.BGR10X2](#) ,
[Afterwarp.BGRA4](#) , [Afterwarp.BGRX4](#) , [Afterwarp.B5G6R5](#) , [Afterwarp.BGR5A1](#) ,
[Afterwarp.BGR5X1](#) , [Afterwarp.RGB8](#) , [Afterwarp.BGR8](#) , [Afterwarp.A8](#) ,
[Afterwarp.L8](#) , [Afterwarp.L16](#) , [Afterwarp.LA4](#) , [Afterwarp.LA8](#) ,
[Afterwarp.I8](#) , [Afterwarp.R_BC](#) , [Afterwarp.RG_BC](#) , [Afterwarp.RGB_BC](#) ,
[Afterwarp.RGBA_BC](#) , [Afterwarp.RGB_BC_SRGB](#) , [Afterwarp.RGBA_BC_SRGB](#) , [Afterwarp.D16](#) ,
[Afterwarp.D24S8](#) , [Afterwarp.D32F](#) , [Afterwarp.D32S8F](#) }

Defines how individual pixels and their colors are encoded in images and textures.

- enum [Afterwarp.AlphaFormatRequest](#) : byte { [Afterwarp.DontCare](#) , [Afterwarp.NonPremultiplied](#) , [Afterwarp.Premultiplied](#) }
Defines how alpha-channel should be handled in the loaded image.
- enum [Afterwarp.ElementFormat](#) : ushort { [Afterwarp.Undefined](#) , [Afterwarp.Float](#) , [Afterwarp.HalfFloat](#) , [Afterwarp.Double](#) , [Afterwarp.Int](#) , [Afterwarp.UnsignedInt](#) , [Afterwarp.Short](#) , [Afterwarp.UnsignedShort](#) , [Afterwarp.Byte](#) , [Afterwarp.UnsignedByte](#) , [Afterwarp.Float_11_11_10](#) }
Format in which a single value of the given element is represented.
- enum [Afterwarp.ShaderType](#) : byte { [Afterwarp.Undefined](#) , [Afterwarp.Fragment](#) , [Afterwarp.Vertex](#) , [Afterwarp.Geometry](#) , [Afterwarp.Compute](#) , [Afterwarp.Hull](#) , [Afterwarp.Domain](#) }
Type of shader in graphics rendering pipeline.
- enum [Afterwarp.ShaderElement](#) : byte { [Afterwarp.Undefined](#) , [Afterwarp.Texture](#) , [Afterwarp.ConstantBuffer](#) , [Afterwarp.TypedBuffer](#) , [Afterwarp.RWTypedBuffer](#) , [Afterwarp.StructuredBuffer](#) , [Afterwarp.RWStructuredBuffer](#) }
Type that characterizes shader element.
- enum [Afterwarp.TriangleFace](#) : byte { [Afterwarp.None](#) , [Afterwarp.Back](#) , [Afterwarp.Front](#) , [Afterwarp.Both](#) }
Type of triangle face that should have operation applied to.
- enum [Afterwarp.ComparisonFunc](#) : byte { [Afterwarp.Always](#) , [Afterwarp.Less](#) , [Afterwarp.LessEqual](#) , [Afterwarp.Greater](#) , [Afterwarp.GreaterEqual](#) , [Afterwarp.Equal](#) , [Afterwarp.NotEqual](#) , [Afterwarp.Never](#) }
Function that defines how depth/stencil operands should be compared.
- enum [Afterwarp.StencilOp](#) : byte { [Afterwarp.Keep](#) , [Afterwarp.Zero](#) , [Afterwarp.Replace](#) , [Afterwarp.Invert](#) , [Afterwarp.Increment](#) , [Afterwarp.Decrement](#) , [Afterwarp.IncrementWrap](#) , [Afterwarp.DecrementWrap](#) }
Operation that should be performed on stencil value.
- enum [Afterwarp.BlendFactor](#) : byte { [Afterwarp.One](#) , [Afterwarp.Zero](#) , [Afterwarp.SourceColor](#) , [Afterwarp.InvSourceColor](#) , [Afterwarp.SourceAlpha](#) , [Afterwarp.InvSourceAlpha](#) , [Afterwarp.DestColor](#) , [Afterwarp.InvDestColor](#) , [Afterwarp.DestAlpha](#) , [Afterwarp.InvDestAlpha](#) , [Afterwarp.SourceAlphaSat](#) , [Afterwarp.ConstantColor](#) , [Afterwarp.InvConstantColor](#) , [Afterwarp.ConstantAlpha](#) , [Afterwarp.InvConstantAlpha](#) , [Afterwarp.SourceColor1](#) , [Afterwarp.InvSourceColor1](#) , [Afterwarp.SourceAlpha1](#) , [Afterwarp.InvSourceAlpha1](#) }
Factor that determines how alpha-blending operand is considered.
- enum [Afterwarp.BlendOp](#) : byte { [Afterwarp.Add](#) , [Afterwarp.Subtract](#) , [Afterwarp.InvSubtract](#) , [Afterwarp.Minimum](#) , [Afterwarp.Maximum](#) }
Operation that should be performed between two blending operands.
- enum [Afterwarp.BufferDataType](#) : byte { [Afterwarp.Vertex](#) , [Afterwarp.Index](#) , [Afterwarp.Constant](#) , [Afterwarp.Typed](#) , [Afterwarp.RWTyped](#) , [Afterwarp.Structured](#) , [Afterwarp.RWStructured](#) }
Type of data that a generic buffer contains.
- enum [Afterwarp.BufferAccessType](#) : byte { [Afterwarp.Default](#) , [Afterwarp.Static](#) , [Afterwarp.Dynamic](#) , [Afterwarp.Compute](#) , [Afterwarp.Staging](#) }
Type of access that is performed with mesh buffer.
- enum [Afterwarp.ComputeTextureAccess](#) : byte { [Afterwarp.Read](#) , [Afterwarp.Write](#) , [Afterwarp.ReadWrite](#) }
Compute program texture access type.
- enum [Afterwarp.TextureFilter](#) : byte { [Afterwarp.None](#) , [Afterwarp.Nearest](#) , [Afterwarp.Linear](#) , [Afterwarp.Anisotropic](#) }
Filtering mode that defines how colors are sampled from the texture.
- enum [Afterwarp.TextureAddress](#) : byte { [Afterwarp.Wrap](#) , [Afterwarp.Clamp](#) , [Afterwarp.Mirror](#) , [Afterwarp.MirrorOnce](#) , [Afterwarp.Border](#) }

Addressing mode that defines how texture coordinates outside of [0, 1] range are treated.

- enum [Afterwarp.PrimitiveTopology](#) : byte {
[Afterwarp.Unknown](#) , [Afterwarp.Points](#) , [Afterwarp.Lines](#) , [Afterwarp.LineStrip](#) ,
[Afterwarp.Triangles](#) , [Afterwarp.TriangleStrip](#) , [Afterwarp.LinesAdjacency](#) , [Afterwarp.LineStripAdjacency](#) ,
[Afterwarp.TrianglesAdjacency](#) , [Afterwarp.TriangleStripAdjacency](#) , [Afterwarp.Patches](#) }

Rendering primitive topology.

- enum [Afterwarp.PathJoint](#) : byte {
[Afterwarp.None](#) , [Afterwarp.Simple](#) , [Afterwarp.Miter](#) , [Afterwarp.Bevel](#) ,
[Afterwarp.Round](#) , [Afterwarp.MiterBevel](#) }

Type of joint for connecting path lines.

- enum [Afterwarp.LineCaps](#) : byte { [Afterwarp.Butt](#) , [Afterwarp.Square](#) , [Afterwarp.Round](#) }

Type of caps that covers line end points.

- enum [Afterwarp.PointShape](#) : byte {
[Afterwarp.Square](#) , [Afterwarp.Round](#) , [Afterwarp.Triangle](#) , [Afterwarp.Star](#) ,
[Afterwarp.Cross](#) }

Shape of point primitive.

- enum [Afterwarp.TextureType](#) : byte { [Afterwarp.Surface](#) , [Afterwarp.CubeMap](#) , [Afterwarp.Volume](#) }

Type of texture that defines how it is composed.

- enum [Afterwarp.BlendingEffect](#) : byte {
[Afterwarp.Undefined](#) = 0 , [Afterwarp.Normal](#) , [Afterwarp.Shadow](#) , [Afterwarp.Add](#) ,
[Afterwarp.Subtract](#) , [Afterwarp.Multiply](#) , [Afterwarp.InverseMultiply](#) , [Afterwarp.SourceColor](#) ,
[Afterwarp.SourceColorAdd](#) , [Afterwarp.Copy](#) = 254 , [Afterwarp.Custom](#) = 255 }

Blending effect that defines how primitives are rendered on drawing surface.

- enum [Afterwarp.CanvasContextState](#) : byte { [Afterwarp.FlatScene](#) , [Afterwarp.FlatText](#) , [Afterwarp.Projectected](#) ,
[Afterwarp.Undefined](#) = 255 }

One of possible defined context states that can be pre-configured by the canvas.

- enum [Afterwarp.SuperSampleSDF](#) : byte { [Afterwarp.NoSuperSample](#) , [Afterwarp.SuperSample4x](#) ,
[Afterwarp.SuperSample16x](#) }

Type of super-sampling used for rendering Signed Distance Field (SDF).

- enum [Afterwarp.FontWeight](#) : byte {
[Afterwarp.Thin](#) , [Afterwarp.ExtraLight](#) , [Afterwarp.Light](#) , [Afterwarp.SemiLight](#) ,
[Afterwarp.Normal](#) , [Afterwarp.Medium](#) , [Afterwarp.SemiBold](#) , [Afterwarp.Bold](#) ,
[Afterwarp.ExtraBold](#) , [Afterwarp.Heavy](#) , [Afterwarp.ExtraHeavy](#) }

Weight of the font (apparent thickness of glyphs).

- enum [Afterwarp.FontStretch](#) : byte {
[Afterwarp.UltraCondensed](#) , [Afterwarp.ExtraCondensed](#) , [Afterwarp.Condensed](#) , [Afterwarp.SemiCondensed](#) ,
[Afterwarp.Normal](#) , [Afterwarp.SemiExpanded](#) , [Afterwarp.Expanded](#) , [Afterwarp.ExtraExpanded](#) ,
[Afterwarp.UltraExpanded](#) }

Relative width of the font.

- enum [Afterwarp.FontSlant](#) : byte { [Afterwarp.None](#) , [Afterwarp.Oblique](#) , [Afterwarp.Italic](#) }

Font slant styles.

- enum [Afterwarp.FontBorder](#) : byte { [Afterwarp.None](#) , [Afterwarp.Normal](#) , [Afterwarp.SemiHeavy](#) ,
[Afterwarp.Heavy](#) }

Border that outlines text glyph's shapes.

- enum [Afterwarp.TextAlignment](#) : byte { [Afterwarp.Start](#) , [Afterwarp.Middle](#) , [Afterwarp.End](#) }

Text alignment when drawing with certain functions.

- enum [Afterwarp.ColorDitheringFormat](#) : byte {
[Afterwarp.RGB8](#) , [Afterwarp.RGB6](#) , [Afterwarp.R5G6B5](#) , [Afterwarp.RGB4](#) ,
[Afterwarp.RG3B2](#) }

Format to be used as target for color dithering.

- enum [Afterwarp.FogFormula](#) : byte { [Afterwarp.Linear](#) , [Afterwarp.Exponential](#) , [Afterwarp.Ground](#) }

Defines type of formula used in fog calculation.

- enum [Afterwarp.DepthFogDistance](#) : byte { [Afterwarp.ZBased](#) , [Afterwarp.EyeRelative](#) }

- Defines how distance is calculated for depth fog.*

 - enum [Afterwarp.TechniqueLighting](#) : byte { [Afterwarp.Phong](#) , [Afterwarp.Minnaert](#) , [Afterwarp.CookTorrance](#) , [Afterwarp.OrenNayer](#) }

Lighting technique.
- enum [Afterwarp.TechniqueShadows](#) : byte { [Afterwarp.None](#) , [Afterwarp.ESM](#) , [Afterwarp.ESM_Warp](#) , [Afterwarp.EVSM](#) }

Shadow rendering technique.
- enum [Afterwarp.ClustersCullingMode](#) : byte { [Afterwarp.Quality](#) , [Afterwarp.Performance](#) }

Light culling mode for clustered shading.
- enum [Afterwarp.TextureFidelity](#) : byte { [Afterwarp.Low](#) , [Afterwarp.Medium](#) , [Afterwarp.High](#) , [Afterwarp.Extreme](#) }

Determines the choice of pixel formats for the container.
- enum [Afterwarp.TextureCabinetType](#) : byte { [Afterwarp.Depth](#) = 0 , [Afterwarp.Normal](#)s = 1 , [Afterwarp.ColorHDR](#) = 4 , [Afterwarp.Color](#) = 5 , [Afterwarp.LinearDepths](#) = 11 , [Afterwarp.GlassyColor](#) = 12 , [Afterwarp.GlassyCounts](#) = 13 , [Afterwarp.GlassyComposite](#) = 14 }

Texture type used in the scene.
- enum [Afterwarp.TextureCabinetPass](#) : byte { [Afterwarp.Color](#) , [Afterwarp.Glassy](#) , [Afterwarp.Depth](#) }

Rendering pass that defines what textures are being used and how they are cleared.
- enum [Afterwarp.TextureCabinetFilterType](#) : byte { [Afterwarp.Occlusion](#) , [Afterwarp.Bloom](#) , [Afterwarp.Glassy](#) , [Afterwarp.GlassyFog](#) }

Filter type that defines how textures are processed.
- enum [Afterwarp.SceneTextureType](#) : byte { [Afterwarp.Albedo](#) , [Afterwarp.NormalMap](#) , [Afterwarp.ParallaxMap](#) }

Type of texture used by the scene.
- enum [Afterwarp.SceneSamplerType](#) : byte { [Afterwarp.Albedo](#) , [Afterwarp.NormalMap](#) , [Afterwarp.ParallaxMap](#) , [Afterwarp.ShadowMap](#) }

Type of sampler used by the scene.
- enum [Afterwarp.SelectionHighlightTextureType](#) : byte { [Afterwarp.Highlight](#) , [Afterwarp.HighlightMask](#) }

Texture type used by the technique.
- enum [Afterwarp.SceneMeshTexture](#) : byte { [Afterwarp.Diffuse](#) = 0 , [Afterwarp.Normal](#)s = 1 , [Afterwarp.Parallax](#) = 2 }

Type of texture used in scene mesh material.
- enum [Afterwarp.ModelTransform](#) : byte { [Afterwarp.Local](#) , [Afterwarp.LocalModel](#) , [Afterwarp.LocalVolume](#) , [Afterwarp.Global](#) , [Afterwarp.GlobalModel](#) , [Afterwarp.GlobalVolume](#) }

Type of transform as used by object models.
- enum [Afterwarp.MeshAlign](#) : byte { [Afterwarp.Positive](#) , [Afterwarp.Origin](#) , [Afterwarp.Negative](#) , [Afterwarp.Unaligned](#) }

Axis alignment that determines the placement of mesh around its model.
- enum [Afterwarp.ObjectModelViewCompare](#) : byte { [Afterwarp.Depth](#) , [Afterwarp.DepthReverse](#) , [Afterwarp.Ordered](#) , [Afterwarp.Meshes](#) , [Afterwarp.Objects](#) }

Type of comparison applied to an ordered list of objects.
- enum [Afterwarp.CameraCommand](#) : byte { [Afterwarp.Movement](#) , [Afterwarp.Dragging](#) , [Afterwarp.Rotation](#) , [Afterwarp.Update](#) , [Afterwarp.Stop](#) }

Camera movement/rotation command.
- enum [Afterwarp.ApplicationWindowState](#) : byte { [Afterwarp.Windowed](#) , [Afterwarp.Minimized](#) , [Afterwarp.Maximized](#) , [Afterwarp.FullScreen](#) }

Application's window state enumeration.
- enum [Afterwarp.MouseEvent](#) : byte { [Afterwarp.Down](#) , [Afterwarp.Move](#) , [Afterwarp.Up](#) , [Afterwarp.WheelUp](#) , [Afterwarp.WheelDown](#) , [Afterwarp.Enter](#) , [Afterwarp.Leave](#) , [Afterwarp.DoubleClick](#) }

Type of mouse event corresponding to the notification.

- enum `Afterwarp.MouseButton` : byte { `Afterwarp.None` , `Afterwarp.Left` , `Afterwarp.Right` , `Afterwarp.Middle` }

Type of mouse button that corresponds to the notification.

- enum `Afterwarp.KeyEvent` : byte { `Afterwarp.Pressed` , `Afterwarp.Released` , `Afterwarp.Typing` }

Type of keyboard event corresponding to the notification.

- enum `Afterwarp.ApplicationEvent` : byte { `Afterwarp.Activate` , `Afterwarp.Deactivate` , `Afterwarp.Minimize` , `Afterwarp.Restore` , `Afterwarp.Appearance` }

Type of application event corresponding to the notification.

- enum `Afterwarp.FileChooserDialog` : byte { `Afterwarp.OpenFile` , `Afterwarp.SaveFile` , `Afterwarp.Directory` }

Type of file chooser dialog type.

- enum `Afterwarp.Key` : byte { `Afterwarp.Null` = 0x00 , `Afterwarp.Home` = 0x02 , `Afterwarp.End` = 0x03 , `Afterwarp.PrintScreen` = 0x05 , `Afterwarp.Backspace` = 0x08 , `Afterwarp.Tab` = 0x09 , `Afterwarp.Linefeed` = 0x0A , `Afterwarp.Clear` = 0x0B , `Afterwarp.Return` = 0x0D , `Afterwarp.PageUp` = 0x0E , `Afterwarp.PageDown` = 0x0F , `Afterwarp.CapsLock` = 0x11 , `Afterwarp.NumLock` = 0x12 , `Afterwarp.Pause` = 0x13 , `Afterwarp.ScrollLock` = 0x14 , `Afterwarp.SysReq` = 0x15 , `Afterwarp.Cancel` = 0x18 , `Afterwarp.Insert` = 0x1A , `Afterwarp.Escape` = 0x1B , `Afterwarp.Left` = 0x1C , `Afterwarp.Up` = 0x1D , `Afterwarp.Right` = 0x1E , `Afterwarp.Down` = 0x1F , `Afterwarp.Space` = 0x20 , `Afterwarp.KeyA` = 0x61 , `Afterwarp.KeyB` = 0x62 , `Afterwarp.KeyC` = 0x63 , `Afterwarp.KeyD` = 0x64 , `Afterwarp.KeyE` = 0x65 , `Afterwarp.KeyF` = 0x66 , `Afterwarp.KeyG` = 0x67 , `Afterwarp.KeyH` = 0x68 , `Afterwarp.KeyI` = 0x69 , `Afterwarp.KeyJ` = 0x6A , `Afterwarp.KeyK` = 0x6B , `Afterwarp.KeyL` = 0x6C , `Afterwarp.KeyM` = 0x6D , `Afterwarp.KeyN` = 0x6E , `Afterwarp.KeyO` = 0x6F , `Afterwarp.KeyP` = 0x70 , `Afterwarp.KeyQ` = 0x71 , `Afterwarp.KeyR` = 0x72 , `Afterwarp.KeyS` = 0x73 , `Afterwarp.KeyT` = 0x74 , `Afterwarp.KeyU` = 0x75 , `Afterwarp.KeyV` = 0x76 , `Afterwarp.KeyW` = 0x77 , `Afterwarp.KeyX` = 0x78 , `Afterwarp.KeyY` = 0x79 , `Afterwarp.KeyZ` = 0x7A , `Afterwarp.Delete` = 0x7F , `Afterwarp.F1` = 0x80 , `Afterwarp.F2` = 0x81 , `Afterwarp.F3` = 0x82 , `Afterwarp.F4` = 0x83 , `Afterwarp.F5` = 0x84 , `Afterwarp.F6` = 0x85 , `Afterwarp.F7` = 0x86 , `Afterwarp.F8` = 0x87 , `Afterwarp.F9` = 0x88 , `Afterwarp.F10` = 0x89 , `Afterwarp.F11` = 0x8A , `Afterwarp.F12` = 0x8B , `Afterwarp.ShiftLeft` = 0x98 , `Afterwarp.ShiftRight` = 0x99 , `Afterwarp.CtrlLeft` = 0x9A , `Afterwarp.CtrlRight` = 0x9B , `Afterwarp.AltLeft` = 0x9C , `Afterwarp.AltRight` = 0x9D , `Afterwarp.SuperLeft` = 0x9E , `Afterwarp.SuperRight` = 0x9F , `Afterwarp.Numpad0` = 0xA0 , `Afterwarp.Numpad1` = 0xA1 , `Afterwarp.Numpad2` = 0xA2 , `Afterwarp.Numpad3` = 0xA3 , `Afterwarp.Numpad4` = 0xA4 , `Afterwarp.Numpad5` = 0xA5 , `Afterwarp.Numpad6` = 0xA6 , `Afterwarp.Numpad7` = 0xA7 , `Afterwarp.Numpad8` = 0xA8 , `Afterwarp.Numpad9` = 0xA9 , `Afterwarp.Multiply` = 0xAA , `Afterwarp.Divide` = 0xAB , `Afterwarp.Add` = 0xAC , `Afterwarp.Subtract` = 0xAD , `Afterwarp.Separator` = 0xAE , `Afterwarp.Decimal` = 0xAF , `Afterwarp.Sleep` = 0xC0 , `Afterwarp.VolumeUp` = 0xCC , `Afterwarp.VolumeDown` = 0xCE , `Afterwarp.VolumeMute` = 0xCF , `Afterwarp.MediaNextTrack` = 0xD0 , `Afterwarp.MediaPrevTrack` = 0xD1 , `Afterwarp.MediaStop` = 0xD2 , `Afterwarp.MediaPlayPause` = 0xD3 , `Afterwarp.Select` = 0xE0 , `Afterwarp.Print` = 0xE1 , `Afterwarp.Execute` = 0xE2 , `Afterwarp.Help` = 0xE3 , `Afterwarp.Apps` = 0xE4 }

Storage type for portable key constants.

- enum `Afterwarp.AppCursor` : byte { `Afterwarp.Blank` = 0x00 , `Afterwarp.Arrow` = 0x01 , `Afterwarp.TextSelect` , `Afterwarp.TextSelectVertical` , `Afterwarp.Wait` , `Afterwarp.ArrowWait` , `Afterwarp.CrossHair` , `Afterwarp.Cell` , `Afterwarp.LinkSelect` , `Afterwarp.NotAllowed` , `Afterwarp.Help` , `Afterwarp.Move` , `Afterwarp.Drag` , `Afterwarp.Dragging` , `Afterwarp.ResizeVert` , `Afterwarp.ResizeHoriz` , `Afterwarp.Resize1` , `Afterwarp.Resize2` , `Afterwarp.ResizeTop` , `Afterwarp.ResizeBottom` , `Afterwarp.ResizeLeft` , `Afterwarp.ResizeRight` , `Afterwarp.ResizeTopLeft` , `Afterwarp.ResizeTopRight` , `Afterwarp.ResizeBottomRight` , `Afterwarp.ResizeBottomLeft` , `Afterwarp.Undefined` = 255 }

Application cursor enumeration.

- enum `Afterwarp.WidgetAlignment` : byte {
 `Afterwarp.None` , `Afterwarp.Client` , `Afterwarp.Left` , `Afterwarp.Top` ,
 `Afterwarp.Right` , `Afterwarp.Bottom` }
Alignment of the widget respective its parent.
- enum `Afterwarp.WidgetPropertyType` : byte {
 `Afterwarp.None` , `Afterwarp.Integer` , `Afterwarp.Int64` , `Afterwarp.Float` ,
 `Afterwarp.Real` , `Afterwarp.Boolean` , `Afterwarp.Point` , `Afterwarp.Vector` ,
 `Afterwarp.Rect` , `Afterwarp.Margins` , `Afterwarp.Color` , `Afterwarp.ColorPair` ,
 `Afterwarp.ColorRect` , `Afterwarp.Enumeration` , `Afterwarp.String` , `Afterwarp.Font` }
Type of property that defines its purpose.
- enum `Afterwarp.WidgetPropertyBehavior` : byte {
 `Afterwarp.Normal` , `Afterwarp.Static` , `Afterwarp.Indirect` , `Afterwarp.Volatile` ,
 `Afterwarp.Event` }
Behavior of the property.
- enum `Afterwarp.WidgetManagerTextureType` : byte { `Afterwarp.Canvas` , `Afterwarp.Screen` , `Afterwarp.Stripes`
 }
Texture type used for the compositioning.

Index

- [_handle](#)
 - [Afterwarp.CustomObject, 131](#)
- [A8, 41](#)
- [Afterwarp, 41](#)
- [AABB](#)
 - [Afterwarp.ObjectModel, 231](#)
- [AcceptKey](#)
 - [Afterwarp.Widget, 455](#)
- [AcceptMouse](#)
 - [Afterwarp.Widget, 455](#)
- [Access](#)
 - [Afterwarp.ComputeBindTextureFormat, 123](#)
- [AccessType](#)
 - [Afterwarp.Buffer, 87](#)
- [Accomodate](#)
 - [Afterwarp.Widget, 455](#)
- [Activate](#)
 - [Afterwarp, 26](#)
- [ActiveVertexElements](#)
 - [Afterwarp.Scene, 320](#)
- [ActorCamera](#)
 - [Afterwarp.ActorCamera, 53](#)
- [Add](#)
 - [Afterwarp, 28, 37](#)
 - [Afterwarp.MeshMetaTags, 211](#)
 - [Afterwarp.ObjectMaterials, 228](#)
 - [Afterwarp.ObjectModels, 239](#)
 - [Afterwarp.SceneLights, 328](#)
 - [Afterwarp.SceneMeshes, 338](#)
 - [Afterwarp.SceneMeshLatches, 345](#)
 - [Afterwarp.SceneMeshMaterials, 355](#)
 - [Afterwarp.ShadowCastingAtlas, 372](#)
- [AddColors](#)
 - [Afterwarp.Utility, 436](#)
- [AddFromBuffer](#)
 - [Afterwarp.SceneMeshes, 338](#)
- [AddFromFile](#)
 - [Afterwarp.SceneMeshes, 338](#)
- [AddRange](#)
 - [Afterwarp.SceneMeshMaterial, 350](#)
- [AddressU](#)
 - [Afterwarp.CanvasSamplerState, 113](#)
 - [Afterwarp.SamplerState, 313](#)
- [AddressV](#)
 - [Afterwarp.CanvasSamplerState, 113](#)
 - [Afterwarp.SamplerState, 313](#)
- [AddressW](#)
 - [Afterwarp.SamplerState, 313](#)
- [Afterwarp, 17](#)
- [A8, 41](#)
- [Activate, 26](#)
- [Add, 28, 37](#)
- [Albedo, 44](#)
- [AlphaFormatRequest, 25](#)
- [AltLeft, 36](#)
- [AltRight, 36](#)
- [Always, 30](#)
- [Android, 31](#)
- [Anisotropic, 48](#)
- [AppCursor, 26](#)
- [Appearance, 26](#)
- [ApplicationEvent, 26](#)
- [ApplicationWindowState, 27](#)
- [Apps, 37](#)
- [Arrow, 26](#)
- [ArrowWait, 26](#)
- [B5G6R5, 41](#)
- [Back, 49](#)
- [Backspace, 35](#)
- [Bevel, 39](#)
- [BGR10A2, 41](#)
- [BGR10X2, 41](#)
- [BGR5A1, 41](#)
- [BGR5X1, 41](#)
- [BGR8, 41](#)
- [BGRA4, 41](#)
- [BGRA8, 41](#)
- [BGRA8_SRGB, 41](#)
- [BGRX4, 41](#)
- [BGRX8, 41](#)
- [Blank, 26](#)
- [BlendFactor, 27](#)
- [BlendingEffect, 27](#)
- [BlendOp, 28](#)
- [Bloom, 47](#)
- [Bold, 35](#)
- [Boolean, 50](#)
- [Border, 47](#)
- [Both, 49](#)
- [Bottom, 49](#)
- [BufferAccessType, 28](#)
- [BufferDataType, 28](#)
- [Butt, 38](#)
- [Byte, 32](#)
- [CameraCommand, 29](#)
- [Cancel, 35](#)
- [Canvas, 49](#)
- [CanvasContextState, 29](#)

- CapsLock, 35
- Cell, 26
- Clamp, 47
- Clear, 35
- Client, 49
- ClustersCullingMode, 29
- Color, 47, 50
- ColorDitheringFormat, 30
- ColorHDR, 47
- ColorPair, 50
- ColorRect, 50
- ComparisonFunc, 30
- Compute, 28, 45
- ComputeTextureAccess, 30
- Condensed, 33
- Constant, 29
- ConstantAlpha, 27
- ConstantBuffer, 44
- ConstantColor, 27
- CookTorrance, 46
- Copy, 28
- Cross, 43
- CrossHair, 26
- CtrlLeft, 36
- CtrlRight, 36
- CubeMap, 48
- Custom, 28
- D16, 41
- D24S8, 41
- D32F, 41
- D32S8F, 41
- Deactivate, 26
- Decimal, 37
- Decrement, 45
- DecrementWarp, 45
- Default, 28
- Delete, 36
- Depth, 39, 47
- DepthFogDistance, 31
- DepthReverse, 39
- DestAlpha, 27
- DestColor, 27
- DevicePlatform, 31
- DeviceTechnology, 31
- Diffuse, 43
- Direct3D, 31
- Directory, 32
- Divide, 37
- Domain, 45
- DontCare, 25
- Double, 32
- DoubleClick, 39
- Down, 35, 39
- Drag, 26
- Dragging, 26, 29
- Dynamic, 28
- ElementFormat, 31
- End, 35, 46
- Enter, 39
- Enumeration, 50
- Equal, 30
- Escape, 35
- ESM, 46
- ESM_Warp, 46
- Event, 49
- EVSM, 46
- Execute, 37
- Expanded, 33
- Exponential, 32
- ExtraBold, 35
- ExtraCondensed, 33
- ExtraExpanded, 33
- ExtraHeavy, 35
- ExtraLight, 35
- Extreme, 48
- EyeRelative, 31
- F1, 36
- F10, 36
- F11, 36
- F12, 36
- F2, 36
- F3, 36
- F4, 36
- F5, 36
- F6, 36
- F7, 36
- F8, 36
- F9, 36
- FileChooserDialog, 32
- FlatScene, 29
- FlatText, 29
- Float, 32, 50
- Float_11_11_10, 32
- FogFormula, 32
- Font, 50
- FontBorder, 32
- FontSlant, 33
- FontStretch, 33
- FontWeight, 33
- Fragment, 45
- Front, 49
- FullScreen, 27
- Geometry, 45
- Glassy, 47
- GlassyColor, 47
- GlassyComposite, 47
- GlassyCounts, 47
- GlassyFog, 47
- Global, 38
- GlobalModel, 38
- GlobalVolume, 38
- Greater, 30
- GreaterEqual, 30
- Ground, 32
- HalfFloat, 32
- Heavy, 33, 35

Help, [26](#), [37](#)
High, [48](#)
Highlight, [44](#)
HighlightMask, [44](#)
Home, [35](#)
Hull, [45](#)
I8, [41](#)
Increment, [45](#)
IncrementWarp, [45](#)
Index, [29](#)
Indirect, [49](#)
Insert, [35](#)
Int, [32](#)
Int64, [50](#)
Integer, [50](#)
InvConstantAlpha, [27](#)
InvConstantColor, [27](#)
InvDestAlpha, [27](#)
InvDestColor, [27](#)
InverseMultiply, [28](#)
Invert, [45](#)
InvSourceAlpha, [27](#)
InvSourceAlpha1, [27](#)
InvSourceColor, [27](#)
InvSourceColor1, [27](#)
InvSubtract, [28](#)
iOS, [31](#)
Italic, [33](#)
Keep, [45](#)
Key, [35](#)
KeyA, [35](#)
KeyB, [35](#)
KeyC, [35](#)
KeyD, [36](#)
KeyE, [36](#)
KeyEvent, [37](#)
KeyF, [36](#)
KeyG, [36](#)
KeyH, [36](#)
KeyI, [36](#)
KeyJ, [36](#)
KeyK, [36](#)
KeyL, [36](#)
KeyM, [36](#)
KeyN, [36](#)
KeyO, [36](#)
KeyP, [36](#)
KeyQ, [36](#)
KeyR, [36](#)
KeyS, [36](#)
KeyT, [36](#)
KeyU, [36](#)
KeyV, [36](#)
KeyW, [36](#)
KeyX, [36](#)
KeyY, [36](#)
KeyZ, [36](#)
L16, [41](#)
L8, [41](#)
LA4, [41](#)
LA8, [41](#)
Leave, [39](#)
Left, [35](#), [38](#), [49](#)
Less, [30](#)
LessEqual, [30](#)
Light, [35](#)
Linear, [32](#), [48](#)
LinearDepths, [47](#)
LineCaps, [37](#)
Linefeed, [35](#)
Lines, [43](#)
LinesAdjacency, [43](#)
LineStrip, [43](#)
LineStripAdjacency, [43](#)
LinkSelect, [26](#)
Linux, [31](#)
Local, [38](#)
LocalModel, [38](#)
LocalVolume, [38](#)
Low, [48](#)
Margins, [50](#)
Maximized, [27](#)
Maximum, [28](#)
MediaNextTrack, [37](#)
MediaPlayPause, [37](#)
MediaPrevTrack, [37](#)
MediaStop, [37](#)
Medium, [35](#), [48](#)
MeshAlign, [38](#)
Meshes, [39](#)
Metal, [31](#)
Middle, [38](#), [46](#)
Minimize, [26](#)
Minimized, [27](#)
Minimum, [28](#)
Minnaert, [46](#)
Mirror, [47](#)
MirrorOnce, [47](#)
Miter, [39](#)
MiterBevel, [39](#)
ModelTransform, [38](#)
MouseButton, [38](#)
MouseEvent, [39](#)
Move, [26](#), [39](#)
Movement, [29](#)
Multiply, [28](#), [37](#)
Nearest, [48](#)
Negative, [38](#)
Never, [30](#)
None, [33](#), [38](#), [39](#), [46](#), [48–50](#)
NonPremultiplied, [25](#)
Normal, [28](#), [33](#), [35](#), [49](#)
NormalMap, [44](#)
Normals, [43](#), [47](#)
NoSuperSample, [45](#)
NotAllowed, [26](#)

- NotEqual, [30](#)
- Null, [35](#)
- NumLock, [35](#)
- Numpad0, [36](#)
- Numpad1, [36](#)
- Numpad2, [36](#)
- Numpad3, [36](#)
- Numpad4, [37](#)
- Numpad5, [37](#)
- Numpad6, [37](#)
- Numpad7, [37](#)
- Numpad8, [37](#)
- Numpad9, [37](#)
- ObjectModelViewCompare, [39](#)
- Objects, [39](#)
- Oblique, [33](#)
- Occlusion, [47](#)
- One, [27](#)
- OpenFile, [32](#)
- OpenGL, [31](#)
- OpenGL_ES, [31](#)
- Ordered, [39](#)
- OrenNayer, [46](#)
- Origin, [38](#)
- OSX, [31](#)
- PageDown, [35](#)
- PageUp, [35](#)
- Parallax, [43](#)
- ParallaxMap, [44](#)
- Patches, [43](#)
- PathJoint, [39](#)
- Pause, [35](#)
- Performance, [30](#)
- Phong, [46](#)
- PixelFormat, [39](#)
- Point, [50](#)
- Points, [43](#)
- PointShape, [41](#)
- Positive, [38](#)
- Premultiplied, [25](#)
- Pressed, [37](#)
- PrimitiveTopology, [43](#)
- Print, [37](#)
- PrintScreen, [35](#)
- Projected, [29](#)
- Proprietary, [31](#)
- Quality, [30](#)
- R16, [40](#)
- R16F, [40](#)
- R16I, [40](#)
- R16S, [40](#)
- R16U, [40](#)
- R32F, [40](#)
- R32I, [40](#)
- R32U, [40](#)
- R5G6B5, [30](#)
- R8, [40](#)
- R8I, [40](#)
- R8S, [40](#)
- R8U, [40](#)
- R_BC, [41](#)
- Read, [30](#)
- ReadWrite, [30](#)
- Real, [50](#)
- Rect, [50](#)
- Released, [37](#)
- Replace, [45](#)
- Resize1, [26](#)
- Resize2, [26](#)
- ResizeBottom, [26](#)
- ResizeBottomLeft, [26](#)
- ResizeBottomRight, [26](#)
- ResizeHoriz, [26](#)
- ResizeLeft, [26](#)
- ResizeRight, [26](#)
- ResizeTop, [26](#)
- ResizeTopLeft, [26](#)
- ResizeTopRight, [26](#)
- ResizeVert, [26](#)
- Restore, [26](#)
- Return, [35](#)
- RG11B10F, [40](#)
- RG16, [40](#)
- RG16F, [40](#)
- RG16I, [40](#)
- RG16S, [40](#)
- RG16U, [40](#)
- RG32F, [40](#)
- RG32I, [40](#)
- RG32U, [40](#)
- RG3B2, [30](#)
- RG8, [40](#)
- RG8I, [40](#)
- RG8S, [40](#)
- RG8U, [40](#)
- RG_BC, [41](#)
- RGB10A2, [41](#)
- RGB10A2U, [41](#)
- RGB4, [30](#)
- RGB6, [30](#)
- RGB8, [30, 41](#)
- RGB9E5F, [40](#)
- RGB_BC, [41](#)
- RGB_BC_SRGB, [41](#)
- RGBA16, [40](#)
- RGBA16F, [40](#)
- RGBA16I, [40](#)
- RGBA16S, [40](#)
- RGBA16U, [40](#)
- RGBA32F, [40](#)
- RGBA32I, [40](#)
- RGBA32U, [40](#)
- RGBA8, [40](#)
- RGBA8_SRGB, [41](#)
- RGBA8I, [40](#)
- RGBA8S, [40](#)

RGBA8U, 40
RGBA_BC, 41
RGBA_BC_SRGB, 41
RGBX8, 41
Right, 35, 38, 49
Rotation, 29
Round, 38, 39, 43
RWStructured, 29
RWStructuredBuffer, 44
RWTyped, 29
RWTypedBuffer, 44
SaveFile, 32
SceneMeshTexture, 43
SceneSamplerType, 43
SceneTextureType, 44
Screen, 49
ScrollLock, 35
Select, 37
SelectionHighlightTextureType, 44
SemiBold, 35
SemiCondensed, 33
SemiExpanded, 33
SemiHeavy, 33
SemiLight, 35
Separator, 37
ShaderElement, 44
ShaderType, 45
Shadow, 28
ShadowMap, 44
ShiftLeft, 36
ShiftRight, 36
Short, 32
Simple, 39
Sleep, 37
Software, 31
SourceAlpha, 27
SourceAlpha1, 27
SourceAlphaSat, 27
SourceColor, 27, 28
SourceColor1, 27
SourceColorAdd, 28
Space, 35
Square, 38, 43
Staging, 28
Star, 43
Start, 46
Static, 28, 49
StencilOp, 45
Stop, 29
String, 50
Stripes, 49
Structured, 29
StructuredBuffer, 44
Subtract, 28, 37
SuperLeft, 36
SuperRight, 36
SuperSample16x, 45
SuperSample4x, 45
SuperSampleSDF, 45
Surface, 48
SysReq, 35
Tab, 35
TechniqueLighting, 45
TechniqueShadows, 46
TextAlignment, 46
TextSelect, 26
TextSelectVertical, 26
Texture, 44
TextureAddress, 46
TextureCabinetFilterType, 47
TextureCabinetPass, 47
TextureCabinetType, 47
TextureFidelity, 48
TextureFilter, 48
TextureType, 48
Thin, 35
Top, 49
Triangle, 43
TriangleFace, 48
Triangles, 43
TrianglesAdjacency, 43
TriangleStrip, 43
TriangleStripAdjacency, 43
Typed, 29
TypedBuffer, 44
Typing, 37
UltraCondensed, 33
UltraExpanded, 33
Unaligned, 38
Undefined, 26, 28, 29, 32, 44, 45
Unix, 31
Unknown, 31, 40, 43
UnsignedByte, 32
UnsignedInt, 32
UnsignedShort, 32
Up, 35, 39
Update, 29
Vector, 50
Vertex, 29, 45
Volatile, 49
Volume, 48
VolumeDown, 37
VolumeMute, 37
VolumeUp, 37
Vulkan, 31
Wait, 26
WebGL, 31
WheelDown, 39
WheelUp, 39
WidgetAlignment, 49
WidgetManagerTextureType, 49
WidgetPropertyBehavior, 49
WidgetPropertyType, 50
Windowed, 27
Windows, 31
Wrap, 46

- Write, 30
- ZBased, 31
- Zero, 27, 45
- Afterwarp Framework, 1
- Afterwarp.ActorCamera, 51
 - ActorCamera, 53
 - Ceiling, 55
 - Command, 53
 - Constraints, 55
 - Distance, 55
 - Forward, 55
 - GetPosition, 53
 - GetQuaternion, 54
 - GetRotation, 54
 - Right, 55
 - SetPosition, 54
 - SetQuaternion, 54
 - SetRotation, 54
 - View, 55
 - Zoom, 54
 - ZoomOrtho, 54
- Afterwarp.AmbientOcclusionParameters, 56
 - AmbientOcclusionParameters, 57
 - Attenuation, 57
 - BlurFallOff, 57
 - BlurSigma, 57
 - Contrast, 57
 - Default, 58
 - DepthBias, 57
 - KernelBias, 58
 - Radius, 58
 - Samples, 58
 - Strength, 58
- Afterwarp.API, 59
- Afterwarp.API.cs, 471
- Afterwarp.API.Exception, 139
 - Exception, 140
- Afterwarp.Application, 59
 - Application, 62
 - BoolCallback, 62
 - CaptureMouseInput, 62
 - ClientRect, 65
 - ClientSize, 65
 - ConvertPortableKey, 62
 - Cursor, 65
 - EventCallback, 62
 - EventHookCallback, 63
 - ExecutablePath, 65
 - Execute, 63
 - FileChooserDialog, 63
 - IconTitle, 65
 - Invalidate, 63
 - KeyboardCallback, 63
 - MinimalSize, 66
 - MouseCallback, 63
 - MouseInputCaptured, 66
 - OnCreate, 67
 - OnDestroy, 67
 - OnHook, 67
 - OnIdle, 67
 - OnKeyboard, 67
 - OnMouse, 67
 - OnRender, 67
 - OnResize, 67
 - OnStatus, 68
 - ReadTextFromClipboard, 64
 - ReleaseMouseInput, 64
 - SetIcons, 64
 - StatusCallback, 64
 - Terminate, 64
 - Title, 66
 - TranslateVirtualKey, 65
 - WindowHandle, 66
 - WindowRect, 66
 - WindowScale, 66
 - WindowState, 66
 - WriteTextToClipboard, 65
- Afterwarp.ApplicationConfiguration, 68
 - ApplicationConfiguration, 69
 - IconBig, 69
 - IconSmall, 69
 - Instance, 70
 - Position, 70
 - Size, 70
 - State, 70
 - WindowClassName, 70
 - WindowClassNameBytes, 70
- Afterwarp.AutoDrawOption, 80
 - MaterialIgnore, 81
 - MaterialSkipOpaque, 81
 - MaterialSkipTransparent, 81
 - MaterialTransparency, 81
 - ObjectDisableInstancing, 81
 - ObjectHighlighted, 81
 - ObjectSkipHavingMaterials, 81
 - ObjectSkipNonTransparent, 82
 - ObjectSkipNotHavingMaterials, 82
 - ObjectSkipTransparent, 82
 - PostUnbind, 82
 - ResetTextures, 82
- Afterwarp.Buffer, 84
 - AccessType, 87
 - Buffer, 86
 - Copy, 86
 - DataType, 87
 - Device, 87
 - Format, 87
 - Pitch, 87
 - PlatformHandle, 87
 - Retrieve, 86
 - Size, 87
 - Update, 86
- Afterwarp.CameraConstraints, 88
 - CameraConstraints, 88
 - PositionMax, 88
 - PositionMin, 88

- RotationMax, [88](#)
- RotationMin, [89](#)
- Afterwarp.Canvas, [89](#)
 - Arc, [94](#)
 - Attributes, [107](#)
 - BatchCount, [107](#)
 - Begin, [95](#)
 - Canvas, [94](#)
 - Circle, [95](#)
 - ClipRect, [107](#)
 - ContextState, [108](#)
 - Device, [108](#)
 - Draw, [95](#)
 - Ellipse, [95](#)
 - End, [95](#)
 - FillRect, [96](#)
 - FillRoundRect, [96](#)
 - FillRoundRectBottom, [96](#)
 - FillRoundRectTop, [96](#)
 - FillRoundRectTopInverse, [97](#)
 - Flush, [97](#)
 - FrameRect, [97](#)
 - FrameRoundRect, [97](#)
 - Hexagon, [98](#)
 - HexagonDelta, [107](#)
 - Highlight, [98](#)
 - Line, [98](#)
 - LineCircle, [99](#)
 - LineEllipse, [99](#)
 - LineHexagon, [99](#)
 - LineQuad, [99](#)
 - Lines, [100](#)
 - LineTriangle, [100](#)
 - Pixel, [100](#)
 - Pixels, [100](#)
 - Quad, [100](#), [101](#)
 - QuadImage, [101](#)
 - QuadRegion, [101](#)
 - RectWithHole, [101](#)
 - Reset, [101](#)
 - ResetSamplerState, [102](#)
 - Ribbon, [102](#)
 - RoundRect, [102](#)
 - RoundRectRegion, [103](#)
 - SamplerState, [108](#)
 - SignedDistanceField, [108](#)
 - Tape, [103](#)
 - TexturedTriangles, [104](#)
 - ThickLine, [104](#)
 - ThickLineCircle, [104](#)
 - ThickLineEllipse, [104](#)
 - ThickLineHexagon, [105](#)
 - ThickLineQuad, [105](#)
 - ThickLineTriangle, [105](#)
 - Transform, [108](#)
 - Triangle, [106](#)
 - TriangleRegion, [106](#)
 - Triangles, [107](#)
 - ViewProjection, [108](#)
 - World, [108](#)
- Afterwarp.Canvas.Attribute, [71](#)
 - ColorAdjust, [71](#)
 - Cubic, [71](#)
 - Outline, [71](#)
 - Projected, [71](#)
 - SDF, [72](#)
 - Transform, [72](#)
- Afterwarp.CanvasBuffer, [109](#)
 - CanvasBuffer, [111](#)
 - Clear, [111](#)
 - ClearAndShrink, [111](#)
 - Colors, [111](#)
 - Extents, [111](#)
 - IndexCount, [111](#)
 - Indices, [111](#)
 - VertexCount, [111](#)
 - Vertices, [112](#)
- Afterwarp.CanvasSamplerState, [112](#)
 - AddressU, [113](#)
 - AddressV, [113](#)
 - BorderColor, [113](#)
 - CanvasSamplerState, [113](#)
 - Default, [114](#)
 - FilterMag, [113](#)
 - FilterMin, [113](#)
 - FilterMip, [114](#)
- Afterwarp.ColorDithering, [114](#)
 - ColorDithering, [116](#)
 - Device, [116](#)
 - Execute, [116](#)
 - Format, [116](#)
- Afterwarp.ColorPair, [117](#)
 - Black, [118](#)
 - ColorPair, [118](#)
 - First, [118](#)
 - Second, [118](#)
 - TranslucentBlack, [118](#)
 - TranslucentWhite, [118](#)
 - White, [119](#)
- Afterwarp.ColorRect, [119](#)
 - Black, [122](#)
 - BottomLeft, [121](#)
 - BottomRight, [121](#)
 - ColorRect, [120](#)
 - GradientX, [120](#)
 - GradientY, [121](#)
 - TopLeft, [121](#)
 - TopRight, [121](#)
 - TranslucentBlack, [122](#)
 - TranslucentWhite, [122](#)
 - White, [122](#)
- Afterwarp.ComputeBindTextureFormat, [122](#)
 - Access, [123](#)
 - Channel, [123](#)
 - ComputeBindTextureFormat, [123](#)
 - Format, [123](#)

- Layer, [124](#)
- MipLevel, [124](#)
- Afterwarp.ComputeProgram, [124](#)
 - Begin, [126](#)
 - Bind, [126](#)
 - Commit, [127](#)
 - ComputeProgram, [126](#)
 - Device, [128](#)
 - Dispatch, [127](#)
 - End, [127](#)
 - ResetBindings, [127](#)
 - Unbind, [127](#)
- Afterwarp.CustomObject, [128](#)
 - _handle, [131](#)
 - Dispose, [130](#)
 - Equals, [131](#)
 - GetHashCode, [131](#)
 - Handle, [131](#)
- Afterwarp.Device, [132](#)
 - Attributes, [134](#)
 - Behavior, [134](#)
 - Clear, [134](#)
 - Device, [134](#)
 - LegacyBits, [134](#)
 - MemoryBarrier, [134](#)
 - Platform, [135](#)
 - RenderingState, [135](#)
 - ResetCache, [134](#)
 - Scissor, [135](#)
 - TechFeatureVersion, [135](#)
 - Technology, [135](#)
 - TechVersion, [135](#)
 - Viewport, [135](#)
- Afterwarp.Device.Attribute, [72](#)
 - Debug, [72](#)
 - Legacy, [72](#)
 - LimitedExtensions, [73](#)
 - Software, [73](#)
- Afterwarp.DeviceBehavior, [136](#)
 - Compute, [136](#)
 - DepthClearBadPrecision, [136](#)
 - DepthClipNegative, [137](#)
 - ForceBufferUnbind, [137](#)
 - PerSampleShading, [137](#)
 - PostDepthCoverage, [137](#)
 - SignedNormIntFormatBugged, [137](#)
 - Tessellation, [137](#)
 - VariableRateRefresh, [137](#)
- Afterwarp.DeviceClear, [138](#)
 - Color, [138](#)
 - Depth, [138](#)
 - Stencil, [138](#)
- Afterwarp.FloatColor, [140](#)
 - Alpha, [143](#)
 - Black, [143](#)
 - Blue, [143](#)
 - FloatColor, [141](#)
 - Green, [143](#)
 - operator+, [142](#)
 - operator-, [142](#)
 - operator/, [142](#)
 - operator*, [142](#)
 - Red, [143](#)
 - RGB, [143](#)
 - ToColor, [143](#)
 - TranslucentBlack, [144](#)
 - TranslucentWhite, [144](#)
 - White, [144](#)
- Afterwarp.FloatColorRGB, [144](#)
 - Black, [147](#)
 - Blue, [147](#)
 - Defined, [147](#)
 - FloatColorRGB, [145](#)
 - Green, [147](#)
 - operator+, [146](#)
 - operator-, [146](#)
 - operator/, [146](#)
 - operator*, [146](#)
 - Red, [147](#)
 - ToColor, [147](#)
 - ToColorAlpha, [147](#)
 - White, [147](#)
- Afterwarp.FogParameters, [148](#)
 - Bias, [149](#)
 - Color, [149](#)
 - Default, [150](#)
 - DensityGround, [149](#)
 - Distance, [149](#)
 - Extinction, [150](#)
 - FogParameters, [149](#)
 - GroundPlane, [150](#)
 - Opacity, [150](#)
 - Scattering, [150](#)
- Afterwarp.FontAttribute, [150](#)
 - StrikeOut, [151](#)
 - Underline, [151](#)
- Afterwarp.FontEffect, [151](#)
 - BorderBrightness, [153](#)
 - BorderOpacity, [153](#)
 - BorderThickness, [153](#)
 - BorderType, [153](#)
 - Default, [154](#)
 - FillBrightness, [153](#)
 - FillOpacity, [153](#)
 - FontEffect, [152](#)
 - ShadowBrightness, [153](#)
 - ShadowDistance, [153](#)
 - ShadowOpacity, [154](#)
 - ShadowSmoothness, [154](#)
 - SignedFieldDistance, [154](#)
- Afterwarp.FontParameters, [155](#)
 - Attributes, [156](#)
 - Effect, [156](#)
 - Family, [157](#)
 - FamilyBytes, [156](#)
 - FontParameters, [156](#)

- Size, [156](#)
- Slant, [156](#)
- Stretch, [157](#)
- Weight, [157](#)
- Afterwarp.GaussianBlur, [159](#)
 - Chroma, [161](#)
 - Device, [161](#)
 - FixedSamples, [161](#)
 - GaussianBlur, [160](#)
 - HardwareFiltering, [161](#)
 - Samples, [162](#)
 - Sigma, [162](#)
 - Update, [161](#)
 - UpdateAt, [161](#)
- Afterwarp.Grapher, [162](#)
 - Arrow, [164](#)
 - BatchCount, [167](#)
 - Begin, [164](#)
 - BoundingBox, [165](#)
 - Device, [167](#)
 - DottedLine, [165](#)
 - End, [165](#)
 - Flush, [165](#)
 - Grapher, [164](#)
 - Line, [165](#)
 - Lines, [166](#)
 - Point, [166](#)
 - Points, [166](#)
 - Reset, [166](#)
 - Transform, [167](#)
- Afterwarp.Graphics.cs, [471](#)
- Afterwarp.ImageAtlas, [168](#)
 - ClearRegions, [170](#)
 - ClearTextures, [170](#)
 - CreateRegion, [170](#)
 - CreateTexture, [170](#)
 - Device, [172](#)
 - ImageAtlas, [170](#)
 - MakeRegions, [170](#)
 - PackRegion, [170](#)
 - PackSurface, [171](#)
 - Region, [171](#)
 - RegionCount, [172](#)
 - RemoveRegion, [171](#)
 - RemoveTexture, [171](#)
 - Texture, [171](#)
 - TextureCount, [172](#)
- Afterwarp.ImageRegion, [172](#)
 - Flip, [173](#)
 - Height, [173](#)
 - ImageRegion, [173](#)
 - Index, [173](#)
 - Left, [173](#)
 - Mirror, [174](#)
 - Rotate, [174](#)
 - Top, [174](#)
 - Width, [174](#)
- Afterwarp.KawaseBlur, [178](#)
 - Device, [180](#)
 - KawaseBlur, [179](#)
 - Offset, [180](#)
 - Passes, [180](#)
 - SinglePass, [179](#)
 - Update, [179](#)
- Afterwarp.Library, [180](#)
 - GetVersion, [181](#)
 - SerialCode, [181](#)
- Afterwarp.Margins, [181](#)
 - Bottom, [183](#)
 - BottomLeft, [184](#)
 - BottomRight, [184](#)
 - Empty, [184](#)
 - Horizontal, [184](#)
 - Left, [183](#)
 - Margins, [182](#), [183](#)
 - Right, [183](#)
 - Top, [183](#)
 - TopLeft, [184](#)
 - TopRight, [184](#)
 - Vertical, [184](#)
 - Zero, [184](#)
- Afterwarp.MeshAligns, [185](#)
 - Bias, [186](#)
 - Default, [186](#)
 - MeshAligns, [186](#)
 - X, [186](#)
 - Y, [186](#)
 - Z, [187](#)
- Afterwarp.MeshBuffer, [187](#)
 - CalculateBounds, [191](#)
 - CalculateFlatNormals, [191](#)
 - CalculateNormals, [191](#)
 - CalculateNormalsWeld, [191](#)
 - Centralize, [191](#)
 - Clear, [192](#)
 - Combine, [192](#)
 - Cone, [192](#)
 - Cube, [192](#)
 - CubeMinimal, [192](#)
 - CubeRound, [193](#)
 - Cylinder, [193](#)
 - Disc, [193](#)
 - EliminateUnusedVertices, [194](#)
 - Extrusion, [194](#)
 - FrustumVolume, [194](#)
 - Geosphere, [194](#)
 - IndexCount, [201](#)
 - Indices, [201](#)
 - IndicesPtr, [201](#)
 - InvertIndexOrder, [194](#)
 - InvertNormals, [195](#)
 - JoinDuplicateVertices, [195](#)
 - LoadFromFile, [195](#)
 - LoadFromFileEx, [195](#)
 - LoadSaveFeedback, [195](#)
 - MeshBuffer, [191](#)

- Model, [196](#)
- Plane, [196](#)
- SaveToFile, [196](#)
- SaveToFileEx, [196](#)
- SuperEllipse, [197](#)
- Supertoroid, [197](#)
- Torus, [197](#)
- TorusKnot, [198](#)
- Transfer, [198](#)
- TransferEx, [199](#)
- Transform, [201](#)
- TransformVertices, [199](#)
- TryLoadFromFile, [199](#)
- TryLoadFromFileEx, [200](#)
- TrySaveToFile, [200](#)
- TrySaveToFileEx, [200](#)
- VertexCount, [201](#)
- Vertices, [201](#)
- VerticesPtr, [201](#)
- Voxelize, [200](#)
- Afterwarp.MeshBuffer.VertexEntry, [449](#)
 - Color, [450](#)
 - Normal, [450](#)
 - Position, [450](#)
 - Tangent, [451](#)
 - TexCoord, [451](#)
 - VertexEntry, [450](#)
- Afterwarp.MeshLoadingOption, [202](#)
 - ExportColors, [202](#)
 - ExportNormals, [202](#)
 - ExportTangents, [203](#)
 - ExportTexCoords, [203](#)
 - ExportWeights, [203](#)
 - MaterialTextureMissing, [203](#)
 - MaterialVertexColor, [203](#)
 - MaterialVertexColorPreferred, [203](#)
 - StripGeometryNames, [203](#)
- Afterwarp.MeshMetaTag, [204](#)
 - GetBounds, [206](#)
 - Name, [207](#)
 - Parent, [207](#)
 - PortionAdd, [206](#)
 - PortionCount, [207](#)
 - PortionErase, [206](#)
 - PortionGet, [206](#)
 - PortionsClear, [206](#)
 - PortionsCopy, [206](#)
 - Type, [207](#)
- Afterwarp.MeshMetaTagPortion, [207](#)
 - BoundsMax, [208](#)
 - BoundsMin, [208](#)
 - FirstIndex, [208](#)
 - FirstVertex, [209](#)
 - IndexCount, [209](#)
 - MeshMetaTagPortion, [208](#)
 - VertexCount, [209](#)
- Afterwarp.MeshMetaTags, [209](#)
 - Add, [211](#)
 - Clear, [211](#)
 - Copy, [211](#)
 - Count, [212](#)
 - Erase, [212](#)
 - MeshMetaTags, [211](#)
 - Tag, [212](#)
 - TakeAway, [212](#)
- Afterwarp.MeshMetaTagType, [213](#)
 - Geometry, [213](#)
 - Indeterminate, [213](#)
 - Object, [213](#)
- Afterwarp.MeshModel, [214](#)
 - Channel, [217](#)
 - Dismantle, [216](#)
 - Draw, [216](#)
 - DrawInstances, [216](#)
 - IndexBuffer, [217](#)
 - IndexCount, [217](#)
 - MeshModel, [216](#)
 - Renderable, [217](#)
 - TakeAway, [216](#)
 - VertexBuffer, [217](#)
 - VertexCount, [217](#)
- Afterwarp.MeshVoxel, [218](#)
 - ComputeParameters, [220](#)
 - Extents, [220](#)
 - Intersect, [220](#)
 - LoadFromFile, [220](#)
 - LoadFromFileInMemory, [220](#)
 - MeshVoxel, [219](#)
 - SaveToFile, [220](#)
 - TakeAway, [221](#)
 - TryLoadFromFile, [221](#)
 - TryLoadFromFileInMemory, [221](#)
 - Visualize, [221](#)
 - VisualizeFunc, [221](#)
- Afterwarp.ObjectMaterial, [222](#)
 - AlbedoColor, [224](#)
 - AmbientColor, [224](#)
 - Bitmask, [224](#)
 - Default, [225](#)
 - EmissiveColor, [224](#)
 - FrostedGlass, [224](#)
 - ObjectMaterial, [223](#)
 - Occlusion, [224](#)
 - Payload, [224](#)
 - Roughness, [224](#)
 - Shadows, [225](#)
 - SpecularColor, [225](#)
 - SpecularExponent, [225](#)
 - Technique, [225](#)
- Afterwarp.ObjectMaterials, [226](#)
 - Add, [228](#)
 - Clear, [228](#)
 - Count, [228](#)
 - Erase, [228](#)
 - GetMaterial, [228](#)
 - ObjectMaterials, [227](#)

- SetMaterial, 228
- Afterwarp.ObjectModel, 229
 - AABB, 231
 - Alignments, 234
 - Attributes, 234
 - Child, 231
 - ChildCount, 234
 - Color, 234
 - ConnectLatches, 232
 - DepthBias, 234
 - Description, 234
 - GetConnectedLatches, 232
 - GetLatchTransform, 232
 - GetLatchWaypointCouple, 232
 - GetLatchWaypointCoupleMatrix, 233
 - GetTransform, 233
 - GetWaypointDistance, 233
 - Highlight, 234
 - ID, 234
 - Invalidate, 233
 - Layers, 235
 - Material, 235
 - Mesh, 235
 - MeshName, 235
 - Name, 235
 - OrderIndex, 235
 - Owner, 235
 - Parent, 236
 - Payload, 236
 - Position, 236
 - SetTransform, 233
 - Size, 236
 - Voxel, 236
- Afterwarp.ObjectModel.Attribute, 73
 - DisableRealign, 74
 - Hierarchyless, 74
 - NonViewable, 74
 - Selectable, 74
 - Transparent, 74
 - Visible, 74
- Afterwarp.ObjectModels, 237
 - Add, 239
 - Clear, 239
 - ClearViews, 240
 - Count, 242
 - CreateView, 240
 - Erase, 240
 - EraseView, 240
 - Exists, 240
 - GetEnumerator, 240
 - Meshes, 242
 - Object, 241
 - ObjectModels, 239
 - Payload, 241
 - PayloadExists, 241
 - TryObject, 241
 - TryPayload, 241
- Afterwarp.ObjectModels.Iterator, 175
 - Current, 176
 - MoveNext, 176
 - Reset, 176
- Afterwarp.ObjectModelView, 242
 - AutoDraw, 244
 - CompareFunc, 244
 - IntersectedObjects, 246
 - IntersectedRays, 246
 - Invalidate, 245
 - Layers, 246
 - Object, 245
 - ObjectCount, 246
 - ObjectsNotCulled, 246
 - Owner, 247
 - Projection, 247
 - Select, 245
 - SelectAny, 245
 - Sort, 245
 - Update, 246
 - UpdateNeeded, 246
 - View, 247
 - ViewProjection, 247
- Afterwarp.OceanMaterial, 247
 - AlbedoColor, 249
 - AmbientColor, 249
 - Bitmask, 249
 - Default, 250
 - EmissiveColor, 249
 - Extinction, 249
 - Fresnel, 249
 - OceanMaterial, 248
 - Shadows, 249
 - SpecularColor, 249
 - SpecularExponent, 250
- Afterwarp.OceanSimulation, 250
 - Attributes, 253
 - Device, 253
 - GetWavesTexture, 252
 - Material, 253
 - OceanSimulation, 252
 - Parameters, 253
 - Plane, 253
 - Projection, 253
 - Render, 252
 - SamplerShadow, 254
 - Scale, 254
 - Sections, 254
 - Update, 253
 - View, 254
 - ViewDistance, 254
- Afterwarp.OceanWavesParameters, 254
 - Choppiness, 255
 - Default, 256
 - OceanWavesParameters, 255
 - Resolution, 255
 - WaveSize, 256
 - Wind, 256
- Afterwarp.ParallaxMappingParameters, 256

- Default, [258](#)
- Occlusion, [257](#)
- ParallaxMappingParameters, [257](#)
- SamplesMax, [257](#)
- SamplesMin, [257](#)
- Scale, [257](#)
- Afterwarp.PathBroker, [258](#)
 - Fill, [260](#)
 - PathBroker, [259](#)
 - Reset, [260](#)
 - Stroke, [260](#)
- Afterwarp.PathBuilder, [260](#)
 - Circle, [261](#)
 - Clear, [261](#)
 - ClosePath, [262](#)
 - CurveTo, [262](#)
 - Diamond, [262](#)
 - Elements, [264](#)
 - Flatness, [262](#)
 - Gear, [262](#)
 - LineTo, [262](#)
 - MoveTo, [263](#)
 - PathBuilder, [261](#)
 - QuadCurveTo, [263](#)
 - Rectangle, [263](#)
 - SmoothCurveTo, [263](#)
 - SmoothQuadCurveTo, [263](#)
- Afterwarp.PathCommand, [264](#)
 - Close, [264](#)
 - CurveTo, [264](#)
 - Flatness, [264](#)
 - LimitCusp, [265](#)
 - LineTo, [265](#)
 - MoveTo, [265](#)
 - QuadCurveTo, [265](#)
 - Relative, [265](#)
 - Smooth, [265](#)
 - ToleranceAngle, [266](#)
- Afterwarp.PathElement, [266](#)
 - Command, [267](#)
 - Length, [267](#)
 - PathElement, [267](#)
 - Value, [267](#)
- Afterwarp.Point, [268](#)
 - Angle, [272](#)
 - Average, [270](#)
 - Cross, [270](#)
 - Distance, [270](#)
 - Dot, [270](#)
 - Empty, [272](#)
 - Equals, [270](#)
 - GetHashCode, [270](#)
 - Length, [272](#)
 - One, [273](#)
 - operator!=, [270](#)
 - operator+, [271](#)
 - operator-, [271](#)
 - operator/, [271](#)
 - operator==, [272](#)
 - operator*, [271](#)
 - Point, [269](#)
 - Transpose, [273](#)
 - UnitX, [273](#)
 - UnitY, [273](#)
 - X, [272](#)
 - Y, [272](#)
 - Zero, [273](#)
- Afterwarp.Program, [274](#)
 - Begin, [276](#)
 - Bind, [276](#)
 - Commit, [276](#)
 - Device, [278](#)
 - Draw, [276](#)
 - DrawIndexed, [277](#)
 - DrawInstances, [277](#)
 - DrawInstancesIndexed, [277](#)
 - End, [277](#)
 - Program, [276](#)
 - ResetBindings, [277](#)
 - ResetCache, [277](#)
 - SetPatchVertices, [278](#)
 - Unbind, [278](#)
 - Update, [278](#)
- Afterwarp.ProgramElement, [279](#)
 - Channel, [280](#)
 - Element, [280](#)
 - Index, [280](#)
 - Name, [280](#)
 - NameBytes, [280](#)
 - ProgramElement, [279](#)
 - Shader, [280](#)
 - Size, [280](#)
- Afterwarp.ProgramVariable, [281](#)
 - Count, [282](#)
 - Format, [282](#)
 - Index, [282](#)
 - Name, [283](#)
 - NameBytes, [282](#)
 - Offset, [282](#)
 - ProgramVariable, [281](#)
 - Shader, [282](#)
 - Size, [282](#)
- Afterwarp.Quad, [284](#)
 - BottomLeft, [288](#)
 - BottomRight, [288](#)
 - Flip, [289](#)
 - Mirror, [289](#)
 - Offset, [286](#)
 - Quad, [285](#), [286](#)
 - Rotated, [286](#)
 - RotatedTopLeft, [287](#)
 - Scale, [287](#)
 - Scaled, [287](#)
 - SkewedHoriz, [287](#)
 - SkewedVert, [288](#)
 - TopLeft, [288](#)

- TopRight, [288](#)
- Transform, [288](#)
- Unity, [289](#)
- Zero, [289](#)
- Afterwarp.RandomSequence, [289](#)
 - Gaussian, [291](#)
 - RandomSequence, [290](#)
 - Ranged, [290](#)
 - Raw, [291](#)
 - Raw64, [291](#)
 - State, [291](#)
 - Value, [291](#)
 - ValueDouble, [291](#)
- Afterwarp.Ray, [291](#)
 - Direction, [293](#)
 - IntersectCubeVolume, [292](#)
 - IntersectPlane, [292](#)
 - IntersectTriangle, [293](#)
 - Origin, [293](#)
 - Ray, [292](#)
- Afterwarp.Rect, [293](#)
 - Bottom, [297](#)
 - Bounds, [295](#)
 - Contains, [295](#)
 - Empty, [297](#)
 - Equals, [295](#)
 - GetHashCode, [295](#)
 - Height, [297](#)
 - Inflate, [296](#)
 - Intersect, [296](#)
 - Join, [296](#)
 - Left, [297](#)
 - Offset, [296](#)
 - operator!=, [296](#)
 - operator==, [296](#)
 - Overlaps, [297](#)
 - Rect, [295](#)
 - Right, [298](#)
 - Top, [297](#)
 - Width, [297](#)
 - Zero, [298](#)
- Afterwarp.RectF, [298](#)
 - Bottom, [303](#)
 - BottomLeft, [303](#)
 - BottomRight, [303](#)
 - Bounds, [300](#)
 - Contains, [300](#), [301](#)
 - Empty, [303](#)
 - Height, [302](#)
 - Inflate, [301](#)
 - Intersect, [301](#)
 - Join, [301](#)
 - Left, [302](#)
 - Offset, [301](#), [302](#)
 - Overlaps, [302](#)
 - RectF, [300](#)
 - Right, [303](#)
 - Size, [303](#)
- Top, [302](#)
- TopLeft, [303](#)
- TopRight, [303](#)
- Unity, [304](#)
- Width, [302](#)
- Zero, [304](#)
- Afterwarp.RenderingState, [304](#)
 - BlendAlpha, [306](#)
 - BlendColor, [306](#)
 - BlendConstant, [306](#)
 - ClampDepthBias, [306](#)
 - CullFace, [306](#)
 - Default, [307](#)
 - DepthBias, [307](#)
 - DepthFunc, [307](#)
 - RenderingState, [306](#)
 - SlopeDepthBias, [307](#)
 - States, [307](#)
 - StencilBack, [307](#)
 - StencilFront, [307](#)
 - StencilRefMask, [307](#)
 - StencilRefValue, [308](#)
 - StencilWriteMask, [308](#)
- Afterwarp.RenderingState.Blend, [82](#)
 - Blend, [83](#)
 - Default, [84](#)
 - Dest, [83](#)
 - Op, [83](#)
 - Source, [83](#)
- Afterwarp.RenderingState.State, [381](#)
 - AlphaToCoverage, [382](#)
 - BlendEnable, [382](#)
 - ColorWrite, [382](#)
 - CubeMapSeamless, [383](#)
 - CullClockwise, [383](#)
 - DepthClip, [383](#)
 - DepthTest, [383](#)
 - DepthWrite, [383](#)
 - LineAntialias, [383](#)
 - Multisampling, [383](#)
 - PerSampleShading, [384](#)
 - ScissorClip, [384](#)
 - StencilTest, [384](#)
 - Wireframe, [384](#)
- Afterwarp.RenderingState.StencilState, [384](#)
 - Default, [386](#)
 - DepthFailOp, [385](#)
 - DepthPassOp, [385](#)
 - FailOp, [385](#)
 - Func, [386](#)
 - StencilState, [385](#)
- Afterwarp.Sampler, [308](#)
 - Bind, [310](#)
 - Device, [310](#)
 - Sampler, [310](#)
 - State, [310](#)
 - Unbind, [310](#)
- Afterwarp.SamplerState, [311](#)

- AddressU, [313](#)
- AddressV, [313](#)
- AddressW, [313](#)
- Anisotropy, [313](#)
- BiasLOD, [313](#)
- BorderColor, [313](#)
- CompareFunc, [313](#)
- CompareToRef, [313](#)
- Default, [314](#)
- FilterMag, [314](#)
- FilterMin, [314](#)
- FilterMip, [314](#)
- MaxLOD, [314](#)
- MinLOD, [314](#)
- SamplerState, [312](#)
- Afterwarp.Scene, [315](#)
 - ActiveVertexElements, [320](#)
 - Atlas, [320](#)
 - Attributes, [320](#)
 - Begin, [317](#)
 - CreateDepthNormals, [317](#)
 - CreateModeling, [317](#)
 - Device, [320](#)
 - End, [318](#)
 - GetSampler, [318](#)
 - GetTexture, [318](#)
 - GetVertexElements, [318](#)
 - Instances, [318](#)
 - InstancesCount, [320](#)
 - Lights, [320](#)
 - Material, [320](#)
 - ParallaxMapping, [320](#)
 - Prepare, [318](#)
 - Program, [321](#)
 - Projection, [321](#)
 - Rendering, [321](#)
 - ResetCache, [319](#)
 - SetTexture, [319](#)
 - SetVertexElements, [319](#)
 - SetVertexElementsFromTextModeller, [319](#)
 - TextureCabinet, [321](#)
 - ToneMappingCoefficients, [321](#)
 - TryPrepare, [319](#)
 - View, [321](#)
 - World, [321](#)
- Afterwarp.Scene.Attribute, [74](#)
 - Coloring, [75](#)
 - DepthPrePass, [75](#)
 - Glassy, [75](#)
 - Instancing, [75](#)
 - LinearDepths, [76](#)
 - Modeling, [76](#)
 - Normals, [76](#)
 - ShadowsCubic, [76](#)
 - SinglePass, [76](#)
 - TexturingCubic, [76](#)
- Afterwarp.SceneLight, [322](#)
 - AlbedoColor, [324](#)
 - AmbientColor, [324](#)
 - Angle, [324](#)
 - AngleCutoff, [324](#)
 - AttenuationEnd, [324](#)
 - AttenuationStart, [324](#)
 - Bitmask, [324](#)
 - Default, [326](#)
 - Direction, [324](#)
 - Intensity, [325](#)
 - Payload, [325](#)
 - Position, [325](#)
 - Reserved, [325](#)
 - SceneLight, [323](#)
 - ShadowCaster, [325](#)
 - SpecularColor, [325](#)
 - SpecularExponent, [325](#)
- Afterwarp.SceneLights, [326](#)
 - Add, [328](#)
 - Clear, [328](#)
 - Clusters, [329](#)
 - ClusterSize, [329](#)
 - Count, [329](#)
 - CullingMode, [329](#)
 - DepthSlices, [330](#)
 - Device, [330](#)
 - Erase, [328](#)
 - Execute, [328](#)
 - GetLight, [328](#)
 - Indices, [330](#)
 - RenderDebug, [329](#)
 - SceneLights, [328](#)
 - SetLight, [329](#)
 - Size, [330](#)
- Afterwarp.SceneMesh, [331](#)
 - AutoDraw, [333](#)
 - AutoDrawSliced, [333](#)
 - Bounds, [333](#)
 - GetSlice, [333](#)
 - Latches, [334](#)
 - Materials, [334](#)
 - Model, [334](#)
 - Name, [334](#)
 - Payload, [335](#)
 - Scale, [335](#)
 - SetSlice, [334](#)
 - Size, [335](#)
 - Tags, [335](#)
 - VertexElementsIndex, [335](#)
 - VertexElementsIndexUndefined, [334](#)
 - Voxel, [335](#)
- Afterwarp.SceneMeshes, [336](#)
 - Add, [338](#)
 - AddFromBuffer, [338](#)
 - AddFromFile, [338](#)
 - Clear, [339](#)
 - Count, [341](#)
 - Device, [341](#)
 - Erase, [339](#)

- Exists, [339](#)
- Mesh, [339](#)
- Payload, [339](#)
- PayloadExists, [340](#)
- SceneMeshes, [338](#)
- Slice, [340](#)
- TryAddFromFile, [340](#)
- TryMesh, [340](#)
- TryPayload, [340](#)
- Afterwarp.SceneMeshLatch, [341](#)
 - Group, [342](#)
 - Name, [342](#)
 - NameBytes, [342](#)
 - Orientation, [342](#)
 - Position, [342](#)
 - Type, [342](#)
- Afterwarp.SceneMeshLatches, [343](#)
 - Add, [345](#)
 - Clear, [345](#)
 - Copy, [345](#)
 - Count, [347](#)
 - Erase, [345](#)
 - GetLatch, [345](#)
 - GetLatchIndex, [346](#)
 - GetWaypointDistance, [346](#)
 - InvalidateWaypoints, [346](#)
 - LoadFromFile, [346](#)
 - LoadFromFileInMemory, [346](#)
 - SaveToFile, [346](#)
 - SceneMeshLatches, [345](#)
 - SetLatch, [346](#)
 - TakeAway, [347](#)
 - TryLoadFromFile, [347](#)
 - TryLoadFromFileInMemory, [347](#)
- Afterwarp.SceneMeshLatchType, [347](#)
 - Joint, [348](#)
 - Waypoint, [348](#)
- Afterwarp.SceneMeshMaterial, [348](#)
 - AddRange, [350](#)
 - ClearRanges, [350](#)
 - Commit, [350](#)
 - Copy, [350](#)
 - GetRange, [351](#)
 - GetTexture, [351](#)
 - Name, [352](#)
 - RangeCount, [352](#)
 - ReleaseTextures, [351](#)
 - SetRange, [351](#)
 - SetTexture, [351](#)
 - Shading, [352](#)
- Afterwarp.SceneMeshMaterialRange, [352](#)
 - IndexCount, [353](#)
 - IndexStart, [353](#)
 - SceneMeshMaterialRange, [353](#)
- Afterwarp.SceneMeshMaterials, [353](#)
 - Add, [355](#)
 - Clear, [355](#)
 - Commit, [355](#)
 - Copy, [356](#)
 - Count, [356](#)
 - Erase, [356](#)
 - GetMaterial, [356](#)
 - Name, [356](#)
 - SceneMeshMaterials, [355](#)
 - TakeAway, [356](#)
 - Texturing, [357](#)
- Afterwarp.SceneMeshMaterialShading, [357](#)
 - Ambient, [358](#)
 - Bloom, [358](#)
 - Diffuse, [358](#)
 - Emissive, [359](#)
 - Lighting, [359](#)
 - Reserved, [359](#)
 - Roughness, [359](#)
 - SceneMeshMaterialShading, [358](#)
 - Specular, [359](#)
 - SpecularExponent, [359](#)
- Afterwarp.SelectionHighlight, [360](#)
 - Begin, [362](#)
 - Clear, [362](#)
 - DepthStencil, [363](#)
 - Device, [363](#)
 - End, [362](#)
 - Filter, [362](#)
 - Grayscale, [363](#)
 - Parameters, [363](#)
 - Rendering, [363](#)
 - Samples, [363](#)
 - SelectionHighlight, [362](#)
 - Size, [363](#)
 - Texture, [362](#)
 - TextureCoordinates, [363](#)
- Afterwarp.SelectionHighlightParameters, [364](#)
 - BlurOffset, [365](#)
 - BlurPasses, [365](#)
 - Default, [366](#)
 - GlowIntensity, [365](#)
 - OutlineColor, [365](#)
 - OutlineStart, [365](#)
 - SelectionHighlightParameters, [365](#)
- Afterwarp.ShadowCaster, [366](#)
 - Atlas, [369](#)
 - Begin, [368](#)
 - Clear, [368](#)
 - Device, [369](#)
 - End, [368](#)
 - Filter, [368](#)
 - GetTexture, [368](#)
 - Position, [369](#)
 - Rendering, [369](#)
 - Size, [369](#)
 - ViewProjection, [369](#)
- Afterwarp.ShadowCastingAtlas, [370](#)
 - Add, [372](#)
 - BorderFill, [372](#)
 - Caster, [372](#)

- Clear, [372](#)
- Count, [373](#)
- Device, [373](#)
- Erase, [372](#)
- Padding, [373](#)
- Parameters, [373](#)
- Samples, [373](#)
- ShadowCastingAtlas, [372](#)
- Size, [373](#)
- Technique, [373](#)
- Texture, [374](#)
- Afterwarp.ShadowParameters, [374](#)
 - Bias, [375](#)
 - BleedSigma, [375](#)
 - BlurSamples, [375](#)
 - BlurSigma, [375](#)
 - Default, [376](#)
 - Exponent1, [375](#)
 - Exponent2, [376](#)
 - ShadowParameters, [375](#)
 - Variance, [376](#)
- Afterwarp.SignedDistanceField, [376](#)
 - Default, [378](#)
 - OutlineDistanceMaxSDF, [377](#)
 - OutlineDistanceMinSDF, [377](#)
 - OutlineOffsetSDF, [377](#)
 - SignedDistanceField, [377](#)
 - SignedFieldDistance, [378](#)
 - SuperSampleSDF, [378](#)
- Afterwarp.SpatialFog, [378](#)
 - Device, [381](#)
 - Distance, [381](#)
 - Execute, [380](#)
 - ExecuteGlassy, [380](#)
 - Formula, [381](#)
 - Parameters, [381](#)
 - Projection, [381](#)
 - SpatialFog, [380](#)
 - View, [381](#)
- Afterwarp.Surface, [386](#)
 - Bits, [394](#)
 - ByteSize, [394](#)
 - BytesPerPixel, [394](#)
 - Clear, [389](#)
 - ConvertFormat, [390](#)
 - Copy, [390](#)
 - Empty, [394](#)
 - Flip, [390](#)
 - Format, [394](#)
 - GetPixel, [390](#)
 - GetPixelBilinear, [390](#)
 - GetPixelPtr, [391](#)
 - GetScanline, [391](#)
 - HasAlphaChannel, [394](#)
 - Height, [394](#)
 - Invert, [391](#)
 - MakeSignedDistanceField, [391](#)
 - Mirror, [391](#)
 - Pitch, [394](#)
 - PremultipliedAlpha, [395](#)
 - PremultiplyAlpha, [391](#)
 - ResetAlpha, [392](#)
 - Resize, [392](#)
 - SaveToFile, [392](#)
 - SaveToFileInMemory, [392](#)
 - SetPixel, [392](#)
 - ShrinkFrom, [393](#)
 - Stretch, [393](#)
 - StretchBilinear, [393](#)
 - Surface, [389](#)
 - UnpremultiplyAlpha, [393](#)
 - Width, [395](#)
- Afterwarp.SwapChain, [395](#)
 - Begin, [397](#)
 - DepthStencil, [398](#)
 - Device, [398](#)
 - End, [397](#)
 - Format, [398](#)
 - Height, [398](#)
 - Multisamples, [398](#)
 - Resize, [398](#)
 - SwapChain, [397](#)
 - VSync, [398](#)
 - Width, [399](#)
 - WindowHandle, [399](#)
- Afterwarp.TextEntryRect, [399](#)
 - Position, [400](#)
 - Rectangle, [400](#)
- Afterwarp.TextModeller, [400](#)
 - Clear, [403](#)
 - CopyToMeshBuffer, [403](#)
 - Device, [406](#)
 - Draw, [403](#)
 - DrawCurved, [404](#)
 - DrawDepthCurved, [405](#)
 - Extent, [405](#)
 - ExtentByShape, [405](#)
 - Prepare, [405](#)
 - Provider, [406](#)
 - Render, [406](#)
 - Reset, [406](#)
 - SetFontParameters, [406](#)
 - TextModeller, [403](#)
 - TextRects, [406](#)
 - Transform, [407](#)
- Afterwarp.TextModeller.FontProvider, [157](#)
- Afterwarp.TextRenderer, [407](#)
 - Canvas, [411](#)
 - Draw, [409](#)
 - DrawAligned, [409](#)
 - DrawAlignedByPixels, [410](#)
 - DrawCentered, [410](#)
 - DrawCenteredByPixels, [410](#)
 - Extent, [410](#)
 - ExtentByPixels, [411](#)
 - Parameters, [411](#)

- TextRects, [411](#)
- TextRenderer, [409](#)
- Afterwarp.TextRenderModifiers, [412](#)
 - Effect, [412](#)
 - Interleave, [412](#)
 - Scale, [412](#)
 - TextRenderModifiers, [412](#)
 - VerticalSpace, [413](#)
- Afterwarp.Texture, [413](#)
 - Attach, [416](#)
 - Begin, [416](#)
 - Bind, [417](#)
 - Clear, [417](#)
 - Copy, [417](#)
 - CreateNormalsAndOcclusion, [418](#)
 - CreateParallax, [418](#)
 - Detach, [418](#)
 - Device, [420](#)
 - End, [418](#)
 - GenerateMipMaps, [418](#)
 - LoadFromFile, [419](#)
 - Parameters, [420](#)
 - PlatformHandle, [420](#)
 - ResetCache, [419](#)
 - Retrieve, [419](#)
 - SaveToFile, [419](#)
 - Texture, [415](#), [416](#)
 - Unbind, [420](#)
 - Update, [420](#)
- Afterwarp.Texture.Attribute, [77](#)
 - Compute, [77](#)
 - Drawable, [77](#)
 - Dynamic, [77](#)
 - MipMapping, [77](#)
 - PremultipliedAlpha, [77](#)
 - Scratch, [78](#)
- Afterwarp.TextureCabinet, [421](#)
 - AmbientOcclusionParameters, [425](#)
 - Attributes, [425](#)
 - Begin, [423](#)
 - Clear, [423](#)
 - DepthStencil, [425](#)
 - Device, [425](#)
 - End, [424](#)
 - Fidelity, [425](#)
 - Filter, [424](#)
 - GetTexture, [424](#)
 - Present, [424](#)
 - Rendering, [425](#)
 - Resolve, [424](#)
 - Samples, [425](#)
 - Size, [425](#)
 - Texture, [426](#)
 - TextureCabinet, [423](#)
 - ToneMappingBloom, [426](#)
- Afterwarp.TextureCabinet.Attribute, [78](#)
 - AmbientOcclusion, [79](#)
 - AmbientOcclusionDownscale, [79](#)
- Bloom, [79](#)
- BloomCubic, [79](#)
- BloomDownscale, [79](#)
- Glassy, [79](#)
- GlassyFast, [79](#)
- GlassyFog, [80](#)
- GlassyFrosted, [80](#)
- LinearDepths, [80](#)
- Afterwarp.TextureParameters, [426](#)
 - Attributes, [427](#)
 - DepthStencil, [427](#)
 - Format, [427](#)
 - Height, [427](#)
 - Layers, [427](#)
 - Multisamples, [428](#)
 - TextureParameters, [427](#)
 - Type, [428](#)
 - Width, [428](#)
- Afterwarp.Timer, [428](#)
 - ExtractTokens, [431](#)
 - FrameRate, [431](#)
 - Latency, [431](#)
 - NextSlice, [430](#)
 - Reset, [430](#)
 - SkippedTimeSlices, [431](#)
 - Speed, [431](#)
 - Timer, [430](#)
 - TimeSlice, [431](#)
 - TrimSkippedTimeSlices, [430](#)
 - Update, [430](#)
 - UpdateNextSlice, [431](#)
- Afterwarp.ToneMappingBloom, [432](#)
 - BloomBlurSamples, [433](#)
 - BloomBlurSigma, [433](#)
 - BloomCoefficients, [433](#)
 - BloomColorShift, [433](#)
 - BloomGamma, [433](#)
 - BloomThreshold, [434](#)
 - Default, [434](#)
 - FrostedPower, [434](#)
 - GlassyBuckets, [434](#)
 - ToneFactors, [434](#)
 - ToneMappingBloom, [433](#)
 - ToneWhite, [434](#)
- Afterwarp.Types.cs, [474](#)
- Afterwarp.Utility, [435](#)
 - AddColors, [436](#)
 - AssignPayload< T >, [436](#)
 - AverageColors, [437](#)
 - AverageFourColors, [437](#)
 - AverageSixColors, [437](#)
 - BiasTransform, [437](#)
 - BlendColors, [437](#)
 - BlendFourColors, [438](#)
 - ColorToGray, [438](#)
 - ColorToGrayF, [438](#)
 - ComposeColors, [438](#)
 - DisplaceRB, [438](#)

- FreePayload, [439](#)
- FromUI, [439](#)
- GainTransform, [439](#)
- GetColorAlpha, [439](#)
- GetColorAlphaF, [439](#)
- InvertColor, [440](#)
- MakeColor, [440](#)
- MakeColorAlpha, [441](#)
- MakeColorGray, [441](#), [442](#)
- MakeColorRGB, [442](#)
- MakeColorWithGray, [443](#)
- MatrixLookAt4x4, [444](#)
- MatrixOrthographic4x4, [444](#)
- MatrixPerspective4x4, [444](#)
- MultiplyColors, [444](#)
- PremultiplyAlpha, [444](#)
- RetrievePayload< T >, [445](#)
- SineAccelerate, [445](#)
- SineCycle, [445](#)
- SineDecelerate, [445](#)
- SineTransform, [445](#)
- SineTwoCycle, [445](#)
- SubtractColors, [446](#)
- SystemTicks, [447](#)
- ToUI, [446](#)
- UnpremultiplyAlpha, [446](#)
- VertexElementsEstimatePitch, [446](#)
- Afterwarp.VertexElement, [447](#)
 - Channel, [448](#)
 - ChannelIndexBuffer, [448](#)
 - ChannelNormalized, [448](#)
 - Count, [448](#)
 - Format, [448](#)
 - Name, [449](#)
 - NameBytes, [449](#)
 - Offset, [449](#)
 - VertexElement, [448](#)
- Afterwarp.Volume, [451](#)
 - CalculateNearFarPlanes, [451](#)
 - CalculateVisibleFrame, [451](#)
- Afterwarp.Widget, [452](#)
 - AcceptKey, [455](#)
 - AcceptMouse, [455](#)
 - Accomodate, [455](#)
 - Alignment, [459](#)
 - BringToFront, [456](#)
 - ChildCount, [459](#)
 - ClassName, [459](#)
 - Enabled, [459](#)
 - ExternalEvent, [459](#)
 - ExternalEventFunc, [456](#)
 - FindAt, [456](#)
 - Focused, [459](#)
 - Get, [456](#)
 - Get< T >, [456](#)
 - GetChild, [456](#)
 - GetChildIndex, [456](#)
 - GetClientRect, [457](#)
 - GetEnumerator, [457](#)
 - GetPadding, [457](#)
 - Invalidate, [457](#)
 - InvokeEvent, [457](#)
 - LocalToScreen, [457](#)
 - Manager, [459](#)
 - Margins, [459](#)
 - Name, [460](#)
 - Parent, [460](#)
 - ParentZone, [460](#)
 - Payload, [460](#)
 - PropertyCount, [460](#)
 - Rect, [460](#)
 - ScreenToLocal, [457](#)
 - SendToBack, [458](#)
 - Set, [458](#)
 - Set< T >, [458](#)
 - SetPadding, [458](#)
 - Style, [460](#)
 - Update, [458](#)
 - Visible, [460](#)
 - Widget, [455](#)
 - ZoneCount, [461](#)
- Afterwarp.Widget.Iterator, [176](#)
 - Current, [177](#)
 - MoveNext, [177](#)
 - Reset, [177](#)
- Afterwarp.Widget.PropertyValue, [283](#)
 - Behavior, [284](#)
 - Name, [284](#)
 - PropertyValue, [283](#)
 - Type, [284](#)
 - Value, [284](#)
- Afterwarp.WidgetManager, [461](#)
 - Application, [466](#)
 - Attributes, [466](#)
 - BatchCount, [466](#)
 - Cursor, [466](#)
 - Format, [466](#)
 - GetTexture, [465](#)
 - GetWidget, [465](#)
 - Multisamples, [466](#)
 - Present, [465](#)
 - Scale, [467](#)
 - TextRenderer, [467](#)
 - TryGetWidget, [465](#), [466](#)
 - WidgetManager, [465](#)
- Afterwarp.WidgetManagerAttribute, [467](#)
 - Composition, [467](#)
 - Design, [467](#)
- Afterwarp.WidgetProperty, [468](#)
 - Attributes, [468](#)
 - Behavior, [468](#)
 - DataBytes, [469](#)
 - Location, [469](#)
 - Name, [469](#)
 - NameBytes, [469](#)
 - Reserved, [469](#)

- Type, [469](#)
- Albedo
 - Afterwarp, [44](#)
- AlbedoColor
 - Afterwarp.ObjectMaterial, [224](#)
 - Afterwarp.OceanMaterial, [249](#)
 - Afterwarp.SceneLight, [324](#)
- Alignment
 - Afterwarp.Widget, [459](#)
- Alignments
 - Afterwarp.ObjectModel, [234](#)
- Alpha
 - Afterwarp.FloatColor, [143](#)
- AlphaFormatRequest
 - Afterwarp, [25](#)
- AlphaToCoverage
 - Afterwarp.RenderingState.State, [382](#)
- AltLeft
 - Afterwarp, [36](#)
- AltRight
 - Afterwarp, [36](#)
- Always
 - Afterwarp, [30](#)
- Ambient
 - Afterwarp.SceneMeshMaterialShading, [358](#)
- AmbientColor
 - Afterwarp.ObjectMaterial, [224](#)
 - Afterwarp.OceanMaterial, [249](#)
 - Afterwarp.SceneLight, [324](#)
- AmbientOcclusion
 - Afterwarp.TextureCabinet.Attribute, [79](#)
- AmbientOcclusionDownscale
 - Afterwarp.TextureCabinet.Attribute, [79](#)
- AmbientOcclusionParameters
 - Afterwarp.AmbientOcclusionParameters, [57](#)
 - Afterwarp.TextureCabinet, [425](#)
- Android
 - Afterwarp, [31](#)
- Angle
 - Afterwarp.Point, [272](#)
 - Afterwarp.SceneLight, [324](#)
- AngleCutoff
 - Afterwarp.SceneLight, [324](#)
- Anisotropic
 - Afterwarp, [48](#)
- Anisotropy
 - Afterwarp.SamplerState, [313](#)
- AppCursor
 - Afterwarp, [26](#)
- Appearance
 - Afterwarp, [26](#)
- Application
 - Afterwarp.Application, [62](#)
 - Afterwarp.WidgetManager, [466](#)
- ApplicationConfiguration
 - Afterwarp.ApplicationConfiguration, [69](#)
- ApplicationEvent
 - Afterwarp, [26](#)
- ApplicationWindowState
 - Afterwarp, [27](#)
- Apps
 - Afterwarp, [37](#)
- Arc
 - Afterwarp.Canvas, [94](#)
- Arrow
 - Afterwarp, [26](#)
 - Afterwarp.Grapher, [164](#)
- ArrowWait
 - Afterwarp, [26](#)
- AssignPayload< T >
 - Afterwarp.Utility, [436](#)
- Atlas
 - Afterwarp.Scene, [320](#)
 - Afterwarp.ShadowCaster, [369](#)
- Attach
 - Afterwarp.Texture, [416](#)
- Attenuation
 - Afterwarp.AmbientOcclusionParameters, [57](#)
- AttenuationEnd
 - Afterwarp.SceneLight, [324](#)
- AttenuationStart
 - Afterwarp.SceneLight, [324](#)
- Attributes
 - Afterwarp.Canvas, [107](#)
 - Afterwarp.Device, [134](#)
 - Afterwarp.FontParameters, [156](#)
 - Afterwarp.ObjectModel, [234](#)
 - Afterwarp.OceanSimulation, [253](#)
 - Afterwarp.Scene, [320](#)
 - Afterwarp.TextureCabinet, [425](#)
 - Afterwarp.TextureParameters, [427](#)
 - Afterwarp.WidgetManager, [466](#)
 - Afterwarp.WidgetProperty, [468](#)
- AutoDraw
 - Afterwarp.ObjectModelView, [244](#)
 - Afterwarp.SceneMesh, [333](#)
- AutoDrawSliced
 - Afterwarp.SceneMesh, [333](#)
- Average
 - Afterwarp.Point, [270](#)
- AverageColors
 - Afterwarp.Utility, [437](#)
- AverageFourColors
 - Afterwarp.Utility, [437](#)
- AverageSixColors
 - Afterwarp.Utility, [437](#)
- B5G6R5
 - Afterwarp, [41](#)
- Back
 - Afterwarp, [49](#)
- Backspace
 - Afterwarp, [35](#)
- BatchCount
 - Afterwarp.Canvas, [107](#)
 - Afterwarp.Grapher, [167](#)
 - Afterwarp.WidgetManager, [466](#)

- Begin
 - Afterwarp.Canvas, 95
 - Afterwarp.ComputeProgram, 126
 - Afterwarp.Grapher, 164
 - Afterwarp.Program, 276
 - Afterwarp.Scene, 317
 - Afterwarp.SelectionHighlight, 362
 - Afterwarp.ShadowCaster, 368
 - Afterwarp.SwapChain, 397
 - Afterwarp.Texture, 416
 - Afterwarp.TextureCabinet, 423
- Behavior
 - Afterwarp.Device, 134
 - Afterwarp.Widget.PropertyValue, 284
 - Afterwarp.WidgetProperty, 468
- Bevel
 - Afterwarp, 39
- BGR10A2
 - Afterwarp, 41
- BGR10X2
 - Afterwarp, 41
- BGR5A1
 - Afterwarp, 41
- BGR5X1
 - Afterwarp, 41
- BGR8
 - Afterwarp, 41
- BGRA4
 - Afterwarp, 41
- BGRA8
 - Afterwarp, 41
- BGRA8_SRGB
 - Afterwarp, 41
- BGRX4
 - Afterwarp, 41
- BGRX8
 - Afterwarp, 41
- Bias
 - Afterwarp.FogParameters, 149
 - Afterwarp.MeshAligns, 186
 - Afterwarp.ShadowParameters, 375
- BiasLOD
 - Afterwarp.SamplerState, 313
- BiasTransform
 - Afterwarp.Utility, 437
- Bind
 - Afterwarp.ComputeProgram, 126
 - Afterwarp.Program, 276
 - Afterwarp.Sampler, 310
 - Afterwarp.Texture, 417
- Bitmask
 - Afterwarp.ObjectMaterial, 224
 - Afterwarp.OceanMaterial, 249
 - Afterwarp.SceneLight, 324
- Bits
 - Afterwarp.Surface, 394
- Black
 - Afterwarp.ColorPair, 118
 - Afterwarp.ColorRect, 122
 - Afterwarp.FloatColor, 143
 - Afterwarp.FloatColorRGB, 147
- Blank
 - Afterwarp, 26
- BleedSigma
 - Afterwarp.ShadowParameters, 375
- Blend
 - Afterwarp.RenderingState.Blend, 83
- BlendAlpha
 - Afterwarp.RenderingState, 306
- BlendColor
 - Afterwarp.RenderingState, 306
- BlendColors
 - Afterwarp.Utility, 437
- BlendConstant
 - Afterwarp.RenderingState, 306
- BlendEnable
 - Afterwarp.RenderingState.State, 382
- BlendFactor
 - Afterwarp, 27
- BlendFourColors
 - Afterwarp.Utility, 438
- BlendingEffect
 - Afterwarp, 27
- BlendOp
 - Afterwarp, 28
- Bloom
 - Afterwarp, 47
 - Afterwarp.SceneMeshMaterialShading, 358
 - Afterwarp.TextureCabinet.Attribute, 79
- BloomBlurSamples
 - Afterwarp.ToneMappingBloom, 433
- BloomBlurSigma
 - Afterwarp.ToneMappingBloom, 433
- BloomCoefficients
 - Afterwarp.ToneMappingBloom, 433
- BloomColorShift
 - Afterwarp.ToneMappingBloom, 433
- BloomCubic
 - Afterwarp.TextureCabinet.Attribute, 79
- BloomDownscale
 - Afterwarp.TextureCabinet.Attribute, 79
- BloomGamma
 - Afterwarp.ToneMappingBloom, 433
- BloomThreshold
 - Afterwarp.ToneMappingBloom, 434
- Blue
 - Afterwarp.FloatColor, 143
 - Afterwarp.FloatColorRGB, 147
- BlurFallOff
 - Afterwarp.AmbientOcclusionParameters, 57
- BlurOffset
 - Afterwarp.SelectionHighlightParameters, 365
- BlurPasses
 - Afterwarp.SelectionHighlightParameters, 365
- BlurSamples
 - Afterwarp.ShadowParameters, 375

- BlurSigma
 - Afterwarp.AmbientOcclusionParameters, [57](#)
 - Afterwarp.ShadowParameters, [375](#)
- Bold
 - Afterwarp, [35](#)
- BoolCallback
 - Afterwarp.Application, [62](#)
- Boolean
 - Afterwarp, [50](#)
- Border
 - Afterwarp, [47](#)
- BorderBrightness
 - Afterwarp.FontEffect, [153](#)
- BorderColor
 - Afterwarp.CanvasSamplerState, [113](#)
 - Afterwarp.SamplerState, [313](#)
- BorderFill
 - Afterwarp.ShadowCastingAtlas, [372](#)
- BorderOpacity
 - Afterwarp.FontEffect, [153](#)
- BorderThickness
 - Afterwarp.FontEffect, [153](#)
- BorderType
 - Afterwarp.FontEffect, [153](#)
- Both
 - Afterwarp, [49](#)
- Bottom
 - Afterwarp, [49](#)
 - Afterwarp.Margins, [183](#)
 - Afterwarp.Rect, [297](#)
 - Afterwarp.RectF, [303](#)
- BottomLeft
 - Afterwarp.ColorRect, [121](#)
 - Afterwarp.Margins, [184](#)
 - Afterwarp.Quad, [288](#)
 - Afterwarp.RectF, [303](#)
- BottomRight
 - Afterwarp.ColorRect, [121](#)
 - Afterwarp.Margins, [184](#)
 - Afterwarp.Quad, [288](#)
 - Afterwarp.RectF, [303](#)
- BoundingBox
 - Afterwarp.Grapher, [165](#)
- Bounds
 - Afterwarp.Rect, [295](#)
 - Afterwarp.RectF, [300](#)
 - Afterwarp.SceneMesh, [333](#)
- BoundsMax
 - Afterwarp.MeshMetaTagPortion, [208](#)
- BoundsMin
 - Afterwarp.MeshMetaTagPortion, [208](#)
- BringToFront
 - Afterwarp.Widget, [456](#)
- Buffer
 - Afterwarp.Buffer, [86](#)
- BufferAccessType
 - Afterwarp, [28](#)
- BufferDataType
 - Afterwarp, [28](#)
- Butt
 - Afterwarp, [38](#)
- Byte
 - Afterwarp, [32](#)
- ByteSize
 - Afterwarp.Surface, [394](#)
- BytesPerPixel
 - Afterwarp.Surface, [394](#)
- CalculateBounds
 - Afterwarp.MeshBuffer, [191](#)
- CalculateFlatNormals
 - Afterwarp.MeshBuffer, [191](#)
- CalculateNearFarPlanes
 - Afterwarp.Volume, [451](#)
- CalculateNormals
 - Afterwarp.MeshBuffer, [191](#)
- CalculateNormalsWeld
 - Afterwarp.MeshBuffer, [191](#)
- CalculateVisibleFrame
 - Afterwarp.Volume, [451](#)
- CameraCommand
 - Afterwarp, [29](#)
- CameraConstraints
 - Afterwarp.CameraConstraints, [88](#)
- Cancel
 - Afterwarp, [35](#)
- Canvas
 - Afterwarp, [49](#)
 - Afterwarp.Canvas, [94](#)
 - Afterwarp.TextRenderer, [411](#)
- CanvasBuffer
 - Afterwarp.CanvasBuffer, [111](#)
- CanvasContextState
 - Afterwarp, [29](#)
- CanvasSamplerState
 - Afterwarp.CanvasSamplerState, [113](#)
- CapsLock
 - Afterwarp, [35](#)
- CaptureMouseInput
 - Afterwarp.Application, [62](#)
- Caster
 - Afterwarp.ShadowCastingAtlas, [372](#)
- Ceiling
 - Afterwarp.ActorCamera, [55](#)
- Cell
 - Afterwarp, [26](#)
- Centralize
 - Afterwarp.MeshBuffer, [191](#)
- Channel
 - Afterwarp.ComputeBindTextureFormat, [123](#)
 - Afterwarp.MeshModel, [217](#)
 - Afterwarp.ProgramElement, [280](#)
 - Afterwarp.VertexElement, [448](#)
- ChannelIndexBuffer
 - Afterwarp.VertexElement, [448](#)
- ChannelNormalized
 - Afterwarp.VertexElement, [448](#)

- Child
 - Afterwarp.ObjectModel, [231](#)
- ChildCount
 - Afterwarp.ObjectModel, [234](#)
 - Afterwarp.Widget, [459](#)
- Choppiness
 - Afterwarp.OceanWavesParameters, [255](#)
- Chroma
 - Afterwarp.GaussianBlur, [161](#)
- Circle
 - Afterwarp.Canvas, [95](#)
 - Afterwarp.PathBuilder, [261](#)
- Clamp
 - Afterwarp, [47](#)
- ClampDepthBias
 - Afterwarp.RenderingState, [306](#)
- ClassName
 - Afterwarp.Widget, [459](#)
- Clear
 - Afterwarp, [35](#)
 - Afterwarp.CanvasBuffer, [111](#)
 - Afterwarp.Device, [134](#)
 - Afterwarp.MeshBuffer, [192](#)
 - Afterwarp.MeshMetaTags, [211](#)
 - Afterwarp.ObjectMaterials, [228](#)
 - Afterwarp.ObjectModels, [239](#)
 - Afterwarp.PathBuilder, [261](#)
 - Afterwarp.SceneLights, [328](#)
 - Afterwarp.SceneMeshes, [339](#)
 - Afterwarp.SceneMeshLatches, [345](#)
 - Afterwarp.SceneMeshMaterials, [355](#)
 - Afterwarp.SelectionHighlight, [362](#)
 - Afterwarp.ShadowCaster, [368](#)
 - Afterwarp.ShadowCastingAtlas, [372](#)
 - Afterwarp.Surface, [389](#)
 - Afterwarp.TextModeller, [403](#)
 - Afterwarp.Texture, [417](#)
 - Afterwarp.TextureCabinet, [423](#)
- ClearAndShrink
 - Afterwarp.CanvasBuffer, [111](#)
- ClearRanges
 - Afterwarp.SceneMeshMaterial, [350](#)
- ClearRegions
 - Afterwarp.ImageAtlas, [170](#)
- ClearTextures
 - Afterwarp.ImageAtlas, [170](#)
- ClearViews
 - Afterwarp.ObjectModels, [240](#)
- Client
 - Afterwarp, [49](#)
- ClientRect
 - Afterwarp.Application, [65](#)
- ClientSize
 - Afterwarp.Application, [65](#)
- ClipRect
 - Afterwarp.Canvas, [107](#)
- Close
 - Afterwarp.PathCommand, [264](#)
- ClosePath
 - Afterwarp.PathBuilder, [262](#)
- Clusters
 - Afterwarp.SceneLights, [329](#)
- ClustersCullingMode
 - Afterwarp, [29](#)
- ClusterSize
 - Afterwarp.SceneLights, [329](#)
- Color
 - Afterwarp, [47](#), [50](#)
 - Afterwarp.DeviceClear, [138](#)
 - Afterwarp.FogParameters, [149](#)
 - Afterwarp.MeshBuffer.VertexEntry, [450](#)
 - Afterwarp.ObjectModel, [234](#)
- ColorAdjust
 - Afterwarp.Canvas.Attribute, [71](#)
- ColorDithering
 - Afterwarp.ColorDithering, [116](#)
- ColorDitheringFormat
 - Afterwarp, [30](#)
- ColorHDR
 - Afterwarp, [47](#)
- Coloring
 - Afterwarp.Scene.Attribute, [75](#)
- ColorPair
 - Afterwarp, [50](#)
 - Afterwarp.ColorPair, [118](#)
- ColorRect
 - Afterwarp, [50](#)
 - Afterwarp.ColorRect, [120](#)
- Colors
 - Afterwarp.CanvasBuffer, [111](#)
- ColorToGray
 - Afterwarp.Utility, [438](#)
- ColorToGrayF
 - Afterwarp.Utility, [438](#)
- ColorWrite
 - Afterwarp.RenderingState.State, [382](#)
- Combine
 - Afterwarp.MeshBuffer, [192](#)
- Command
 - Afterwarp.ActorCamera, [53](#)
 - Afterwarp.PathElement, [267](#)
- Commit
 - Afterwarp.ComputeProgram, [127](#)
 - Afterwarp.Program, [276](#)
 - Afterwarp.SceneMeshMaterial, [350](#)
 - Afterwarp.SceneMeshMaterials, [355](#)
- CompareFunc
 - Afterwarp.ObjectModelView, [244](#)
 - Afterwarp.SamplerState, [313](#)
- CompareToRef
 - Afterwarp.SamplerState, [313](#)
- ComparisonFunc
 - Afterwarp, [30](#)
- ComposeColors
 - Afterwarp.Utility, [438](#)
- Composition

- Afterwarp.WidgetManagerAttribute, 467
- Compute
 - Afterwarp, 28, 45
 - Afterwarp.DeviceBehavior, 136
 - Afterwarp.Texture.Attribute, 77
- ComputeBindTextureFormat
 - Afterwarp.ComputeBindTextureFormat, 123
- ComputeParameters
 - Afterwarp.MeshVoxel, 220
- ComputeProgram
 - Afterwarp.ComputeProgram, 126
- ComputeTextureAccess
 - Afterwarp, 30
- Condensed
 - Afterwarp, 33
- Cone
 - Afterwarp.MeshBuffer, 192
- ConnectLatches
 - Afterwarp.ObjectModel, 232
- Constant
 - Afterwarp, 29
- ConstantAlpha
 - Afterwarp, 27
- ConstantBuffer
 - Afterwarp, 44
- ConstantColor
 - Afterwarp, 27
- Constraints
 - Afterwarp.ActorCamera, 55
- Contains
 - Afterwarp.Rect, 295
 - Afterwarp.RectF, 300, 301
- ContextState
 - Afterwarp.Canvas, 108
- Contrast
 - Afterwarp.AmbientOcclusionParameters, 57
- ConvertFormat
 - Afterwarp.Surface, 390
- ConvertPortableKey
 - Afterwarp.Application, 62
- CookTorrance
 - Afterwarp, 46
- Copy
 - Afterwarp, 28
 - Afterwarp.Buffer, 86
 - Afterwarp.MeshMetaTags, 211
 - Afterwarp.SceneMeshLatches, 345
 - Afterwarp.SceneMeshMaterial, 350
 - Afterwarp.SceneMeshMaterials, 356
 - Afterwarp.Surface, 390
 - Afterwarp.Texture, 417
- CopyToMeshBuffer
 - Afterwarp.TextModeller, 403
- Count
 - Afterwarp.MeshMetaTags, 212
 - Afterwarp.ObjectMaterials, 228
 - Afterwarp.ObjectModels, 242
 - Afterwarp.ProgramVariable, 282
 - Afterwarp.SceneLights, 329
 - Afterwarp.SceneMeshes, 341
 - Afterwarp.SceneMeshLatches, 347
 - Afterwarp.SceneMeshMaterials, 356
 - Afterwarp.ShadowCastingAtlas, 373
 - Afterwarp.VertexElement, 448
- CreateDepthNormals
 - Afterwarp.Scene, 317
- CreateModeling
 - Afterwarp.Scene, 317
- CreateNormalsAndOcclusion
 - Afterwarp.Texture, 418
- CreateParallax
 - Afterwarp.Texture, 418
- CreateRegion
 - Afterwarp.ImageAtlas, 170
- CreateTexture
 - Afterwarp.ImageAtlas, 170
- CreateView
 - Afterwarp.ObjectModels, 240
- Cross
 - Afterwarp, 43
 - Afterwarp.Point, 270
- CrossHair
 - Afterwarp, 26
- CtrlLeft
 - Afterwarp, 36
- CtrlRight
 - Afterwarp, 36
- Cube
 - Afterwarp.MeshBuffer, 192
- CubeMap
 - Afterwarp, 48
- CubeMapSeamless
 - Afterwarp.RenderingState.State, 383
- CubeMinimal
 - Afterwarp.MeshBuffer, 192
- CubeRound
 - Afterwarp.MeshBuffer, 193
- Cubic
 - Afterwarp.Canvas.Attribute, 71
- CullClockwise
 - Afterwarp.RenderingState.State, 383
- CullFace
 - Afterwarp.RenderingState, 306
- CullingMode
 - Afterwarp.SceneLights, 329
- Current
 - Afterwarp.ObjectModels.Iterator, 176
 - Afterwarp.Widget.Iterator, 177
- Cursor
 - Afterwarp.Application, 65
 - Afterwarp.WidgetManager, 466
- CurveTo
 - Afterwarp.PathBuilder, 262
 - Afterwarp.PathCommand, 264
- Custom
 - Afterwarp, 28

- Cylinder
 - Afterwarp.MeshBuffer, [193](#)
- D16
 - Afterwarp, [41](#)
- D24S8
 - Afterwarp, [41](#)
- D32F
 - Afterwarp, [41](#)
- D32S8F
 - Afterwarp, [41](#)
- DataBytes
 - Afterwarp.WidgetProperty, [469](#)
- Data Type
 - Afterwarp.Buffer, [87](#)
- Deactivate
 - Afterwarp, [26](#)
- Debug
 - Afterwarp.Device.Attribute, [72](#)
- Decimal
 - Afterwarp, [37](#)
- Decrement
 - Afterwarp, [45](#)
- DecrementWarp
 - Afterwarp, [45](#)
- Default
 - Afterwarp, [28](#)
 - Afterwarp.AmbientOcclusionParameters, [58](#)
 - Afterwarp.CanvasSamplerState, [114](#)
 - Afterwarp.FogParameters, [150](#)
 - Afterwarp.FontEffect, [154](#)
 - Afterwarp.MeshAligns, [186](#)
 - Afterwarp.ObjectMaterial, [225](#)
 - Afterwarp.OceanMaterial, [250](#)
 - Afterwarp.OceanWavesParameters, [256](#)
 - Afterwarp.ParallaxMappingParameters, [258](#)
 - Afterwarp.RenderingState, [307](#)
 - Afterwarp.RenderingState.Blend, [84](#)
 - Afterwarp.RenderingState.StencilState, [386](#)
 - Afterwarp.SamplerState, [314](#)
 - Afterwarp.SceneLight, [326](#)
 - Afterwarp.SelectionHighlightParameters, [366](#)
 - Afterwarp.ShadowParameters, [376](#)
 - Afterwarp.SignedDistanceField, [378](#)
 - Afterwarp.ToneMappingBloom, [434](#)
- Defined
 - Afterwarp.FloatColorRGB, [147](#)
- Delete
 - Afterwarp, [36](#)
- DensityGround
 - Afterwarp.FogParameters, [149](#)
- Depth
 - Afterwarp, [39](#), [47](#)
 - Afterwarp.DeviceClear, [138](#)
- DepthBias
 - Afterwarp.AmbientOcclusionParameters, [57](#)
 - Afterwarp.ObjectModel, [234](#)
 - Afterwarp.RenderingState, [307](#)
- DepthClearBadPrecision
 - Afterwarp.DeviceBehavior, [136](#)
- DepthClip
 - Afterwarp.RenderingState.State, [383](#)
- DepthClipNegative
 - Afterwarp.DeviceBehavior, [137](#)
- DepthFailOp
 - Afterwarp.RenderingState.StencilState, [385](#)
- DepthFogDistance
 - Afterwarp, [31](#)
- DepthFunc
 - Afterwarp.RenderingState, [307](#)
- DepthPassOp
 - Afterwarp.RenderingState.StencilState, [385](#)
- DepthPrePass
 - Afterwarp.Scene.Attribute, [75](#)
- DepthReverse
 - Afterwarp, [39](#)
- DepthSlices
 - Afterwarp.SceneLights, [330](#)
- DepthStencil
 - Afterwarp.SelectionHighlight, [363](#)
 - Afterwarp.SwapChain, [398](#)
 - Afterwarp.TextureCabinet, [425](#)
 - Afterwarp.TextureParameters, [427](#)
- DepthTest
 - Afterwarp.RenderingState.State, [383](#)
- DepthWrite
 - Afterwarp.RenderingState.State, [383](#)
- Description
 - Afterwarp.ObjectModel, [234](#)
- Design
 - Afterwarp.WidgetManagerAttribute, [467](#)
- Dest
 - Afterwarp.RenderingState.Blend, [83](#)
- DestAlpha
 - Afterwarp, [27](#)
- DestColor
 - Afterwarp, [27](#)
- Detach
 - Afterwarp.Texture, [418](#)
- Device
 - Afterwarp.Buffer, [87](#)
 - Afterwarp.Canvas, [108](#)
 - Afterwarp.ColorDithering, [116](#)
 - Afterwarp.ComputeProgram, [128](#)
 - Afterwarp.Device, [134](#)
 - Afterwarp.GaussianBlur, [161](#)
 - Afterwarp.Grapher, [167](#)
 - Afterwarp.ImageAtlas, [172](#)
 - Afterwarp.KawaseBlur, [180](#)
 - Afterwarp.OceanSimulation, [253](#)
 - Afterwarp.Program, [278](#)
 - Afterwarp.Sampler, [310](#)
 - Afterwarp.Scene, [320](#)
 - Afterwarp.SceneLights, [330](#)
 - Afterwarp.SceneMeshes, [341](#)
 - Afterwarp.SelectionHighlight, [363](#)
 - Afterwarp.ShadowCaster, [369](#)

- Afterwarp.ShadowCastingAtlas, [373](#)
- Afterwarp.SpatialFog, [381](#)
- Afterwarp.SwapChain, [398](#)
- Afterwarp.TextModeller, [406](#)
- Afterwarp.Texture, [420](#)
- Afterwarp.TextureCabinet, [425](#)
- DevicePlatform
 - Afterwarp, [31](#)
- DeviceTechnology
 - Afterwarp, [31](#)
- Diamond
 - Afterwarp.PathBuilder, [262](#)
- Diffuse
 - Afterwarp, [43](#)
 - Afterwarp.SceneMeshMaterialShading, [358](#)
- Direct3D
 - Afterwarp, [31](#)
- Direction
 - Afterwarp.Ray, [293](#)
 - Afterwarp.SceneLight, [324](#)
- Directory
 - Afterwarp, [32](#)
- DisableRealign
 - Afterwarp.ObjectModel.Attribute, [74](#)
- Disc
 - Afterwarp.MeshBuffer, [193](#)
- Dismantle
 - Afterwarp.MeshModel, [216](#)
- Dispatch
 - Afterwarp.ComputeProgram, [127](#)
- DisplaceRB
 - Afterwarp.Utility, [438](#)
- Dispose
 - Afterwarp.CustomObject, [130](#)
- Distance
 - Afterwarp.ActorCamera, [55](#)
 - Afterwarp.FogParameters, [149](#)
 - Afterwarp.Point, [270](#)
 - Afterwarp.SpatialFog, [381](#)
- Divide
 - Afterwarp, [37](#)
- Domain
 - Afterwarp, [45](#)
- DontCare
 - Afterwarp, [25](#)
- Dot
 - Afterwarp.Point, [270](#)
- DottedLine
 - Afterwarp.Grapher, [165](#)
- Double
 - Afterwarp, [32](#)
- DoubleClick
 - Afterwarp, [39](#)
- Down
 - Afterwarp, [35](#), [39](#)
- Drag
 - Afterwarp, [26](#)
- Dragging
 - Afterwarp, [26](#), [29](#)
- Draw
 - Afterwarp.Canvas, [95](#)
 - Afterwarp.MeshModel, [216](#)
 - Afterwarp.Program, [276](#)
 - Afterwarp.TextModeller, [403](#)
 - Afterwarp.TextRenderer, [409](#)
- Drawable
 - Afterwarp.Texture.Attribute, [77](#)
- DrawAligned
 - Afterwarp.TextRenderer, [409](#)
- DrawAlignedByPixels
 - Afterwarp.TextRenderer, [410](#)
- DrawCentered
 - Afterwarp.TextRenderer, [410](#)
- DrawCenteredByPixels
 - Afterwarp.TextRenderer, [410](#)
- DrawCurved
 - Afterwarp.TextModeller, [404](#)
- DrawDepthCurved
 - Afterwarp.TextModeller, [405](#)
- DrawIndexed
 - Afterwarp.Program, [277](#)
- DrawInstances
 - Afterwarp.MeshModel, [216](#)
 - Afterwarp.Program, [277](#)
- DrawInstancesIndexed
 - Afterwarp.Program, [277](#)
- Dynamic
 - Afterwarp, [28](#)
 - Afterwarp.Texture.Attribute, [77](#)
- Effect
 - Afterwarp.FontParameters, [156](#)
 - Afterwarp.TextRenderModifiers, [412](#)
- Element
 - Afterwarp.ProgramElement, [280](#)
- ElementFormat
 - Afterwarp, [31](#)
- Elements
 - Afterwarp.PathBuilder, [264](#)
- EliminateUnusedVertices
 - Afterwarp.MeshBuffer, [194](#)
- Ellipse
 - Afterwarp.Canvas, [95](#)
- Emissive
 - Afterwarp.SceneMeshMaterialShading, [359](#)
- EmissiveColor
 - Afterwarp.ObjectMaterial, [224](#)
 - Afterwarp.OceanMaterial, [249](#)
- Empty
 - Afterwarp.Margins, [184](#)
 - Afterwarp.Point, [272](#)
 - Afterwarp.Rect, [297](#)
 - Afterwarp.RectF, [303](#)
 - Afterwarp.Surface, [394](#)
- Enabled
 - Afterwarp.Widget, [459](#)
- End

- Afterwarp, [35](#), [46](#)
- Afterwarp.Canvas, [95](#)
- Afterwarp.ComputeProgram, [127](#)
- Afterwarp.Grapher, [165](#)
- Afterwarp.Program, [277](#)
- Afterwarp.Scene, [318](#)
- Afterwarp.SelectionHighlight, [362](#)
- Afterwarp.ShadowCaster, [368](#)
- Afterwarp.SwapChain, [397](#)
- Afterwarp.Texture, [418](#)
- Afterwarp.TextureCabinet, [424](#)
- Enter
 - Afterwarp, [39](#)
- Enumeration
 - Afterwarp, [50](#)
- Equal
 - Afterwarp, [30](#)
- Equals
 - Afterwarp.CustomObject, [131](#)
 - Afterwarp.Point, [270](#)
 - Afterwarp.Rect, [295](#)
- Erase
 - Afterwarp.MeshMetaTags, [212](#)
 - Afterwarp.ObjectMaterials, [228](#)
 - Afterwarp.ObjectModels, [240](#)
 - Afterwarp.SceneLights, [328](#)
 - Afterwarp.SceneMeshes, [339](#)
 - Afterwarp.SceneMeshLatches, [345](#)
 - Afterwarp.SceneMeshMaterials, [356](#)
 - Afterwarp.ShadowCastingAtlas, [372](#)
- EraseView
 - Afterwarp.ObjectModels, [240](#)
- Escape
 - Afterwarp, [35](#)
- ESM
 - Afterwarp, [46](#)
- ESM_Warp
 - Afterwarp, [46](#)
- Event
 - Afterwarp, [49](#)
- EventCallback
 - Afterwarp.Application, [62](#)
- EventHookCallback
 - Afterwarp.Application, [63](#)
- EVSM
 - Afterwarp, [46](#)
- Exception
 - Afterwarp.API.Exception, [140](#)
- ExecutablePath
 - Afterwarp.Application, [65](#)
- Execute
 - Afterwarp, [37](#)
 - Afterwarp.Application, [63](#)
 - Afterwarp.ColorDithering, [116](#)
 - Afterwarp.SceneLights, [328](#)
 - Afterwarp.SpatialFog, [380](#)
- ExecuteGlassy
 - Afterwarp.SpatialFog, [380](#)
- Exists
 - Afterwarp.ObjectModels, [240](#)
 - Afterwarp.SceneMeshes, [339](#)
- Expanded
 - Afterwarp, [33](#)
- Exponent1
 - Afterwarp.ShadowParameters, [375](#)
- Exponent2
 - Afterwarp.ShadowParameters, [376](#)
- Exponential
 - Afterwarp, [32](#)
- ExportColors
 - Afterwarp.MeshLoadingOption, [202](#)
- ExportNormals
 - Afterwarp.MeshLoadingOption, [202](#)
- ExportTangents
 - Afterwarp.MeshLoadingOption, [203](#)
- ExportTexCoords
 - Afterwarp.MeshLoadingOption, [203](#)
- ExportWeights
 - Afterwarp.MeshLoadingOption, [203](#)
- Extent
 - Afterwarp.TextModeller, [405](#)
 - Afterwarp.TextRenderer, [410](#)
- ExtentByPixels
 - Afterwarp.TextRenderer, [411](#)
- ExtentByShape
 - Afterwarp.TextModeller, [405](#)
- Extents
 - Afterwarp.CanvasBuffer, [111](#)
 - Afterwarp.MeshVoxel, [220](#)
- ExternalEvent
 - Afterwarp.Widget, [459](#)
- ExternalEventFunc
 - Afterwarp.Widget, [456](#)
- Extinction
 - Afterwarp.FogParameters, [150](#)
 - Afterwarp.OceanMaterial, [249](#)
- ExtraBold
 - Afterwarp, [35](#)
- ExtraCondensed
 - Afterwarp, [33](#)
- ExtractTokens
 - Afterwarp.Timer, [431](#)
- ExtraExpanded
 - Afterwarp, [33](#)
- ExtraHeavy
 - Afterwarp, [35](#)
- ExtraLight
 - Afterwarp, [35](#)
- Extreme
 - Afterwarp, [48](#)
- Extrusion
 - Afterwarp.MeshBuffer, [194](#)
- EyeRelative
 - Afterwarp, [31](#)
- F1
 - Afterwarp, [36](#)

- F10
 - Afterwarp, [36](#)
- F11
 - Afterwarp, [36](#)
- F12
 - Afterwarp, [36](#)
- F2
 - Afterwarp, [36](#)
- F3
 - Afterwarp, [36](#)
- F4
 - Afterwarp, [36](#)
- F5
 - Afterwarp, [36](#)
- F6
 - Afterwarp, [36](#)
- F7
 - Afterwarp, [36](#)
- F8
 - Afterwarp, [36](#)
- F9
 - Afterwarp, [36](#)
- FailOp
 - Afterwarp.RenderingState.StencilState, [385](#)
- Family
 - Afterwarp.FontParameters, [157](#)
- FamilyBytes
 - Afterwarp.FontParameters, [156](#)
- Fidelity
 - Afterwarp.TextureCabinet, [425](#)
- FileChooserDialog
 - Afterwarp, [32](#)
 - Afterwarp.Application, [63](#)
- Fill
 - Afterwarp.PathBroker, [260](#)
- FillBrightness
 - Afterwarp.FontEffect, [153](#)
- FillOpacity
 - Afterwarp.FontEffect, [153](#)
- FillRect
 - Afterwarp.Canvas, [96](#)
- FillRoundRect
 - Afterwarp.Canvas, [96](#)
- FillRoundRectBottom
 - Afterwarp.Canvas, [96](#)
- FillRoundRectTop
 - Afterwarp.Canvas, [96](#)
- FillRoundRectTopInverse
 - Afterwarp.Canvas, [97](#)
- Filter
 - Afterwarp.SelectionHighlight, [362](#)
 - Afterwarp.ShadowCaster, [368](#)
 - Afterwarp.TextureCabinet, [424](#)
- FilterMag
 - Afterwarp.CanvasSamplerState, [113](#)
 - Afterwarp.SamplerState, [314](#)
- FilterMin
 - Afterwarp.CanvasSamplerState, [113](#)
 - Afterwarp.SamplerState, [314](#)
- FilterMip
 - Afterwarp.CanvasSamplerState, [114](#)
 - Afterwarp.SamplerState, [314](#)
- FindAt
 - Afterwarp.Widget, [456](#)
- First
 - Afterwarp.ColorPair, [118](#)
- FirstIndex
 - Afterwarp.MeshMetaTagPortion, [208](#)
- FirstVertex
 - Afterwarp.MeshMetaTagPortion, [209](#)
- FixedSamples
 - Afterwarp.GaussianBlur, [161](#)
- Flatness
 - Afterwarp.PathBuilder, [262](#)
 - Afterwarp.PathCommand, [264](#)
- FlatScene
 - Afterwarp, [29](#)
- FlatText
 - Afterwarp, [29](#)
- Flip
 - Afterwarp.ImageRegion, [173](#)
 - Afterwarp.Quad, [289](#)
 - Afterwarp.Surface, [390](#)
- Float
 - Afterwarp, [32](#), [50](#)
- Float_11_11_10
 - Afterwarp, [32](#)
- FloatColor
 - Afterwarp.FloatColor, [141](#)
- FloatColorRGB
 - Afterwarp.FloatColorRGB, [145](#)
- Flush
 - Afterwarp.Canvas, [97](#)
 - Afterwarp.Grapher, [165](#)
- Focused
 - Afterwarp.Widget, [459](#)
- FogFormula
 - Afterwarp, [32](#)
- FogParameters
 - Afterwarp.FogParameters, [149](#)
- Font
 - Afterwarp, [50](#)
- FontBorder
 - Afterwarp, [32](#)
- FontEffect
 - Afterwarp.FontEffect, [152](#)
- FontParameters
 - Afterwarp.FontParameters, [156](#)
- FontSlant
 - Afterwarp, [33](#)
- FontStretch
 - Afterwarp, [33](#)
- FontWeight
 - Afterwarp, [33](#)
- ForceBufferUnbind
 - Afterwarp.DeviceBehavior, [137](#)

- Format
 - Afterwarp.Buffer, 87
 - Afterwarp.ColorDithering, 116
 - Afterwarp.ComputeBindTextureFormat, 123
 - Afterwarp.ProgramVariable, 282
 - Afterwarp.Surface, 394
 - Afterwarp.SwapChain, 398
 - Afterwarp.TextureParameters, 427
 - Afterwarp.VertexElement, 448
 - Afterwarp.WidgetManager, 466
- Formula
 - Afterwarp.SpatialFog, 381
- Forward
 - Afterwarp.ActorCamera, 55
- Fragment
 - Afterwarp, 45
- FrameRate
 - Afterwarp.Timer, 431
- FrameRect
 - Afterwarp.Canvas, 97
- FrameRoundRect
 - Afterwarp.Canvas, 97
- FreePayload
 - Afterwarp.Utility, 439
- Fresnel
 - Afterwarp.OceanMaterial, 249
- FromUI
 - Afterwarp.Utility, 439
- Front
 - Afterwarp, 49
- FrostedGlass
 - Afterwarp.ObjectMaterial, 224
- FrostedPower
 - Afterwarp.ToneMappingBloom, 434
- FrustumVolume
 - Afterwarp.MeshBuffer, 194
- FullScreen
 - Afterwarp, 27
- Func
 - Afterwarp.RenderingState.StencilState, 386
- GainTransform
 - Afterwarp.Utility, 439
- Gaussian
 - Afterwarp.RandomSequence, 291
- GaussianBlur
 - Afterwarp.GaussianBlur, 160
- Gear
 - Afterwarp.PathBuilder, 262
- GenerateMipMaps
 - Afterwarp.Texture, 418
- Geometry
 - Afterwarp, 45
 - Afterwarp.MeshMetaTagType, 213
- Geosphere
 - Afterwarp.MeshBuffer, 194
- Get
 - Afterwarp.Widget, 456
- Get< T >
 - Afterwarp.Widget, 456
- GetBounds
 - Afterwarp.MeshMetaTag, 206
- GetChild
 - Afterwarp.Widget, 456
- GetChildIndex
 - Afterwarp.Widget, 456
- GetClientRect
 - Afterwarp.Widget, 457
- GetColorAlpha
 - Afterwarp.Utility, 439
- GetColorAlphaF
 - Afterwarp.Utility, 439
- GetConnectedLatches
 - Afterwarp.ObjectModel, 232
- GetEnumerator
 - Afterwarp.ObjectModels, 240
 - Afterwarp.Widget, 457
- GetHashCode
 - Afterwarp.CustomObject, 131
 - Afterwarp.Point, 270
 - Afterwarp.Rect, 295
- GetLatch
 - Afterwarp.SceneMeshLatches, 345
- GetLatchIndex
 - Afterwarp.SceneMeshLatches, 346
- GetLatchTransform
 - Afterwarp.ObjectModel, 232
- GetLatchWaypointCouple
 - Afterwarp.ObjectModel, 232
- GetLatchWaypointCoupleMatrix
 - Afterwarp.ObjectModel, 233
- GetLight
 - Afterwarp.SceneLights, 328
- GetMaterial
 - Afterwarp.ObjectMaterials, 228
 - Afterwarp.SceneMeshMaterials, 356
- GetPadding
 - Afterwarp.Widget, 457
- GetPixel
 - Afterwarp.Surface, 390
- GetPixelBilinear
 - Afterwarp.Surface, 390
- GetPixelPtr
 - Afterwarp.Surface, 391
- GetPosition
 - Afterwarp.ActorCamera, 53
- GetQuaternion
 - Afterwarp.ActorCamera, 54
- GetRange
 - Afterwarp.SceneMeshMaterial, 351
- GetRotation
 - Afterwarp.ActorCamera, 54
- GetSampler
 - Afterwarp.Scene, 318
- GetScanline
 - Afterwarp.Surface, 391
- GetSlice

- Afterwarp.SceneMesh, 333
- GetTexture
 - Afterwarp.Scene, 318
 - Afterwarp.SceneMeshMaterial, 351
 - Afterwarp.ShadowCaster, 368
 - Afterwarp.TextureCabinet, 424
 - Afterwarp.WidgetManager, 465
- GetTransform
 - Afterwarp.ObjectModel, 233
- GetVersion
 - Afterwarp.Library, 181
- GetVertexElements
 - Afterwarp.Scene, 318
- GetWavesTexture
 - Afterwarp.OceanSimulation, 252
- GetWaypointDistance
 - Afterwarp.ObjectModel, 233
 - Afterwarp.SceneMeshLatches, 346
- GetWidget
 - Afterwarp.WidgetManager, 465
- Glassy
 - Afterwarp, 47
 - Afterwarp.Scene.Attribute, 75
 - Afterwarp.TextureCabinet.Attribute, 79
- GlassyBuckets
 - Afterwarp.ToneMappingBloom, 434
- GlassyColor
 - Afterwarp, 47
- GlassyComposite
 - Afterwarp, 47
- GlassyCounts
 - Afterwarp, 47
- GlassyFast
 - Afterwarp.TextureCabinet.Attribute, 79
- GlassyFog
 - Afterwarp, 47
 - Afterwarp.TextureCabinet.Attribute, 80
- GlassyFrosted
 - Afterwarp.TextureCabinet.Attribute, 80
- Global
 - Afterwarp, 38
- GlobalModel
 - Afterwarp, 38
- GlobalVolume
 - Afterwarp, 38
- GlowIntensity
 - Afterwarp.SelectionHighlightParameters, 365
- GradientX
 - Afterwarp.ColorRect, 120
- GradientY
 - Afterwarp.ColorRect, 121
- Grapher
 - Afterwarp.Grapher, 164
- Grayscale
 - Afterwarp.SelectionHighlight, 363
- Greater
 - Afterwarp, 30
- GreaterEqual
 - Afterwarp, 30
- Green
 - Afterwarp.FloatColor, 143
 - Afterwarp.FloatColorRGB, 147
- Ground
 - Afterwarp, 32
- GroundPlane
 - Afterwarp.FogParameters, 150
- Group
 - Afterwarp.SceneMeshLatch, 342
- HalfFloat
 - Afterwarp, 32
- Handle
 - Afterwarp.CustomObject, 131
- HardwareFiltering
 - Afterwarp.GaussianBlur, 161
- HasAlphaChannel
 - Afterwarp.Surface, 394
- Heavy
 - Afterwarp, 33, 35
- Height
 - Afterwarp.ImageRegion, 173
 - Afterwarp.Rect, 297
 - Afterwarp.RectF, 302
 - Afterwarp.Surface, 394
 - Afterwarp.SwapChain, 398
 - Afterwarp.TextureParameters, 427
- Help
 - Afterwarp, 26, 37
- Hexagon
 - Afterwarp.Canvas, 98
- HexagonDelta
 - Afterwarp.Canvas, 107
- Hierarchyless
 - Afterwarp.ObjectModel.Attribute, 74
- High
 - Afterwarp, 48
- Highlight
 - Afterwarp, 44
 - Afterwarp.Canvas, 98
 - Afterwarp.ObjectModel, 234
- HighlightMask
 - Afterwarp, 44
- Home
 - Afterwarp, 35
- Horizontal
 - Afterwarp.Margins, 184
- Hull
 - Afterwarp, 45
- I8
 - Afterwarp, 41
- IconBig
 - Afterwarp.ApplicationConfiguration, 69
- IconSmall
 - Afterwarp.ApplicationConfiguration, 69
- IconTitle
 - Afterwarp.Application, 65

- ID
 - Afterwarp.ObjectModel, [234](#)
- ImageAtlas
 - Afterwarp.ImageAtlas, [170](#)
- ImageRegion
 - Afterwarp.ImageRegion, [173](#)
- Increment
 - Afterwarp, [45](#)
- IncrementWarp
 - Afterwarp, [45](#)
- Indeterminate
 - Afterwarp.MeshMetaTagType, [213](#)
- Index
 - Afterwarp, [29](#)
 - Afterwarp.ImageRegion, [173](#)
 - Afterwarp.ProgramElement, [280](#)
 - Afterwarp.ProgramVariable, [282](#)
- IndexBuffer
 - Afterwarp.MeshModel, [217](#)
- IndexCount
 - Afterwarp.CanvasBuffer, [111](#)
 - Afterwarp.MeshBuffer, [201](#)
 - Afterwarp.MeshMetaTagPortion, [209](#)
 - Afterwarp.MeshModel, [217](#)
 - Afterwarp.SceneMeshMaterialRange, [353](#)
- IndexStart
 - Afterwarp.SceneMeshMaterialRange, [353](#)
- Indices
 - Afterwarp.CanvasBuffer, [111](#)
 - Afterwarp.MeshBuffer, [201](#)
 - Afterwarp.SceneLights, [330](#)
- IndicesPtr
 - Afterwarp.MeshBuffer, [201](#)
- Indirect
 - Afterwarp, [49](#)
- Inflate
 - Afterwarp.Rect, [296](#)
 - Afterwarp.RectF, [301](#)
- Insert
 - Afterwarp, [35](#)
- Instance
 - Afterwarp.ApplicationConfiguration, [70](#)
- Instances
 - Afterwarp.Scene, [318](#)
- InstancesCount
 - Afterwarp.Scene, [320](#)
- Instancing
 - Afterwarp.Scene.Attribute, [75](#)
- Int
 - Afterwarp, [32](#)
- Int64
 - Afterwarp, [50](#)
- Integer
 - Afterwarp, [50](#)
- Intensity
 - Afterwarp.SceneLight, [325](#)
- Interleave
 - Afterwarp.TextRenderModifiers, [412](#)
- Intersect
 - Afterwarp.MeshVoxel, [220](#)
 - Afterwarp.Rect, [296](#)
 - Afterwarp.RectF, [301](#)
- IntersectCubeVolume
 - Afterwarp.Ray, [292](#)
- IntersectedObjects
 - Afterwarp.ObjectModelView, [246](#)
- IntersectedRays
 - Afterwarp.ObjectModelView, [246](#)
- IntersectPlane
 - Afterwarp.Ray, [292](#)
- IntersectTriangle
 - Afterwarp.Ray, [293](#)
- Invalidate
 - Afterwarp.Application, [63](#)
 - Afterwarp.ObjectModel, [233](#)
 - Afterwarp.ObjectModelView, [245](#)
 - Afterwarp.Widget, [457](#)
- InvalidateWaypoints
 - Afterwarp.SceneMeshLatches, [346](#)
- InvConstantAlpha
 - Afterwarp, [27](#)
- InvConstantColor
 - Afterwarp, [27](#)
- InvDestAlpha
 - Afterwarp, [27](#)
- InvDestColor
 - Afterwarp, [27](#)
- InverseMultiply
 - Afterwarp, [28](#)
- Invert
 - Afterwarp, [45](#)
 - Afterwarp.Surface, [391](#)
- InvertColor
 - Afterwarp.Utility, [440](#)
- InvertIndexOrder
 - Afterwarp.MeshBuffer, [194](#)
- InvertNormals
 - Afterwarp.MeshBuffer, [195](#)
- InvokeEvent
 - Afterwarp.Widget, [457](#)
- InvSourceAlpha
 - Afterwarp, [27](#)
- InvSourceAlpha1
 - Afterwarp, [27](#)
- InvSourceColor
 - Afterwarp, [27](#)
- InvSourceColor1
 - Afterwarp, [27](#)
- InvSubtract
 - Afterwarp, [28](#)
- iOS
 - Afterwarp, [31](#)
- Italic
 - Afterwarp, [33](#)
- Join
 - Afterwarp.Rect, [296](#)

- Afterwarp.RectF, [301](#)
- JoinDuplicateVertices
 - Afterwarp.MeshBuffer, [195](#)
- Joint
 - Afterwarp.SceneMeshLatchType, [348](#)
- KawaseBlur
 - Afterwarp.KawaseBlur, [179](#)
- Keep
 - Afterwarp, [45](#)
- KernelBias
 - Afterwarp.AmbientOcclusionParameters, [58](#)
- Key
 - Afterwarp, [35](#)
- KeyA
 - Afterwarp, [35](#)
- KeyB
 - Afterwarp, [35](#)
- KeyboardCallback
 - Afterwarp.Application, [63](#)
- KeyC
 - Afterwarp, [35](#)
- KeyD
 - Afterwarp, [36](#)
- KeyE
 - Afterwarp, [36](#)
- KeyEvent
 - Afterwarp, [37](#)
- KeyF
 - Afterwarp, [36](#)
- KeyG
 - Afterwarp, [36](#)
- KeyH
 - Afterwarp, [36](#)
- KeyI
 - Afterwarp, [36](#)
- KeyJ
 - Afterwarp, [36](#)
- KeyK
 - Afterwarp, [36](#)
- KeyL
 - Afterwarp, [36](#)
- KeyM
 - Afterwarp, [36](#)
- KeyN
 - Afterwarp, [36](#)
- KeyO
 - Afterwarp, [36](#)
- KeyP
 - Afterwarp, [36](#)
- KeyQ
 - Afterwarp, [36](#)
- KeyR
 - Afterwarp, [36](#)
- KeyS
 - Afterwarp, [36](#)
- KeyT
 - Afterwarp, [36](#)
- KeyU
 - Afterwarp, [36](#)
- KeyV
 - Afterwarp, [36](#)
- KeyW
 - Afterwarp, [36](#)
- KeyX
 - Afterwarp, [36](#)
- KeyY
 - Afterwarp, [36](#)
- KeyZ
 - Afterwarp, [36](#)
- L16
 - Afterwarp, [41](#)
- L8
 - Afterwarp, [41](#)
- LA4
 - Afterwarp, [41](#)
- LA8
 - Afterwarp, [41](#)
- Latches
 - Afterwarp.SceneMesh, [334](#)
- Latency
 - Afterwarp.Timer, [431](#)
- Layer
 - Afterwarp.ComputeBindTextureFormat, [124](#)
- Layers
 - Afterwarp.ObjectModel, [235](#)
 - Afterwarp.ObjectModelView, [246](#)
 - Afterwarp.TextureParameters, [427](#)
- Leave
 - Afterwarp, [39](#)
- Left
 - Afterwarp, [35](#), [38](#), [49](#)
 - Afterwarp.ImageRegion, [173](#)
 - Afterwarp.Margins, [183](#)
 - Afterwarp.Rect, [297](#)
 - Afterwarp.RectF, [302](#)
- Legacy
 - Afterwarp.Device.Attribute, [72](#)
- LegacyBits
 - Afterwarp.Device, [134](#)
- Length
 - Afterwarp.PathElement, [267](#)
 - Afterwarp.Point, [272](#)
- Less
 - Afterwarp, [30](#)
- LessEqual
 - Afterwarp, [30](#)
- Light
 - Afterwarp, [35](#)
- Lighting
 - Afterwarp.SceneMeshMaterialShading, [359](#)
- Lights
 - Afterwarp.Scene, [320](#)
- LimitCusp
 - Afterwarp.PathCommand, [265](#)
- LimitedExtensions
 - Afterwarp.Device.Attribute, [73](#)

- Line
 - Afterwarp.Canvas, 98
 - Afterwarp.Grapher, 165
- LineAntialias
 - Afterwarp.RenderingState.State, 383
- Linear
 - Afterwarp, 32, 48
- LinearDepths
 - Afterwarp, 47
 - Afterwarp.Scene.Attribute, 76
 - Afterwarp.TextureCabinet.Attribute, 80
- LineCaps
 - Afterwarp, 37
- LineCircle
 - Afterwarp.Canvas, 99
- LineEllipse
 - Afterwarp.Canvas, 99
- Linefeed
 - Afterwarp, 35
- LineHexagon
 - Afterwarp.Canvas, 99
- LineQuad
 - Afterwarp.Canvas, 99
- Lines
 - Afterwarp, 43
 - Afterwarp.Canvas, 100
 - Afterwarp.Grapher, 166
- LinesAdjacency
 - Afterwarp, 43
- LineStrip
 - Afterwarp, 43
- LineStripAdjacency
 - Afterwarp, 43
- LineTo
 - Afterwarp.PathBuilder, 262
 - Afterwarp.PathCommand, 265
- LineTriangle
 - Afterwarp.Canvas, 100
- LinkSelect
 - Afterwarp, 26
- Linux
 - Afterwarp, 31
- LoadFromFile
 - Afterwarp.MeshBuffer, 195
 - Afterwarp.MeshVoxel, 220
 - Afterwarp.SceneMeshLatches, 346
 - Afterwarp.Texture, 419
- LoadFromFileEx
 - Afterwarp.MeshBuffer, 195
- LoadFromFileInMemory
 - Afterwarp.MeshVoxel, 220
 - Afterwarp.SceneMeshLatches, 346
- LoadSaveFeedback
 - Afterwarp.MeshBuffer, 195
- Local
 - Afterwarp, 38
- LocalModel
 - Afterwarp, 38
- LocalToScreen
 - Afterwarp.Widget, 457
- LocalVolume
 - Afterwarp, 38
- Location
 - Afterwarp.WidgetProperty, 469
- Low
 - Afterwarp, 48
- MakeColor
 - Afterwarp.Utility, 440
- MakeColorAlpha
 - Afterwarp.Utility, 441
- MakeColorGray
 - Afterwarp.Utility, 441, 442
- MakeColorRGB
 - Afterwarp.Utility, 442
- MakeColorWithGray
 - Afterwarp.Utility, 443
- MakeRegions
 - Afterwarp.ImageAtlas, 170
- MakeSignedDistanceField
 - Afterwarp.Surface, 391
- Manager
 - Afterwarp.Widget, 459
- Margins
 - Afterwarp, 50
 - Afterwarp.Margins, 182, 183
 - Afterwarp.Widget, 459
- Material
 - Afterwarp.ObjectModel, 235
 - Afterwarp.OceanSimulation, 253
 - Afterwarp.Scene, 320
- MaterialIgnore
 - Afterwarp.AutoDrawOption, 81
- Materials
 - Afterwarp.SceneMesh, 334
- MaterialSkipOpaque
 - Afterwarp.AutoDrawOption, 81
- MaterialSkipTransparent
 - Afterwarp.AutoDrawOption, 81
- MaterialTextureMissing
 - Afterwarp.MeshLoadingOption, 203
- MaterialTransparency
 - Afterwarp.AutoDrawOption, 81
- MaterialVertexColor
 - Afterwarp.MeshLoadingOption, 203
- MaterialVertexColorPreferred
 - Afterwarp.MeshLoadingOption, 203
- MatrixLookAt4x4
 - Afterwarp.Utility, 444
- MatrixOrthographic4x4
 - Afterwarp.Utility, 444
- MatrixPerspective4x4
 - Afterwarp.Utility, 444
- Maximized
 - Afterwarp, 27
- Maximum
 - Afterwarp, 28

- MaxLOD
 - Afterwarp.SamplerState, [314](#)
- MediaNextTrack
 - Afterwarp, [37](#)
- MediaPlayPause
 - Afterwarp, [37](#)
- MediaPrevTrack
 - Afterwarp, [37](#)
- MediaStop
 - Afterwarp, [37](#)
- Medium
 - Afterwarp, [35](#), [48](#)
- MemoryBarrier
 - Afterwarp.Device, [134](#)
- Mesh
 - Afterwarp.ObjectModel, [235](#)
 - Afterwarp.SceneMeshes, [339](#)
- MeshAlign
 - Afterwarp, [38](#)
- MeshAligns
 - Afterwarp.MeshAligns, [186](#)
- MeshBuffer
 - Afterwarp.MeshBuffer, [191](#)
- Meshes
 - Afterwarp, [39](#)
 - Afterwarp.ObjectModels, [242](#)
- MeshMetaTagPortion
 - Afterwarp.MeshMetaTagPortion, [208](#)
- MeshMetaTags
 - Afterwarp.MeshMetaTags, [211](#)
- MeshModel
 - Afterwarp.MeshModel, [216](#)
- MeshName
 - Afterwarp.ObjectModel, [235](#)
- MeshVoxel
 - Afterwarp.MeshVoxel, [219](#)
- Metal
 - Afterwarp, [31](#)
- Middle
 - Afterwarp, [38](#), [46](#)
- MinimalSize
 - Afterwarp.Application, [66](#)
- Minimize
 - Afterwarp, [26](#)
- Minimized
 - Afterwarp, [27](#)
- Minimum
 - Afterwarp, [28](#)
- MinLOD
 - Afterwarp.SamplerState, [314](#)
- Minnaert
 - Afterwarp, [46](#)
- MipLevel
 - Afterwarp.ComputeBindTextureFormat, [124](#)
- MipMapping
 - Afterwarp.Texture.Attribute, [77](#)
- Mirror
 - Afterwarp, [47](#)
 - Afterwarp.ImageRegion, [174](#)
 - Afterwarp.Quad, [289](#)
 - Afterwarp.Surface, [391](#)
- MirrorOnce
 - Afterwarp, [47](#)
- Miter
 - Afterwarp, [39](#)
- MiterBevel
 - Afterwarp, [39](#)
- Model
 - Afterwarp.MeshBuffer, [196](#)
 - Afterwarp.SceneMesh, [334](#)
- Modeling
 - Afterwarp.Scene.Attribute, [76](#)
- ModelTransform
 - Afterwarp, [38](#)
- MouseButton
 - Afterwarp, [38](#)
- MouseCallback
 - Afterwarp.Application, [63](#)
- MouseEvent
 - Afterwarp, [39](#)
- MouseInputCaptured
 - Afterwarp.Application, [66](#)
- Move
 - Afterwarp, [26](#), [39](#)
- Movement
 - Afterwarp, [29](#)
- MoveNext
 - Afterwarp.ObjectModels.Iterator, [176](#)
 - Afterwarp.Widget.Iterator, [177](#)
- MoveTo
 - Afterwarp.PathBuilder, [263](#)
 - Afterwarp.PathCommand, [265](#)
- Multiply
 - Afterwarp, [28](#), [37](#)
- MultiplyColors
 - Afterwarp.Utility, [444](#)
- Multisamples
 - Afterwarp.SwapChain, [398](#)
 - Afterwarp.TextureParameters, [428](#)
 - Afterwarp.WidgetManager, [466](#)
- Multisampling
 - Afterwarp.RenderingState.State, [383](#)
- Name
 - Afterwarp.MeshMetaTag, [207](#)
 - Afterwarp.ObjectModel, [235](#)
 - Afterwarp.ProgramElement, [280](#)
 - Afterwarp.ProgramVariable, [283](#)
 - Afterwarp.SceneMesh, [334](#)
 - Afterwarp.SceneMeshLatch, [342](#)
 - Afterwarp.SceneMeshMaterial, [352](#)
 - Afterwarp.SceneMeshMaterials, [356](#)
 - Afterwarp.VertexElement, [449](#)
 - Afterwarp.Widget, [460](#)
 - Afterwarp.Widget.PropertyValue, [284](#)
 - Afterwarp.WidgetProperty, [469](#)
- NameBytes

- Afterwarp.ProgramElement, 280
- Afterwarp.ProgramVariable, 282
- Afterwarp.SceneMeshLatch, 342
- Afterwarp.VertexElement, 449
- Afterwarp.WidgetProperty, 469
- Nearest
 - Afterwarp, 48
- Negative
 - Afterwarp, 38
- Never
 - Afterwarp, 30
- NextSlice
 - Afterwarp.Timer, 430
- None
 - Afterwarp, 33, 38, 39, 46, 48–50
- NonPremultiplied
 - Afterwarp, 25
- NonViewable
 - Afterwarp.ObjectModel.Attribute, 74
- Normal
 - Afterwarp, 28, 33, 35, 49
 - Afterwarp.MeshBuffer.VertexEntry, 450
- NormalMap
 - Afterwarp, 44
- Normals
 - Afterwarp, 43, 47
 - Afterwarp.Scene.Attribute, 76
- NoSuperSample
 - Afterwarp, 45
- NotAllowed
 - Afterwarp, 26
- NotEqual
 - Afterwarp, 30
- Null
 - Afterwarp, 35
- NumLock
 - Afterwarp, 35
- Numpad0
 - Afterwarp, 36
- Numpad1
 - Afterwarp, 36
- Numpad2
 - Afterwarp, 36
- Numpad3
 - Afterwarp, 36
- Numpad4
 - Afterwarp, 37
- Numpad5
 - Afterwarp, 37
- Numpad6
 - Afterwarp, 37
- Numpad7
 - Afterwarp, 37
- Numpad8
 - Afterwarp, 37
- Numpad9
 - Afterwarp, 37
- Object
 - Afterwarp.MeshMetaTagType, 213
 - Afterwarp.ObjectModels, 241
 - Afterwarp.ObjectModelView, 245
 - ObjectCount
 - Afterwarp.ObjectModelView, 246
 - ObjectDisableInstancing
 - Afterwarp.AutoDrawOption, 81
 - ObjectHighlighted
 - Afterwarp.AutoDrawOption, 81
 - ObjectMaterial
 - Afterwarp.ObjectMaterial, 223
 - ObjectMaterials
 - Afterwarp.ObjectMaterials, 227
 - ObjectModels
 - Afterwarp.ObjectModels, 239
 - ObjectModelViewCompare
 - Afterwarp, 39
 - Objects
 - Afterwarp, 39
 - ObjectSkipHavingMaterials
 - Afterwarp.AutoDrawOption, 81
 - ObjectSkipNonTransparent
 - Afterwarp.AutoDrawOption, 82
 - ObjectSkipNotHavingMaterials
 - Afterwarp.AutoDrawOption, 82
 - ObjectSkipTransparent
 - Afterwarp.AutoDrawOption, 82
 - ObjectsNotCulled
 - Afterwarp.ObjectModelView, 246
 - Oblique
 - Afterwarp, 33
 - Occlusion
 - Afterwarp, 47
 - Afterwarp.ObjectMaterial, 224
 - Afterwarp.ParallaxMappingParameters, 257
 - OceanMaterial
 - Afterwarp.OceanMaterial, 248
 - OceanSimulation
 - Afterwarp.OceanSimulation, 252
 - OceanWavesParameters
 - Afterwarp.OceanWavesParameters, 255
 - Offset
 - Afterwarp.KawaseBlur, 180
 - Afterwarp.ProgramVariable, 282
 - Afterwarp.Quad, 286
 - Afterwarp.Rect, 296
 - Afterwarp.RectF, 301, 302
 - Afterwarp.VertexElement, 449
 - OnCreate
 - Afterwarp.Application, 67
 - OnDestroy
 - Afterwarp.Application, 67
 - One
 - Afterwarp, 27
 - Afterwarp.Point, 273
 - OnHook
 - Afterwarp.Application, 67
 - OnIdle

- Afterwarp.Application, [67](#)
- OnKeyboard
 - Afterwarp.Application, [67](#)
- OnMouse
 - Afterwarp.Application, [67](#)
- OnRender
 - Afterwarp.Application, [67](#)
- OnResize
 - Afterwarp.Application, [67](#)
- OnStatus
 - Afterwarp.Application, [68](#)
- Op
 - Afterwarp.RenderingState.Blend, [83](#)
- Opacity
 - Afterwarp.FogParameters, [150](#)
- OpenFile
 - Afterwarp, [32](#)
- OpenGL
 - Afterwarp, [31](#)
- OpenGL_ES
 - Afterwarp, [31](#)
- operator!=
 - Afterwarp.Point, [270](#)
 - Afterwarp.Rect, [296](#)
- operator+
 - Afterwarp.FloatColor, [142](#)
 - Afterwarp.FloatColorRGB, [146](#)
 - Afterwarp.Point, [271](#)
- operator-
 - Afterwarp.FloatColor, [142](#)
 - Afterwarp.FloatColorRGB, [146](#)
 - Afterwarp.Point, [271](#)
- operator/
 - Afterwarp.FloatColor, [142](#)
 - Afterwarp.FloatColorRGB, [146](#)
 - Afterwarp.Point, [271](#)
- operator==
 - Afterwarp.Point, [272](#)
 - Afterwarp.Rect, [296](#)
- operator*
 - Afterwarp.FloatColor, [142](#)
 - Afterwarp.FloatColorRGB, [146](#)
 - Afterwarp.Point, [271](#)
- Ordered
 - Afterwarp, [39](#)
- OrderIndex
 - Afterwarp.ObjectModel, [235](#)
- OrenNayer
 - Afterwarp, [46](#)
- Orientation
 - Afterwarp.SceneMeshLatch, [342](#)
- Origin
 - Afterwarp, [38](#)
 - Afterwarp.Ray, [293](#)
- OSX
 - Afterwarp, [31](#)
- Outline
 - Afterwarp.Canvas.Attribute, [71](#)
- OutlineColor
 - Afterwarp.SelectionHighlightParameters, [365](#)
- OutlineDistanceMaxSDF
 - Afterwarp.SignedDistanceField, [377](#)
- OutlineDistanceMinSDF
 - Afterwarp.SignedDistanceField, [377](#)
- OutlineOffsetSDF
 - Afterwarp.SignedDistanceField, [377](#)
- OutlineStart
 - Afterwarp.SelectionHighlightParameters, [365](#)
- Overlaps
 - Afterwarp.Rect, [297](#)
 - Afterwarp.RectF, [302](#)
- Owner
 - Afterwarp.ObjectModel, [235](#)
 - Afterwarp.ObjectModelView, [247](#)
- PackRegion
 - Afterwarp.ImageAtlas, [170](#)
- PackSurface
 - Afterwarp.ImageAtlas, [171](#)
- Padding
 - Afterwarp.ShadowCastingAtlas, [373](#)
- PageDown
 - Afterwarp, [35](#)
- PageUp
 - Afterwarp, [35](#)
- Parallax
 - Afterwarp, [43](#)
- ParallaxMap
 - Afterwarp, [44](#)
- ParallaxMapping
 - Afterwarp.Scene, [320](#)
- ParallaxMappingParameters
 - Afterwarp.ParallaxMappingParameters, [257](#)
- Parameters
 - Afterwarp.OceanSimulation, [253](#)
 - Afterwarp.SelectionHighlight, [363](#)
 - Afterwarp.ShadowCastingAtlas, [373](#)
 - Afterwarp.SpatialFog, [381](#)
 - Afterwarp.TextRenderer, [411](#)
 - Afterwarp.Texture, [420](#)
- Parent
 - Afterwarp.MeshMetaTag, [207](#)
 - Afterwarp.ObjectModel, [236](#)
 - Afterwarp.Widget, [460](#)
- ParentZone
 - Afterwarp.Widget, [460](#)
- Passes
 - Afterwarp.KawaseBlur, [180](#)
- Patches
 - Afterwarp, [43](#)
- PathBroker
 - Afterwarp.PathBroker, [259](#)
- PathBuilder
 - Afterwarp.PathBuilder, [261](#)
- PathElement
 - Afterwarp.PathElement, [267](#)
- PathJoint

- Afterwarp, [39](#)
- Pause
 - Afterwarp, [35](#)
- Payload
 - Afterwarp.ObjectMaterial, [224](#)
 - Afterwarp.ObjectModel, [236](#)
 - Afterwarp.ObjectModels, [241](#)
 - Afterwarp.SceneLight, [325](#)
 - Afterwarp.SceneMesh, [335](#)
 - Afterwarp.SceneMeshes, [339](#)
 - Afterwarp.Widget, [460](#)
- PayloadExists
 - Afterwarp.ObjectModels, [241](#)
 - Afterwarp.SceneMeshes, [340](#)
- Performance
 - Afterwarp, [30](#)
- PerSampleShading
 - Afterwarp.DeviceBehavior, [137](#)
 - Afterwarp.RenderingState.State, [384](#)
- Phong
 - Afterwarp, [46](#)
- Pitch
 - Afterwarp.Buffer, [87](#)
 - Afterwarp.Surface, [394](#)
- Pixel
 - Afterwarp.Canvas, [100](#)
- PixelFormat
 - Afterwarp, [39](#)
- Pixels
 - Afterwarp.Canvas, [100](#)
- Plane
 - Afterwarp.MeshBuffer, [196](#)
 - Afterwarp.OceanSimulation, [253](#)
- Platform
 - Afterwarp.Device, [135](#)
- PlatformHandle
 - Afterwarp.Buffer, [87](#)
 - Afterwarp.Texture, [420](#)
- Point
 - Afterwarp, [50](#)
 - Afterwarp.Grapher, [166](#)
 - Afterwarp.Point, [269](#)
- Points
 - Afterwarp, [43](#)
 - Afterwarp.Grapher, [166](#)
- PointShape
 - Afterwarp, [41](#)
- PortionAdd
 - Afterwarp.MeshMetaTag, [206](#)
- PortionCount
 - Afterwarp.MeshMetaTag, [207](#)
- PortionErase
 - Afterwarp.MeshMetaTag, [206](#)
- PortionGet
 - Afterwarp.MeshMetaTag, [206](#)
- PortionsClear
 - Afterwarp.MeshMetaTag, [206](#)
- PortionsCopy
 - Afterwarp.MeshMetaTag, [206](#)
- Position
 - Afterwarp.ApplicationConfiguration, [70](#)
 - Afterwarp.MeshBuffer.VertexEntry, [450](#)
 - Afterwarp.ObjectModel, [236](#)
 - Afterwarp.SceneLight, [325](#)
 - Afterwarp.SceneMeshLatch, [342](#)
 - Afterwarp.ShadowCaster, [369](#)
 - Afterwarp.TextEntryRect, [400](#)
- PositionMax
 - Afterwarp.CameraConstraints, [88](#)
- PositionMin
 - Afterwarp.CameraConstraints, [88](#)
- Positive
 - Afterwarp, [38](#)
- PostDepthCoverage
 - Afterwarp.DeviceBehavior, [137](#)
- PostUnbind
 - Afterwarp.AutoDrawOption, [82](#)
- Premultiplied
 - Afterwarp, [25](#)
- PremultipliedAlpha
 - Afterwarp.Surface, [395](#)
 - Afterwarp.Texture.Attribute, [77](#)
- PremultiplyAlpha
 - Afterwarp.Surface, [391](#)
 - Afterwarp.Utility, [444](#)
- Prepare
 - Afterwarp.Scene, [318](#)
 - Afterwarp.TextModeller, [405](#)
- Present
 - Afterwarp.TextureCabinet, [424](#)
 - Afterwarp.WidgetManager, [465](#)
- Pressed
 - Afterwarp, [37](#)
- PrimitiveTopology
 - Afterwarp, [43](#)
- Print
 - Afterwarp, [37](#)
- PrintScreen
 - Afterwarp, [35](#)
- Program
 - Afterwarp.Program, [276](#)
 - Afterwarp.Scene, [321](#)
- ProgramElement
 - Afterwarp.ProgramElement, [279](#)
- ProgramVariable
 - Afterwarp.ProgramVariable, [281](#)
- Projected
 - Afterwarp, [29](#)
 - Afterwarp.Canvas.Attribute, [71](#)
- Projection
 - Afterwarp.ObjectModelView, [247](#)
 - Afterwarp.OceanSimulation, [253](#)
 - Afterwarp.Scene, [321](#)
 - Afterwarp.SpatialFog, [381](#)
- PropertyCount
 - Afterwarp.Widget, [460](#)

- PropertyValue
 - Afterwarp.Widget.PropertyValue, [283](#)
- Proprietary
 - Afterwarp, [31](#)
- Provider
 - Afterwarp.TextModeller, [406](#)
- Quad
 - Afterwarp.Canvas, [100](#), [101](#)
 - Afterwarp.Quad, [285](#), [286](#)
- QuadCurveTo
 - Afterwarp.PathBuilder, [263](#)
 - Afterwarp.PathCommand, [265](#)
- QuadImage
 - Afterwarp.Canvas, [101](#)
- QuadRegion
 - Afterwarp.Canvas, [101](#)
- Quality
 - Afterwarp, [30](#)
- R16
 - Afterwarp, [40](#)
- R16F
 - Afterwarp, [40](#)
- R16I
 - Afterwarp, [40](#)
- R16S
 - Afterwarp, [40](#)
- R16U
 - Afterwarp, [40](#)
- R32F
 - Afterwarp, [40](#)
- R32I
 - Afterwarp, [40](#)
- R32U
 - Afterwarp, [40](#)
- R5G6B5
 - Afterwarp, [30](#)
- R8
 - Afterwarp, [40](#)
- R8I
 - Afterwarp, [40](#)
- R8S
 - Afterwarp, [40](#)
- R8U
 - Afterwarp, [40](#)
- R_BC
 - Afterwarp, [41](#)
- Radius
 - Afterwarp.AmbientOcclusionParameters, [58](#)
- RandomSequence
 - Afterwarp.RandomSequence, [290](#)
- RangeCount
 - Afterwarp.SceneMeshMaterial, [352](#)
- Ranged
 - Afterwarp.RandomSequence, [290](#)
- Raw
 - Afterwarp.RandomSequence, [291](#)
- Raw64
 - Afterwarp.RandomSequence, [291](#)
- Ray
 - Afterwarp.Ray, [292](#)
- Read
 - Afterwarp, [30](#)
- ReadTextFromClipboard
 - Afterwarp.Application, [64](#)
- ReadWrite
 - Afterwarp, [30](#)
- Real
 - Afterwarp, [50](#)
- Rect
 - Afterwarp, [50](#)
 - Afterwarp.Rect, [295](#)
 - Afterwarp.Widget, [460](#)
- Rectangle
 - Afterwarp.PathBuilder, [263](#)
 - Afterwarp.TextEntryRect, [400](#)
- RectF
 - Afterwarp.RectF, [300](#)
- RectWithHole
 - Afterwarp.Canvas, [101](#)
- Red
 - Afterwarp.FloatColor, [143](#)
 - Afterwarp.FloatColorRGB, [147](#)
- Region
 - Afterwarp.ImageAtlas, [171](#)
- RegionCount
 - Afterwarp.ImageAtlas, [172](#)
- Relative
 - Afterwarp.PathCommand, [265](#)
- Released
 - Afterwarp, [37](#)
- ReleaseMouseInput
 - Afterwarp.Application, [64](#)
- ReleaseTextures
 - Afterwarp.SceneMeshMaterial, [351](#)
- RemoveRegion
 - Afterwarp.ImageAtlas, [171](#)
- RemoveTexture
 - Afterwarp.ImageAtlas, [171](#)
- Render
 - Afterwarp.OceanSimulation, [252](#)
 - Afterwarp.TextModeller, [406](#)
- Renderable
 - Afterwarp.MeshModel, [217](#)
- RenderDebug
 - Afterwarp.SceneLights, [329](#)
- Rendering
 - Afterwarp.Scene, [321](#)
 - Afterwarp.SelectionHighlight, [363](#)
 - Afterwarp.ShadowCaster, [369](#)
 - Afterwarp.TextureCabinet, [425](#)
- RenderingState
 - Afterwarp.Device, [135](#)
 - Afterwarp.RenderingState, [306](#)
- Replace
 - Afterwarp, [45](#)

- Reserved
 - Afterwarp.SceneLight, 325
 - Afterwarp.SceneMeshMaterialShading, 359
 - Afterwarp.WidgetProperty, 469
- Reset
 - Afterwarp.Canvas, 101
 - Afterwarp.Grapher, 166
 - Afterwarp.ObjectModels.Iterator, 176
 - Afterwarp.PathBroker, 260
 - Afterwarp.TextModeller, 406
 - Afterwarp.Timer, 430
 - Afterwarp.Widget.Iterator, 177
- ResetAlpha
 - Afterwarp.Surface, 392
- ResetBindings
 - Afterwarp.ComputeProgram, 127
 - Afterwarp.Program, 277
- ResetCache
 - Afterwarp.Device, 134
 - Afterwarp.Program, 277
 - Afterwarp.Scene, 319
 - Afterwarp.Texture, 419
- ResetSamplerState
 - Afterwarp.Canvas, 102
- ResetTextures
 - Afterwarp.AutoDrawOption, 82
- Resize
 - Afterwarp.Surface, 392
 - Afterwarp.SwapChain, 398
- Resize1
 - Afterwarp, 26
- Resize2
 - Afterwarp, 26
- ResizeBottom
 - Afterwarp, 26
- ResizeBottomLeft
 - Afterwarp, 26
- ResizeBottomRight
 - Afterwarp, 26
- ResizeHoriz
 - Afterwarp, 26
- ResizeLeft
 - Afterwarp, 26
- ResizeRight
 - Afterwarp, 26
- ResizeTop
 - Afterwarp, 26
- ResizeTopLeft
 - Afterwarp, 26
- ResizeTopRight
 - Afterwarp, 26
- ResizeVert
 - Afterwarp, 26
- Resolution
 - Afterwarp.OceanWavesParameters, 255
- Resolve
 - Afterwarp.TextureCabinet, 424
- Restore
 - Afterwarp, 26
- Retrieve
 - Afterwarp.Buffer, 86
 - Afterwarp.Texture, 419
- RetrievePayload< T >
 - Afterwarp.Utility, 445
- Return
 - Afterwarp, 35
- RG11B10F
 - Afterwarp, 40
- RG16
 - Afterwarp, 40
- RG16F
 - Afterwarp, 40
- RG16I
 - Afterwarp, 40
- RG16S
 - Afterwarp, 40
- RG16U
 - Afterwarp, 40
- RG32F
 - Afterwarp, 40
- RG32I
 - Afterwarp, 40
- RG32U
 - Afterwarp, 40
- RG3B2
 - Afterwarp, 30
- RG8
 - Afterwarp, 40
- RG8I
 - Afterwarp, 40
- RG8S
 - Afterwarp, 40
- RG8U
 - Afterwarp, 40
- RG_BC
 - Afterwarp, 41
- RGB
 - Afterwarp.FloatColor, 143
- RGB10A2
 - Afterwarp, 41
- RGB10A2U
 - Afterwarp, 41
- RGB4
 - Afterwarp, 30
- RGB6
 - Afterwarp, 30
- RGB8
 - Afterwarp, 30, 41
- RGB9E5F
 - Afterwarp, 40
- RGB_BC
 - Afterwarp, 41
- RGB_BC_SRGB
 - Afterwarp, 41
- RGBA16
 - Afterwarp, 40

- RGBA16F
 - Afterwarp, [40](#)
- RGBA16I
 - Afterwarp, [40](#)
- RGBA16S
 - Afterwarp, [40](#)
- RGBA16U
 - Afterwarp, [40](#)
- RGBA32F
 - Afterwarp, [40](#)
- RGBA32I
 - Afterwarp, [40](#)
- RGBA32U
 - Afterwarp, [40](#)
- RGBA8
 - Afterwarp, [40](#)
- RGBA8_SRGB
 - Afterwarp, [41](#)
- RGBA8I
 - Afterwarp, [40](#)
- RGBA8S
 - Afterwarp, [40](#)
- RGBA8U
 - Afterwarp, [40](#)
- RGBA_BC
 - Afterwarp, [41](#)
- RGBA_BC_SRGB
 - Afterwarp, [41](#)
- RGBX8
 - Afterwarp, [41](#)
- Ribbon
 - Afterwarp.Canvas, [102](#)
- Right
 - Afterwarp, [35](#), [38](#), [49](#)
 - Afterwarp.ActorCamera, [55](#)
 - Afterwarp.Margins, [183](#)
 - Afterwarp.Rect, [298](#)
 - Afterwarp.RectF, [303](#)
- Rotate
 - Afterwarp.ImageRegion, [174](#)
- Rotated
 - Afterwarp.Quad, [286](#)
- RotatedTopLeft
 - Afterwarp.Quad, [287](#)
- Rotation
 - Afterwarp, [29](#)
- RotationMax
 - Afterwarp.CameraConstraints, [88](#)
- RotationMin
 - Afterwarp.CameraConstraints, [89](#)
- Roughness
 - Afterwarp.ObjectMaterial, [224](#)
 - Afterwarp.SceneMeshMaterialShading, [359](#)
- Round
 - Afterwarp, [38](#), [39](#), [43](#)
- RoundRect
 - Afterwarp.Canvas, [102](#)
- RoundRectRegion
 - Afterwarp.Canvas, [103](#)
- RWStructured
 - Afterwarp, [29](#)
- RWStructuredBuffer
 - Afterwarp, [44](#)
- RWTyped
 - Afterwarp, [29](#)
- RWTypedBuffer
 - Afterwarp, [44](#)
- Sampler
 - Afterwarp.Sampler, [310](#)
- SamplerShadow
 - Afterwarp.OceanSimulation, [254](#)
- SamplerState
 - Afterwarp.Canvas, [108](#)
 - Afterwarp.SamplerState, [312](#)
- Samples
 - Afterwarp.AmbientOcclusionParameters, [58](#)
 - Afterwarp.GaussianBlur, [162](#)
 - Afterwarp.SelectionHighlight, [363](#)
 - Afterwarp.ShadowCastingAtlas, [373](#)
 - Afterwarp.TextureCabinet, [425](#)
- SamplesMax
 - Afterwarp.ParallaxMappingParameters, [257](#)
- SamplesMin
 - Afterwarp.ParallaxMappingParameters, [257](#)
- SaveFile
 - Afterwarp, [32](#)
- SaveToFile
 - Afterwarp.MeshBuffer, [196](#)
 - Afterwarp.MeshVoxel, [220](#)
 - Afterwarp.SceneMeshLatches, [346](#)
 - Afterwarp.Surface, [392](#)
 - Afterwarp.Texture, [419](#)
- SaveToFileEx
 - Afterwarp.MeshBuffer, [196](#)
- SaveToFileInMemory
 - Afterwarp.Surface, [392](#)
- Scale
 - Afterwarp.OceanSimulation, [254](#)
 - Afterwarp.ParallaxMappingParameters, [257](#)
 - Afterwarp.Quad, [287](#)
 - Afterwarp.SceneMesh, [335](#)
 - Afterwarp.TextRenderModifiers, [412](#)
 - Afterwarp.WidgetManager, [467](#)
- Scaled
 - Afterwarp.Quad, [287](#)
- Scattering
 - Afterwarp.FogParameters, [150](#)
- SceneLight
 - Afterwarp.SceneLight, [323](#)
- SceneLights
 - Afterwarp.SceneLights, [328](#)
- SceneMeshes
 - Afterwarp.SceneMeshes, [338](#)
- SceneMeshLatches
 - Afterwarp.SceneMeshLatches, [345](#)
- SceneMeshMaterialRange

- Afterwarp.SceneMeshMaterialRange, 353
- SceneMeshMaterials
 - Afterwarp.SceneMeshMaterials, 355
- SceneMeshMaterialShading
 - Afterwarp.SceneMeshMaterialShading, 358
- SceneMeshTexture
 - Afterwarp, 43
- SceneSamplerType
 - Afterwarp, 43
- SceneTextureType
 - Afterwarp, 44
- Scissor
 - Afterwarp.Device, 135
- ScissorClip
 - Afterwarp.RenderingState.State, 384
- Scratch
 - Afterwarp.Texture.Attribute, 78
- Screen
 - Afterwarp, 49
- ScreenToLocal
 - Afterwarp.Widget, 457
- ScrollLock
 - Afterwarp, 35
- SDF
 - Afterwarp.Canvas.Attribute, 72
- Second
 - Afterwarp.ColorPair, 118
- Sections
 - Afterwarp.OceanSimulation, 254
- Select
 - Afterwarp, 37
 - Afterwarp.ObjectModelView, 245
- Selectable
 - Afterwarp.ObjectModel.Attribute, 74
- SelectAny
 - Afterwarp.ObjectModelView, 245
- SelectionHighlight
 - Afterwarp.SelectionHighlight, 362
- SelectionHighlightParameters
 - Afterwarp.SelectionHighlightParameters, 365
- SelectionHighlightTextureType
 - Afterwarp, 44
- SemiBold
 - Afterwarp, 35
- SemiCondensed
 - Afterwarp, 33
- SemiExpanded
 - Afterwarp, 33
- SemiHeavy
 - Afterwarp, 33
- SemiLight
 - Afterwarp, 35
- SendToBack
 - Afterwarp.Widget, 458
- Separator
 - Afterwarp, 37
- SerialCode
 - Afterwarp.Library, 181
- Set
 - Afterwarp.Widget, 458
- Set< T >
 - Afterwarp.Widget, 458
- SetFontParameters
 - Afterwarp.TextModeller, 406
- SetIcons
 - Afterwarp.Application, 64
- SetLatch
 - Afterwarp.SceneMeshLatches, 346
- SetLight
 - Afterwarp.SceneLights, 329
- SetMaterial
 - Afterwarp.ObjectMaterials, 228
- SetPadding
 - Afterwarp.Widget, 458
- SetPatchVertices
 - Afterwarp.Program, 278
- SetPixel
 - Afterwarp.Surface, 392
- SetPosition
 - Afterwarp.ActorCamera, 54
- SetQuaternion
 - Afterwarp.ActorCamera, 54
- SetRange
 - Afterwarp.SceneMeshMaterial, 351
- SetRotation
 - Afterwarp.ActorCamera, 54
- SetSlice
 - Afterwarp.SceneMesh, 334
- SetTexture
 - Afterwarp.Scene, 319
 - Afterwarp.SceneMeshMaterial, 351
- SetTransform
 - Afterwarp.ObjectModel, 233
- SetVertexElements
 - Afterwarp.Scene, 319
- SetVertexElementsFromTextModeller
 - Afterwarp.Scene, 319
- Shader
 - Afterwarp.ProgramElement, 280
 - Afterwarp.ProgramVariable, 282
- ShaderElement
 - Afterwarp, 44
- ShaderType
 - Afterwarp, 45
- Shading
 - Afterwarp.SceneMeshMaterial, 352
- Shadow
 - Afterwarp, 28
- ShadowBrightness
 - Afterwarp.FontEffect, 153
- ShadowCaster
 - Afterwarp.SceneLight, 325
- ShadowCastingAtlas
 - Afterwarp.ShadowCastingAtlas, 372
- ShadowDistance
 - Afterwarp.FontEffect, 153

- ShadowMap
 - Afterwarp, [44](#)
- ShadowOpacity
 - Afterwarp.FontEffect, [154](#)
- ShadowParameters
 - Afterwarp.ShadowParameters, [375](#)
- Shadows
 - Afterwarp.ObjectMaterial, [225](#)
 - Afterwarp.OceanMaterial, [249](#)
- ShadowsCubic
 - Afterwarp.Scene.Attribute, [76](#)
- ShadowSmoothness
 - Afterwarp.FontEffect, [154](#)
- ShiftLeft
 - Afterwarp, [36](#)
- ShiftRight
 - Afterwarp, [36](#)
- Short
 - Afterwarp, [32](#)
- ShrinkFrom
 - Afterwarp.Surface, [393](#)
- Sigma
 - Afterwarp.GaussianBlur, [162](#)
- SignedDistanceField
 - Afterwarp.Canvas, [108](#)
 - Afterwarp.SignedDistanceField, [377](#)
- SignedFieldDistance
 - Afterwarp.FontEffect, [154](#)
 - Afterwarp.SignedDistanceField, [378](#)
- SignedNormIntFormatBugged
 - Afterwarp.DeviceBehavior, [137](#)
- Simple
 - Afterwarp, [39](#)
- SineAccelerate
 - Afterwarp.Utility, [445](#)
- SineCycle
 - Afterwarp.Utility, [445](#)
- SineDecelerate
 - Afterwarp.Utility, [445](#)
- SineTransform
 - Afterwarp.Utility, [445](#)
- SineTwoCycle
 - Afterwarp.Utility, [445](#)
- SinglePass
 - Afterwarp.KawaseBlur, [179](#)
 - Afterwarp.Scene.Attribute, [76](#)
- Size
 - Afterwarp.ApplicationConfiguration, [70](#)
 - Afterwarp.Buffer, [87](#)
 - Afterwarp.FontParameters, [156](#)
 - Afterwarp.ObjectModel, [236](#)
 - Afterwarp.ProgramElement, [280](#)
 - Afterwarp.ProgramVariable, [282](#)
 - Afterwarp.RectF, [303](#)
 - Afterwarp.SceneLights, [330](#)
 - Afterwarp.SceneMesh, [335](#)
 - Afterwarp.SelectionHighlight, [363](#)
 - Afterwarp.ShadowCaster, [369](#)
 - Afterwarp.ShadowCastingAtlas, [373](#)
 - Afterwarp.TextureCabinet, [425](#)
- SkewedHoriz
 - Afterwarp.Quad, [287](#)
- SkewedVert
 - Afterwarp.Quad, [288](#)
- SkippedTimeSlices
 - Afterwarp.Timer, [431](#)
- Slant
 - Afterwarp.FontParameters, [156](#)
- Sleep
 - Afterwarp, [37](#)
- Slice
 - Afterwarp.SceneMeshes, [340](#)
- SlopeDepthBias
 - Afterwarp.RenderingState, [307](#)
- Smooth
 - Afterwarp.PathCommand, [265](#)
- SmoothCurveTo
 - Afterwarp.PathBuilder, [263](#)
- SmoothQuadCurveTo
 - Afterwarp.PathBuilder, [263](#)
- Software
 - Afterwarp, [31](#)
 - Afterwarp.Device.Attribute, [73](#)
- Sort
 - Afterwarp.ObjectModelView, [245](#)
- Source
 - Afterwarp.RenderingState.Blend, [83](#)
- SourceAlpha
 - Afterwarp, [27](#)
- SourceAlpha1
 - Afterwarp, [27](#)
- SourceAlphaSat
 - Afterwarp, [27](#)
- SourceColor
 - Afterwarp, [27](#), [28](#)
- SourceColor1
 - Afterwarp, [27](#)
- SourceColorAdd
 - Afterwarp, [28](#)
- Space
 - Afterwarp, [35](#)
- SpatialFog
 - Afterwarp.SpatialFog, [380](#)
- Specular
 - Afterwarp.SceneMeshMaterialShading, [359](#)
- SpecularColor
 - Afterwarp.ObjectMaterial, [225](#)
 - Afterwarp.OceanMaterial, [249](#)
 - Afterwarp.SceneLight, [325](#)
- SpecularExponent
 - Afterwarp.ObjectMaterial, [225](#)
 - Afterwarp.OceanMaterial, [250](#)
 - Afterwarp.SceneLight, [325](#)
 - Afterwarp.SceneMeshMaterialShading, [359](#)
- Speed
 - Afterwarp.Timer, [431](#)

- Square
 - Afterwarp, [38, 43](#)
- Staging
 - Afterwarp, [28](#)
- Star
 - Afterwarp, [43](#)
- Start
 - Afterwarp, [46](#)
- State
 - Afterwarp.ApplicationConfiguration, [70](#)
 - Afterwarp.RandomSequence, [291](#)
 - Afterwarp.Sampler, [310](#)
- States
 - Afterwarp.RenderingState, [307](#)
- Static
 - Afterwarp, [28, 49](#)
- StatusCallback
 - Afterwarp.Application, [64](#)
- Stencil
 - Afterwarp.DeviceClear, [138](#)
- StencilBack
 - Afterwarp.RenderingState, [307](#)
- StencilFront
 - Afterwarp.RenderingState, [307](#)
- StencilOp
 - Afterwarp, [45](#)
- StencilRefMask
 - Afterwarp.RenderingState, [307](#)
- StencilRefValue
 - Afterwarp.RenderingState, [308](#)
- StencilState
 - Afterwarp.RenderingState.StencilState, [385](#)
- StencilTest
 - Afterwarp.RenderingState.State, [384](#)
- StencilWriteMask
 - Afterwarp.RenderingState, [308](#)
- Stop
 - Afterwarp, [29](#)
- Strength
 - Afterwarp.AmbientOcclusionParameters, [58](#)
- Stretch
 - Afterwarp.FontParameters, [157](#)
 - Afterwarp.Surface, [393](#)
- StretchBilinear
 - Afterwarp.Surface, [393](#)
- StrikeOut
 - Afterwarp.FontAttribute, [151](#)
- String
 - Afterwarp, [50](#)
- Stripes
 - Afterwarp, [49](#)
- StripGeometryNames
 - Afterwarp.MeshLoadingOption, [203](#)
- Stroke
 - Afterwarp.PathBroker, [260](#)
- Structured
 - Afterwarp, [29](#)
- StructuredBuffer
 - Afterwarp, [44](#)
- Style
 - Afterwarp.Widget, [460](#)
- Subtract
 - Afterwarp, [28, 37](#)
- SubtractColors
 - Afterwarp.Utility, [446](#)
- SuperEllipse
 - Afterwarp.MeshBuffer, [197](#)
- SuperLeft
 - Afterwarp, [36](#)
- SuperRight
 - Afterwarp, [36](#)
- SuperSample16x
 - Afterwarp, [45](#)
- SuperSample4x
 - Afterwarp, [45](#)
- SuperSampleSDF
 - Afterwarp, [45](#)
 - Afterwarp.SignedDistanceField, [378](#)
- Supertoroid
 - Afterwarp.MeshBuffer, [197](#)
- Surface
 - Afterwarp, [48](#)
 - Afterwarp.Surface, [389](#)
- SwapChain
 - Afterwarp.SwapChain, [397](#)
- SysReq
 - Afterwarp, [35](#)
- SystemTicks
 - Afterwarp.Utility, [447](#)
- Tab
 - Afterwarp, [35](#)
- Tag
 - Afterwarp.MeshMetaTags, [212](#)
- Tags
 - Afterwarp.SceneMesh, [335](#)
- TakeAway
 - Afterwarp.MeshMetaTags, [212](#)
 - Afterwarp.MeshModel, [216](#)
 - Afterwarp.MeshVoxel, [221](#)
 - Afterwarp.SceneMeshLatches, [347](#)
 - Afterwarp.SceneMeshMaterials, [356](#)
- Tangent
 - Afterwarp.MeshBuffer.VertexEntry, [451](#)
- Tape
 - Afterwarp.Canvas, [103](#)
- TechFeatureVersion
 - Afterwarp.Device, [135](#)
- Technique
 - Afterwarp.ObjectMaterial, [225](#)
 - Afterwarp.ShadowCastingAtlas, [373](#)
- TechniqueLighting
 - Afterwarp, [45](#)
- TechniqueShadows
 - Afterwarp, [46](#)
- Technology
 - Afterwarp.Device, [135](#)

- TechVersion
 - Afterwarp.Device, [135](#)
- Terminate
 - Afterwarp.Application, [64](#)
- Tessellation
 - Afterwarp.DeviceBehavior, [137](#)
- TexCoord
 - Afterwarp.MeshBuffer.VertexEntry, [451](#)
- TextAlignment
 - Afterwarp, [46](#)
- TextModeller
 - Afterwarp.TextModeller, [403](#)
- TextRects
 - Afterwarp.TextModeller, [406](#)
 - Afterwarp.TextRenderer, [411](#)
- TextRenderer
 - Afterwarp.TextRenderer, [409](#)
 - Afterwarp.WidgetManager, [467](#)
- TextRenderModifiers
 - Afterwarp.TextRenderModifiers, [412](#)
- TextSelect
 - Afterwarp, [26](#)
- TextSelectVertical
 - Afterwarp, [26](#)
- Texture
 - Afterwarp, [44](#)
 - Afterwarp.ImageAtlas, [171](#)
 - Afterwarp.SelectionHighlight, [362](#)
 - Afterwarp.ShadowCastingAtlas, [374](#)
 - Afterwarp.Texture, [415](#), [416](#)
 - Afterwarp.TextureCabinet, [426](#)
- TextureAddress
 - Afterwarp, [46](#)
- TextureCabinet
 - Afterwarp.Scene, [321](#)
 - Afterwarp.TextureCabinet, [423](#)
- TextureCabinetFilterType
 - Afterwarp, [47](#)
- TextureCabinetPass
 - Afterwarp, [47](#)
- TextureCabinetType
 - Afterwarp, [47](#)
- TextureCoordinates
 - Afterwarp.SelectionHighlight, [363](#)
- TextureCount
 - Afterwarp.ImageAtlas, [172](#)
- TexturedTriangles
 - Afterwarp.Canvas, [104](#)
- TextureFidelity
 - Afterwarp, [48](#)
- TextureFilter
 - Afterwarp, [48](#)
- TextureParameters
 - Afterwarp.TextureParameters, [427](#)
- TextureType
 - Afterwarp, [48](#)
- Texturing
 - Afterwarp.Scene.MeshMaterials, [357](#)
- TexturingCubic
 - Afterwarp.Scene.Attribute, [76](#)
- ThickLine
 - Afterwarp.Canvas, [104](#)
- ThickLineCircle
 - Afterwarp.Canvas, [104](#)
- ThickLineEllipse
 - Afterwarp.Canvas, [104](#)
- ThickLineHexagon
 - Afterwarp.Canvas, [105](#)
- ThickLineQuad
 - Afterwarp.Canvas, [105](#)
- ThickLineTriangle
 - Afterwarp.Canvas, [105](#)
- Thin
 - Afterwarp, [35](#)
- Timer
 - Afterwarp.Timer, [430](#)
- TimeSlice
 - Afterwarp.Timer, [431](#)
- Title
 - Afterwarp.Application, [66](#)
- ToColor
 - Afterwarp.FloatColor, [143](#)
 - Afterwarp.FloatColorRGB, [147](#)
- ToColorAlpha
 - Afterwarp.FloatColorRGB, [147](#)
- ToleranceAngle
 - Afterwarp.PathCommand, [266](#)
- ToneFactors
 - Afterwarp.ToneMappingBloom, [434](#)
- ToneMappingBloom
 - Afterwarp.TextureCabinet, [426](#)
 - Afterwarp.ToneMappingBloom, [433](#)
- ToneMappingCoefficients
 - Afterwarp.Scene, [321](#)
- ToneWhite
 - Afterwarp.ToneMappingBloom, [434](#)
- Top
 - Afterwarp, [49](#)
 - Afterwarp.ImageRegion, [174](#)
 - Afterwarp.Margins, [183](#)
 - Afterwarp.Rect, [297](#)
 - Afterwarp.RectF, [302](#)
- TopLeft
 - Afterwarp.ColorRect, [121](#)
 - Afterwarp.Margins, [184](#)
 - Afterwarp.Quad, [288](#)
 - Afterwarp.RectF, [303](#)
- TopRight
 - Afterwarp.ColorRect, [121](#)
 - Afterwarp.Margins, [184](#)
 - Afterwarp.Quad, [288](#)
 - Afterwarp.RectF, [303](#)
- Torus
 - Afterwarp.MeshBuffer, [197](#)
- TorusKnot
 - Afterwarp.MeshBuffer, [198](#)

- ToUI
 - Afterwarp.Utility, 446
- Transfer
 - Afterwarp.MeshBuffer, 198
- TransferEx
 - Afterwarp.MeshBuffer, 199
- Transform
 - Afterwarp.Canvas, 108
 - Afterwarp.Canvas.Attribute, 72
 - Afterwarp.Grapher, 167
 - Afterwarp.MeshBuffer, 201
 - Afterwarp.Quad, 288
 - Afterwarp.TextModeller, 407
- TransformVertices
 - Afterwarp.MeshBuffer, 199
- TranslateVirtualKey
 - Afterwarp.Application, 65
- TranslucentBlack
 - Afterwarp.ColorPair, 118
 - Afterwarp.ColorRect, 122
 - Afterwarp.FloatColor, 144
- TranslucentWhite
 - Afterwarp.ColorPair, 118
 - Afterwarp.ColorRect, 122
 - Afterwarp.FloatColor, 144
- Transparent
 - Afterwarp.ObjectModel.Attribute, 74
- Transpose
 - Afterwarp.Point, 273
- Triangle
 - Afterwarp, 43
 - Afterwarp.Canvas, 106
- TriangleFace
 - Afterwarp, 48
- TriangleRegion
 - Afterwarp.Canvas, 106
- Triangles
 - Afterwarp, 43
 - Afterwarp.Canvas, 107
- TrianglesAdjacency
 - Afterwarp, 43
- TriangleStrip
 - Afterwarp, 43
- TriangleStripAdjacency
 - Afterwarp, 43
- TrimSkippedTimeSlices
 - Afterwarp.Timer, 430
- TryAddFromFile
 - Afterwarp.SceneMeshes, 340
- TryGetWidget
 - Afterwarp.WidgetManager, 465, 466
- TryLoadFromFile
 - Afterwarp.MeshBuffer, 199
 - Afterwarp.MeshVoxel, 221
 - Afterwarp.SceneMeshLatches, 347
- TryLoadFromFileEx
 - Afterwarp.MeshBuffer, 200
- TryLoadFromFileInMemory
 - Afterwarp.MeshVoxel, 221
 - Afterwarp.SceneMeshLatches, 347
- TryMesh
 - Afterwarp.SceneMeshes, 340
- TryObject
 - Afterwarp.ObjectModels, 241
- TryPayload
 - Afterwarp.ObjectModels, 241
 - Afterwarp.SceneMeshes, 340
- TryPrepare
 - Afterwarp.Scene, 319
- TrySaveToFile
 - Afterwarp.MeshBuffer, 200
- TrySaveToFileEx
 - Afterwarp.MeshBuffer, 200
- Type
 - Afterwarp.MeshMetaTag, 207
 - Afterwarp.SceneMeshLatch, 342
 - Afterwarp.TextureParameters, 428
 - Afterwarp.Widget.PropertyValue, 284
 - Afterwarp.WidgetProperty, 469
- Typed
 - Afterwarp, 29
- TypedBuffer
 - Afterwarp, 44
- Typing
 - Afterwarp, 37
- UltraCondensed
 - Afterwarp, 33
- UltraExpanded
 - Afterwarp, 33
- Unaligned
 - Afterwarp, 38
- Unbind
 - Afterwarp.ComputeProgram, 127
 - Afterwarp.Program, 278
 - Afterwarp.Sampler, 310
 - Afterwarp.Texture, 420
- Undefined
 - Afterwarp, 26, 28, 29, 32, 44, 45
- Underline
 - Afterwarp.FontAttribute, 151
- UnitX
 - Afterwarp.Point, 273
- UnitY
 - Afterwarp.Point, 273
- Unity
 - Afterwarp.Quad, 289
 - Afterwarp.RectF, 304
- Unix
 - Afterwarp, 31
- Unknown
 - Afterwarp, 31, 40, 43
- UnpremultiplyAlpha
 - Afterwarp.Surface, 393
 - Afterwarp.Utility, 446
- UnsignedByte
 - Afterwarp, 32

- UnsignedInt
 - Afterwarp, [32](#)
- UnsignedShort
 - Afterwarp, [32](#)
- Up
 - Afterwarp, [35](#), [39](#)
- Update
 - Afterwarp, [29](#)
 - Afterwarp.Buffer, [86](#)
 - Afterwarp.GaussianBlur, [161](#)
 - Afterwarp.KawaseBlur, [179](#)
 - Afterwarp.ObjectModelView, [246](#)
 - Afterwarp.OceanSimulation, [253](#)
 - Afterwarp.Program, [278](#)
 - Afterwarp.Texture, [420](#)
 - Afterwarp.Timer, [430](#)
 - Afterwarp.Widget, [458](#)
- UpdateAt
 - Afterwarp.GaussianBlur, [161](#)
- UpdateNeeded
 - Afterwarp.ObjectModelView, [246](#)
- UpdateNextSlice
 - Afterwarp.Timer, [431](#)
- Value
 - Afterwarp.PathElement, [267](#)
 - Afterwarp.RandomSequence, [291](#)
 - Afterwarp.Widget.PropertyValue, [284](#)
- ValueDouble
 - Afterwarp.RandomSequence, [291](#)
- VariableRateRefresh
 - Afterwarp.DeviceBehavior, [137](#)
- Variance
 - Afterwarp.ShadowParameters, [376](#)
- Vector
 - Afterwarp, [50](#)
- Vertex
 - Afterwarp, [29](#), [45](#)
- VertexBuffer
 - Afterwarp.MeshModel, [217](#)
- VertexCount
 - Afterwarp.CanvasBuffer, [111](#)
 - Afterwarp.MeshBuffer, [201](#)
 - Afterwarp.MeshMetaTagPortion, [209](#)
 - Afterwarp.MeshModel, [217](#)
- VertexElement
 - Afterwarp.VertexElement, [448](#)
- VertexElementsEstimatePitch
 - Afterwarp.Utility, [446](#)
- VertexElementsIndex
 - Afterwarp.SceneMesh, [335](#)
- VertexElementsIndexUndefined
 - Afterwarp.SceneMesh, [334](#)
- VertexEntry
 - Afterwarp.MeshBuffer.VertexEntry, [450](#)
- Vertical
 - Afterwarp.Margins, [184](#)
- VerticalSpace
 - Afterwarp.TextRenderModifiers, [413](#)
- Vertices
 - Afterwarp.CanvasBuffer, [112](#)
 - Afterwarp.MeshBuffer, [201](#)
- VerticesPtr
 - Afterwarp.MeshBuffer, [201](#)
- View
 - Afterwarp.ActorCamera, [55](#)
 - Afterwarp.ObjectModelView, [247](#)
 - Afterwarp.OceanSimulation, [254](#)
 - Afterwarp.Scene, [321](#)
 - Afterwarp.SpatialFog, [381](#)
- ViewDistance
 - Afterwarp.OceanSimulation, [254](#)
- Viewport
 - Afterwarp.Device, [135](#)
- ViewProjection
 - Afterwarp.Canvas, [108](#)
 - Afterwarp.ObjectModelView, [247](#)
 - Afterwarp.ShadowCaster, [369](#)
- Visible
 - Afterwarp.ObjectModel.Attribute, [74](#)
 - Afterwarp.Widget, [460](#)
- Visualize
 - Afterwarp.MeshVoxel, [221](#)
- VisualizeFunc
 - Afterwarp.MeshVoxel, [221](#)
- Volatile
 - Afterwarp, [49](#)
- Volume
 - Afterwarp, [48](#)
- VolumeDown
 - Afterwarp, [37](#)
- VolumeMute
 - Afterwarp, [37](#)
- VolumeUp
 - Afterwarp, [37](#)
- Voxel
 - Afterwarp.ObjectModel, [236](#)
 - Afterwarp.SceneMesh, [335](#)
- Voxelize
 - Afterwarp.MeshBuffer, [200](#)
- VSync
 - Afterwarp.SwapChain, [398](#)
- Vulkan
 - Afterwarp, [31](#)
- Wait
 - Afterwarp, [26](#)
- WaveSize
 - Afterwarp.OceanWavesParameters, [256](#)
- Waypoint
 - Afterwarp.SceneMeshLatchType, [348](#)
- WebGL
 - Afterwarp, [31](#)
- Weight
 - Afterwarp.FontParameters, [157](#)
- WheelDown
 - Afterwarp, [39](#)
- WheelUp

- Afterwarp, [39](#)
- White
 - Afterwarp.ColorPair, [119](#)
 - Afterwarp.ColorRect, [122](#)
 - Afterwarp.FloatColor, [144](#)
 - Afterwarp.FloatColorRGB, [147](#)
- Widget
 - Afterwarp.Widget, [455](#)
- WidgetAlignment
 - Afterwarp, [49](#)
- WidgetManager
 - Afterwarp.WidgetManager, [465](#)
- WidgetManagerTextureType
 - Afterwarp, [49](#)
- WidgetPropertyBehavior
 - Afterwarp, [49](#)
- WidgetPropertyType
 - Afterwarp, [50](#)
- Width
 - Afterwarp.ImageRegion, [174](#)
 - Afterwarp.Rect, [297](#)
 - Afterwarp.RectF, [302](#)
 - Afterwarp.Surface, [395](#)
 - Afterwarp.SwapChain, [399](#)
 - Afterwarp.TextureParameters, [428](#)
- Wind
 - Afterwarp.OceanWavesParameters, [256](#)
- WindowClassName
 - Afterwarp.ApplicationConfiguration, [70](#)
- WindowClassNameBytes
 - Afterwarp.ApplicationConfiguration, [70](#)
- Windowed
 - Afterwarp, [27](#)
- WindowHandle
 - Afterwarp.Application, [66](#)
 - Afterwarp.SwapChain, [399](#)
- WindowRect
 - Afterwarp.Application, [66](#)
- Windows
 - Afterwarp, [31](#)
- WindowScale
 - Afterwarp.Application, [66](#)
- WindowState
 - Afterwarp.Application, [66](#)
- Wireframe
 - Afterwarp.RenderingState.State, [384](#)
- World
 - Afterwarp.Canvas, [108](#)
 - Afterwarp.Scene, [321](#)
- Wrap
 - Afterwarp, [46](#)
- Write
 - Afterwarp, [30](#)
- WriteTextToClipboard
 - Afterwarp.Application, [65](#)
- X
 - Afterwarp.MeshAligns, [186](#)
 - Afterwarp.Point, [272](#)
- Y
 - Afterwarp.MeshAligns, [186](#)
 - Afterwarp.Point, [272](#)
- Z
 - Afterwarp.MeshAligns, [187](#)
- ZBased
 - Afterwarp, [31](#)
- Zero
 - Afterwarp, [27](#), [45](#)
 - Afterwarp.Margins, [184](#)
 - Afterwarp.Point, [273](#)
 - Afterwarp.Quad, [289](#)
 - Afterwarp.Rect, [298](#)
 - Afterwarp.RectF, [304](#)
- ZoneCount
 - Afterwarp.Widget, [461](#)
- Zoom
 - Afterwarp.ActorCamera, [54](#)
- ZoomOrtho
 - Afterwarp.ActorCamera, [54](#)