

VectSharp

1.7.0

Generated by Doxygen 1.8.18

1 VectSharp: a light library for C# vector graphics	1
1.1 Introduction	1
1.2 Installing VectSharp	1
1.3 Usage	2
1.4 Creating new output layers	2
1.5 Compiling VectSharp from source	3
1.5.1 Windows	3
1.5.2 macOS and Linux	3
1.6 Note about VectSharp.MuPDFUtils and .NET Framework	3
2 Namespace Index	5
2.1 Packages	5
3 Hierarchical Index	7
3.1 Class Hierarchy	7
4 Class Index	9
4.1 Class List	9
5 Namespace Documentation	13
5.1 VectSharp Namespace Reference	13
5.1.1 Enumeration Type Documentation	14
5.1.1.1 LineCaps	14
5.1.1.2 LineJoins	15
5.1.1.3 PixelFormats	15
5.1.1.4 SegmentType	15
5.1.1.5 TextAnchors	16
5.1.1.6 TextBaselines	16
5.1.1.7 UnbalancedStackActions	16
5.2 VectSharp.Canvas Namespace Reference	17
5.3 VectSharp.MuPDFUtils Namespace Reference	17
5.4 VectSharp.PDF Namespace Reference	17
5.5 VectSharp.Raster Namespace Reference	17
5.6 VectSharp.SVG Namespace Reference	17
5.7 VectSharp.ThreeD Namespace Reference	18
6 Class Documentation	19
6.1 VectSharp.ThreeD.AmbientLightSource Class Reference	19
6.1.1 Detailed Description	20
6.1.2 Constructor & Destructor Documentation	20
6.1.2.1 AmbientLightSource()	20
6.1.3 Property Documentation	20
6.1.3.1 Intensity	20
6.2 VectSharp.ThreeD.AreaLightSource Class Reference	21

6.2.1 Detailed Description	22
6.2.2 Constructor & Destructor Documentation	22
6.2.2.1 AreaLightSource()	22
6.2.3 Property Documentation	22
6.2.3.1 Center	22
6.2.3.2 Direction	23
6.2.3.3 DistanceAttenuationExponent	23
6.2.3.4 Intensity	23
6.2.3.5 PenumbraAttenuationExponent	23
6.2.3.6 PenumbraRadius	23
6.2.3.7 Radius	24
6.2.3.8 ShadowSamplingPointCount	24
6.2.3.9 SourceDistance	24
6.3 VectSharp.Canvas.AvaloniaContextInterpreter Class Reference	24
6.3.1 Detailed Description	25
6.3.2 Member Enumeration Documentation	25
6.3.2.1 TextOptions	25
6.3.3 Member Function Documentation	25
6.3.3.1 PaintToCanvas() [1/4]	25
6.3.3.2 PaintToCanvas() [2/4]	26
6.3.3.3 PaintToCanvas() [3/4]	27
6.3.3.4 PaintToCanvas() [4/4]	27
6.4 VectSharp.TrueTypeFile.Bearings Struct Reference	28
6.4.1 Detailed Description	28
6.4.2 Member Data Documentation	28
6.4.2.1 LeftSideBearing	28
6.4.2.2 RightSideBearing	29
6.5 VectSharp.Colour Struct Reference	29
6.5.1 Detailed Description	31
6.5.2 Member Function Documentation	31
6.5.2.1 FromCSSString()	31
6.5.2.2 FromHSL()	31
6.5.2.3 FromLab()	32
6.5.2.4 FromRgb() [1/3]	32
6.5.2.5 FromRgb() [2/3]	33
6.5.2.6 FromRgb() [3/3]	33
6.5.2.7 FromRgba() [1/6]	34
6.5.2.8 FromRgba() [2/6]	34
6.5.2.9 FromRgba() [3/6]	34
6.5.2.10 FromRgba() [4/6]	35
6.5.2.11 FromRgba() [5/6]	35
6.5.2.12 FromRgba() [6/6]	36

6.5.2.13 FromXYZ()	36
6.5.2.14 ToCSSString()	37
6.5.2.15 WithAlpha() [1/4]	37
6.5.2.16 WithAlpha() [2/4]	38
6.5.2.17 WithAlpha() [3/4]	38
6.5.2.18 WithAlpha() [4/4]	39
6.5.3 Member Data Documentation	39
6.5.3.1 A	39
6.5.3.2 B	39
6.5.3.3 G	40
6.5.3.4 H	40
6.5.3.5 L	40
6.5.3.6 R	40
6.5.3.7 X	41
6.6 VectSharp.ThreeD.ColourMaterial Class Reference	41
6.6.1 Detailed Description	42
6.6.2 Constructor & Destructor Documentation	42
6.6.2.1 ColourMaterial()	42
6.6.3 Property Documentation	42
6.6.3.1 Colour	42
6.7 VectSharp.Colours Class Reference	42
6.7.1 Detailed Description	48
6.7.2 Member Data Documentation	49
6.7.2.1 AliceBlue	49
6.7.2.2 AntiqueWhite	49
6.7.2.3 Aqua	49
6.7.2.4 Aquamarine	49
6.7.2.5 Azure	49
6.7.2.6 Beige	50
6.7.2.7 Bisque	50
6.7.2.8 Black	50
6.7.2.9 BlanchedAlmond	50
6.7.2.10 Blue	50
6.7.2.11 BlueViolet	51
6.7.2.12 Brown	51
6.7.2.13 BurlyWood	51
6.7.2.14 CadetBlue	51
6.7.2.15 Chartreuse	51
6.7.2.16 Chocolate	52
6.7.2.17 Coral	52
6.7.2.18 CornflowerBlue	52
6.7.2.19 Cornsilk	52

6.7.2.20 Crimson	52
6.7.2.21 Cyan	53
6.7.2.22 DarkBlue	53
6.7.2.23 DarkCyan	53
6.7.2.24 DarkGoldenRod	53
6.7.2.25 DarkGray	53
6.7.2.26 DarkGreen	54
6.7.2.27 DarkGrey	54
6.7.2.28 DarkKhaki	54
6.7.2.29 DarkMagenta	54
6.7.2.30 DarkOliveGreen	54
6.7.2.31 DarkOrange	55
6.7.2.32 DarkOrchid	55
6.7.2.33 DarkRed	55
6.7.2.34 DarkSalmon	55
6.7.2.35 DarkSeaGreen	55
6.7.2.36 DarkSlateBlue	56
6.7.2.37 DarkSlateGray	56
6.7.2.38 DarkSlateGrey	56
6.7.2.39 DarkTurquoise	56
6.7.2.40 DarkViolet	56
6.7.2.41 DeepPink	57
6.7.2.42 DeepSkyBlue	57
6.7.2.43 DimGray	57
6.7.2.44 DimGrey	57
6.7.2.45 DodgerBlue	57
6.7.2.46 FireBrick	58
6.7.2.47 FloralWhite	58
6.7.2.48 ForestGreen	58
6.7.2.49 Fuchsia	58
6.7.2.50 Gainsboro	58
6.7.2.51 GhostWhite	59
6.7.2.52 Gold	59
6.7.2.53 GoldenRod	59
6.7.2.54 Gray	59
6.7.2.55 Green	59
6.7.2.56 GreenYellow	60
6.7.2.57 Grey	60
6.7.2.58 HoneyDew	60
6.7.2.59 HotPink	60
6.7.2.60 IndianRed	60
6.7.2.61 Indigo	61

6.7.2.62 Ivory	61
6.7.2.63 Khaki	61
6.7.2.64 Lavender	61
6.7.2.65 LavenderBlush	61
6.7.2.66 LawnGreen	62
6.7.2.67 LemonChiffon	62
6.7.2.68 LightBlue	62
6.7.2.69 LightCoral	62
6.7.2.70 LightCyan	62
6.7.2.71 LightGoldenRodYellow	63
6.7.2.72 LightGray	63
6.7.2.73 LightGreen	63
6.7.2.74 LightGrey	63
6.7.2.75 LightPink	63
6.7.2.76 LightSalmon	64
6.7.2.77 LightSeaGreen	64
6.7.2.78 LightSkyBlue	64
6.7.2.79 LightSlateGray	64
6.7.2.80 LightSlateGrey	64
6.7.2.81 LightSteelBlue	65
6.7.2.82 LightYellow	65
6.7.2.83 Lime	65
6.7.2.84 LimeGreen	65
6.7.2.85 Linen	65
6.7.2.86 Magenta	66
6.7.2.87 Maroon	66
6.7.2.88 MediumAquaMarine	66
6.7.2.89 MediumBlue	66
6.7.2.90 MediumOrchid	66
6.7.2.91 MediumPurple	67
6.7.2.92 MediumSeaGreen	67
6.7.2.93 MediumSlateBlue	67
6.7.2.94 MediumSpringGreen	67
6.7.2.95 MediumTurquoise	67
6.7.2.96 MediumVioletRed	68
6.7.2.97 MidnightBlue	68
6.7.2.98 MintCream	68
6.7.2.99 MistyRose	68
6.7.2.100 Moccasin	68
6.7.2.101 NavajoWhite	69
6.7.2.102 Navy	69
6.7.2.103 OldLace	69

6.7.2.104 Olive	69
6.7.2.105 OliveDrab	69
6.7.2.106 Orange	70
6.7.2.107 OrangeRed	70
6.7.2.108 Orchid	70
6.7.2.109 PaleGoldenRod	70
6.7.2.110 PaleGreen	70
6.7.2.111 PaleTurquoise	71
6.7.2.112 PaleVioletRed	71
6.7.2.113 PapayaWhip	71
6.7.2.114 PeachPuff	71
6.7.2.115 Peru	71
6.7.2.116 Pink	72
6.7.2.117 Plum	72
6.7.2.118 PowderBlue	72
6.7.2.119 Purple	72
6.7.2.120 RebeccaPurple	72
6.7.2.121 Red	73
6.7.2.122 RosyBrown	73
6.7.2.123 RoyalBlue	73
6.7.2.124 SaddleBrown	73
6.7.2.125 Salmon	73
6.7.2.126 SandyBrown	74
6.7.2.127 SeaGreen	74
6.7.2.128 SeaShell	74
6.7.2.129 Sienna	74
6.7.2.130 Silver	74
6.7.2.131 SkyBlue	75
6.7.2.132 SlateBlue	75
6.7.2.133 SlateGray	75
6.7.2.134 SlateGrey	75
6.7.2.135 Snow	75
6.7.2.136 SpringGreen	76
6.7.2.137 SteelBlue	76
6.7.2.138 Tan	76
6.7.2.139 Teal	76
6.7.2.140 Thistle	76
6.7.2.141 Tomato	77
6.7.2.142 Turquoise	77
6.7.2.143 Violet	77
6.7.2.144 Wheat	77
6.7.2.145 White	77

6.7.2.146 WhiteSmoke	78
6.7.2.147 Yellow	78
6.7.2.148 YellowGreen	78
6.8 VectSharp.Font.DetailedFontMetrics Class Reference	78
6.8.1 Detailed Description	79
6.8.2 Property Documentation	79
6.8.2.1 Bottom	79
6.8.2.2 Height	79
6.8.2.3 LeftSideBearing	79
6.8.2.4 RightSideBearing	79
6.8.2.5 Top	80
6.8.2.6 Width	80
6.9 VectSharp.DisposableIntPtr Class Reference	80
6.9.1 Detailed Description	81
6.9.2 Constructor & Destructor Documentation	81
6.9.2.1 DisposableIntPtr()	81
6.9.3 Member Data Documentation	81
6.9.3.1 InternalPointer	81
6.10 VectSharp.Document Class Reference	82
6.10.1 Detailed Description	82
6.10.2 Constructor & Destructor Documentation	82
6.10.2.1 Document()	82
6.10.3 Member Data Documentation	82
6.10.3.1 Pages	82
6.11 VectSharp.Font Class Reference	83
6.11.1 Detailed Description	83
6.11.2 Constructor & Destructor Documentation	83
6.11.2.1 Font()	83
6.11.3 Member Function Documentation	84
6.11.3.1 MeasureText()	84
6.11.3.2 MeasureTextAdvanced()	84
6.11.4 Property Documentation	85
6.11.4.1 Ascent	85
6.11.4.2 Descent	85
6.11.4.3 FontFamily	85
6.11.4.4 FontSize	85
6.11.4.5 YMax	86
6.11.4.6 YMin	86
6.12 VectSharp.FontFamily Class Reference	86
6.12.1 Detailed Description	87
6.12.2 Member Enumeration Documentation	87
6.12.2.1 StandardFontFamilies	87

6.12.3 Constructor & Destructor Documentation	88
6.12.3.1 <code>FontFamily()</code> [1/3]	88
6.12.3.2 <code>FontFamily()</code> [2/3]	88
6.12.3.3 <code>FontFamily()</code> [3/3]	89
6.12.4 Member Data Documentation	89
6.12.4.1 <code>StandardFamilies</code>	89
6.12.4.2 <code>StandardFontFamilyResources</code>	89
6.12.5 Property Documentation	90
6.12.5.1 <code>FileName</code>	90
6.12.5.2 <code>IsBold</code>	90
6.12.5.3 <code>IsItalic</code>	90
6.12.5.4 <code>IsOblique</code>	90
6.12.5.5 <code>IsStandardFamily</code>	91
6.12.5.6 <code>TrueTypeFile</code>	91
6.13 VectSharp.Graphics Class Reference	91
6.13.1 Detailed Description	93
6.13.2 Member Function Documentation	93
6.13.2.1 <code>CopyToIGraphicsContext()</code>	93
6.13.2.2 <code>DrawGraphics()</code> [1/2]	94
6.13.2.3 <code>DrawGraphics()</code> [2/2]	94
6.13.2.4 <code>DrawRasterImage()</code> [1/5]	94
6.13.2.5 <code>DrawRasterImage()</code> [2/5]	95
6.13.2.6 <code>DrawRasterImage()</code> [3/5]	95
6.13.2.7 <code>DrawRasterImage()</code> [4/5]	97
6.13.2.8 <code>DrawRasterImage()</code> [5/5]	97
6.13.2.9 <code>FillPath()</code>	98
6.13.2.10 <code>FillRectangle()</code> [1/2]	98
6.13.2.11 <code>FillRectangle()</code> [2/2]	99
6.13.2.12 <code>FillText()</code> [1/2]	99
6.13.2.13 <code>FillText()</code> [2/2]	100
6.13.2.14 <code>FillTextOnPath()</code>	100
6.13.2.15 <code>Linearise()</code>	101
6.13.2.16 <code>MeasureText()</code>	101
6.13.2.17 <code>Restore()</code>	102
6.13.2.18 <code>Rotate()</code>	102
6.13.2.19 <code>RotateAt()</code>	102
6.13.2.20 <code>Save()</code>	102
6.13.2.21 <code>Scale()</code>	103
6.13.2.22 <code>SetClippingPath()</code> [1/3]	103
6.13.2.23 <code>SetClippingPath()</code> [2/3]	103
6.13.2.24 <code>SetClippingPath()</code> [3/3]	104
6.13.2.25 <code>StrokePath()</code>	104

6.13.2.26 StrokeRectangle() [1/2]	105
6.13.2.27 StrokeRectangle() [2/2]	105
6.13.2.28 StrokeText() [1/2]	106
6.13.2.29 StrokeText() [2/2]	106
6.13.2.30 StrokeTextOnPath()	107
6.13.2.31 Transform() [1/2]	108
6.13.2.32 Transform() [2/2]	108
6.13.2.33 Translate() [1/2]	109
6.13.2.34 Translate() [2/2]	109
6.13.3 Property Documentation	109
6.13.3.1 UnbalancedStackAction	110
6.14 VectSharp.GraphicsPath Class Reference	110
6.14.1 Detailed Description	111
6.14.2 Member Function Documentation	111
6.14.2.1 AddSmoothSpline()	111
6.14.2.2 AddText() [1/2]	112
6.14.2.3 AddText() [2/2]	112
6.14.2.4 AddTextOnPath()	113
6.14.2.5 Arc() [1/2]	113
6.14.2.6 Arc() [2/2]	114
6.14.2.7 Close()	115
6.14.2.8 CubicBezierTo() [1/2]	115
6.14.2.9 CubicBezierTo() [2/2]	115
6.14.2.10 EllipticalArc()	116
6.14.2.11 GetLinearisationPointsNormals()	116
6.14.2.12 GetNormalAtAbsolute()	117
6.14.2.13 GetNormalAtRelative()	117
6.14.2.14 GetPointAtAbsolute()	118
6.14.2.15 GetPointAtRelative()	118
6.14.2.16 GetPoints()	118
6.14.2.17 GetTangentAtAbsolute()	119
6.14.2.18 GetTangentAtRelative()	119
6.14.2.19 Linearise()	119
6.14.2.20 LineTo() [1/2]	120
6.14.2.21 LineTo() [2/2]	120
6.14.2.22 MeasureLength()	121
6.14.2.23 MoveTo() [1/2]	121
6.14.2.24 MoveTo() [2/2]	121
6.14.2.25 Transform()	122
6.14.2.26 Triangulate()	122
6.14.3 Property Documentation	122
6.14.3.1 Segments	123

6.15 VectSharp.IGraphicsContext Interface Reference	123
6.15.1 Detailed Description	124
6.15.2 Member Function Documentation	124
6.15.2.1 Close()	125
6.15.2.2 CubicBezierTo()	125
6.15.2.3 DrawRasterImage()	125
6.15.2.4 Fill()	126
6.15.2.5 FillText()	126
6.15.2.6 LineTo()	126
6.15.2.7 MoveTo()	127
6.15.2.8 Rectangle()	127
6.15.2.9 Restore()	127
6.15.2.10 Rotate()	128
6.15.2.11 Save()	128
6.15.2.12 Scale()	128
6.15.2.13 SetClippingPath()	128
6.15.2.14 SetFillStyle() [1/2]	128
6.15.2.15 SetFillStyle() [2/2]	129
6.15.2.16 SetLineDash()	129
6.15.2.17 SetStrokeStyle() [1/2]	129
6.15.2.18 SetStrokeStyle() [2/2]	130
6.15.2.19 Stroke()	130
6.15.2.20 StrokeText()	130
6.15.2.21 Transform()	130
6.15.2.22 Translate()	131
6.15.3 Property Documentation	131
6.15.3.1 FillStyle	131
6.15.3.2 Font	131
6.15.3.3 Height	132
6.15.3.4 LineCap	132
6.15.3.5 LineJoin	132
6.15.3.6 LineWidth	132
6.15.3.7 StrokeStyle	132
6.15.3.8 Tag	133
6.15.3.9 TextBaseline	133
6.15.3.10 Width	133
6.16 VectSharp.ThreeD.ILightSource Interface Reference	133
6.16.1 Detailed Description	134
6.16.2 Member Function Documentation	134
6.16.2.1 GetLightAt()	134
6.16.2.2 GetObstruction()	135
6.16.3 Property Documentation	135

6.16.3.1 CastsShadow	135
6.17 VectSharp.MuPDFUtils.ImageURIParser Class Reference	136
6.17.1 Detailed Description	136
6.17.2 Member Function Documentation	136
6.17.2.1 Parser()	136
6.18 VectSharp.ThreeD.IMaterial Interface Reference	137
6.18.1 Detailed Description	137
6.18.2 Member Function Documentation	137
6.18.2.1 GetColour()	137
6.19 VectSharp.ThreeD.IScene Interface Reference	138
6.19.1 Detailed Description	139
6.19.2 Member Function Documentation	139
6.19.2.1 AddElement()	139
6.19.2.2 AddRange()	139
6.19.2.3 Replace() [1/2]	139
6.19.2.4 Replace() [2/2]	140
6.19.3 Property Documentation	140
6.19.3.1 SceneElements	140
6.19.3.2 SceneLock	140
6.20 VectSharp.ThreeD.LightIntensity Struct Reference	141
6.20.1 Detailed Description	141
6.20.2 Constructor & Destructor Documentation	141
6.20.2.1 LightIntensity()	141
6.20.3 Member Function Documentation	142
6.20.3.1 Deconstruct()	142
6.20.4 Member Data Documentation	142
6.20.4.1 Direction	142
6.20.4.2 Intensity	142
6.21 VectSharp.LineDash Struct Reference	143
6.21.1 Detailed Description	143
6.21.2 Constructor & Destructor Documentation	143
6.21.2.1 LineDash()	143
6.21.3 Member Data Documentation	144
6.21.3.1 Phase	144
6.21.3.2 SolidLine	144
6.21.3.3 UnitsOff	144
6.21.3.4 UnitsOn	144
6.22 VectSharp.ThreeD.MaskedLightSource Class Reference	145
6.22.1 Detailed Description	146
6.22.2 Constructor & Destructor Documentation	146
6.22.2.1 MaskedLightSource() [1/2]	146
6.22.2.2 MaskedLightSource() [2/2]	146

6.22.3 Property Documentation	147
6.22.3.1 AngleAttenuationExponent	147
6.22.3.2 Direction	147
6.22.3.3 Distance	147
6.22.3.4 DistanceAttenuationExponent	148
6.22.3.5 Intensity	148
6.22.3.6 Origin	148
6.22.3.7 Position	148
6.23 VectSharp.ThreeD.ObjectFactory Class Reference	148
6.23.1 Detailed Description	149
6.23.2 Member Function Documentation	149
6.23.2.1 CreateCube()	149
6.23.2.2 CreateCuboid()	150
6.23.2.3 CreatePoints()	151
6.23.2.4 CreatePolygon()	151
6.23.2.5 CreatePrism()	152
6.23.2.6 CreateRectangle() [1/2]	152
6.23.2.7 CreateRectangle() [2/2]	153
6.23.2.8 CreateSphere()	154
6.23.2.9 CreateTetrahedron()	154
6.23.2.10 CreateWireframe()	155
6.24 VectSharp.Page Class Reference	156
6.24.1 Detailed Description	156
6.24.2 Constructor & Destructor Documentation	156
6.24.2.1 Page()	156
6.24.3 Member Function Documentation	157
6.24.3.1 Crop()	157
6.24.4 Property Documentation	157
6.24.4.1 Background	157
6.24.4.2 Graphics	157
6.24.4.3 Height	158
6.24.4.4 Width	158
6.25 VectSharp.ThreeD.ParallelLightSource Class Reference	158
6.25.1 Detailed Description	159
6.25.2 Constructor & Destructor Documentation	159
6.25.2.1 ParallelLightSource()	159
6.25.3 Property Documentation	159
6.25.3.1 Direction	159
6.25.3.2 Intensity	160
6.25.3.3 ReverseDirection	160
6.26 VectSharp.SVG.Parser Class Reference	160
6.26.1 Detailed Description	161

6.26.2 Member Function Documentation	161
6.26.2.1 FromFile()	161
6.26.2.2 FromStream()	161
6.26.2.3 FromString()	162
6.26.2.4 ParseSVGURI()	162
6.26.3 Member Data Documentation	162
6.26.3.1 ParseImageURI	162
6.27 VectSharp.PDF.PDFContextInterpreter Class Reference	163
6.27.1 Detailed Description	163
6.27.2 Member Enumeration Documentation	163
6.27.2.1 TextOptions	163
6.27.3 Member Function Documentation	164
6.27.3.1 SaveAsPDF() [1/2]	164
6.27.3.2 SaveAsPDF() [2/2]	164
6.28 VectSharp.ThreeD.PhongMaterial Class Reference	165
6.28.1 Detailed Description	166
6.28.2 Constructor & Destructor Documentation	166
6.28.2.1 PhongMaterial()	166
6.28.3 Property Documentation	166
6.28.3.1 AmbientReflectionCoefficient	166
6.28.3.2 Colour	166
6.28.3.3 DiffuseReflectionCoefficient	167
6.28.3.4 SpecularReflectionCoefficient	167
6.28.3.5 SpecularShininess	167
6.29 VectSharp.Point Struct Reference	167
6.29.1 Detailed Description	168
6.29.2 Constructor & Destructor Documentation	168
6.29.2.1 Point()	168
6.29.3 Member Function Documentation	168
6.29.3.1 IsEqual()	168
6.29.3.2 Modulus()	169
6.29.3.3 Normalize()	169
6.29.4 Member Data Documentation	169
6.29.4.1 X	170
6.29.4.2 Y	170
6.30 VectSharp.ThreeD.PointLightSource Class Reference	170
6.30.1 Detailed Description	171
6.30.2 Constructor & Destructor Documentation	171
6.30.2.1 PointLightSource()	171
6.30.3 Property Documentation	171
6.30.3.1 DistanceAttenuationExponent	171
6.30.3.2 Intensity	172

6.30.3.3 Position	172
6.31 VectSharp.Raster.Raster Class Reference	172
6.31.1 Detailed Description	172
6.31.2 Member Function Documentation	172
6.31.2.1 SaveAsPNG() [1/2]	172
6.31.2.2 SaveAsPNG() [2/2]	173
6.32 VectSharp.RasterImage Class Reference	173
6.32.1 Detailed Description	175
6.32.2 Constructor & Destructor Documentation	175
6.32.2.1 RasterImage() [1/3]	175
6.32.2.2 RasterImage() [2/3]	175
6.32.2.3 RasterImage() [3/3]	176
6.32.3 Member Function Documentation	176
6.32.3.1 ClearPNGCache()	176
6.32.4 Property Documentation	176
6.32.4.1 DataHolder	177
6.32.4.2 HasAlpha	177
6.32.4.3 Height	177
6.32.4.4 Id	177
6.32.4.5 ImageDataAddress	177
6.32.4.6 Interpolate	178
6.32.4.7 PNGStream	178
6.32.4.8 Width	178
6.33 VectSharp.MuPDFUtils.RasterImageFile Class Reference	178
6.33.1 Detailed Description	179
6.33.2 Constructor & Destructor Documentation	179
6.33.2.1 RasterImageFile()	179
6.34 VectSharp.MuPDFUtils.RasterImageStream Class Reference	179
6.34.1 Detailed Description	180
6.34.2 Constructor & Destructor Documentation	180
6.34.2.1 RasterImageStream() [1/2]	180
6.34.2.2 RasterImageStream() [2/2]	181
6.35 VectSharp.Canvas.RenderAction Class Reference	181
6.35.1 Detailed Description	183
6.35.2 Member Enumeration Documentation	183
6.35.2.1 ActionTypes	183
6.35.3 Member Function Documentation	184
6.35.3.1 BringToFront()	184
6.35.3.2 ImageAction()	184
6.35.3.3 PathAction()	184
6.35.3.4 SendToBack()	185
6.35.3.5 TextAction()	185

6.35.4 Property Documentation	186
6.35.4.1 ActionType	186
6.35.4.2 ClippingPath	186
6.35.4.3 Fill	186
6.35.4.4 Geometry	186
6.35.4.5 ImageDestination	187
6.35.4.6 Imageld	187
6.35.4.7 ImageSource	187
6.35.4.8 InverseTransform	187
6.35.4.9 Parent	187
6.35.4.10 Stroke	188
6.35.4.11 Tag	188
6.35.4.12 Text	188
6.35.4.13 Transform	188
6.35.5 Event Documentation	188
6.35.5.1 PointerEnter	188
6.35.5.2 PointerLeave	189
6.35.5.3 PointerPressed	189
6.35.5.4 PointerReleased	189
6.36 VectSharp.Canvas.ResourceFontFamily Class Reference	189
6.36.1 Detailed Description	190
6.36.2 Constructor & Destructor Documentation	190
6.36.2.1 ResourceFontFamily()	190
6.37 VectSharp.ThreeD.Scene Class Reference	190
6.37.1 Detailed Description	191
6.37.2 Constructor & Destructor Documentation	191
6.37.2.1 Scene()	192
6.38 VectSharp.Segment Class Reference	192
6.38.1 Detailed Description	192
6.38.2 Member Function Documentation	193
6.38.2.1 Clone()	193
6.38.2.2 GetLinearisationTangents()	193
6.38.2.3 GetPointAt()	193
6.38.2.4 GetTangentAt()	194
6.38.2.5 Linearise()	194
6.38.2.6 Measure()	194
6.38.2.7 Transform()	195
6.38.3 Property Documentation	195
6.38.3.1 Point	195
6.38.3.2 Points	195
6.38.3.3 Type	196
6.39 VectSharp.Size Struct Reference	196

6.39.1 Detailed Description	196
6.39.2 Constructor & Destructor Documentation	196
6.39.2.1 Size()	196
6.39.3 Member Data Documentation	197
6.39.3.1 Height	197
6.39.3.2 Width	197
6.40 VectSharp.ThreeD.SpotlightLightSource Class Reference	197
6.40.1 Detailed Description	198
6.40.2 Constructor & Destructor Documentation	198
6.40.2.1 SpotlightLightSource()	198
6.40.3 Property Documentation	199
6.40.3.1 AngleAttenuationExponent	199
6.40.3.2 BeamWidthAngle	199
6.40.3.3 CutoffAngle	199
6.40.3.4 Direction	200
6.40.3.5 DistanceAttenuationExponent	200
6.40.3.6 Intensity	200
6.40.3.7 Position	200
6.41 VectSharp.SVG.SVGContextInterpreter Class Reference	200
6.41.1 Detailed Description	201
6.41.2 Member Enumeration Documentation	201
6.41.2.1 TextOptions	201
6.41.3 Member Function Documentation	201
6.41.3.1 SaveAsSVG() [1/2]	202
6.41.3.2 SaveAsSVG() [2/2]	202
6.42 VectSharp.TrueTypeFile Class Reference	202
6.42.1 Detailed Description	204
6.42.2 Member Function Documentation	204
6.42.2.1 Destroy()	204
6.42.2.2 Get1000EmAscent()	205
6.42.2.3 Get1000EmDescent()	205
6.42.2.4 Get1000EmGlyphBearings()	205
6.42.2.5 Get1000EmGlyphVerticalMetrics()	206
6.42.2.6 Get1000EmGlyphWidth() [1/2]	206
6.42.2.7 Get1000EmGlyphWidth() [2/2]	206
6.42.2.8 Get1000EmXMax()	207
6.42.2.9 Get1000EmXMin()	207
6.42.2.10 Get1000EmYMax()	207
6.42.2.11 Get1000EmYMin()	208
6.42.2.12 GetFirstCharIndex()	208
6.42.2.13 GetFontFamilyName()	208
6.42.2.14 GetFontName()	208

6.42.2.15 GetGlyphIndex()	208
6.42.2.16 GetGlyphPath() [1/2]	209
6.42.2.17 GetGlyphPath() [2/2]	209
6.42.2.18 GetLastCharIndex()	210
6.42.2.19 IsBold()	210
6.42.2.20 IsFixedPitch()	210
6.42.2.21 IsItalic()	211
6.42.2.22 IsOblique()	211
6.42.2.23 IsScript()	211
6.42.2.24 IsSerif()	211
6.42.2.25 SubsetFont()	211
6.42.3 Property Documentation	212
6.42.3.1 FontStream	212
6.43 VectSharp.TrueTypeFile.TrueTypePoint Struct Reference	212
6.43.1 Detailed Description	213
6.43.2 Member Data Documentation	213
6.43.2.1 IsOnCurve	213
6.43.2.2 X	213
6.43.2.3 Y	213
6.44 VectSharp.UnbalancedStackException Class Reference	214
6.44.1 Detailed Description	214
6.45 VectSharp.TrueTypeFile.VerticalMetrics Struct Reference	214
6.45.1 Detailed Description	214
6.45.2 Member Data Documentation	215
6.45.2.1 YMax	215
6.45.2.2 YMin	215
Index	217

Chapter 1

VectSharp: a light library for C# vector graphics

1.1 Introduction

VectSharp is a library to create vector graphics (including text) in C#, without too many dependencies.

It includes an abstract layer on top of which output layers can be written. Currently, there are four available output layers: **VectSharp.PDF** produces PDF documents, **VectSharp.Canvas** produces an `Avalonia.Controls.Canvas` object (<https://avaloniaui.net/docs/controls/canvas>) containing the rendered graphics objects, **VectSharp.Raster** produces raster images in PNG format, and **VectSharp.SVG** produces vector graphics in SVG format.

VectSharp.ThreeD adds support for 3D vector and raster graphics.

VectSharp is written using .NET Core, and is available for Mac, Windows and Linux. It is released under a GPLv3 license. It includes 14 standard fonts, also released under a GPL license.

Since version 2.0.0, **VectSharp.Raster** is released under an AGPLv3 license.

VectSharp.MuPDFUtils, also released under an AGPLv3 license, contains some utility functions that use **MuPDFCore** to make it possible to include in **VectSharp** graphics images in various formats.

1.2 Installing VectSharp

To include **VectSharp** in your project, you will need one of the output layer NuGet packages: **VectSharp.PDF**, **VectSharp.Canvas**, **VectSharp.Raster**, or **VectSharp.SVG**. You will need **VectSharp.ThreeD** to work with 3D graphics. You may want the **VectSharp.MuPDFUtils** package if you wish to manipulate raster images.

1.3 Usage

You can find the full documentation for the [VectSharp](#) library at the [documentation website](#). A [PDF reference manual](#) is also available.

In general, working with [VectSharp](#) involves: creating a Document, adding Pages, drawing to the Pages' Graphics objects and, finally, exporting them to a PDF document, Canvas, PNG image or SVG document.

- **Create a Document:**

```
using VectSharp;
// ...
Document doc = new Document();
```
- **Add a Page:**

```
doc.Pages.Add(new Page(1000, 1000));
```
- **Draw to the Page's Graphics object:**

```
Graphics gpr = doc.Pages.Last().Graphics;
gpr.FillRectangle(100, 100, 800, 800, Colour.FromRgb(128, 128, 128));
```
- **Save as PDF document:**

```
using VectSharp.PDF;
// ...
doc.SaveAsPDF(@"Test.pdf");
```
- **Export the graphics to a Canvas:**

```
using VectSharp.Canvas;
// ...
Avalonia.Controls.Canvas can = doc.Pages.Last().PaintToCanvas();
```
- **Save as a PNG image:**

```
using VectSharp.Raster;
// ...
doc.Pages.Last().SaveAsPNG(@"Sample.png");
```
- **Save as an SVG document:**

```
using VectSharp.SVG;
// ...
doc.Pages.Last().SaveAsSVG(@"Sample.svg");
```

The public classes and methods are [fully documented](#), and you can find a (much) more detailed code example in [MainWindow.xaml.cs](#). A detailed guide about 3D graphics in [VectSharp.ThreeD](#) is available in the [VectSharp.ThreeD](#) folder.

1.4 Creating new output layers

[VectSharp](#) can be easily extended to provide additional output layers. To do so:

1. Create a new class implementing the `IGraphicsContext` interface.
2. Provide an extension method to either the `Page` or `Document` types.
3. Somewhere in the extension method, call the `CopyToIGraphicsContext` method on the `Graphics` object of the `Pages`.
4. Opportunely save or return the rendered result.

1.5 Compiling VectSharp from source

The [VectSharp](#) source code includes an example project (*VectSharp.Demo*) presenting how [VectSharp](#) can be used to produce graphics.

To be able to compile [VectSharp](#) from source, you will need to install the latest [.NET SDK](#) for your operating system.

You can use [Microsoft Visual Studio](#) to compile the program. The following instructions will cover compiling [VectSharp](#) from the command line, instead.

First of all, you will need to download the [VectSharp](#) source code: [VectSharp.tar.gz](#) and extract it somewhere.

1.5.1 Windows

Open a command-line window in the folder where you have extracted the source code, and type:

```
BuildDemo <Target>
```

Where `<Target>` can be one of `Win-x64`, `Linux-x64` or `Mac-x64` depending on which platform you wish to generate executables for.

In the Release folder and in the appropriate subfolder for the target platform you selected, you will find the compiled program.

1.5.2 macOS and Linux

Open a terminal in the folder where you have extracted the source code, and type:

```
./BuildDemo.sh <Target>
```

Where `<Target>` can be one of `Win-x64`, `Linux-x64` or `Mac-x64` depending on which platform you wish to generate executables for.

In the Release folder and in the appropriate subfolder for the target platform you selected, you will find the compiled program.

If you receive an error about permissions being denied, try typing `chmod +x BuildDemo.sh` first.

1.6 Note about VectSharp.MuPDFUtils and .NET Framework

If you wish to use [VectSharp.MuPDFUtils](#) in a .NET Framework project, you will need to manually copy the native MuPDFWrapper library for the platform you are using to the executable directory (this is done automatically if you target .NET core).

One way to obtain the appropriate library files is:

1. Manually download the NuGet package for [MuPDFCore](#) (click on the "Download package" link on the right).
2. Rename the `.nupkg` file so that it has a `.zip` extension.
3. Extract the zip file.
4. Within the extracted folder, the library files are in the `runtimes/xxx-x64/native/` folder, where `xxx` is either `linux`, `osx` or `win`, depending on the platform you are using.

Make sure you copy the appropriate file to the same folder as the executable!

Chapter 2

Namespace Index

2.1 Packages

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

VectSharp	13
VectSharp.Canvas	17
VectSharp.MuPDFUtils	17
VectSharp.PDF	17
VectSharp.Raster	17
VectSharp.SVG	17
VectSharp.ThreeD	18

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

VectSharp.Canvas.AvaloniaContextInterpreter	24
VectSharp.TrueTypeFile.Bearings	28
VectSharp.Colours	42
VectSharp.Font.DetailedFontMetrics	78
VectSharp.Document	82
Exception	
VectSharp.UnbalancedStackException	214
VectSharp.Font	83
VectSharp.FontFamily	86
VectSharp.Canvas.ResourceFontFamily	189
VectSharp.Graphics	91
VectSharp.GraphicsPath	110
IDisposable	
VectSharp.DisposableIntPtr	80
VectSharp.RasterImage	173
VectSharp.MuPDFUtils.RasterImageFile	178
VectSharp.MuPDFUtils.RasterImageStream	179
IEquatable	
VectSharp.Colour	29
VectSharp.IGraphicsContext	123
VectSharp.ThreeD.ILightSource	133
VectSharp.ThreeD.AmbientLightSource	19
VectSharp.ThreeD.AreaLightSource	21
VectSharp.ThreeD.MaskedLightSource	145
VectSharp.ThreeD.ParallelLightSource	158
VectSharp.ThreeD.PointLightSource	170
VectSharp.ThreeD.SpotlightLightSource	197
VectSharp.MuPDFUtils.ImageURIParser	136
VectSharp.ThreeD.IMaterial	137
VectSharp.ThreeD.ColourMaterial	41
VectSharp.ThreeD.PhongMaterial	165
VectSharp.ThreeD.IScene	138
VectSharp.ThreeD.Scene	190
VectSharp.ThreeD.LightIntensity	141

VectSharp.LineDash	143
VectSharp.ThreeD.ObjectFactory	148
VectSharp.Page	156
VectSharp.SVG.Parser	160
VectSharp.PDF.PDFContextInterpreter	163
VectSharp.Point	167
VectSharp.Raster.Raster	172
VectSharp.Canvas.RenderAction	181
VectSharp.Segment	192
VectSharp.Size	196
VectSharp.SVG.SVGContextInterpreter	200
VectSharp.TrueTypeFile	202
VectSharp.TrueTypeFile.TrueTypePoint	212
VectSharp.TrueTypeFile.VerticalMetrics	214

Chapter 4

Class Index

4.1 Class List

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

VectSharp.ThreeD.AmbientLightSource	
Represents a uniform ambien light source	19
VectSharp.ThreeD.AreaLightSource	
Represents a light source emitting light from a circular area	21
VectSharp.Canvas.AvaloniaContextInterpreter	
Contains methods to render a Page to an Avalonia.Controls.Canvas	24
VectSharp.TrueTypeFile.Bearings	
Represents the left- and right-side bearings of a glyph	28
VectSharp.Colour	
Represents an RGB colour	29
VectSharp.ThreeD.ColourMaterial	
Represents a material that always has the same colour, regardless of light	41
VectSharp.Colours	
Standard colours	42
VectSharp.Font.DetailedFontMetrics	
Represents detailed information about the metrics of a text string when drawn with a certain font	78
VectSharp.DisposableIntPtr	
An IDisposable wrapper around an IntPtr that frees the allocated memory when it is disposed	80
VectSharp.Document	
Represents a collection of pages	82
VectSharp.Font	
Represents a typeface with a specific size	83
VectSharp.FontFamily	
Represents a typeface	86
VectSharp.Graphics	
Represents an abstract drawing surface	91
VectSharp.GraphicsPath	
Represents a graphics path that can be filled or stroked	110
VectSharp.IGraphicsContext	
This interface should be implemented by classes intended to provide graphics output capability to a Graphics object	123
VectSharp.ThreeD.ILightSource	
Represents a light source	133
VectSharp.MuPDFUtils.ImageURIParser	
Provides a method to parse an image URI into a page	136

VectSharp.ThreeD.IMaterial	Represents a material used to determine the appearance of Triangle3DElement	137
VectSharp.ThreeD.IScene	Represents a 3D scene	138
VectSharp.ThreeD.LightIntensity	Represents the intensity of a light source at a particular point	141
VectSharp.LineDash	Represents instructions on how to paint a dashed line	143
VectSharp.ThreeD.MaskedLightSource	Represents a point light source with a stencil in front of it	145
VectSharp.ThreeD.ObjectFactory	A static class containing methods to create complex 3D objects	148
VectSharp.Page	Represents a Graphics object with a width and height	156
VectSharp.ThreeD.ParallelLightSource	Represents a parallel light source	158
VectSharp.SVG.Parser	Contains methods to read an SVG image file	160
VectSharp.PDF.PDFContextInterpreter	Contains methods to render a Document as a PDF document	163
VectSharp.ThreeD.PhongMaterial	Represents a material that uses a Phong reflection model to determine the colour of the material based on the light sources that hit it	165
VectSharp.Point	Represents a point relative to an origin in the top-left corner	167
VectSharp.ThreeD.PointLightSource	Represents a point light source	170
VectSharp.Raster.Raster	Contains methods to render a page to a PNG image	172
VectSharp.RasterImage	Represents a raster image, created from raw pixel data. Consider using the derived classes included in the NuGet package "VectSharp.MuPDFUtils" if you need to load a raster image from a file or a Stream	173
VectSharp.MuPDFUtils.RasterImageFile	A RasterImage created from a file	178
VectSharp.MuPDFUtils.RasterImageStream	A RasterImage created from a stream	179
VectSharp.Canvas.RenderAction	Represents a light-weight rendering action	181
VectSharp.Canvas.ResourceFontFamily	Represents a FontFamily created from a resource stream	189
VectSharp.ThreeD.Scene	Represents a 3D scene	190
VectSharp.Segment	Represents a segment as part of a GraphicsPath	192
VectSharp.Size	Represents the size of an object	196
VectSharp.ThreeD.SpotlightLightSource	Represents a conic spotlight	197
VectSharp.SVG.SVGContextInterpreter	Contains methods to render a Page as an SVG file	200
VectSharp.TrueTypeFile	Represents a font file in TrueType format. Reference: http://stevehanov.ca/blog/?id=143 , https://developer.apple.com/fonts/TrueType-Reference-Manual/ , https://docs.microsoft.com/en-us/typography/opentype/spec/202	
VectSharp.TrueTypeFile.TrueTypePoint	Represents a point in a TrueType path description	212

VectSharp.UnbalancedStackException	
The exception that is thrown when an unbalanced graphics state stack occurs	214
VectSharp.TrueTypeFile.VerticalMetrics	
Represents the maximum heigth above and depth below the baseline of a glyph	214

Chapter 5

Namespace Documentation

5.1 VectSharp Namespace Reference

Classes

- struct [Colour](#)
Represents an RGB colour.
- class [Colours](#)
Standard colours.
- class [DisposableIntPtr](#)
An IDisposable wrapper around an IntPtr that frees the allocated memory when it is disposed.
- class [Document](#)
Represents a collection of pages.
- class [Font](#)
Represents a typeface with a specific size.
- class [FontFamily](#)
Represents a typeface.
- class [Graphics](#)
Represents an abstract drawing surface.
- class [GraphicsPath](#)
Represents a graphics path that can be filled or stroked.
- interface [IGraphicsContext](#)
This interface should be implemented by classes intended to provide graphics output capability to a [Graphics](#) object.
- struct [LineDash](#)
Represents instructions on how to paint a dashed line.
- class [Page](#)
Represents a [Graphics](#) object with a width and height.
- struct [Point](#)
Represents a point relative to an origin in the top-left corner.
- class [RasterImage](#)
Represents a raster image, created from raw pixel data. Consider using the derived classes included in the NuGet package "VectSharp.MuPDFUtils" if you need to load a raster image from a file or a Stream.
- class [Segment](#)
Represents a segment as part of a [GraphicsPath](#).
- struct [Size](#)
Represents the size of an object.

- class [TrueTypeFile](#)

Represents a font file in TrueType format. Reference: <http://stevehanov.ca/blog/?id=143>, <https://developer.apple.com/fonts/TrueType-Reference-Manual/>, <https://docs.microsoft.com/en-us/typography/opentype/spec/>

- class [UnbalancedStackException](#)

The exception that is thrown when an unbalanced graphics state stack occurs.

Enumerations

- enum [TextBaselines](#) { [TextBaselines.Top](#), [TextBaselines.Bottom](#), [TextBaselines.Middle](#), [TextBaselines.Baseline](#) }

Represent text baselines.

- enum [TextAnchors](#) { [TextAnchors.Left](#), [TextAnchors.Center](#), [TextAnchors.Right](#) }

Represents text anchors.

- enum [LineCaps](#) { [LineCaps.Butt](#) = 0, [LineCaps.Round](#) = 1, [LineCaps.Square](#) = 2 }

Represents line caps.

- enum [LineJoins](#) { [LineJoins.Bevel](#) = 2, [LineJoins.Miter](#) = 0, [LineJoins.Round](#) = 1 }

Represents line joining options.

- enum [SegmentType](#) { [SegmentType.Move](#), [SegmentType.Line](#), [SegmentType.CubicBezier](#), [SegmentType.Arc](#), [SegmentType.Close](#) }

Types of Segment.

- enum [UnbalancedStackActions](#) { [UnbalancedStackActions.Throw](#), [UnbalancedStackActions.SilentlyFix](#), [UnbalancedStackActions.Ignore](#) }

Represents ways to deal with unbalanced graphics state stacks.

- enum [PixelFormat](#) { [PixelFormat.RGB](#), [PixelFormat.RGBA](#), [PixelFormat.BGR](#), [PixelFormat.BGRA](#) }

Represents the pixel format of a raster image.

5.1.1 Enumeration Type Documentation

5.1.1.1 LineCaps

```
enum VectSharp.LineCaps [strong]
```

Represents line caps.

Enumerator

Butt	The ends of the line are squared off at the endpoints.
Round	The ends of the lines are rounded.
Square	The ends of the lines are squared off by adding an half square box at each end.

Definition at line 88 of file Graphics.cs.

5.1.1.2 LineJoins

```
enum VectSharp.LineJoins [strong]
```

Represents line joining options.

Enumerator

Bevel	Consecutive segments are joined by straight corners.
Miter	Consecutive segments are joined by extending their outside edges until they meet.
Round	Consecutive segments are joined by arc segments.

Definition at line 109 of file Graphics.cs.

5.1.1.3 PixelFormats

```
enum VectSharp.PixelFormats [strong]
```

Represents the pixel format of a raster image.

Enumerator

RGB	RGB 24bpp format.
RGBA	RGBA 32bpp format.
BGR	BGR 24bpp format.
BGRA	BGR 32bpp format.

Definition at line 27 of file RasterImage.cs.

5.1.1.4 SegmentType

```
enum VectSharp.SegmentType [strong]
```

Types of [Segment](#).

Enumerator

Move	The segment represents a move from the current point to a new point.
Line	The segment represents a straight line from the current point to a new point.
CubicBezier	The segment represents a cubic bezier curve from the current point to a new point.
Arc	The segment represents a circular arc from the current point to a new point.
Close	The segment represents the closing segment of a figure.

Definition at line 1312 of file Graphics.cs.

5.1.1.5 TextAnchors

enum `VectSharp.TextAnchors` [strong]

Represents text anchors.

Enumerator

Left	The current coordinate will determine the position of the left side of the text string.
Center	The current coordinate will determine the position of the center of the text string.
Right	The current coordinate will determine the position of the right side of the text string.

Definition at line 67 of file Graphics.cs.

5.1.1.6 TextBaselines

enum `VectSharp.TextBaselines` [strong]

Represent text baselines.

Enumerator

Top	The current vertical coordinate determines where the top of the text string will be placed.
Bottom	The current vertical coordinate determines where the bottom of the text string will be placed.
Middle	The current vertical coordinate determines where the middle of the text string will be placed.
Baseline	The current vertical coordinate determines where the baseline of the text string will be placed.

Definition at line 41 of file Graphics.cs.

5.1.1.7 UnbalancedStackActions

enum `VectSharp.UnbalancedStackActions` [strong]

Represents ways to deal with unbalanced graphics state stacks.

Enumerator

Throw	If the graphics state stack is unbalanced, an exception will be thrown.
SilentlyFix	The graphics state stack will be automatically balanced by adding or removing calls to Graphics.Restore as necessary.
Ignore	No attempt will be made at correcting an unbalanced graphics state stack. This may cause issues with some consumers.

Definition at line 2292 of file Graphics.cs.

5.2 VectSharp.Canvas Namespace Reference

Classes

- class [AvaloniaContextInterpreter](#)
Contains methods to render a [Page](#) to an [Avalonia.Controls.Canvas](#).
- class [RenderAction](#)
Represents a light-weight rendering action.
- class [ResourceFontFamily](#)
Represents a [FontFamily](#) created from a resource stream.

5.3 VectSharp.MuPDFUtils Namespace Reference

Classes

- class [ImageURIParser](#)
Provides a method to parse an image URI into a page.
- class [RasterImageFile](#)
A [RasterImage](#) created from a file.
- class [RasterImageStream](#)
A [RasterImage](#) created from a stream.

5.4 VectSharp.PDF Namespace Reference

Classes

- class [PDFContextInterpreter](#)
Contains methods to render a [Document](#) as a [PDF](#) document.

5.5 VectSharp.Raster Namespace Reference

Classes

- class [Raster](#)
Contains methods to render a page to a PNG image.

5.6 VectSharp.SVG Namespace Reference

Classes

- class [Parser](#)
Contains methods to read an [SVG](#) image file.
- class [SVGContextInterpreter](#)
Contains methods to render a [Page](#) as an [SVG](#) file.

5.7 VectSharp.ThreeD Namespace Reference

Classes

- class [AmbientLightSource](#)
Represents a uniform ambien light source.
- class [AreaLightSource](#)
Represents a light source emitting light from a circular area.
- class [ColourMaterial](#)
Represents a material that always has the same colour, regardless of light.
- interface [ILightSource](#)
Represents a light source.
- interface [IMaterial](#)
Represents a material used to the determine the appearance of Triangle3DElement.
- interface [IScene](#)
Represents a 3D scene.
- struct [LightIntensity](#)
Represents the intensity of a light source at a particular point.
- class [MaskedLightSource](#)
Represents a point light source with a stencil in front of it.
- class [ObjectFactory](#)
A static class containing methods to create complex 3D objects.
- class [ParallelLightSource](#)
Represents a parallel light source.
- class [PhongMaterial](#)
Represents a material that uses a Phong reflection model to determine the colour of the material based on the light sources that hit it.
- class [PointLightSource](#)
Represents a point light source.
- class [Scene](#)
Represents a 3D scene.
- class [SpotlightLightSource](#)
Represents a conic spotlight.

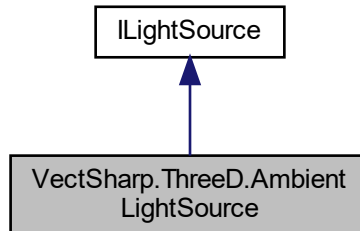
Chapter 6

Class Documentation

6.1 VectSharp.ThreeD.AmbientLightSource Class Reference

Represents a uniform ambien light source.

Inheritance diagram for VectSharp.ThreeD.AmbientLightSource:



Public Member Functions

- [AmbientLightSource](#) (double intensity)
Creates a new [AmbientLightSource](#) instance.
- [LightIntensity GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< Triangle3DElement > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles .

Public Attributes

- bool [CastsShadow](#) => false

Properties

- double [Intensity](#) [get, set]
The intensity of the light.

6.1.1 Detailed Description

Represents a uniform ambien light source.

Definition at line 74 of file Lights.cs.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 AmbientLightSource()

```
VectSharp.ThreeD.AmbientLightSource.AmbientLightSource (  
    double intensity )
```

Creates a new [AmbientLightSource](#) instance.

Parameters

<i>intensity</i>	The intensity of the light.
------------------	-----------------------------

Definition at line 88 of file Lights.cs.

6.1.3 Property Documentation

6.1.3.1 Intensity

```
double VectSharp.ThreeD.AmbientLightSource.Intensity [get], [set]
```

The intensity of the light.

Definition at line 79 of file Lights.cs.

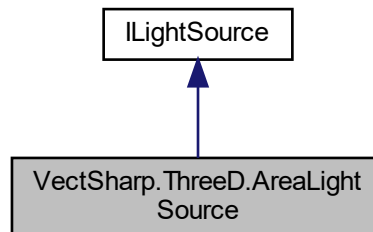
The documentation for this class was generated from the following file:

- VectSharp.ThreeD/Lights.cs

6.2 VectSharp.ThreeD.AreaLightSource Class Reference

Represents a light source emitting light from a circular area.

Inheritance diagram for VectSharp.ThreeD.AreaLightSource:



Public Member Functions

- [AreaLightSource](#) (double intensity, Point3D center, double radius, double penumbraRadius, NormalizedVector3D direction, double sourceDistance, int shadowSamplingPointCount)
Creates a new [AreaLightSource](#) instance.
- [LightIntensity](#) [GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< Triangle3DElement > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles .

Properties

- bool [CastsShadow](#) = true [get, set]
- Point3D [Center](#) [get]
The centre of the light-emitting area.
- NormalizedVector3D [Direction](#) [get]
The direction of the light's main axis, i.e. the normal to the plane containing the light-emitting area.
- double [Radius](#) [get]
The radius of the light emitting area.
- double [PenumbraRadius](#) [get]
The radius of the penumbra area.
- double [Intensity](#) [get, set]
The base intensity of the light.
- double [SourceDistance](#) [get]
The distance between the focal point of the light and the light's [Center](#).
- double [DistanceAttenuationExponent](#) = 2 [get, set]
An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.
- double [PenumbraAttenuationExponent](#) = 1 [get, set]
An exponent determining how fast the light attenuates between the light-emitting area radius and the penumbra radius.
- int [ShadowSamplingPointCount](#) [get]
The number of points to use when determining the amount of light that is obstructed at a certain point.

6.2.1 Detailed Description

Represents a light source emitting light from a circular area.

Definition at line 562 of file Lights.cs.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 AreaLightSource()

```
VectSharp.ThreeD.AreaLightSource.AreaLightSource (
    double intensity,
    Point3D center,
    double radius,
    double penumbraRadius,
    NormalizedVector3D direction,
    double sourceDistance,
    int shadowSamplingPointCount )
```

Creates a new [AreaLightSource](#) instance.

Parameters

<i>intensity</i>	The base intensity of the light.
<i>center</i>	The centre of the light-emitting area.
<i>radius</i>	The radius of the light-emitting area.
<i>penumbraRadius</i>	The radius of the penumbra area.
<i>direction</i>	The direction of the light.
<i>sourceDistance</i>	The distance between the focal point of the light and the light's center.
<i>shadowSamplingPointCount</i>	The number of points to use when determining the amount of light that is obstructed at a certain point.

Definition at line 626 of file Lights.cs.

6.2.3 Property Documentation

6.2.3.1 Center

```
Point3D VectSharp.ThreeD.AreaLightSource.Center [get]
```

The centre of the light-emitting area.

Definition at line 570 of file Lights.cs.

6.2.3.2 Direction

```
NormalizedVector3D VectSharp.ThreeD.AreaLightSource.Direction [get]
```

The direction of the light's main axis, i.e. the normal to the plane containing the light-emitting area.

Definition at line 577 of file Lights.cs.

6.2.3.3 DistanceAttenuationExponent

```
double VectSharp.ThreeD.AreaLightSource.DistanceAttenuationExponent = 2 [get], [set]
```

An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.

Definition at line 602 of file Lights.cs.

6.2.3.4 Intensity

```
double VectSharp.ThreeD.AreaLightSource.Intensity [get], [set]
```

The base intensity of the light.

Definition at line 592 of file Lights.cs.

6.2.3.5 PenumbraAttenuationExponent

```
double VectSharp.ThreeD.AreaLightSource.PenumbraAttenuationExponent = 1 [get], [set]
```

An exponent determining how fast the light attenuates between the light-emitting area radius and the penumbra radius.

Definition at line 607 of file Lights.cs.

6.2.3.6 PenumbraRadius

```
double VectSharp.ThreeD.AreaLightSource.PenumbraRadius [get]
```

The radius of the penumbra area.

Definition at line 587 of file Lights.cs.

6.2.3.7 Radius

```
double VectSharp.ThreeD.AreaLightSource.Radius [get]
```

The radius of the light emitting area.

Definition at line 582 of file Lights.cs.

6.2.3.8 ShadowSamplingPointCount

```
int VectSharp.ThreeD.AreaLightSource.ShadowSamplingPointCount [get]
```

The number of points to use when determining the amount of light that is obstructed at a certain point.

Definition at line 612 of file Lights.cs.

6.2.3.9 SourceDistance

```
double VectSharp.ThreeD.AreaLightSource.SourceDistance [get]
```

The distance between the focal point of the light and the light's [Center](#).

Definition at line 597 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.3 VectSharp.Canvas.AvaloniaContextInterpreter Class Reference

Contains methods to render a [Page](#) to an Avalonia.Controls.Canvas.

Public Types

- enum [TextOptions](#) { [TextOptions.AlwaysConvert](#), [TextOptions.ConvertIfNecessary](#), [TextOptions.NeverConvert](#) }

Defines whether text items should be converted into paths when drawing.

Static Public Member Functions

- static Avalonia.Controls.Canvas [PaintToCanvas](#) (this [Page](#) page, [TextOptions](#) textOption=[TextOptions.ConvertIfNecessary](#))
Render a [Page](#) to an Avalonia.Controls.Canvas.
- static Avalonia.Controls.Canvas [PaintToCanvas](#) (this [Page](#) page, bool graphicsAsControls, [TextOptions](#) text↵
Option=[TextOptions.ConvertIfNecessary](#))
Render a [Page](#) to an Avalonia.Controls.Canvas.
- static Avalonia.Controls.Canvas [PaintToCanvas](#) (this [Page](#) page, bool graphicsAsControls, Dictionary<
string, Delegate > taggedActions, bool removeTaggedActionsAfterExecution=true, [TextOptions](#) text↵
Option=[TextOptions.ConvertIfNecessary](#))
Render a [Page](#) to an Avalonia.Controls.Canvas.
- static Avalonia.Controls.Canvas [PaintToCanvas](#) (this [Page](#) page, Dictionary< string, Delegate > tagged↵
Actions, bool removeTaggedActionsAfterExecution=true, [TextOptions](#) textOption=[TextOptions.ConvertIfNecessary](#))
Render a [Page](#) to an Avalonia.Controls.Canvas.

6.3.1 Detailed Description

Contains methods to render a [Page](#) to an Avalonia.Controls.Canvas.

Definition at line 1905 of file AvaloniaContext.cs.

6.3.2 Member Enumeration Documentation

6.3.2.1 TextOptions

```
enum VectSharp.Canvas.AvaloniaContextInterpreter.TextOptions [strong]
```

Defines whether text items should be converted into paths when drawing.

Enumerator

AlwaysConvert	Converts all text items into paths.
ConvertIfNecessary	Converts all text items into paths, with the exception of those that use a standard font.
NeverConvert	Does not convert any text items into paths.

Definition at line 1910 of file AvaloniaContext.cs.

6.3.3 Member Function Documentation

6.3.3.1 PaintToCanvas() [1/4]

```
static Avalonia.Controls.Canvas VectSharp.Canvas.AvaloniaContextInterpreter.PaintToCanvas (
    this Page page,
```

```

bool graphicsAsControls,
Dictionary< string, Delegate > taggedActions,
bool removeTaggedActionsAfterExecution = true,
TextOptions textOption = TextOptions.ConvertIfNecessary ) [static]

```

Render a [Page](#) to an Avalonia.Controls.Canvas.

Parameters

<i>page</i>	The Page to render.
<i>graphicsAsControls</i>	If this is true, each graphics object (e.g. paths, text...) is rendered as a separate Avalonia.Controls.Control. Otherwise, they are directly rendered onto the drawing context (which is faster, but does not allow interactivity).
<i>taggedActions</i>	A Dictionary<String, Delegate> containing the Actions that will be performed on items with the corresponding tag. If <i>graphicsAsControls</i> is true, the delegates should be voids that accept one parameter of type TextBlock or Path (depending on the tagged item), otherwise, they should accept one parameter of type RenderAction and return an IEnumerable<RenderAction> of the actions that will actually be performed.
<i>removeTaggedActionsAfterExecution</i>	Whether the Actions should be removed from <i>taggedActions</i> after their execution. Set to false if the same Action should be performed on multiple items with the same tag.
<i>textOption</i>	Defines whether text items should be converted into paths when drawing.

Returns

An Avalonia.Controls.Canvas containing the rendered graphics objects.

Definition at line 1973 of file AvaloniaContext.cs.

6.3.3.2 PaintToCanvas() [2/4]

```

static Avalonia.Controls.Canvas VectSharp.Canvas.AvaloniaContextInterpreter.PaintToCanvas (
    this Page page,
    bool graphicsAsControls,
    TextOptions textOption = TextOptions.ConvertIfNecessary ) [static]

```

Render a [Page](#) to an Avalonia.Controls.Canvas.

Parameters

<i>page</i>	The Page to render.
<i>graphicsAsControls</i>	If this is true, each graphics object (e.g. paths, text...) is rendered as a separate Avalonia.Controls.Control. Otherwise, they are directly rendered onto the drawing context (which is faster, but does not allow interactivity).
<i>textOption</i>	Defines whether text items should be converted into paths when drawing.

Returns

An Avalonia.Controls.Canvas containing the rendered graphics objects.

Definition at line 1949 of file AvaloniaContext.cs.

6.3.3.3 PaintToCanvas() [3/4]

```
static Avalonia.Controls.Canvas VectSharp.Canvas.AvaloniaContextInterpreter.PaintToCanvas (
    this Page page,
    Dictionary< string, Delegate > taggedActions,
    bool removeTaggedActionsAfterExecution = true,
    TextOptions textOption = TextOptions.ConvertIfNecessary ) [static]
```

Render a [Page](#) to an Avalonia.Controls.Canvas.

Parameters

<i>page</i>	The Page to render.
<i>taggedActions</i>	A Dictionary<String, Delegate> containing the Actions that will be performed on items with the corresponding tag. The delegates should accept one parameter of type TextBlock or Path (depending on the tagged item).
<i>removeTaggedActionsAfterExecution</i>	Whether the Actions should be removed from <i>taggedActions</i> after their execution. Set to false if the same Action should be performed on multiple items with the same tag.
<i>textOption</i>	Defines whether text items should be converted into paths when drawing.

Returns

An Avalonia.Controls.Canvas containing the rendered graphics objects.

Definition at line 1996 of file AvaloniaContext.cs.

6.3.3.4 PaintToCanvas() [4/4]

```
static Avalonia.Controls.Canvas VectSharp.Canvas.AvaloniaContextInterpreter.PaintToCanvas (
    this Page page,
    TextOptions textOption = TextOptions.ConvertIfNecessary ) [static]
```

Render a [Page](#) to an Avalonia.Controls.Canvas.

Parameters

<i>page</i>	The Page to render.
<i>textOption</i>	Defines whether text items should be converted into paths when drawing.

Returns

An Avalonia.Controls.Canvas containing the rendered graphics objects.

Definition at line 1934 of file AvaloniaContext.cs.

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

- VectSharp.Canvas/AvaloniaContext.cs

6.4 VectSharp.TrueTypeFile.Bearings Struct Reference

Represents the left- and right-side bearings of a glyph.

Public Attributes

- int [LeftSideBearing](#)
The left-side bearing of the glyph.
- int [RightSideBearing](#)
The right-side bearing of the glyph.

6.4.1 Detailed Description

Represents the left- and right-side bearings of a glyph.

Definition at line 2115 of file TrueType.cs.

6.4.2 Member Data Documentation

6.4.2.1 LeftSideBearing

```
int VectSharp.TrueTypeFile.Bearings.LeftSideBearing
```

The left-side bearing of the glyph.

Definition at line 2120 of file TrueType.cs.

6.4.2.2 RightSideBearing

```
int VectSharp.TrueTypeFile.Bearings.RightSideBearing
```

The right-side bearing of the glyph.

Definition at line 2125 of file TrueType.cs.

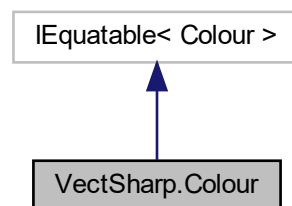
The documentation for this struct was generated from the following file:

- VectSharp/TrueType.cs

6.5 VectSharp.Colour Struct Reference

Represents an RGB colour.

Inheritance diagram for VectSharp.Colour:



Public Member Functions

- override bool [Equals](#) (object obj)
- bool [Equals](#) (Colour col)
- override int [GetHashCode](#) ()
- string [ToCSSString](#) (bool includeAlpha)

Convert the [Colour](#) object into a hex string that is constituted by a "#" followed by two-digit hexadecimal representations of the red, green and blue components of the colour (in the range 0x00 - 0xFF). Optionally also includes opacity (alpha channel) data.

- [Colour WithAlpha](#) (double alpha)

Create a new [Colour](#) with the same RGB components as the current [Colour](#), but with the specified alpha .

- [Colour WithAlpha](#) (byte alpha)

Create a new [Colour](#) with the same RGB components as the current [Colour](#), but with the specified alpha .

- double double double Z [ToXYZ](#) ()
- double double double b [ToLab](#) ()
- double double double L [ToHSL](#) ()

Static Public Member Functions

- static [Colour FromRgb](#) (double r, double g, double b)
Create a new colour from RGB (red, green and blue) values.
- static [Colour FromRgb](#) (byte r, byte g, byte b)
Create a new colour from RGB (red, green and blue) values.
- static [Colour FromRgb](#) (int r, int g, int b)
Create a new colour from RGB (red, green and blue) values.
- static [Colour FromRgba](#) (double r, double g, double b, double a)
Create a new colour from RGBA (red, green, blue and alpha) values.
- static [Colour FromRgba](#) (byte r, byte g, byte b, byte a)
Create a new colour from RGBA (red, green, blue and alpha) values.
- static [Colour FromRgba](#) (byte r, byte g, byte b, double a)
Create a new colour from RGBA (red, green, blue and alpha) values.
- static [Colour FromRgba](#) (int r, int g, int b, int a)
Create a new colour from RGBA (red, green, blue and alpha) values.
- static [Colour FromRgba](#) (int r, int g, int b, double a)
Create a new colour from RGBA (red, green, blue and alpha) values.
- static [Colour FromRgba](#) ((int r, int g, int b, double a) colour)
Create a new colour from RGBA (red, green, blue and alpha) values.
- static bool [operator==](#) ([Colour](#) col1, [Colour](#) col2)
- static bool [operator!=](#) ([Colour](#) col1, [Colour](#) col2)
- static ? [Colour FromCSSString](#) (string cssString)
Convert a CSS colour string into a [Colour](#) object.
- static [Colour WithAlpha](#) ([Colour](#) original, double alpha)
Create a new [Colour](#) with the same RGB components as the original [Colour](#), but with the specified alpha .
- static [Colour WithAlpha](#) ([Colour](#) original, byte alpha)
Create a new [Colour](#) with the same RGB components as the original [Colour](#), but with the specified alpha .
- static [Colour FromXYZ](#) (double x, double y, double z)
Creates a [Colour](#) from CIE XYZ coordinates.
- static [Colour FromLab](#) (double L, double a, double b)
Creates a [Colour](#) from CIE Lab coordinates (under Illuminant D65).
- static [Colour FromHSL](#) (double h, double s, double l)
Creates a [Colour](#) from HSL coordinates.

Public Attributes

- double [R](#)
Red component of the colour. Range: [0, 1].
- double [G](#)
Green component of the colour. Range: [0, 1].
- double [B](#)
Blue component of the colour. Range: [0, 1].
- double [A](#)
Alpha component of the colour. Range: [0, 1].
- double [X](#)
Converts a [Colour](#) to the CIE XYZ colour space.
- double double [Y](#)
- double [L](#)
Converts a [Colour](#) to the CIE Lab colour space (under Illuminant D65).
- double double [a](#)
- double [H](#)
Converts a [Colour](#) to the HSL colour space.
- double double [S](#)

6.5.1 Detailed Description

Represents an RGB colour.

Definition at line 169 of file Graphics.cs.

6.5.2 Member Function Documentation

6.5.2.1 FromCSSString()

```
static ? Colour VectSharp.Colour.FromCSSString (  
    string cssString ) [static]
```

Convert a CSS colour string into a [Colour](#) object.

Parameters

<i>cssString</i>	The CSS colour string. In addition to 148 standard colour names (case-insensitive), #RGB, #RGBA, #RRGGBB and #RRGGBBAA hex strings and rgb(r, g, b) and rgba(r, g, b, a) functional colour notations are supported.
------------------	---

Returns

Definition at line 369 of file Graphics.cs.

6.5.2.2 FromHSL()

```
static Colour VectSharp.Colour.FromHSL (  
    double h,  
    double s,  
    double l ) [static]
```

Creates a [Colour](#) from HSL coordinates.

Parameters

<i>h</i>	The H component. Should be in range [0, 1].
<i>s</i>	The S component. Should be in range [0, 1].
<i>l</i>	The L component. Should be in range [0, 1].

Returns

A [Colour](#) created from the specified components.

Definition at line 719 of file Graphics.cs.

6.5.2.3 FromLab()

```
static Colour VectSharp.Colour.FromLab (
    double L,
    double a,
    double b ) [static]
```

Creates a [Colour](#) from CIE Lab coordinates (under Illuminant D65).

Parameters

<i>L</i>	The L* component.
<i>a</i>	The a* component.
<i>b</i>	The b* component.

Returns

An sRGB [Colour](#) created from the specified components.

Definition at line 641 of file Graphics.cs.

6.5.2.4 FromRgb() [1/3]

```
static Colour VectSharp.Colour.FromRgb (
    byte r,
    byte g,
    byte b ) [static]
```

Create a new colour from RGB (red, green and blue) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 255].
<i>g</i>	The green component of the colour. Range: [0, 255].
<i>b</i>	The blue component of the colour. Range: [0, 255].

Returns

A [Colour](#) struct with the specified components and an alpha component of 1.

Definition at line 218 of file Graphics.cs.

6.5.2.5 FromRgb() [2/3]

```
static Colour VectSharp.Colour.FromRgb (  
    double r,  
    double g,  
    double b ) [static]
```

Create a new colour from RGB (red, green and blue) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 1].
<i>g</i>	The green component of the colour. Range: [0, 1].
<i>b</i>	The blue component of the colour. Range: [0, 1].

Returns

A [Colour](#) struct with the specified components and an alpha component of 1.

Definition at line 206 of file Graphics.cs.

6.5.2.6 FromRgb() [3/3]

```
static Colour VectSharp.Colour.FromRgb (  
    int r,  
    int g,  
    int b ) [static]
```

Create a new colour from RGB (red, green and blue) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 255].
<i>g</i>	The green component of the colour. Range: [0, 255].
<i>b</i>	The blue component of the colour. Range: [0, 255].

Returns

A [Colour](#) struct with the specified components and an alpha component of 1.

Definition at line 230 of file Graphics.cs.

6.5.2.7 FromRgba() [1/6]

```
static Colour VectSharp.Colour.FromRgba (
    (int r, int g, int b, double a) colour ) [static]
```

Create a new colour from RGBA (red, green, blue and alpha) values.

Parameters

<i>colour</i>	A ValueTuple<Int32, Int32, Int32, Double> containing component information for the colour. For r, g, and b, range: [0, 255]; for a, range: [0, 1].
---------------	--

Returns

A [Colour](#) struct with the specified components.

Definition at line 304 of file Graphics.cs.

6.5.2.8 FromRgba() [2/6]

```
static Colour VectSharp.Colour.FromRgba (
    byte r,
    byte g,
    byte b,
    byte a ) [static]
```

Create a new colour from RGBA (red, green, blue and alpha) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 255].
<i>g</i>	The green component of the colour. Range: [0, 255].
<i>b</i>	The blue component of the colour. Range: [0, 255].
<i>a</i>	The alpha component of the colour. Range: [0, 255].

Returns

A [ColourColour](#) struct with the specified components.

Definition at line 256 of file Graphics.cs.

6.5.2.9 FromRgba() [3/6]

```
static Colour VectSharp.Colour.FromRgba (
    byte r,
```

```
byte g,  
byte b,  
double a ) [static]
```

Create a new colour from RGBA (red, green, blue and alpha) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 255].
<i>g</i>	The green component of the colour. Range: [0, 255].
<i>b</i>	The blue component of the colour. Range: [0, 255].
<i>a</i>	The alpha component of the colour. Range: [0, 1].

Returns

A [Colour](#) struct with the specified components.

Definition at line 269 of file Graphics.cs.

6.5.2.10 FromRgba() [4/6]

```
static Colour VectSharp.Colour.FromRgba (  
    double r,  
    double g,  
    double b,  
    double a ) [static]
```

Create a new colour from RGBA (red, green, blue and alpha) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 1].
<i>g</i>	The green component of the colour. Range: [0, 1].
<i>b</i>	The blue component of the colour. Range: [0, 1].
<i>a</i>	The alpha component of the colour. Range: [0, 1].

Returns

A [Colour](#) struct with the specified components.

Definition at line 243 of file Graphics.cs.

6.5.2.11 FromRgba() [5/6]

```
static Colour VectSharp.Colour.FromRgba (  
    int r,
```

```

    int g,
    int b,
    double a ) [static]

```

Create a new colour from RGBA (red, green, blue and alpha) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 255].
<i>g</i>	The green component of the colour. Range: [0, 255].
<i>b</i>	The blue component of the colour. Range: [0, 255].
<i>a</i>	The alpha component of the colour. Range: [0, 1].

Returns

A [Colour](#) struct with the specified components.

Definition at line 294 of file Graphics.cs.

6.5.2.12 FromRgba() [6/6]

```

static Colour VectSharp.Colour.FromRgba (
    int r,
    int g,
    int b,
    int a ) [static]

```

Create a new colour from RGBA (red, green, blue and alpha) values.

Parameters

<i>r</i>	The red component of the colour. Range: [0, 255].
<i>g</i>	The green component of the colour. Range: [0, 255].
<i>b</i>	The blue component of the colour. Range: [0, 255].
<i>a</i>	The alpha component of the colour. Range: [0, 255].

Returns

A [Colour](#) struct with the specified components.

Definition at line 281 of file Graphics.cs.

6.5.2.13 FromXYZ()

```

static Colour VectSharp.Colour.FromXYZ (
    double x,

```



```
double y,  
double z ) [static]
```

Creates a [Colour](#) from CIE XYZ coordinates.

Parameters

<i>x</i>	The X coordinate.
<i>y</i>	The Y coordinate.
<i>z</i>	The Z coordinate.

Returns

An sRGB [Colour](#) created from the specified components.

Definition at line 559 of file Graphics.cs.

6.5.2.14 ToCSSString()

```
string VectSharp.Colour.ToCSSString (  
    bool includeAlpha )
```

Convert the [Colour](#) object into a hex string that is constituted by a "#" followed by two-digit hexadecimal representations of the red, green and blue components of the colour (in the range 0x00 - 0xFF). Optionally also includes opacity (alpha channel) data.

Parameters

<i>includeAlpha</i>	Whether two additional hex digits representing the colour's opacity (alpha channel) should be included in the string.
---------------------	---

Returns

A hex colour string.

Definition at line 352 of file Graphics.cs.

6.5.2.15 WithAlpha() [1/4]

```
Colour VectSharp.Colour.WithAlpha (  
    byte alpha )
```

Create a new [Colour](#) with the same RGB components as the current [Colour](#), but with the specified *alpha* .

Parameters

<i>alpha</i>	The alpha component of the new Colour .
--------------	---

Returns

A [Colour](#) struct with the same RGB components as the current [Colour](#) and the specified *alpha* .

Definition at line 505 of file Graphics.cs.

6.5.2.16 WithAlpha() [2/4]

```
static Colour VectSharp.Colour.WithAlpha (  
    Colour original,  
    byte alpha ) [static]
```

Create a new [Colour](#) with the same RGB components as the *original* [Colour](#), but with the specified *alpha* .

Parameters

<i>original</i>	The original Colour from which the RGB components will be taken.
<i>alpha</i>	The alpha component of the new Colour .

Returns

A [Colour](#) struct with the same RGB components as the *original* [Colour](#) and the specified *alpha* .

Definition at line 485 of file Graphics.cs.

6.5.2.17 WithAlpha() [3/4]

```
static Colour VectSharp.Colour.WithAlpha (  
    Colour original,  
    double alpha ) [static]
```

Create a new [Colour](#) with the same RGB components as the *original* [Colour](#), but with the specified *alpha* .

Parameters

<i>original</i>	The original Colour from which the RGB components will be taken.
<i>alpha</i>	The alpha component of the new Colour .

Returns

A [Colour](#) struct with the same RGB components as the *original* [Colour](#) and the specified *alpha* .

Definition at line 474 of file Graphics.cs.

6.5.2.18 WithAlpha() [4/4]

```
Colour VectSharp.Colour.WithAlpha (
    double alpha )
```

Create a new [Colour](#) with the same RGB components as the current [Colour](#), but with the specified *alpha* .

Parameters

<i>alpha</i>	The alpha component of the new Colour .
--------------	---

Returns

A [Colour](#) struct with the same RGB components as the current [Colour](#) and the specified *alpha* .

Definition at line 495 of file Graphics.cs.

6.5.3 Member Data Documentation**6.5.3.1 A**

```
double VectSharp.Colour.A
```

Alpha component of the colour. Range: [0, 1].

Definition at line 189 of file Graphics.cs.

6.5.3.2 B

```
double VectSharp.Colour.B
```

Blue component of the colour. Range: [0, 1].

Definition at line 184 of file Graphics.cs.

6.5.3.3 G

```
double VectSharp.Colour.G
```

Green component of the colour. Range: [0, 1].

Definition at line 179 of file Graphics.cs.

6.5.3.4 H

```
double VectSharp.Colour.H
```

Converts a [Colour](#) to the HSL colour space.

Returns

A ValueType containing the H, S and L components of the [Colour](#). Each component has range [0, 1].

Definition at line 672 of file Graphics.cs.

6.5.3.5 L

```
double VectSharp.Colour.L
```

Converts a [Colour](#) to the CIE Lab colour space (under Illuminant D65).

Returns

A ValueType containing the L*, a* and b* components of the [Colour](#).

Definition at line 603 of file Graphics.cs.

6.5.3.6 R

```
double VectSharp.Colour.R
```

Red component of the colour. Range: [0, 1].

Definition at line 174 of file Graphics.cs.

6.5.3.7 X

```
double VectSharp.Colour.X
```

Converts a [Colour](#) to the CIE XYZ colour space.

Returns

A ValueTuple containing the X, Y and Z components of the [Colour](#).

Definition at line 514 of file Graphics.cs.

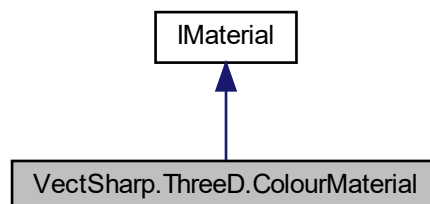
The documentation for this struct was generated from the following files:

- VectSharp/Graphics.cs
- VectSharp/StandardColours.cs

6.6 VectSharp.ThreeD.ColourMaterial Class Reference

Represents a material that always has the same colour, regardless of light.

Inheritance diagram for VectSharp.ThreeD.ColourMaterial:



Public Member Functions

- [ColourMaterial](#) ([Colour](#) colour)
Creates a new [ColourMaterial](#) instance.
- [Colour](#) [GetColour](#) (Point3D point, NormalizedVector3D surfaceNormal, Camera camera, IList< [ILightSource](#) > lights, IList< double > obstructions)
Obtains the [Colour](#) at the specified point.

Properties

- [Colour](#) [Colour](#) [get]
The colour of the material.

6.6.1 Detailed Description

Represents a material that always has the same colour, regardless of light.

Definition at line 31 of file Materials.cs.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 ColourMaterial()

```
VectSharp.ThreeD.ColourMaterial.ColourMaterial (
    Colour colour )
```

Creates a new [ColourMaterial](#) instance.

Parameters

<i>colour</i>	The colour of the material.
---------------	-----------------------------

Definition at line 42 of file Materials.cs.

6.6.3 Property Documentation

6.6.3.1 Colour

```
Colour VectSharp.ThreeD.ColourMaterial.Colour [get]
```

The colour of the material.

Definition at line 36 of file Materials.cs.

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

- VectSharp.ThreeD/Materials.cs

6.7 VectSharp.Colours Class Reference

Standard colours.

Static Public Attributes

- static [Colour Black](#) = [Colour.FromRgb](#)(0, 0, 0)
Black #000000
- static [Colour Navy](#) = [Colour.FromRgb](#)(0, 0, 128)
Navy #000080
- static [Colour DarkBlue](#) = [Colour.FromRgb](#)(0, 0, 139)
DarkBlue #00008B
- static [Colour MediumBlue](#) = [Colour.FromRgb](#)(0, 0, 205)
MediumBlue #0000CD
- static [Colour Blue](#) = [Colour.FromRgb](#)(0, 0, 255)
Blue #0000FF
- static [Colour DarkGreen](#) = [Colour.FromRgb](#)(0, 100, 0)
DarkGreen #006400
- static [Colour Green](#) = [Colour.FromRgb](#)(0, 128, 0)
Green #008000
- static [Colour Teal](#) = [Colour.FromRgb](#)(0, 128, 128)
Teal #008080
- static [Colour DarkCyan](#) = [Colour.FromRgb](#)(0, 139, 139)
DarkCyan #008B8B
- static [Colour DeepSkyBlue](#) = [Colour.FromRgb](#)(0, 191, 255)
DeepSkyBlue #00BFFF
- static [Colour DarkTurquoise](#) = [Colour.FromRgb](#)(0, 206, 209)
DarkTurquoise #00CED1
- static [Colour MediumSpringGreen](#) = [Colour.FromRgb](#)(0, 250, 154)
MediumSpringGreen #00FA9A
- static [Colour Lime](#) = [Colour.FromRgb](#)(0, 255, 0)
Lime #00FF00
- static [Colour SpringGreen](#) = [Colour.FromRgb](#)(0, 255, 127)
SpringGreen #00FF7F
- static [Colour Aqua](#) = [Colour.FromRgb](#)(0, 255, 255)
Aqua #00FFFF
- static [Colour Cyan](#) = [Colour.FromRgb](#)(0, 255, 255)
Cyan #00FFFF
- static [Colour MidnightBlue](#) = [Colour.FromRgb](#)(25, 25, 112)
MidnightBlue #191970
- static [Colour DodgerBlue](#) = [Colour.FromRgb](#)(30, 144, 255)
DodgerBlue #1E90FF
- static [Colour LightSeaGreen](#) = [Colour.FromRgb](#)(32, 178, 170)
LightSeaGreen #20B2AA
- static [Colour ForestGreen](#) = [Colour.FromRgb](#)(34, 139, 34)
ForestGreen #228B22
- static [Colour SeaGreen](#) = [Colour.FromRgb](#)(46, 139, 87)
SeaGreen #2E8B57
- static [Colour DarkSlateGray](#) = [Colour.FromRgb](#)(47, 79, 79)
DarkSlateGray #2F4F4F
- static [Colour DarkSlateGrey](#) = [Colour.FromRgb](#)(47, 79, 79)
DarkSlateGrey #2F4F4F
- static [Colour LimeGreen](#) = [Colour.FromRgb](#)(50, 205, 50)
LimeGreen #32CD32
- static [Colour MediumSeaGreen](#) = [Colour.FromRgb](#)(60, 179, 113)

- MediumSeaGreen #3CB371*
 - static **Colour Turquoise** = **Colour.FromRgb**(64, 224, 208)
- Turquoise #40E0D0*
 - static **Colour RoyalBlue** = **Colour.FromRgb**(65, 105, 225)
- RoyalBlue #4169E1*
 - static **Colour SteelBlue** = **Colour.FromRgb**(70, 130, 180)
- SteelBlue #4682B4*
 - static **Colour DarkSlateBlue** = **Colour.FromRgb**(72, 61, 139)
- DarkSlateBlue #483D8B*
 - static **Colour MediumTurquoise** = **Colour.FromRgb**(72, 209, 204)
- MediumTurquoise #48D1CC*
 - static **Colour Indigo** = **Colour.FromRgb**(75, 0, 130)
- Indigo #4B0082*
 - static **Colour DarkOliveGreen** = **Colour.FromRgb**(85, 107, 47)
- DarkOliveGreen #556B2F*
 - static **Colour CadetBlue** = **Colour.FromRgb**(95, 158, 160)
- CadetBlue #5F9EA0*
 - static **Colour CornflowerBlue** = **Colour.FromRgb**(100, 149, 237)
- CornflowerBlue #6495ED*
 - static **Colour RebeccaPurple** = **Colour.FromRgb**(102, 51, 153)
- RebeccaPurple #663399*
 - static **Colour MediumAquaMarine** = **Colour.FromRgb**(102, 205, 170)
- MediumAquaMarine #66CDAA*
 - static **Colour DimGray** = **Colour.FromRgb**(105, 105, 105)
- DimGray #696969*
 - static **Colour DimGrey** = **Colour.FromRgb**(105, 105, 105)
- DimGrey #696969*
 - static **Colour SlateBlue** = **Colour.FromRgb**(106, 90, 205)
- SlateBlue #6A5ACD*
 - static **Colour OliveDrab** = **Colour.FromRgb**(107, 142, 35)
- OliveDrab #6B8E23*
 - static **Colour SlateGray** = **Colour.FromRgb**(112, 128, 144)
- SlateGray #708090*
 - static **Colour SlateGrey** = **Colour.FromRgb**(112, 128, 144)
- SlateGrey #708090*
 - static **Colour LightSlateGray** = **Colour.FromRgb**(119, 136, 153)
- LightSlateGray #778899*
 - static **Colour LightSlateGrey** = **Colour.FromRgb**(119, 136, 153)
- LightSlateGrey #778899*
 - static **Colour MediumSlateBlue** = **Colour.FromRgb**(123, 104, 238)
- MediumSlateBlue #7B68EE*
 - static **Colour LawnGreen** = **Colour.FromRgb**(124, 252, 0)
- LawnGreen #7CFC00*
 - static **Colour Chartreuse** = **Colour.FromRgb**(127, 255, 0)
- Chartreuse #7FFF00*
 - static **Colour Aquamarine** = **Colour.FromRgb**(127, 255, 212)
- Aquamarine #7FFFD4*
 - static **Colour Maroon** = **Colour.FromRgb**(128, 0, 0)
- Maroon #800000*
 - static **Colour Purple** = **Colour.FromRgb**(128, 0, 128)
- Purple #800080*

- static **Colour Olive** = **Colour.FromRgb**(128, 128, 0)
Olive #808000
- static **Colour Gray** = **Colour.FromRgb**(128, 128, 128)
Gray #808080
- static **Colour Grey** = **Colour.FromRgb**(128, 128, 128)
Grey #808080
- static **Colour SkyBlue** = **Colour.FromRgb**(135, 206, 235)
SkyBlue #87CEEB
- static **Colour LightSkyBlue** = **Colour.FromRgb**(135, 206, 250)
LightSkyBlue #87CEFA
- static **Colour BlueViolet** = **Colour.FromRgb**(138, 43, 226)
BlueViolet #8A2BE2
- static **Colour DarkRed** = **Colour.FromRgb**(139, 0, 0)
DarkRed #8B0000
- static **Colour DarkMagenta** = **Colour.FromRgb**(139, 0, 139)
DarkMagenta #8B008B
- static **Colour SaddleBrown** = **Colour.FromRgb**(139, 69, 19)
SaddleBrown #8B4513
- static **Colour DarkSeaGreen** = **Colour.FromRgb**(143, 188, 143)
DarkSeaGreen #8FBC8F
- static **Colour LightGreen** = **Colour.FromRgb**(144, 238, 144)
LightGreen #90EE90
- static **Colour MediumPurple** = **Colour.FromRgb**(147, 112, 219)
MediumPurple #9370DB
- static **Colour DarkViolet** = **Colour.FromRgb**(148, 0, 211)
DarkViolet #9400D3
- static **Colour PaleGreen** = **Colour.FromRgb**(152, 251, 152)
PaleGreen #98FB98
- static **Colour DarkOrchid** = **Colour.FromRgb**(153, 50, 204)
DarkOrchid #9932CC
- static **Colour YellowGreen** = **Colour.FromRgb**(154, 205, 50)
YellowGreen #9ACD32
- static **Colour Sienna** = **Colour.FromRgb**(160, 82, 45)
Sienna #A0522D
- static **Colour Brown** = **Colour.FromRgb**(165, 42, 42)
Brown #A52A2A
- static **Colour DarkGray** = **Colour.FromRgb**(169, 169, 169)
DarkGray #A9A9A9
- static **Colour DarkGrey** = **Colour.FromRgb**(169, 169, 169)
DarkGrey #A9A9A9
- static **Colour LightBlue** = **Colour.FromRgb**(173, 216, 230)
LightBlue #ADD8E6
- static **Colour GreenYellow** = **Colour.FromRgb**(173, 255, 47)
GreenYellow #ADFF2F
- static **Colour PaleTurquoise** = **Colour.FromRgb**(175, 238, 238)
PaleTurquoise #AFEEEE
- static **Colour LightSteelBlue** = **Colour.FromRgb**(176, 196, 222)
LightSteelBlue #B0C4DE
- static **Colour PowderBlue** = **Colour.FromRgb**(176, 224, 230)
PowderBlue #B0E0E6
- static **Colour FireBrick** = **Colour.FromRgb**(178, 34, 34)

- FireBrick #B22222*
- static `Colour DarkGoldenRod = Colour.FromRgb(184, 134, 11)`
DarkGoldenRod #B8860B
- static `Colour MediumOrchid = Colour.FromRgb(186, 85, 211)`
MediumOrchid #BA55D3
- static `Colour RosyBrown = Colour.FromRgb(188, 143, 143)`
RosyBrown #BC8F8F
- static `Colour DarkKhaki = Colour.FromRgb(189, 183, 107)`
DarkKhaki #BDB76B
- static `Colour Silver = Colour.FromRgb(192, 192, 192)`
Silver #C0C0C0
- static `Colour MediumVioletRed = Colour.FromRgb(199, 21, 133)`
MediumVioletRed #C71585
- static `Colour IndianRed = Colour.FromRgb(205, 92, 92)`
IndianRed #CD5C5C
- static `Colour Peru = Colour.FromRgb(205, 133, 63)`
Peru #CD853F
- static `Colour Chocolate = Colour.FromRgb(210, 105, 30)`
Chocolate #D2691E
- static `Colour Tan = Colour.FromRgb(210, 180, 140)`
Tan #D2B48C
- static `Colour LightGray = Colour.FromRgb(211, 211, 211)`
LightGray #D3D3D3
- static `Colour LightGrey = Colour.FromRgb(211, 211, 211)`
LightGrey #D3D3D3
- static `Colour Thistle = Colour.FromRgb(216, 191, 216)`
Thistle #D8BFD8
- static `Colour Orchid = Colour.FromRgb(218, 112, 214)`
Orchid #DA70D6
- static `Colour GoldenRod = Colour.FromRgb(218, 165, 32)`
GoldenRod #DAA520
- static `Colour PaleVioletRed = Colour.FromRgb(219, 112, 147)`
PaleVioletRed #DB7093
- static `Colour Crimson = Colour.FromRgb(220, 20, 60)`
Crimson #DC143C
- static `Colour Gainsboro = Colour.FromRgb(220, 220, 220)`
Gainsboro #DCDCDC
- static `Colour Plum = Colour.FromRgb(221, 160, 221)`
Plum #DDA0DD
- static `Colour BurllyWood = Colour.FromRgb(222, 184, 135)`
BurllyWood #DEB887
- static `Colour LightCyan = Colour.FromRgb(224, 255, 255)`
LightCyan #E0FFFF
- static `Colour Lavender = Colour.FromRgb(230, 230, 250)`
Lavender #E6E6FA
- static `Colour DarkSalmon = Colour.FromRgb(233, 150, 122)`
DarkSalmon #E9967A
- static `Colour Violet = Colour.FromRgb(238, 130, 238)`
Violet #EE82EE
- static `Colour PaleGoldenRod = Colour.FromRgb(238, 232, 170)`
PaleGoldenRod #EEE8AA

- static `Colour LightCoral` = `Colour.FromRgb(240, 128, 128)`
LightCoral #F08080
- static `Colour Khaki` = `Colour.FromRgb(240, 230, 140)`
Khaki #F0E68C
- static `Colour AliceBlue` = `Colour.FromRgb(240, 248, 255)`
AliceBlue #F0F8FF
- static `Colour HoneyDew` = `Colour.FromRgb(240, 255, 240)`
HoneyDew #F0FFF0
- static `Colour Azure` = `Colour.FromRgb(240, 255, 255)`
Azure #F0FFFF
- static `Colour SandyBrown` = `Colour.FromRgb(244, 164, 96)`
SandyBrown #F4A460
- static `Colour Wheat` = `Colour.FromRgb(245, 222, 179)`
Wheat #F5DEB3
- static `Colour Beige` = `Colour.FromRgb(245, 245, 220)`
Beige #F5F5DC
- static `Colour WhiteSmoke` = `Colour.FromRgb(245, 245, 245)`
WhiteSmoke #F5F5F5
- static `Colour MintCream` = `Colour.FromRgb(245, 255, 250)`
MintCream #F5FFFA
- static `Colour GhostWhite` = `Colour.FromRgb(248, 248, 255)`
GhostWhite #F8F8FF
- static `Colour Salmon` = `Colour.FromRgb(250, 128, 114)`
Salmon #FA8072
- static `Colour AntiqueWhite` = `Colour.FromRgb(250, 235, 215)`
AntiqueWhite #FAEBD7
- static `Colour Linen` = `Colour.FromRgb(250, 240, 230)`
Linen #FAF0E6
- static `Colour LightGoldenRodYellow` = `Colour.FromRgb(250, 250, 210)`
LightGoldenRodYellow #FAFAD2
- static `Colour OldLace` = `Colour.FromRgb(253, 245, 230)`
OldLace #FDF5E6
- static `Colour Red` = `Colour.FromRgb(255, 0, 0)`
Red #FF0000
- static `Colour Fuchsia` = `Colour.FromRgb(255, 0, 255)`
Fuchsia #FF00FF
- static `Colour Magenta` = `Colour.FromRgb(255, 0, 255)`
Magenta #FF00FF
- static `Colour DeepPink` = `Colour.FromRgb(255, 20, 147)`
DeepPink #FF1493
- static `Colour OrangeRed` = `Colour.FromRgb(255, 69, 0)`
OrangeRed #FF4500
- static `Colour Tomato` = `Colour.FromRgb(255, 99, 71)`
Tomato #FF6347
- static `Colour HotPink` = `Colour.FromRgb(255, 105, 180)`
HotPink #FF69B4
- static `Colour Coral` = `Colour.FromRgb(255, 127, 80)`
Coral #FF7F50
- static `Colour DarkOrange` = `Colour.FromRgb(255, 140, 0)`
DarkOrange #FF8C00
- static `Colour LightSalmon` = `Colour.FromRgb(255, 160, 122)`

- LightSalmon #FFA07A*
- static **Colour Orange** = **Colour.FromRgb**(255, 165, 0)
- Orange #FFA500*
- static **Colour LightPink** = **Colour.FromRgb**(255, 182, 193)
- LightPink #FFB6C1*
- static **Colour Pink** = **Colour.FromRgb**(255, 192, 203)
- Pink #FFC0CB*
- static **Colour Gold** = **Colour.FromRgb**(255, 215, 0)
- Gold #FFD700*
- static **Colour PeachPuff** = **Colour.FromRgb**(255, 218, 185)
- PeachPuff #FFDAB9*
- static **Colour NavajoWhite** = **Colour.FromRgb**(255, 222, 173)
- NavajoWhite #FFDEAD*
- static **Colour Moccasin** = **Colour.FromRgb**(255, 228, 181)
- Moccasin #FFE4B5*
- static **Colour Bisque** = **Colour.FromRgb**(255, 228, 196)
- Bisque #FFE4C4*
- static **Colour MistyRose** = **Colour.FromRgb**(255, 228, 225)
- MistyRose #FFE4E1*
- static **Colour BlanchedAlmond** = **Colour.FromRgb**(255, 235, 205)
- BlanchedAlmond #FFEBCD*
- static **Colour PapayaWhip** = **Colour.FromRgb**(255, 239, 213)
- PapayaWhip #FFEFD5*
- static **Colour LavenderBlush** = **Colour.FromRgb**(255, 240, 245)
- LavenderBlush #FFF0F5*
- static **Colour SeaShell** = **Colour.FromRgb**(255, 245, 238)
- SeaShell #FFF5EE*
- static **Colour Cornsilk** = **Colour.FromRgb**(255, 248, 220)
- Cornsilk #FFF8DC*
- static **Colour LemonChiffon** = **Colour.FromRgb**(255, 250, 205)
- LemonChiffon #FFFACD*
- static **Colour FloralWhite** = **Colour.FromRgb**(255, 250, 240)
- FloralWhite #FFFAF0*
- static **Colour Snow** = **Colour.FromRgb**(255, 250, 250)
- Snow #FFFAFA*
- static **Colour Yellow** = **Colour.FromRgb**(255, 255, 0)
- Yellow #FFFF00*
- static **Colour LightYellow** = **Colour.FromRgb**(255, 255, 224)
- LightYellow #FFFFE0*
- static **Colour Ivory** = **Colour.FromRgb**(255, 255, 240)
- Ivory #FFFFF0*
- static **Colour White** = **Colour.FromRgb**(255, 255, 255)
- White #FFFFFF*

6.7.1 Detailed Description

Standard colours.

Definition at line 182 of file StandardColours.cs.

6.7.2 Member Data Documentation

6.7.2.1 AliceBlue

```
Colour VectSharp.Colours.AliceBlue = Colour.FromRgb(240, 248, 255) [static]
```

AliceBlue #F0F8FF

Definition at line 599 of file StandardColours.cs.

6.7.2.2 AntiqueWhite

```
Colour VectSharp.Colours.AntiqueWhite = Colour.FromRgb(250, 235, 215) [static]
```

AntiqueWhite #FAEBD7

Definition at line 639 of file StandardColours.cs.

6.7.2.3 Aqua

```
Colour VectSharp.Colours.Aqua = Colour.FromRgb(0, 255, 255) [static]
```

Aqua #00FFFF

Definition at line 243 of file StandardColours.cs.

6.7.2.4 Aquamarine

```
Colour VectSharp.Colours.Aquamarine = Colour.FromRgb(127, 255, 212) [static]
```

Aquamarine #7FFFD4

Definition at line 375 of file StandardColours.cs.

6.7.2.5 Azure

```
Colour VectSharp.Colours.Azure = Colour.FromRgb(240, 255, 255) [static]
```

Azure #F0FFFF

Definition at line 607 of file StandardColours.cs.

6.7.2.6 Beige

```
Colour VectSharp.Colours.Beige = Colour.FromRgb(245, 245, 220) [static]
```

Beige #F5F5DC

Definition at line 619 of file StandardColours.cs.

6.7.2.7 Bisque

```
Colour VectSharp.Colours.Bisque = Colour.FromRgb(255, 228, 196) [static]
```

Bisque #FFE4C4

Definition at line 723 of file StandardColours.cs.

6.7.2.8 Black

```
Colour VectSharp.Colours.Black = Colour.FromRgb(0, 0, 0) [static]
```

Black #000000

Definition at line 187 of file StandardColours.cs.

6.7.2.9 BlanchedAlmond

```
Colour VectSharp.Colours.BlanchedAlmond = Colour.FromRgb(255, 235, 205) [static]
```

BlanchedAlmond #FFEBCD

Definition at line 731 of file StandardColours.cs.

6.7.2.10 Blue

```
Colour VectSharp.Colours.Blue = Colour.FromRgb(0, 0, 255) [static]
```

Blue #0000FF

Definition at line 203 of file StandardColours.cs.

6.7.2.11 BlueViolet

```
Colour VectSharp.Colours.BlueViolet = Colour.FromRgb(138, 43, 226) [static]
```

BlueViolet #8A2BE2

Definition at line 407 of file StandardColours.cs.

6.7.2.12 Brown

```
Colour VectSharp.Colours.Brown = Colour.FromRgb(165, 42, 42) [static]
```

Brown #A52A2A

Definition at line 455 of file StandardColours.cs.

6.7.2.13 BurlyWood

```
Colour VectSharp.Colours.BurlyWood = Colour.FromRgb(222, 184, 135) [static]
```

BurlyWood #DEB887

Definition at line 567 of file StandardColours.cs.

6.7.2.14 CadetBlue

```
Colour VectSharp.Colours.CadetBlue = Colour.FromRgb(95, 158, 160) [static]
```

CadetBlue #5F9EA0

Definition at line 315 of file StandardColours.cs.

6.7.2.15 Chartreuse

```
Colour VectSharp.Colours.Chartreuse = Colour.FromRgb(127, 255, 0) [static]
```

Chartreuse #7FFF00

Definition at line 371 of file StandardColours.cs.

6.7.2.16 Chocolate

```
Colour VectSharp.Colours.Chocolate = Colour.FromRgb(210, 105, 30) [static]
```

Chocolate #D2691E

Definition at line 523 of file StandardColours.cs.

6.7.2.17 Coral

```
Colour VectSharp.Colours.Coral = Colour.FromRgb(255, 127, 80) [static]
```

Coral #FF7F50

Definition at line 683 of file StandardColours.cs.

6.7.2.18 CornflowerBlue

```
Colour VectSharp.Colours.CornflowerBlue = Colour.FromRgb(100, 149, 237) [static]
```

CornflowerBlue #6495ED

Definition at line 319 of file StandardColours.cs.

6.7.2.19 Cornsilk

```
Colour VectSharp.Colours.Cornsilk = Colour.FromRgb(255, 248, 220) [static]
```

Cornsilk #FFF8DC

Definition at line 747 of file StandardColours.cs.

6.7.2.20 Crimson

```
Colour VectSharp.Colours.Crimson = Colour.FromRgb(220, 20, 60) [static]
```

Crimson #DC143C

Definition at line 555 of file StandardColours.cs.

6.7.2.21 Cyan

```
Colour VectSharp.Colours.Cyan = Colour.FromRgb(0, 255, 255) [static]
```

Cyan #00FFFF

Definition at line 247 of file StandardColours.cs.

6.7.2.22 DarkBlue

```
Colour VectSharp.Colours.DarkBlue = Colour.FromRgb(0, 0, 139) [static]
```

DarkBlue #00008B

Definition at line 195 of file StandardColours.cs.

6.7.2.23 DarkCyan

```
Colour VectSharp.Colours.DarkCyan = Colour.FromRgb(0, 139, 139) [static]
```

DarkCyan #008B8B

Definition at line 219 of file StandardColours.cs.

6.7.2.24 DarkGoldenRod

```
Colour VectSharp.Colours.DarkGoldenRod = Colour.FromRgb(184, 134, 11) [static]
```

DarkGoldenRod #B8860B

Definition at line 491 of file StandardColours.cs.

6.7.2.25 DarkGray

```
Colour VectSharp.Colours.DarkGray = Colour.FromRgb(169, 169, 169) [static]
```

DarkGray #A9A9A9

Definition at line 459 of file StandardColours.cs.

6.7.2.26 DarkGreen

```
Colour VectSharp.Colours.DarkGreen = Colour.FromRgb(0, 100, 0) [static]
```

DarkGreen #006400

Definition at line 207 of file StandardColours.cs.

6.7.2.27 DarkGrey

```
Colour VectSharp.Colours.DarkGrey = Colour.FromRgb(169, 169, 169) [static]
```

DarkGrey #A9A9A9

Definition at line 463 of file StandardColours.cs.

6.7.2.28 DarkKhaki

```
Colour VectSharp.Colours.DarkKhaki = Colour.FromRgb(189, 183, 107) [static]
```

DarkKhaki #BDB76B

Definition at line 503 of file StandardColours.cs.

6.7.2.29 DarkMagenta

```
Colour VectSharp.Colours.DarkMagenta = Colour.FromRgb(139, 0, 139) [static]
```

DarkMagenta #8B008B

Definition at line 415 of file StandardColours.cs.

6.7.2.30 DarkOliveGreen

```
Colour VectSharp.Colours.DarkOliveGreen = Colour.FromRgb(85, 107, 47) [static]
```

DarkOliveGreen #556B2F

Definition at line 311 of file StandardColours.cs.

6.7.2.31 DarkOrange

```
Colour VectSharp.Colours.DarkOrange = Colour.FromRgb(255, 140, 0) [static]
```

DarkOrange #FF8C00

Definition at line 687 of file StandardColours.cs.

6.7.2.32 DarkOrchid

```
Colour VectSharp.Colours.DarkOrchid = Colour.FromRgb(153, 50, 204) [static]
```

DarkOrchid #9932CC

Definition at line 443 of file StandardColours.cs.

6.7.2.33 DarkRed

```
Colour VectSharp.Colours.DarkRed = Colour.FromRgb(139, 0, 0) [static]
```

DarkRed #8B0000

Definition at line 411 of file StandardColours.cs.

6.7.2.34 DarkSalmon

```
Colour VectSharp.Colours.DarkSalmon = Colour.FromRgb(233, 150, 122) [static]
```

DarkSalmon #E9967A

Definition at line 579 of file StandardColours.cs.

6.7.2.35 DarkSeaGreen

```
Colour VectSharp.Colours.DarkSeaGreen = Colour.FromRgb(143, 188, 143) [static]
```

DarkSeaGreen #8FBC8F

Definition at line 423 of file StandardColours.cs.

6.7.2.36 DarkSlateBlue

```
Colour VectSharp.Colours.DarkSlateBlue = Colour.FromRgb(72, 61, 139) [static]
```

DarkSlateBlue #483D8B

Definition at line 299 of file StandardColours.cs.

6.7.2.37 DarkSlateGray

```
Colour VectSharp.Colours.DarkSlateGray = Colour.FromRgb(47, 79, 79) [static]
```

DarkSlateGray #2F4F4F

Definition at line 271 of file StandardColours.cs.

6.7.2.38 DarkSlateGrey

```
Colour VectSharp.Colours.DarkSlateGrey = Colour.FromRgb(47, 79, 79) [static]
```

DarkSlateGrey #2F4F4F

Definition at line 275 of file StandardColours.cs.

6.7.2.39 DarkTurquoise

```
Colour VectSharp.Colours.DarkTurquoise = Colour.FromRgb(0, 206, 209) [static]
```

DarkTurquoise #00CED1

Definition at line 227 of file StandardColours.cs.

6.7.2.40 DarkViolet

```
Colour VectSharp.Colours.DarkViolet = Colour.FromRgb(148, 0, 211) [static]
```

DarkViolet #9400D3

Definition at line 435 of file StandardColours.cs.

6.7.2.41 DeepPink

```
Colour VectSharp.Colours.DeepPink = Colour.FromRgb(255, 20, 147) [static]
```

DeepPink #FF1493

Definition at line 667 of file StandardColours.cs.

6.7.2.42 DeepSkyBlue

```
Colour VectSharp.Colours.DeepSkyBlue = Colour.FromRgb(0, 191, 255) [static]
```

DeepSkyBlue #00BFFF

Definition at line 223 of file StandardColours.cs.

6.7.2.43 DimGray

```
Colour VectSharp.Colours.DimGray = Colour.FromRgb(105, 105, 105) [static]
```

DimGray #696969

Definition at line 331 of file StandardColours.cs.

6.7.2.44 DimGrey

```
Colour VectSharp.Colours.DimGrey = Colour.FromRgb(105, 105, 105) [static]
```

DimGrey #696969

Definition at line 335 of file StandardColours.cs.

6.7.2.45 DodgerBlue

```
Colour VectSharp.Colours.DodgerBlue = Colour.FromRgb(30, 144, 255) [static]
```

DodgerBlue #1E90FF

Definition at line 255 of file StandardColours.cs.

6.7.2.46 FireBrick

```
Colour VectSharp.Colours.FireBrick = Colour.FromRgb(178, 34, 34) [static]
```

FireBrick #B22222

Definition at line 487 of file StandardColours.cs.

6.7.2.47 FloralWhite

```
Colour VectSharp.Colours.FloralWhite = Colour.FromRgb(255, 250, 240) [static]
```

FloralWhite #FFFAF0

Definition at line 755 of file StandardColours.cs.

6.7.2.48 ForestGreen

```
Colour VectSharp.Colours.ForestGreen = Colour.FromRgb(34, 139, 34) [static]
```

ForestGreen #228B22

Definition at line 263 of file StandardColours.cs.

6.7.2.49 Fuchsia

```
Colour VectSharp.Colours.Fuchsia = Colour.FromRgb(255, 0, 255) [static]
```

Fuchsia #FF00FF

Definition at line 659 of file StandardColours.cs.

6.7.2.50 Gainsboro

```
Colour VectSharp.Colours.Gainsboro = Colour.FromRgb(220, 220, 220) [static]
```

Gainsboro #DCDCDC

Definition at line 559 of file StandardColours.cs.

6.7.2.51 GhostWhite

```
Colour VectSharp.Colours.GhostWhite = Colour.FromRgb(248, 248, 255) [static]
```

GhostWhite #F8F8FF

Definition at line 631 of file StandardColours.cs.

6.7.2.52 Gold

```
Colour VectSharp.Colours.Gold = Colour.FromRgb(255, 215, 0) [static]
```

Gold #FFD700

Definition at line 707 of file StandardColours.cs.

6.7.2.53 GoldenRod

```
Colour VectSharp.Colours.GoldenRod = Colour.FromRgb(218, 165, 32) [static]
```

GoldenRod #DAA520

Definition at line 547 of file StandardColours.cs.

6.7.2.54 Gray

```
Colour VectSharp.Colours.Gray = Colour.FromRgb(128, 128, 128) [static]
```

Gray #808080

Definition at line 391 of file StandardColours.cs.

6.7.2.55 Green

```
Colour VectSharp.Colours.Green = Colour.FromRgb(0, 128, 0) [static]
```

Green #008000

Definition at line 211 of file StandardColours.cs.

6.7.2.56 GreenYellow

```
Colour VectSharp.Colours.GreenYellow = Colour.FromRgb(173, 255, 47) [static]
```

GreenYellow #ADFF2F

Definition at line 471 of file StandardColours.cs.

6.7.2.57 Grey

```
Colour VectSharp.Colours.Grey = Colour.FromRgb(128, 128, 128) [static]
```

Grey #808080

Definition at line 395 of file StandardColours.cs.

6.7.2.58 HoneyDew

```
Colour VectSharp.Colours.HoneyDew = Colour.FromRgb(240, 255, 240) [static]
```

HoneyDew #F0FFF0

Definition at line 603 of file StandardColours.cs.

6.7.2.59 HotPink

```
Colour VectSharp.Colours.HotPink = Colour.FromRgb(255, 105, 180) [static]
```

HotPink #FF69B4

Definition at line 679 of file StandardColours.cs.

6.7.2.60 IndianRed

```
Colour VectSharp.Colours.IndianRed = Colour.FromRgb(205, 92, 92) [static]
```

IndianRed #CD5C5C

Definition at line 515 of file StandardColours.cs.

6.7.2.61 Indigo

```
Colour VectSharp.Colours.Indigo = Colour.FromRgb(75, 0, 130) [static]
```

Indigo #4B0082

Definition at line 307 of file StandardColours.cs.

6.7.2.62 Ivory

```
Colour VectSharp.Colours.Ivory = Colour.FromRgb(255, 255, 240) [static]
```

Ivory #FFFFFF0

Definition at line 771 of file StandardColours.cs.

6.7.2.63 Khaki

```
Colour VectSharp.Colours.Khaki = Colour.FromRgb(240, 230, 140) [static]
```

Khaki #F0E68C

Definition at line 595 of file StandardColours.cs.

6.7.2.64 Lavender

```
Colour VectSharp.Colours.Lavender = Colour.FromRgb(230, 230, 250) [static]
```

Lavender #E6E6FA

Definition at line 575 of file StandardColours.cs.

6.7.2.65 LavenderBlush

```
Colour VectSharp.Colours.LavenderBlush = Colour.FromRgb(255, 240, 245) [static]
```

LavenderBlush #FFF0F5

Definition at line 739 of file StandardColours.cs.

6.7.2.66 LawnGreen

```
Colour VectSharp.Colours.LawnGreen = Colour.FromRgb(124, 252, 0) [static]
```

LawnGreen #7CFC00

Definition at line 367 of file StandardColours.cs.

6.7.2.67 LemonChiffon

```
Colour VectSharp.Colours.LemonChiffon = Colour.FromRgb(255, 250, 205) [static]
```

LemonChiffon #FFFACD

Definition at line 751 of file StandardColours.cs.

6.7.2.68 LightBlue

```
Colour VectSharp.Colours.LightBlue = Colour.FromRgb(173, 216, 230) [static]
```

LightBlue #ADD8E6

Definition at line 467 of file StandardColours.cs.

6.7.2.69 LightCoral

```
Colour VectSharp.Colours.LightCoral = Colour.FromRgb(240, 128, 128) [static]
```

LightCoral #F08080

Definition at line 591 of file StandardColours.cs.

6.7.2.70 LightCyan

```
Colour VectSharp.Colours.LightCyan = Colour.FromRgb(224, 255, 255) [static]
```

LightCyan #E0FFFF

Definition at line 571 of file StandardColours.cs.

6.7.2.71 LightGoldenRodYellow

```
Colour VectSharp.Colours.LightGoldenRodYellow = Colour.FromRgb(250, 250, 210) [static]
```

LightGoldenRodYellow #FAFAD2

Definition at line 647 of file StandardColours.cs.

6.7.2.72 LightGray

```
Colour VectSharp.Colours.LightGray = Colour.FromRgb(211, 211, 211) [static]
```

LightGray #D3D3D3

Definition at line 531 of file StandardColours.cs.

6.7.2.73 LightGreen

```
Colour VectSharp.Colours.LightGreen = Colour.FromRgb(144, 238, 144) [static]
```

LightGreen #90EE90

Definition at line 427 of file StandardColours.cs.

6.7.2.74 LightGrey

```
Colour VectSharp.Colours.LightGrey = Colour.FromRgb(211, 211, 211) [static]
```

LightGrey #D3D3D3

Definition at line 535 of file StandardColours.cs.

6.7.2.75 LightPink

```
Colour VectSharp.Colours.LightPink = Colour.FromRgb(255, 182, 193) [static]
```

LightPink #FFB6C1

Definition at line 699 of file StandardColours.cs.

6.7.2.76 LightSalmon

```
Colour VectSharp.Colours.LightSalmon = Colour.FromRgb(255, 160, 122) [static]
```

LightSalmon #FFA07A

Definition at line 691 of file StandardColours.cs.

6.7.2.77 LightSeaGreen

```
Colour VectSharp.Colours.LightSeaGreen = Colour.FromRgb(32, 178, 170) [static]
```

LightSeaGreen #20B2AA

Definition at line 259 of file StandardColours.cs.

6.7.2.78 LightSkyBlue

```
Colour VectSharp.Colours.LightSkyBlue = Colour.FromRgb(135, 206, 250) [static]
```

LightSkyBlue #87CEFA

Definition at line 403 of file StandardColours.cs.

6.7.2.79 LightSlateGray

```
Colour VectSharp.Colours.LightSlateGray = Colour.FromRgb(119, 136, 153) [static]
```

LightSlateGray #778899

Definition at line 355 of file StandardColours.cs.

6.7.2.80 LightSlateGrey

```
Colour VectSharp.Colours.LightSlateGrey = Colour.FromRgb(119, 136, 153) [static]
```

LightSlateGrey #778899

Definition at line 359 of file StandardColours.cs.

6.7.2.81 LightSteelBlue

```
Colour VectSharp.Colours.LightSteelBlue = Colour.FromRgb(176, 196, 222) [static]
```

LightSteelBlue #B0C4DE

Definition at line 479 of file StandardColours.cs.

6.7.2.82 LightYellow

```
Colour VectSharp.Colours.LightYellow = Colour.FromRgb(255, 255, 224) [static]
```

LightYellow #FFFFE0

Definition at line 767 of file StandardColours.cs.

6.7.2.83 Lime

```
Colour VectSharp.Colours.Lime = Colour.FromRgb(0, 255, 0) [static]
```

Lime #00FF00

Definition at line 235 of file StandardColours.cs.

6.7.2.84 LimeGreen

```
Colour VectSharp.Colours.LimeGreen = Colour.FromRgb(50, 205, 50) [static]
```

LimeGreen #32CD32

Definition at line 279 of file StandardColours.cs.

6.7.2.85 Linen

```
Colour VectSharp.Colours.Linen = Colour.FromRgb(250, 240, 230) [static]
```

Linen #FAF0E6

Definition at line 643 of file StandardColours.cs.

6.7.2.86 Magenta

```
Colour VectSharp.Colours.Magenta = Colour.FromRgb(255, 0, 255) [static]
```

Magenta #FF00FF

Definition at line 663 of file StandardColours.cs.

6.7.2.87 Maroon

```
Colour VectSharp.Colours.Maroon = Colour.FromRgb(128, 0, 0) [static]
```

Maroon #800000

Definition at line 379 of file StandardColours.cs.

6.7.2.88 MediumAquaMarine

```
Colour VectSharp.Colours.MediumAquaMarine = Colour.FromRgb(102, 205, 170) [static]
```

MediumAquaMarine #66CDAA

Definition at line 327 of file StandardColours.cs.

6.7.2.89 MediumBlue

```
Colour VectSharp.Colours.MediumBlue = Colour.FromRgb(0, 0, 205) [static]
```

MediumBlue #0000CD

Definition at line 199 of file StandardColours.cs.

6.7.2.90 MediumOrchid

```
Colour VectSharp.Colours.MediumOrchid = Colour.FromRgb(186, 85, 211) [static]
```

MediumOrchid #BA55D3

Definition at line 495 of file StandardColours.cs.

6.7.2.91 MediumPurple

```
Colour VectSharp.Colours.MediumPurple = Colour.FromRgb(147, 112, 219) [static]
```

MediumPurple #9370DB

Definition at line 431 of file StandardColours.cs.

6.7.2.92 MediumSeaGreen

```
Colour VectSharp.Colours.MediumSeaGreen = Colour.FromRgb(60, 179, 113) [static]
```

MediumSeaGreen #3CB371

Definition at line 283 of file StandardColours.cs.

6.7.2.93 MediumSlateBlue

```
Colour VectSharp.Colours.MediumSlateBlue = Colour.FromRgb(123, 104, 238) [static]
```

MediumSlateBlue #7B68EE

Definition at line 363 of file StandardColours.cs.

6.7.2.94 MediumSpringGreen

```
Colour VectSharp.Colours.MediumSpringGreen = Colour.FromRgb(0, 250, 154) [static]
```

MediumSpringGreen #00FA9A

Definition at line 231 of file StandardColours.cs.

6.7.2.95 MediumTurquoise

```
Colour VectSharp.Colours.MediumTurquoise = Colour.FromRgb(72, 209, 204) [static]
```

MediumTurquoise #48D1CC

Definition at line 303 of file StandardColours.cs.

6.7.2.96 MediumVioletRed

```
Colour VectSharp.Colours.MediumVioletRed = Colour.FromRgb(199, 21, 133) [static]
```

MediumVioletRed #C71585

Definition at line 511 of file StandardColours.cs.

6.7.2.97 MidnightBlue

```
Colour VectSharp.Colours.MidnightBlue = Colour.FromRgb(25, 25, 112) [static]
```

MidnightBlue #191970

Definition at line 251 of file StandardColours.cs.

6.7.2.98 MintCream

```
Colour VectSharp.Colours.MintCream = Colour.FromRgb(245, 255, 250) [static]
```

MintCream #F5FFFA

Definition at line 627 of file StandardColours.cs.

6.7.2.99 MistyRose

```
Colour VectSharp.Colours.MistyRose = Colour.FromRgb(255, 228, 225) [static]
```

MistyRose #FFE4E1

Definition at line 727 of file StandardColours.cs.

6.7.2.100 Moccasin

```
Colour VectSharp.Colours.Moccasin = Colour.FromRgb(255, 228, 181) [static]
```

Moccasin #FFE4B5

Definition at line 719 of file StandardColours.cs.

6.7.2.101 NavajoWhite

```
Colour VectSharp.Colours.NavajoWhite = Colour.FromRgb(255, 222, 173) [static]
```

NavajoWhite #FFDEAD

Definition at line 715 of file StandardColours.cs.

6.7.2.102 Navy

```
Colour VectSharp.Colours.Navy = Colour.FromRgb(0, 0, 128) [static]
```

Navy #000080

Definition at line 191 of file StandardColours.cs.

6.7.2.103 OldLace

```
Colour VectSharp.Colours.OldLace = Colour.FromRgb(253, 245, 230) [static]
```

OldLace #FDF5E6

Definition at line 651 of file StandardColours.cs.

6.7.2.104 Olive

```
Colour VectSharp.Colours.Olive = Colour.FromRgb(128, 128, 0) [static]
```

Olive #808000

Definition at line 387 of file StandardColours.cs.

6.7.2.105 OliveDrab

```
Colour VectSharp.Colours.OliveDrab = Colour.FromRgb(107, 142, 35) [static]
```

OliveDrab #6B8E23

Definition at line 343 of file StandardColours.cs.

6.7.2.106 Orange

```
Colour VectSharp.Colours.Orange = Colour.FromRgb(255, 165, 0) [static]
```

Orange #FFA500

Definition at line 695 of file StandardColours.cs.

6.7.2.107 OrangeRed

```
Colour VectSharp.Colours.OrangeRed = Colour.FromRgb(255, 69, 0) [static]
```

OrangeRed #FF4500

Definition at line 671 of file StandardColours.cs.

6.7.2.108 Orchid

```
Colour VectSharp.Colours.Orchid = Colour.FromRgb(218, 112, 214) [static]
```

Orchid #DA70D6

Definition at line 543 of file StandardColours.cs.

6.7.2.109 PaleGoldenRod

```
Colour VectSharp.Colours.PaleGoldenRod = Colour.FromRgb(238, 232, 170) [static]
```

PaleGoldenRod #EEE8AA

Definition at line 587 of file StandardColours.cs.

6.7.2.110 PaleGreen

```
Colour VectSharp.Colours.PaleGreen = Colour.FromRgb(152, 251, 152) [static]
```

PaleGreen #98FB98

Definition at line 439 of file StandardColours.cs.

6.7.2.111 PaleTurquoise

```
Colour VectSharp.Colours.PaleTurquoise = Colour.FromRgb(175, 238, 238) [static]
```

PaleTurquoise #AFEEEE

Definition at line 475 of file StandardColours.cs.

6.7.2.112 PaleVioletRed

```
Colour VectSharp.Colours.PaleVioletRed = Colour.FromRgb(219, 112, 147) [static]
```

PaleVioletRed #DB7093

Definition at line 551 of file StandardColours.cs.

6.7.2.113 PapayaWhip

```
Colour VectSharp.Colours.PapayaWhip = Colour.FromRgb(255, 239, 213) [static]
```

PapayaWhip #FFEFD5

Definition at line 735 of file StandardColours.cs.

6.7.2.114 PeachPuff

```
Colour VectSharp.Colours.PeachPuff = Colour.FromRgb(255, 218, 185) [static]
```

PeachPuff #FFDAB9

Definition at line 711 of file StandardColours.cs.

6.7.2.115 Peru

```
Colour VectSharp.Colours.Peru = Colour.FromRgb(205, 133, 63) [static]
```

Peru #CD853F

Definition at line 519 of file StandardColours.cs.

6.7.2.116 Pink

```
Colour VectSharp.Colours.Pink = Colour.FromRgb(255, 192, 203) [static]
```

Pink #FFC0CB

Definition at line 703 of file StandardColours.cs.

6.7.2.117 Plum

```
Colour VectSharp.Colours.Plum = Colour.FromRgb(221, 160, 221) [static]
```

Plum #DDA0DD

Definition at line 563 of file StandardColours.cs.

6.7.2.118 PowderBlue

```
Colour VectSharp.Colours.PowderBlue = Colour.FromRgb(176, 224, 230) [static]
```

PowderBlue #B0E0E6

Definition at line 483 of file StandardColours.cs.

6.7.2.119 Purple

```
Colour VectSharp.Colours.Purple = Colour.FromRgb(128, 0, 128) [static]
```

Purple #800080

Definition at line 383 of file StandardColours.cs.

6.7.2.120 RebeccaPurple

```
Colour VectSharp.Colours.RebeccaPurple = Colour.FromRgb(102, 51, 153) [static]
```

RebeccaPurple #663399

Definition at line 323 of file StandardColours.cs.

6.7.2.121 Red

```
Colour VectSharp.Colours.Red = Colour.FromRgb(255, 0, 0) [static]
```

Red #FF0000

Definition at line 655 of file StandardColours.cs.

6.7.2.122 RosyBrown

```
Colour VectSharp.Colours.RosyBrown = Colour.FromRgb(188, 143, 143) [static]
```

RosyBrown #BC8F8F

Definition at line 499 of file StandardColours.cs.

6.7.2.123 RoyalBlue

```
Colour VectSharp.Colours.RoyalBlue = Colour.FromRgb(65, 105, 225) [static]
```

RoyalBlue #4169E1

Definition at line 291 of file StandardColours.cs.

6.7.2.124 SaddleBrown

```
Colour VectSharp.Colours.SaddleBrown = Colour.FromRgb(139, 69, 19) [static]
```

SaddleBrown #8B4513

Definition at line 419 of file StandardColours.cs.

6.7.2.125 Salmon

```
Colour VectSharp.Colours.Salmon = Colour.FromRgb(250, 128, 114) [static]
```

Salmon #FA8072

Definition at line 635 of file StandardColours.cs.

6.7.2.126 SandyBrown

```
Colour VectSharp.Colours.SandyBrown = Colour.FromRgb(244, 164, 96) [static]
```

SandyBrown #F4A460

Definition at line 611 of file StandardColours.cs.

6.7.2.127 SeaGreen

```
Colour VectSharp.Colours.SeaGreen = Colour.FromRgb(46, 139, 87) [static]
```

SeaGreen #2E8B57

Definition at line 267 of file StandardColours.cs.

6.7.2.128 SeaShell

```
Colour VectSharp.Colours.SeaShell = Colour.FromRgb(255, 245, 238) [static]
```

SeaShell #FFF5EE

Definition at line 743 of file StandardColours.cs.

6.7.2.129 Sienna

```
Colour VectSharp.Colours.Sienna = Colour.FromRgb(160, 82, 45) [static]
```

Sienna #A0522D

Definition at line 451 of file StandardColours.cs.

6.7.2.130 Silver

```
Colour VectSharp.Colours.Silver = Colour.FromRgb(192, 192, 192) [static]
```

Silver #C0C0C0

Definition at line 507 of file StandardColours.cs.

6.7.2.131 SkyBlue

```
Colour VectSharp.Colours.SkyBlue = Colour.FromRgb(135, 206, 235) [static]
```

SkyBlue #87CEEB

Definition at line 399 of file StandardColours.cs.

6.7.2.132 SlateBlue

```
Colour VectSharp.Colours.SlateBlue = Colour.FromRgb(106, 90, 205) [static]
```

SlateBlue #6A5ACD

Definition at line 339 of file StandardColours.cs.

6.7.2.133 SlateGray

```
Colour VectSharp.Colours.SlateGray = Colour.FromRgb(112, 128, 144) [static]
```

SlateGray #708090

Definition at line 347 of file StandardColours.cs.

6.7.2.134 SlateGrey

```
Colour VectSharp.Colours.SlateGrey = Colour.FromRgb(112, 128, 144) [static]
```

SlateGrey #708090

Definition at line 351 of file StandardColours.cs.

6.7.2.135 Snow

```
Colour VectSharp.Colours.Snow = Colour.FromRgb(255, 250, 250) [static]
```

Snow #FFFAFA

Definition at line 759 of file StandardColours.cs.

6.7.2.136 SpringGreen

```
Colour VectSharp.Colours.SpringGreen = Colour.FromRgb(0, 255, 127) [static]
```

SpringGreen #00FF7F

Definition at line 239 of file StandardColours.cs.

6.7.2.137 SteelBlue

```
Colour VectSharp.Colours.SteelBlue = Colour.FromRgb(70, 130, 180) [static]
```

SteelBlue #4682B4

Definition at line 295 of file StandardColours.cs.

6.7.2.138 Tan

```
Colour VectSharp.Colours.Tan = Colour.FromRgb(210, 180, 140) [static]
```

Tan #D2B48C

Definition at line 527 of file StandardColours.cs.

6.7.2.139 Teal

```
Colour VectSharp.Colours.Teal = Colour.FromRgb(0, 128, 128) [static]
```

Teal #008080

Definition at line 215 of file StandardColours.cs.

6.7.2.140 Thistle

```
Colour VectSharp.Colours.Thistle = Colour.FromRgb(216, 191, 216) [static]
```

Thistle #D8BFD8

Definition at line 539 of file StandardColours.cs.

6.7.2.141 Tomato

```
Colour VectSharp.Colours.Tomato = Colour.FromRgb(255, 99, 71) [static]
```

Tomato #FF6347

Definition at line 675 of file StandardColours.cs.

6.7.2.142 Turquoise

```
Colour VectSharp.Colours.Turquoise = Colour.FromRgb(64, 224, 208) [static]
```

Turquoise #40E0D0

Definition at line 287 of file StandardColours.cs.

6.7.2.143 Violet

```
Colour VectSharp.Colours.Violet = Colour.FromRgb(238, 130, 238) [static]
```

Violet #EE82EE

Definition at line 583 of file StandardColours.cs.

6.7.2.144 Wheat

```
Colour VectSharp.Colours.Wheat = Colour.FromRgb(245, 222, 179) [static]
```

Wheat #F5DEB3

Definition at line 615 of file StandardColours.cs.

6.7.2.145 White

```
Colour VectSharp.Colours.White = Colour.FromRgb(255, 255, 255) [static]
```

White #FFFFFF

Definition at line 775 of file StandardColours.cs.

6.7.2.146 WhiteSmoke

`Colour VectSharp.Colours.WhiteSmoke = Colour.FromRgb(245, 245, 245) [static]`

WhiteSmoke #F5F5F5

Definition at line 623 of file StandardColours.cs.

6.7.2.147 Yellow

`Colour VectSharp.Colours.Yellow = Colour.FromRgb(255, 255, 0) [static]`

Yellow #FFFF00

Definition at line 763 of file StandardColours.cs.

6.7.2.148 YellowGreen

`Colour VectSharp.Colours.YellowGreen = Colour.FromRgb(154, 205, 50) [static]`

YellowGreen #9ACD32

Definition at line 447 of file StandardColours.cs.

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

- VectSharp/StandardColours.cs

6.8 VectSharp.Font.DetailedFontMetrics Class Reference

Represents detailed information about the metrics of a text string when drawn with a certain font.

Properties

- double `Width` [get]
Width of the text (measured on the actual glyph outlines).
- double `Height` [get]
Height of the text (measured on the actual glyph outlines).
- double `LeftSideBearing` [get]
How much the leftmost glyph in the string overhangs the glyph origin on the left. Positive for glyphs that hang past the origin (e.g. italic 'f').
- double `RightSideBearing` [get]
How much the rightmost glyph in the string overhangs the glyph end on the right. Positive for glyphs that hang past the end (e.g. italic 'f').
- double `Top` [get]
Height of the tallest glyph in the string over the baseline. Always ≥ 0 .
- double `Bottom` [get]
Depth of the deepest glyph in the string below the baseline. Always ≤ 0 .

6.8.1 Detailed Description

Represents detailed information about the metrics of a text string when drawn with a certain font.

Definition at line 786 of file Graphics.cs.

6.8.2 Property Documentation

6.8.2.1 Bottom

```
double VectSharp.Font.DetailedFontMetrics.Bottom [get]
```

Depth of the deepest glyph in the string below the baseline. Always ≤ 0 .

Definition at line 816 of file Graphics.cs.

6.8.2.2 Height

```
double VectSharp.Font.DetailedFontMetrics.Height [get]
```

Height of the text (measured on the actual glyph outlines).

Definition at line 796 of file Graphics.cs.

6.8.2.3 LeftSideBearing

```
double VectSharp.Font.DetailedFontMetrics.LeftSideBearing [get]
```

How much the leftmost glyph in the string overhangs the glyph origin on the left. Positive for glyphs that hang past the origin (e.g. italic 'f').

Definition at line 801 of file Graphics.cs.

6.8.2.4 RightSideBearing

```
double VectSharp.Font.DetailedFontMetrics.RightSideBearing [get]
```

How much the rightmost glyph in the string overhangs the glyph end on the right. Positive for glyphs that hang past the end (e.g. italic 'f').

Definition at line 806 of file Graphics.cs.

6.8.2.5 Top

```
double VectSharp.Font.DetailedFontMetrics.Top [get]
```

Height of the tallest glyph in the string over the baseline. Always ≥ 0 .

Definition at line 811 of file Graphics.cs.

6.8.2.6 Width

```
double VectSharp.Font.DetailedFontMetrics.Width [get]
```

Width of the text (measured on the actual glyph outlines).

Definition at line 791 of file Graphics.cs.

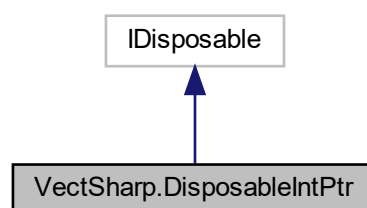
The documentation for this class was generated from the following file:

- VectSharp/Graphics.cs

6.9 VectSharp.DisposableIntPtr Class Reference

An IDisposable wrapper around an IntPtr that frees the allocated memory when it is disposed.

Inheritance diagram for VectSharp.DisposableIntPtr:



Public Member Functions

- [DisposableIntPtr](#) (IntPtr pointer)
Create a new [DisposableIntPtr](#).
- void **Dispose** ()

Public Attributes

- readonly IntPtr [InternalPointer](#)
The pointer to the unmanaged memory.

6.9.1 Detailed Description

An IDisposable wrapper around an IntPtr that frees the allocated memory when it is disposed.

Definition at line 53 of file RasterImage.cs.

6.9.2 Constructor & Destructor Documentation

6.9.2.1 DisposableIntPtr()

```
VectSharp.DisposableIntPtr.DisposableIntPtr (
    IntPtr pointer )
```

Create a new [DisposableIntPtr](#).

Parameters

<i>pointer</i>	The pointer that should be freed upon disposing of this object.
----------------	---

Definition at line 64 of file RasterImage.cs.

6.9.3 Member Data Documentation

6.9.3.1 InternalPointer

```
readonly IntPtr VectSharp.DisposableIntPtr.InternalPointer
```

The pointer to the unmanaged memory.

Definition at line 58 of file RasterImage.cs.

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

- VectSharp/RasterImage.cs

6.10 VectSharp.Document Class Reference

Represents a collection of pages.

Public Member Functions

- [Document](#) ()
Create a new document.

Public Attributes

- List< [Page](#) > [Pages](#) = new List<[Page](#)>()
The pages in the document.

6.10.1 Detailed Description

Represents a collection of pages.

Definition at line 27 of file Document.cs.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 Document()

```
VectSharp.Document.Document ( )
```

Create a new document.

Definition at line 38 of file Document.cs.

6.10.3 Member Data Documentation

6.10.3.1 Pages

```
List<Page> VectSharp.Document.Pages = new List<Page>()
```

The pages in the document.

Definition at line 32 of file Document.cs.

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

- VectSharp/Document.cs

6.11 VectSharp.Font Class Reference

Represents a typeface with a specific size.

Classes

- class [DetailedFontMetrics](#)

Represents detailed information about the metrics of a text string when drawn with a certain font.

Public Member Functions

- [Font](#) ([FontFamily](#) fontFamily, double fontSize)
Create a new [Font](#) object, given the base typeface and the font size.
- [Size MeasureText](#) (string text)
Measure the size of a text string when typeset with this font.
- [DetailedFontMetrics MeasureTextAdvanced](#) (string text)
Measure all the metrics of a text string when typeset with this font.

Properties

- double [FontSize](#) [get]
Font size, in graphics units.
- [FontFamily](#) [FontFamily](#) [get]
Font typeface.
- double [Ascent](#) [get]
Maximum height over the baseline of the usual glyphs in the font (there may be glyphs taller than this). Always ≥ 0 .
- double [Descent](#) [get]
Maximum depth below the baseline of the usual glyphs in the font (there may be glyphs deeper than this). Always ≤ 0 .
- double [YMax](#) [get]
Absolute maximum height over the baseline of the glyphs in the font. Always ≥ 0 .
- double [YMin](#) [get]
Absolute maximum depth below the baseline of the glyphs in the font. Always ≤ 0 .

6.11.1 Detailed Description

Represents a typeface with a specific size.

Definition at line 781 of file Graphics.cs.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Font()

```
VectSharp.Font.Font (
    FontFamily fontFamily,
    double fontSize )
```

Create a new [Font](#) object, given the base typeface and the font size.

Parameters

<i>fontFamily</i>	Base typeface. See FontFamily .
<i>fontSize</i>	The font size, in graphics units.

Definition at line 844 of file Graphics.cs.

6.11.3 Member Function Documentation

6.11.3.1 MeasureText()

```
Size VectSharp.Font.MeasureText (
    string text )
```

Measure the size of a text string when typeset with this font.

Parameters

<i>text</i>	The string to measure.
-------------	------------------------

Returns

A [Size](#) object representing the width and height of the text.

Definition at line 927 of file Graphics.cs.

6.11.3.2 MeasureTextAdvanced()

```
DetailedFontMetrics VectSharp.Font.MeasureTextAdvanced (
    string text )
```

Measure all the metrics of a text string when typeset with this font.

Parameters

<i>text</i>	The string to measure.
-------------	------------------------

Returns

A [DetailedFontMetrics](#) object representing the metrics of the text.

Definition at line 960 of file Graphics.cs.

6.11.4 Property Documentation

6.11.4.1 Ascent

```
double VectSharp.Font.Ascent [get]
```

Maximum height over the baseline of the usual glyphs in the font (there may be glyphs taller than this). Always ≥ 0 .

Definition at line 853 of file Graphics.cs.

6.11.4.2 Descent

```
double VectSharp.Font.Descent [get]
```

Maximum depth below the baseline of the usual glyphs in the font (there may be glyphs deeper than this). Always ≤ 0 .

Definition at line 871 of file Graphics.cs.

6.11.4.3 FontFamily

```
FontFamily VectSharp.Font.FontFamily [get]
```

Font typeface.

Definition at line 837 of file Graphics.cs.

6.11.4.4 FontSize

```
double VectSharp.Font.FontSize [get]
```

Font size, in graphics units.

Definition at line 832 of file Graphics.cs.

6.11.4.5 YMax

```
double VectSharp.Font.YMax [get]
```

Absolute maximum height over the baseline of the glyphs in the font. Always ≥ 0 .

Definition at line 889 of file Graphics.cs.

6.11.4.6 YMin

```
double VectSharp.Font.YMin [get]
```

Absolute maximum depth below the baseline of the glyphs in the font. Always ≤ 0 .

Definition at line 907 of file Graphics.cs.

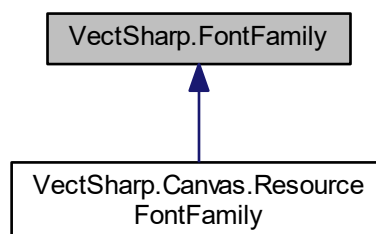
The documentation for this class was generated from the following file:

- VectSharp/Graphics.cs

6.12 VectSharp.FontFamily Class Reference

Represents a typeface.

Inheritance diagram for VectSharp.FontFamily:



Public Types

- enum [StandardFontFamilies](#) {
 [StandardFontFamilies.TimesRoman](#), [StandardFontFamilies.TimesBold](#), [StandardFontFamilies.TimesItalic](#),
 [StandardFontFamilies.TimesBoldItalic](#),
 [StandardFontFamilies.Helvetica](#), [StandardFontFamilies.HelveticaBold](#), [StandardFontFamilies.HelveticaOblique](#),
 [StandardFontFamilies.HelveticaBoldOblique](#),
 [StandardFontFamilies.Courier](#), [StandardFontFamilies.CourierBold](#), [StandardFontFamilies.CourierOblique](#),
 [StandardFontFamilies.CourierBoldOblique](#),
 [StandardFontFamilies.Symbol](#), [StandardFontFamilies.ZapfDingbats](#) }

The 14 standard font families.

Public Member Functions

- [FontFamily](#) (string fileName)
Create a new [FontFamily](#).
- [FontFamily](#) (Stream ttfStream)
Create a new [FontFamily](#).
- [FontFamily](#) ([StandardFontFamilies](#) standardFontFamily)
Create a new standard [FontFamily](#).

Static Public Attributes

- static string[] [StandardFamilies](#) = new string[] { "Times-Roman", "Times-Bold", "Times-Italic", "Times-Bold↵Italic", "Helvetica", "Helvetica-Bold", "Helvetica-Oblique", "Helvetica-BoldOblique", "Courier", "Courier-Bold", "Courier-Oblique", "Courier-BoldOblique", "Symbol", "ZapfDingbats" }
The names of the 14 standard families that are guaranteed to be displayed correctly.
- static string[] [StandardFontFamilyResources](#)
The names of the resource streams pointing to the included TrueType font files for each of the standard 14 font families.

Properties

- bool [IsStandardFamily](#) [get]
Whether this is one of the 14 standard font families or not.
- string [FileName](#) [get]
Full path to the TrueType font file for this font family (or, if this is a standard font family, name of the font family).
- [TrueTypeFile](#) [TrueTypeFile](#) [get]
*Parsed TrueType font file for this font family. See also:
See also
[VectSharp.TrueTypeFile](#)*
- bool [IsBold](#) [get]
Whether this font is bold or not. This is set based on the information included in the OS/2 table of the TrueType file.
- bool [IsItalic](#) [get]
Whether this font is italic or oblique or not. This is set based on the information included in the OS/2 table of the TrueType file.
- bool [IsOblique](#) [get]
Whether this font is oblique or not. This is set based on the information included in the OS/2 table of the TrueType file.

6.12.1 Detailed Description

Represents a typeface.

Definition at line 996 of file Graphics.cs.

6.12.2 Member Enumeration Documentation

6.12.2.1 StandardFontFamilies

```
enum VectSharp.FontFamily.StandardFontFamilies [strong]
```

The 14 standard font families.

Enumerator

TimesRoman	Serif normal regular face.
TimesBold	Serif bold regular face.
TimesItalic	Serif normal italic face.
TimesBoldItalic	Serif bold italic face.
Helvetica	Sans-serif normal regular face.
HelveticaBold	Sans-serif bold regular face.
HelveticaOblique	Sans-serif normal oblique face.
HelveticaBoldOblique	Sans-serif bold oblique face.
Courier	Monospace normal regular face.
CourierBold	Monospace bold regular face.
CourierOblique	Monospace normal oblique face.
CourierBoldOblique	Monospace bold oblique face.
Symbol	Symbol font.
ZapfDingbats	Dingbat font.

Definition at line 1035 of file Graphics.cs.

6.12.3 Constructor & Destructor Documentation

6.12.3.1 FontFamily() [1/3]

```
VectSharp.FontFamily.FontFamily (
    string fileName )
```

Create a new [FontFamily](#).

Parameters

<i>fileName</i>	The full path to the TrueType font file for this font family or the name of a standard font family.
-----------------	---

Definition at line 1138 of file Graphics.cs.

6.12.3.2 FontFamily() [2/3]

```
VectSharp.FontFamily.FontFamily (
    Stream ttfStream )
```

Create a new [FontFamily](#).

Parameters

<i>ttfStream</i>	A stream containing a file in TTF format.
------------------	---

Definition at line 1187 of file Graphics.cs.

6.12.3.3 FontFamily() [3/3]

```
VectSharp.FontFamily.FontFamily (
    StandardFontFamilies standardFontFamily )
```

Create a new standard [FontFamily](#).

Parameters

<i>standardFontFamily</i>	The standard font family.
---------------------------	---------------------------

Definition at line 1203 of file Graphics.cs.

6.12.4 Member Data Documentation

6.12.4.1 StandardFamilies

```
string [] VectSharp.FontFamily.StandardFamilies = new string[] { "Times-Roman", "Times-Bold",
"Times-Italic", "Times-BoldItalic", "Helvetica", "Helvetica-Bold", "Helvetica-Oblique", "Helvetica-Bold↵
Oblique", "Courier", "Courier-Bold", "Courier-Oblique", "Courier-BoldOblique", "Symbol", "Zapf↵
Dingbats" } [static]
```

The names of the 14 standard families that are guaranteed to be displayed correctly.

Definition at line 1014 of file Graphics.cs.

6.12.4.2 StandardFontFamilyResources

```
string [] VectSharp.FontFamily.StandardFontFamilyResources [static]
```

Initial value:

```
= new string[]
{
    "VectSharp.StandardFonts.NimbusRomNo9L-Reg.ttf",
    "VectSharp.StandardFonts.NimbusRomNo9L-Med.ttf", "VectSharp.StandardFonts.NimbusRomNo9L-RegIta.ttf",
    "VectSharp.StandardFonts.NimbusRomNo9L-MedIta.ttf",
    "VectSharp.StandardFonts.NimbusSanL-Reg.ttf", "VectSharp.StandardFonts.NimbusSanL-Bol.ttf",
    "VectSharp.StandardFonts.NimbusSanL-RegIta.ttf", "VectSharp.StandardFonts.NimbusSanL-BolIta.ttf",
    "VectSharp.StandardFonts.NimbusMono-Regular.ttf", "VectSharp.StandardFonts.NimbusMono-Bold.ttf",
    "VectSharp.StandardFonts.NimbusMono-Oblique.ttf",
    "VectSharp.StandardFonts.NimbusMono-BoldOblique.ttf",
    "VectSharp.StandardFonts.StandardSymbolsPS.ttf", "VectSharp.StandardFonts.D050000L.ttf"
}
```

The names of the resource streams pointing to the included TrueType font files for each of the standard 14 font families.

Definition at line 1019 of file Graphics.cs.

6.12.5 Property Documentation

6.12.5.1 FileName

```
string VectSharp.FontFamily.FileName [get]
```

Full path to the TrueType font file for this font family (or, if this is a standard font family, name of the font family).

Definition at line 1111 of file Graphics.cs.

6.12.5.2 IsBold

```
bool VectSharp.FontFamily.IsBold [get]
```

Whether this font is bold or not. This is set based on the information included in the OS/2 table of the TrueType file.

Definition at line 1122 of file Graphics.cs.

6.12.5.3 IsItalic

```
bool VectSharp.FontFamily.IsItalic [get]
```

Whether this font is italic or oblique or not. This is set based on the information included in the OS/2 table of the TrueType file.

Definition at line 1127 of file Graphics.cs.

6.12.5.4 IsOblique

```
bool VectSharp.FontFamily.IsOblique [get]
```

Whether this font is oblique or not. This is set based on the information included in the OS/2 table of the TrueType file.

Definition at line 1132 of file Graphics.cs.

6.12.5.5 IsStandardFamily

```
bool VectSharp.FontFamily.IsStandardFamily [get]
```

Whether this is one of the 14 standard font families or not.

Definition at line 1030 of file Graphics.cs.

6.12.5.6 TrueTypeFile

```
TrueTypeFile VectSharp.FontFamily.TrueTypeFile [get]
```

Parsed TrueType font file for this font family. See also:

See also

[VectSharp.TrueTypeFile](#)

.

Definition at line 1117 of file Graphics.cs.

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

- VectSharp/Graphics.cs

6.13 VectSharp.Graphics Class Reference

Represents an abstract drawing surface.

Public Member Functions

- void [FillPath](#) ([GraphicsPath](#) path, [Colour](#) fillColour, string tag=null)
Fill a [GraphicsPath](#).
- void [StrokePath](#) ([GraphicsPath](#) path, [Colour](#) strokeColour, double lineWidth=1, [LineCaps](#) lineCap=LineCaps.Butt, [LineJoins](#) lineJoin=LineJoins.Miter, [LineDash](#)? lineDash=null, string tag=null)
Stroke a [GraphicsPath](#).
- void [SetClippingPath](#) ([GraphicsPath](#) path)
Intersect the current clipping path with the specified [GraphicsPath](#).
- void [SetClippingPath](#) (double leftX, double topY, double width, double height)
Intersect the current clipping path with the specified rectangle.
- void [SetClippingPath](#) ([Point](#) topLeft, [Size](#) size)
Intersect the current clipping path with the specified rectangle.
- void [Rotate](#) (double angle)
Rotate the coordinate system around the origin.
- void [RotateAt](#) (double angle, [Point](#) pivot)
Rotate the coordinate system around a pivot point.

- void [Transform](#) (double a, double b, double c, double d, double e, double f)
Transform the coordinate system with the specified transformation matrix $\begin{bmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{bmatrix}$.
- void [Translate](#) (double x, double y)
Translate the coordinate system origin.
- void [Translate](#) ([Point](#) delta)
Translate the coordinate system origin.
- void [Scale](#) (double scaleX, double scaleY)
Scale the coordinate system with respect to the origin.
- void [FillRectangle](#) ([Point](#) topLeft, [Size](#) size, [Colour](#) fillColour, string tag=null)
Fill a rectangle.
- void [FillRectangle](#) (double leftX, double topY, double width, double height, [Colour](#) fillColour, string tag=null)
Fill a rectangle.
- void [StrokeRectangle](#) ([Point](#) topLeft, [Size](#) size, [Colour](#) strokeColour, double lineWidth=1, [LineCaps](#) lineCap=[LineCaps.Butt](#), [LineJoins](#) lineJoin=[LineJoins.Miter](#), [LineDash?](#) lineDash=null, string tag=null)
Stroke a rectangle.
- void [StrokeRectangle](#) (double leftX, double topY, double width, double height, [Colour](#) strokeColour, double lineWidth=1, [LineCaps](#) lineCap=[LineCaps.Butt](#), [LineJoins](#) lineJoin=[LineJoins.Miter](#), [LineDash?](#) lineDash=null, string tag=null)
Stroke a rectangle.
- void [DrawRasterImage](#) (int sourceX, int sourceY, int sourceWidth, int sourceHeight, double destinationX, double destinationY, double destinationWidth, double destinationHeight, [RasterImage](#) image, string tag=null)
Draw a raster image.
- void [DrawRasterImage](#) (double x, double y, [RasterImage](#) image, string tag=null)
Draw a raster image.
- void [DrawRasterImage](#) ([Point](#) position, [RasterImage](#) image, string tag=null)
Draw a raster image.
- void [DrawRasterImage](#) (double x, double y, double width, double height, [RasterImage](#) image, string tag=null)
Draw a raster image.
- void [DrawRasterImage](#) ([Point](#) position, [Size](#) size, [RasterImage](#) image, string tag=null)
Draw a raster image.
- void [FillText](#) ([Point](#) origin, string text, [Font](#) font, [Colour](#) fillColour, [TextBaselines](#) textBaseline=[TextBaselines.Top](#), string tag=null)
Fill a text string.
- void [FillText](#) (double originX, double originY, string text, [Font](#) font, [Colour](#) fillColour, [TextBaselines](#) textBaseline=[TextBaselines.Top](#), string tag=null)
Fill a text string.
- void [StrokeText](#) ([Point](#) origin, string text, [Font](#) font, [Colour](#) strokeColour, [TextBaselines](#) textBaseline=[TextBaselines.Top](#), double lineWidth=1, [LineCaps](#) lineCap=[LineCaps.Butt](#), [LineJoins](#) lineJoin=[LineJoins.Miter](#), [LineDash?](#) lineDash=null, string tag=null)
Stroke a text string.
- void [StrokeText](#) (double originX, double originY, string text, [Font](#) font, [Colour](#) strokeColour, [TextBaselines](#) textBaseline=[TextBaselines.Top](#), double lineWidth=1, [LineCaps](#) lineCap=[LineCaps.Butt](#), [LineJoins](#) lineJoin=[LineJoins.Miter](#), [LineDash?](#) lineDash=null, string tag=null)
Stroke a text string.
- void [FillTextOnPath](#) ([GraphicsPath](#) path, string text, [Font](#) font, [Colour](#) fillColour, double reference=0, [TextAnchors](#) anchor=[TextAnchors.Left](#), [TextBaselines](#) textBaseline=[TextBaselines.Top](#), string tag=null)
Fill a text string along a [GraphicsPath](#).
- void [StrokeTextOnPath](#) ([GraphicsPath](#) path, string text, [Font](#) font, [Colour](#) strokeColour, double reference=0, [TextAnchors](#) anchor=[TextAnchors.Left](#), [TextBaselines](#) textBaseline=[TextBaselines.Top](#), double lineWidth=1, [LineCaps](#) lineCap=[LineCaps.Butt](#), [LineJoins](#) lineJoin=[LineJoins.Miter](#), [LineDash?](#) lineDash=null, string tag=null)
Stroke a text string along a [GraphicsPath](#).
- [Size MeasureText](#) (string text, [Font](#) font)
Measure a text string. See also

See also

[Font.MeasureText\(string\)](#), [Font.MeasureTextAdvanced\(string\)](#)

and .

- void [Save](#) ()
Save the current transform state (rotation, translation, scale).
- void [Restore](#) ()
Restore the previous transform state (rotation, translation scale).
- void [CopyToIGraphicsContext](#) ([IGraphicsContext](#) destinationContext)
Copy the current graphics to an instance of a class implementing [IGraphicsContext](#).
- void [DrawGraphics](#) ([Point](#) origin, [Graphics](#) graphics)
Draws a [Graphics](#) object on the current [Graphics](#) object.
- void [DrawGraphics](#) (double originX, double originY, [Graphics](#) graphics)
Draws a [Graphics](#) object on the current [Graphics](#) object.
- [Graphics Transform](#) (Func< [Point](#), [Point](#) > transformationFunction, double linearisationResolution)
Creates a new [Graphics](#) object in which all the graphics actions have been transformed using an arbitrary transformation function. [Raster](#) images are replaced by grey rectangles.
- [Graphics Linearise](#) (double resolution)
Creates a new [Graphics](#) object by linearising all of the elements of the current instance, i.e. replacing curve segments with series of line segments that approximate them. [Raster](#) images are left unchanged.

Properties

- static [UnbalancedStackActions](#) [UnbalancedStackAction](#) = [UnbalancedStackActions.Throw](#) [get, set]
Determines how an unbalanced graphics state stack (which occurs if the number of calls to [Save](#) and [Restore](#) is not equal) will be treated. The default is [UnbalancedStackActions.Throw](#).

6.13.1 Detailed Description

Represents an abstract drawing surface.

Definition at line 2321 of file Graphics.cs.

6.13.2 Member Function Documentation

6.13.2.1 CopyToIGraphicsContext()

```
void VectSharp.Graphics.CopyToIGraphicsContext (
    IGraphicsContext destinationContext )
```

Copy the current graphics to an instance of a class implementing [IGraphicsContext](#).

Parameters

<i>destinationContext</i>	The IGraphicsContext on which the graphics are to be copied.
---------------------------	--

Definition at line 2945 of file Graphics.cs.

6.13.2.2 DrawGraphics() [1/2]

```
void VectSharp.Graphics.DrawGraphics (
    double originX,
    double originY,
    Graphics graphics )
```

Draws a [Graphics](#) object on the current [Graphics](#) object.

Parameters

<i>originX</i>	The horizontal coordinate at which to place the origin of <i>graphics</i> .
<i>originY</i>	The vertical coordinate at which to place the origin of <i>graphics</i> .
<i>graphics</i>	The Graphics object to draw on the current Graphics object.

Definition at line 3161 of file Graphics.cs.

6.13.2.3 DrawGraphics() [2/2]

```
void VectSharp.Graphics.DrawGraphics (
    Point origin,
    Graphics graphics )
```

Draws a [Graphics](#) object on the current [Graphics](#) object.

Parameters

<i>origin</i>	The point at which to place the origin of <i>graphics</i> .
<i>graphics</i>	The Graphics object to draw on the current Graphics object.

Definition at line 3143 of file Graphics.cs.

6.13.2.4 DrawRasterImage() [1/5]

```
void VectSharp.Graphics.DrawRasterImage (
    double x,
    double y,
    double width,
    double height,
    RasterImage image,
    string tag = null )
```

Draw a raster image.

Parameters

<i>x</i>	The horizontal coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>y</i>	The vertical coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>width</i>	The width of the rectangle delimiting the destination area of the image.
<i>height</i>	The height of the rectangle delimiting the destination area of the image.
<i>image</i>	The image to draw.
<i>tag</i>	A tag to identify the drawn image.

Definition at line 2564 of file Graphics.cs.

6.13.2.5 DrawRasterImage() [2/5]

```
void VectSharp.Graphics.DrawRasterImage (
    double x,
    double y,
    RasterImage image,
    string tag = null )
```

Draw a raster image.

Parameters

<i>x</i>	The horizontal coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>y</i>	The vertical coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>image</i>	The image to draw.
<i>tag</i>	A tag to identify the drawn image.

Definition at line 2539 of file Graphics.cs.

6.13.2.6 DrawRasterImage() [3/5]

```
void VectSharp.Graphics.DrawRasterImage (
    int sourceX,
    int sourceY,
    int sourceWidth,
    int sourceHeight,
    double destinationX,
    double destinationY,
    double destinationWidth,
    double destinationHeight,
```

```
RasterImage image,  
string tag = null )
```

Draw a raster image.

Parameters

<i>sourceX</i>	The horizontal coordinate of the top-left corner of the rectangle delimiting the source area of the image.
<i>sourceY</i>	The vertical coordinate of the top-left corner of the rectangle delimiting the source area of the image.
<i>sourceWidth</i>	The width of the rectangle delimiting the source area of the image.
<i>sourceHeight</i>	The height of the rectangle delimiting the source area of the image.
<i>destinationX</i>	The horizontal coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>destinationY</i>	The vertical coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>destinationWidth</i>	The width of the rectangle delimiting the destination area of the image.
<i>destinationHeight</i>	The height of the rectangle delimiting the destination area of the image.
<i>image</i>	The image to draw.
<i>tag</i>	A tag to identify the drawn image.

Definition at line 2527 of file Graphics.cs.

6.13.2.7 DrawRasterImage() [4/5]

```
void VectSharp.Graphics.DrawRasterImage (
    Point position,
    RasterImage image,
    string tag = null )
```

Draw a raster image.

Parameters

<i>position</i>	The the top-left corner of the rectangle delimiting the destination area of the image.
<i>image</i>	The image to draw.
<i>tag</i>	A tag to identify the drawn image.

Definition at line 2550 of file Graphics.cs.

6.13.2.8 DrawRasterImage() [5/5]

```
void VectSharp.Graphics.DrawRasterImage (
    Point position,
    Size size,
    RasterImage image,
    string tag = null )
```

Draw a raster image.

Parameters

<i>position</i>	The the top-left corner of the rectangle delimiting the destination area of the image.
<i>size</i>	The size of the rectangle delimiting the destination area of the image.
<i>image</i>	The image to draw.
<i>tag</i>	A tag to identify the drawn image.

Definition at line 2576 of file Graphics.cs.

6.13.2.9 FillPath()

```
void VectSharp.Graphics.FillPath (
    GraphicsPath path,
    Colour fillColour,
    string tag = null )
```

Fill a [GraphicsPath](#).

Parameters

<i>path</i>	The GraphicsPath to fill.
<i>fillColour</i>	The Colour with which to fill the GraphicsPath .
<i>tag</i>	A tag to identify the filled path.

Definition at line 2336 of file Graphics.cs.

6.13.2.10 FillRectangle() [1/2]

```
void VectSharp.Graphics.FillRectangle (
    double leftX,
    double topY,
    double width,
    double height,
    Colour fillColour,
    string tag = null )
```

Fill a rectangle.

Parameters

<i>leftX</i>	The horizontal coordinate of the top-left corner of the rectangle.
<i>topY</i>	The vertical coordinate of the top-left corner of the rectangle.
<i>width</i>	The width of the rectangle.
<i>height</i>	The height of the rectangle.
<i>fillColour</i>	The colour with which to fill the rectangle.
<i>tag</i>	A tag to identify the filled rectangle.

Definition at line 2475 of file Graphics.cs.

6.13.2.11 FillRectangle() [2/2]

```
void VectSharp.Graphics.FillRectangle (
    Point topLeft,
    Size size,
    Colour fillColour,
    string tag = null )
```

Fill a rectangle.

Parameters

<i>topLeft</i>	The top-left corner of the rectangle.
<i>size</i>	The size of the rectangle.
<i>fillColour</i>	The colour with which to fill the rectangle.
<i>tag</i>	A tag to identify the filled rectangle.

Definition at line 2461 of file Graphics.cs.

6.13.2.12 FillText() [1/2]

```
void VectSharp.Graphics.FillText (
    double originX,
    double originY,
    string text,
    Font font,
    Colour fillColour,
    TextBaselines textBaseline = TextBaselines.Top,
    string tag = null )
```

Fill a text string.

Parameters

<i>originX</i>	The horizontal coordinate of the text origin.
<i>originY</i>	The vertical coordinate of the text origin. See <i>textBaseline</i> .
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>fillColour</i>	The colour to use to fill the text.
<i>textBaseline</i>	The text baseline (determines what <i>originY</i> represents).
<i>tag</i>	A tag to identify the filled text.

Definition at line 2605 of file Graphics.cs.

6.13.2.13 FillText() [2/2]

```
void VectSharp.Graphics.FillText (
    Point origin,
    string text,
    Font font,
    Colour fillColour,
    TextBaselines textBaseline = TextBaselines.Top,
    string tag = null )
```

Fill a text string.

Parameters

<i>origin</i>	The text origin. See <i>textBaseline</i> .
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>fillColour</i>	The colour to use to fill the text.
<i>textBaseline</i>	The text baseline (determines what the vertical component of <i>origin</i> represents).
<i>tag</i>	A tag to identify the filled text.

Definition at line 2590 of file Graphics.cs.

6.13.2.14 FillTextOnPath()

```
void VectSharp.Graphics.FillTextOnPath (
    GraphicsPath path,
    string text,
    Font font,
    Colour fillColour,
    double reference = 0,
    TextAnchors anchor = TextAnchors.Left,
    TextBaselines textBaseline = TextBaselines.Top,
    string tag = null )
```

Fill a text string along a [GraphicsPath](#).

Parameters

<i>path</i>	The GraphicsPath along which the text will flow.
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>fillColour</i>	The colour to use to fill the text.
<i>reference</i>	The (relative) starting point on the path starting from which the text should be drawn (0 is the start of the path, 1 is the end of the path).
<i>anchor</i>	The anchor in the text string that will correspond to the point specified by the <i>reference</i> .
<i>textBaseline</i>	The text baseline (determines which the position of the text in relation to the <i>path</i> .
<i>tag</i>	A tag to identify the filled text.

Definition at line 2658 of file Graphics.cs.

6.13.2.15 Linearise()

```
Graphics VectSharp.Graphics.Linearise (
    double resolution )
```

Creates a new [Graphics](#) object by linearising all of the elements of the current instance, i.e. replacing curve segments with series of line segments that approximate them. [Raster](#) images are left unchanged.

Parameters

<i>resolution</i>	The resolution that will be used to linearise curve segments.
-------------------	---

Returns

A new [Graphics](#) object containing the linearised elements.

Definition at line 3347 of file Graphics.cs.

6.13.2.16 MeasureText()

```
Size VectSharp.Graphics.MeasureText (
    string text,
    Font font )
```

Measure a text string. See also

See also

[Font.MeasureText\(string\)](#), [Font.MeasureTextAdvanced\(string\)](#)

and .

Parameters

<i>text</i>	The string to measure.
<i>font</i>	The font to use to measure the string.

Returns

Definition at line 2862 of file Graphics.cs.

6.13.2.17 Restore()

```
void VectSharp.Graphics.Restore ( )
```

Restore the previous transform state (rotation, translation scale).

Definition at line 2878 of file Graphics.cs.

6.13.2.18 Rotate()

```
void VectSharp.Graphics.Rotate (
    double angle )
```

Rotate the coordinate system around the origin.

Parameters

<i>angle</i>	The angle (in radians) by which to rotate the coordinate system.
--------------	--

Definition at line 2392 of file Graphics.cs.

6.13.2.19 RotateAt()

```
void VectSharp.Graphics.RotateAt (
    double angle,
    Point pivot )
```

Rotate the coordinate system around a pivot point.

Parameters

<i>angle</i>	The angle (in radians) by which to rotate the coordinate system.
<i>pivot</i>	The pivot around which the coordinate system is to be rotated.

Definition at line 2402 of file Graphics.cs.

6.13.2.20 Save()

```
void VectSharp.Graphics.Save ( )
```

Save the current transform state (rotation, translation, scale).

Definition at line 2870 of file Graphics.cs.

6.13.2.21 Scale()

```
void VectSharp.Graphics.Scale (
    double scaleX,
    double scaleY )
```

Scale the coordinate system with respect to the origin.

Parameters

<i>scaleX</i>	The horizontal scale.
<i>scaleY</i>	The vertical scale.

Definition at line 2449 of file Graphics.cs.

6.13.2.22 SetClippingPath() [1/3]

```
void VectSharp.Graphics.SetClippingPath (
    double leftX,
    double topY,
    double width,
    double height )
```

Intersect the current clipping path with the specified rectangle.

Parameters

<i>leftX</i>	The horizontal coordinate of the top-left corner of the rectangle.
<i>topY</i>	The vertical coordinate of the top-left corner of the rectangle.
<i>width</i>	The width of the rectangle.
<i>height</i>	The height of the rectangle.

Definition at line 2373 of file Graphics.cs.

6.13.2.23 SetClippingPath() [2/3]

```
void VectSharp.Graphics.SetClippingPath (
    GraphicsPath path )
```

Intersect the current clipping path with the specified [GraphicsPath](#).

Parameters

<i>path</i>	The GraphicsPath to intersect with the current clipping path.
-------------	---

Definition at line 2361 of file Graphics.cs.

6.13.2.24 SetClippingPath() [3/3]

```
void VectSharp.Graphics.SetClippingPath (
    Point topLeft,
    Size size )
```

Intersect the current clipping path with the specified rectangle.

Parameters

<i>topLeft</i>	The top-left corner of the rectangle.
<i>size</i>	The size of the rectangle.

Definition at line 2383 of file Graphics.cs.

6.13.2.25 StrokePath()

```
void VectSharp.Graphics.StrokePath (
    GraphicsPath path,
    Colour strokeColour,
    double lineWidth = 1,
    LineCaps lineCap = LineCaps.Butt,
    LineJoins lineJoin = LineJoins.Miter,
    LineDash? lineDash = null,
    string tag = null )
```

Stroke a [GraphicsPath](#).

Parameters

<i>path</i>	The GraphicsPath to stroke.
<i>strokeColour</i>	The Colour with which to stroke the GraphicsPath .
<i>lineWidth</i>	The width of the line with which the path is stroked.
<i>lineCap</i>	The line cap to use to stroke the path.
<i>lineJoin</i>	The line join to use to stroke the path.
<i>lineDash</i>	The line dash to use to stroke the path.
<i>tag</i>	A tag to identify the stroked path.

Definition at line 2352 of file Graphics.cs.

6.13.2.26 StrokeRectangle() [1/2]

```
void VectSharp.Graphics.StrokeRectangle (
    double leftX,
    double topY,
    double width,
    double height,
    Colour strokeColour,
    double lineWidth = 1,
    LineCaps lineCap = LineCaps.Butt,
    LineJoins lineJoin = LineJoins.Miter,
    LineDash? lineDash = null,
    string tag = null )
```

Stroke a rectangle.

Parameters

<i>leftX</i>	The horizontal coordinate of the top-left corner of the rectangle.
<i>topY</i>	The vertical coordinate of the top-left corner of the rectangle.
<i>width</i>	The width of the rectangle.
<i>height</i>	The height of the rectangle.
<i>strokeColour</i>	The colour with which to stroke the rectangle.
<i>lineWidth</i>	The width of the line with which the rectangle is stroked.
<i>lineCap</i>	The line cap to use to stroke the rectangle.
<i>lineJoin</i>	The line join to use to stroke the rectangle.
<i>lineDash</i>	The line dash to use to stroke the rectangle.
<i>tag</i>	A tag to identify the filled rectangle.

Definition at line 2509 of file Graphics.cs.

6.13.2.27 StrokeRectangle() [2/2]

```
void VectSharp.Graphics.StrokeRectangle (
    Point topLeft,
    Size size,
    Colour strokeColour,
    double lineWidth = 1,
    LineCaps lineCap = LineCaps.Butt,
    LineJoins lineJoin = LineJoins.Miter,
    LineDash? lineDash = null,
    string tag = null )
```

Stroke a rectangle.

Parameters

<i>topLeft</i>	The top-left corner of the rectangle.
<i>size</i>	The size of the rectangle.
<i>strokeColour</i>	The colour with which to stroke the rectangle.

Parameters

<i>lineWidth</i>	The width of the line with which the rectangle is stroked.
<i>lineCap</i>	The line cap to use to stroke the rectangle.
<i>lineJoin</i>	The line join to use to stroke the rectangle.
<i>lineDash</i>	The line dash to use to stroke the rectangle.
<i>tag</i>	A tag to identify the filled rectangle.

Definition at line 2491 of file Graphics.cs.

6.13.2.28 StrokeText() [1/2]

```
void VectSharp.Graphics.StrokeText (
    double originX,
    double originY,
    string text,
    Font font,
    Colour strokeColour,
    TextBaselines textBaseline = TextBaselines.Top,
    double lineWidth = 1,
    LineCaps lineCap = LineCaps.Butt,
    LineJoins lineJoin = LineJoins.Miter,
    LineDash? lineDash = null,
    string tag = null )
```

Stroke a text string.

Parameters

<i>originX</i>	The horizontal coordinate of the text origin.
<i>originY</i>	The vertical coordinate of the text origin. See <i>textBaseline</i> .
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>strokeColour</i>	The colour with which to stroke the text.
<i>lineWidth</i>	The width of the line with which the text is stroked.
<i>lineCap</i>	The line cap to use to stroke the text.
<i>lineJoin</i>	The line join to use to stroke the text.
<i>lineDash</i>	The line dash to use to stroke the text.
<i>textBaseline</i>	The text baseline (determines what <i>originY</i> represents).
<i>tag</i>	A tag to identify the stroked text.

Definition at line 2642 of file Graphics.cs.

6.13.2.29 StrokeText() [2/2]

```
void VectSharp.Graphics.StrokeText (
    Point origin,
```

```

string text,
Font font,
Colour strokeColour,
TextBaselines textBaseline = TextBaselines.Top,
double lineWidth = 1,
LineCaps lineCap = LineCaps.Butt,
LineJoins lineJoin = LineJoins.Miter,
LineDash? lineDash = null,
string tag = null )

```

Stroke a text string.

Parameters

<i>origin</i>	The text origin. See <i>textBaseline</i> .
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>strokeColour</i>	The colour with which to stroke the text.
<i>lineWidth</i>	The width of the line with which the text is stroked.
<i>lineCap</i>	The line cap to use to stroke the text.
<i>lineJoin</i>	The line join to use to stroke the text.
<i>lineDash</i>	The line dash to use to stroke the text.
<i>textBaseline</i>	The text baseline (determines what the vertical component of <i>origin</i> represents).
<i>tag</i>	A tag to identify the stroked text.

Definition at line 2623 of file Graphics.cs.

6.13.2.30 StrokeTextOnPath()

```

void VectSharp.Graphics.StrokeTextOnPath (
    GraphicsPath path,
    string text,
    Font font,
    Colour strokeColour,
    double reference = 0,
    TextAnchors anchor = TextAnchors.Left,
    TextBaselines textBaseline = TextBaselines.Top,
    double lineWidth = 1,
    LineCaps lineCap = LineCaps.Butt,
    LineJoins lineJoin = LineJoins.Miter,
    LineDash? lineDash = null,
    string tag = null )

```

Stroke a text string along a [GraphicsPath](#).

Parameters

<i>path</i>	The GraphicsPath along which the text will flow.
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>strokeColour</i>	The colour with which to stroke the text.

Parameters

<i>lineWidth</i>	The width of the line with which the text is stroked.
<i>lineCap</i>	The line cap to use to stroke the text.
<i>lineJoin</i>	The line join to use to stroke the text.
<i>lineDash</i>	The line dash to use to stroke the text.
<i>reference</i>	The (relative) starting point on the path starting from which the text should be drawn (0 is the start of the path, 1 is the end of the path).
<i>anchor</i>	The anchor in the text string that will correspond to the point specified by the <i>reference</i> .
<i>textBaseline</i>	The text baseline (determines which the position of the text in relation to the <i>path</i> .
<i>tag</i>	A tag to identify the stroked text.

Definition at line 2764 of file Graphics.cs.

6.13.2.31 Transform() [1/2]

```
void VectSharp.Graphics.Transform (
    double a,
    double b,
    double c,
    double d,
    double e,
    double f )
```

Transform the coordinate system with the specified transformation matrix [[a, c, e], [b, d, f], [0, 0, 1]].

Parameters

<i>a</i>	The first element of the first column.
<i>b</i>	The second element of the first column.
<i>c</i>	The first element of the second column.
<i>d</i>	The second element of the second column.
<i>e</i>	The first element of the third column.
<i>f</i>	The second element of the third column.

Definition at line 2419 of file Graphics.cs.

6.13.2.32 Transform() [2/2]

```
Graphics VectSharp.Graphics.Transform (
    Func< Point, Point > transformationFunction,
    double linearisationResolution )
```

Creates a new [Graphics](#) object in which all the graphics actions have been transformed using an arbitrary transformation function. [Raster](#) images are replaced by grey rectangles.

Parameters

<i>transformationFunction</i>	An arbitrary transformation function.
<i>linearisationResolution</i>	The resolution that will be used to linearise curve segments.

Returns

A new [Graphics](#) object in which all graphics actions have been linearised and transformed using the *transformationFunction*.

Definition at line 3222 of file Graphics.cs.

6.13.2.33 Translate() [1/2]

```
void VectSharp.Graphics.Translate (
    double x,
    double y )
```

Translate the coordinate system origin.

Parameters

<i>x</i>	The horizontal translation.
<i>y</i>	The vertical translation.

Definition at line 2430 of file Graphics.cs.

6.13.2.34 Translate() [2/2]

```
void VectSharp.Graphics.Translate (
    Point delta )
```

Translate the coordinate system origin.

Parameters

<i>delta</i>	The new origin point.
--------------	-----------------------

Definition at line 2439 of file Graphics.cs.

6.13.3 Property Documentation

6.13.3.1 UnbalancedStackAction

`UnbalancedStackActions` VectSharp.Graphics.UnbalancedStackAction = `UnbalancedStackActions.Throw`
 [static], [get], [set]

Determines how an unbalanced graphics state stack (which occurs if the number of calls to [Save](#) and [Restore](#) is not equal) will be treated. The default is [UnbalancedStackActions.Throw](#).

Definition at line 2326 of file Graphics.cs.

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

- VectSharp/Graphics.cs

6.14 VectSharp.GraphicsPath Class Reference

Represents a graphics path that can be filled or stroked.

Public Member Functions

- [GraphicsPath MoveTo](#) ([Point](#) p)
Move the current point without tracing a segment from the previous point.
- [GraphicsPath MoveTo](#) (double x, double y)
Move the current point without tracing a segment from the previous point.
- [GraphicsPath LineTo](#) ([Point](#) p)
Move the current point and trace a segment from the previous point.
- [GraphicsPath LineTo](#) (double x, double y)
Move the current point and trace a segment from the previous point.
- [GraphicsPath Arc](#) ([Point](#) center, double radius, double startAngle, double endAngle)
Trace an arc segment from a circle with the specified center and radius , starting at startAngle and ending at endAngle . The current point is updated to the end point of the arc.
- [GraphicsPath Arc](#) (double centerX, double centerY, double radius, double startAngle, double endAngle)
Trace an arc segment from a circle with the specified center and radius , starting at startAngle and ending at endAngle . The current point is updated to the end point of the arc.
- [GraphicsPath EllipticalArc](#) (double radiusX, double radiusY, double axisAngle, bool largeArc, bool sweep↔ Clockwise, [Point](#) endPoint)
Trace an arc from an ellipse with the specified radii, rotated by axisAngle with respect to the x-axis, starting at the current point and ending at the endPoint .
- [GraphicsPath CubicBezierTo](#) ([Point](#) control1, [Point](#) control2, [Point](#) endPoint)
Trace a cubic Bezier curve from the current point to a destination point, with two control points. The current point is updated to the end point of the Bezier curve.
- [GraphicsPath CubicBezierTo](#) (double control1X, double control1Y, double control2X, double control2Y, double endPointX, double endPointY)
Trace a cubic Bezier curve from the current point to a destination point, with two control points. The current point is updated to the end point of the Bezier curve.
- [GraphicsPath Close](#) ()
Trace a segment from the current point to the start point of the figure and flag the figure as closed.
- [GraphicsPath AddText](#) (double originX, double originY, string text, [Font](#) font, [TextBaselines](#) text↔ Baseline=[TextBaselines.Top](#))
Add the contour of a text string to the current path.

- [GraphicsPath AddText](#) ([Point](#) origin, string text, [Font](#) font, [TextBaselines](#) textBaseline=[TextBaselines.Top](#))
Add the contour of a text string to the current path.
- [GraphicsPath AddTextOnPath](#) ([GraphicsPath](#) path, string text, [Font](#) font, double reference=0, [TextAnchors](#) anchor=[TextAnchors.Left](#), [TextBaselines](#) textBaseline=[TextBaselines.Top](#))
Add the contour of a text string flowing along a [GraphicsPath](#) to the current path.
- [GraphicsPath AddSmoothSpline](#) (params [Point](#)[] points)
Adds a smooth spline composed of cubic bezier segments that pass through the specified points.
- double [MeasureLength](#) ()
Measures the length of the [GraphicsPath](#).
- [Point GetPointAtRelative](#) (double position)
Gets the point at the relative position specified on the [GraphicsPath](#).
- [Point GetPointAtAbsolute](#) (double length)
Gets the point at the absolute position specified on the [GraphicsPath](#).
- [Point GetTangentAtRelative](#) (double position)
Gets the tangent to the point at the relative position specified on the [GraphicsPath](#).
- [Point GetTangentAtAbsolute](#) (double length)
Gets the tangent to the point at the absolute position specified on the [GraphicsPath](#).
- [Point GetNormalAtAbsolute](#) (double length)
Gets the normal to the point at the absolute position specified on the [GraphicsPath](#).
- [Point GetNormalAtRelative](#) (double position)
Gets the normal to the point at the relative position specified on the [GraphicsPath](#).
- [GraphicsPath Linearise](#) (double resolution)
Linearises a [GraphicsPath](#), replacing curve segments with series of line segments that approximate them.
- [IEnumerable< List< Point > > GetPoints](#) ()
Gets a collection of the end points of all the segments in the [GraphicsPath](#), divided by figure.
- [IEnumerable< List< Point > > GetLinearisationPointsNormals](#) (double resolution)
Gets a collection of the tangents at the end point of the segments in which the [GraphicsPath](#) would be linearised, divided by figure.
- [IEnumerable< GraphicsPath > Triangulate](#) (double resolution, bool clockwise)
Divides a [GraphicsPath](#) into triangles.
- [GraphicsPath Transform](#) (Func< [Point](#), [Point](#) > transformationFunction)
Transforms all of the [Points](#) in the [GraphicsPath](#) with an arbitrary transformation function.

Properties

- [List< Segment > Segments](#) = new List<[Segment](#)>() [get, set]
The segments that make up the path.

6.14.1 Detailed Description

Represents a graphics path that can be filled or stroked.

Definition at line 3583 of file Graphics.cs.

6.14.2 Member Function Documentation

6.14.2.1 AddSmoothSpline()

```
GraphicsPath VectSharp.GraphicsPath.AddSmoothSpline (
    params Point[] points )
```

Adds a smooth spline composed of cubic bezier segments that pass through the specified points.

Parameters

<i>points</i>	The points through which the spline should pass.
---------------	--

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 4034 of file Graphics.cs.

6.14.2.2 AddText() [1/2]

```
GraphicsPath VectSharp.GraphicsPath.AddText (
    double originX,
    double originY,
    string text,
    Font font,
    TextBaselines textBaseline = TextBaselines.Top )
```

Add the contour of a text string to the current path.

Parameters

<i>originX</i>	The horizontal coordinate of the text origin.
<i>originY</i>	The vertical coordinate of the text origin. See <i>textBaseline</i> .
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>textBaseline</i>	The text baseline (determines what <i>originY</i> represents).

///

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3832 of file Graphics.cs.

6.14.2.3 AddText() [2/2]

```
GraphicsPath VectSharp.GraphicsPath.AddText (
    Point origin,
    string text,
    Font font,
    TextBaselines textBaseline = TextBaselines.Top )
```

Add the contour of a text string to the current path.

Parameters

<i>origin</i>	The text origin. See <i>textBaseline</i> .
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>textBaseline</i>	The text baseline (determines what the vertical component of <i>origin</i> represents).

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3845 of file Graphics.cs.

6.14.2.4 AddTextOnPath()

```
GraphicsPath VectSharp.GraphicsPath.AddTextOnPath (
    GraphicsPath path,
    string text,
    Font font,
    double reference = 0,
    TextAnchors anchor = TextAnchors.Left,
    TextBaselines textBaseline = TextBaselines.Top )
```

Add the contour of a text string flowing along a [GraphicsPath](#) to the current path.

Parameters

<i>path</i>	The GraphicsPath along which the text will flow.
<i>text</i>	The string to draw.
<i>font</i>	The font with which to draw the text.
<i>reference</i>	The (relative) starting point on the path starting from which the text should be drawn (0 is the start of the path, 1 is the end of the path).
<i>anchor</i>	The anchor in the text string that will correspond to the point specified by the <i>reference</i> .
<i>textBaseline</i>	The text baseline (determines which the position of the text in relation to the <i>path</i> .

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3922 of file Graphics.cs.

6.14.2.5 Arc() [1/2]

```
GraphicsPath VectSharp.GraphicsPath.Arc (
    double centerX,
```

```
double centerY,
double radius,
double startAngle,
double endAngle )
```

Trace an arc segment from a circle with the specified center and *radius* , starting at *startAngle* and ending at *endAngle* . The current point is updated to the end point of the arc.

Parameters

<i>centerX</i>	The horizontal coordinate of the center of the arc.
<i>centerY</i>	The vertical coordinate of the center of the arc.
<i>radius</i>	The radius of the arc.
<i>startAngle</i>	The start angle (in radians) of the arc.
<i>endAngle</i>	The end angle (in radians) of the arc.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3673 of file Graphics.cs.

6.14.2.6 Arc() [2/2]

```
GraphicsPath VectSharp.GraphicsPath.Arc (
    Point center,
    double radius,
    double startAngle,
    double endAngle )
```

Trace an arc segment from a circle with the specified *center* and *radius* , starting at *startAngle* and ending at *endAngle* . The current point is updated to the end point of the arc.

Parameters

<i>center</i>	The center of the arc.
<i>radius</i>	The radius of the arc.
<i>startAngle</i>	The start angle (in radians) of the arc.
<i>endAngle</i>	The end angle (in radians) of the arc.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3653 of file Graphics.cs.

6.14.2.7 Close()

```
GraphicsPath VectSharp.GraphicsPath.Close ( )
```

Trace a segment from the current point to the start point of the figure and flag the figure as closed.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3817 of file Graphics.cs.

6.14.2.8 CubicBezierTo() [1/2]

```
GraphicsPath VectSharp.GraphicsPath.CubicBezierTo (
    double control1X,
    double control1Y,
    double control2X,
    double control2Y,
    double endPointX,
    double endPointY )
```

Trace a cubic Bezier curve from the current point to a destination point, with two control points. The current point is updated to the end point of the Bezier curve.

Parameters

<i>control1X</i>	The horizontal coordinate of the first control point.
<i>control1Y</i>	The vertical coordinate of the first control point.
<i>control2X</i>	The horizontal coordinate of the second control point.
<i>control2Y</i>	The vertical coordinate of the second control point.
<i>endPointX</i>	The horizontal coordinate of the destination point.
<i>endPointY</i>	The vertical coordinate of the destination point.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3807 of file Graphics.cs.

6.14.2.9 CubicBezierTo() [2/2]

```
GraphicsPath VectSharp.GraphicsPath.CubicBezierTo (
    Point control1,
    Point control2,
    Point endPoint )
```

Trace a cubic Bezier curve from the current point to a destination point, with two control points. The current point is updated to the end point of the Bezier curve.

Parameters

<i>control1</i>	The first control point.
<i>control2</i>	The second control point.
<i>endPoint</i>	The destination point.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3786 of file Graphics.cs.

6.14.2.10 EllipticalArc()

```
GraphicsPath VectSharp.GraphicsPath.EllipticalArc (
    double radiusX,
    double radiusY,
    double axisAngle,
    bool largeArc,
    bool sweepClockwise,
    Point endPoint )
```

Trace an arc from an ellipse with the specified radii, rotated by *axisAngle* with respect to the x-axis, starting at the current point and ending at the *endPoint*.

Parameters

<i>radiusX</i>	The horizontal radius of the ellipse.
<i>radiusY</i>	The vertical radius of the ellipse.
<i>axisAngle</i>	The angle of the horizontal axis of the ellipse with respect to the horizontal axis.
<i>largeArc</i>	Determines whether the large or the small arc is drawn.
<i>sweepClockwise</i>	Determines whether the clockwise or anticlockwise arc is drawn.
<i>endPoint</i>	The end point of the arc.

Returns

Definition at line 3689 of file Graphics.cs.

6.14.2.11 GetLinearisationPointsNormals()

```
IEnumerable<List<Point> > VectSharp.GraphicsPath.GetLinearisationPointsNormals (
    double resolution )
```

Gets a collection of the tangents at the end point of the segments in which the [GraphicsPath](#) would be linearised, divided by figure.

Parameters

<i>resolution</i>	The absolute length between successive samples in curve segments.
-------------------	---

Returns

A collection of the tangents at the end point of the segments in which the [GraphicsPath](#) would be linearised, divided by figure.

Definition at line 4828 of file Graphics.cs.

6.14.2.12 GetNormalAtAbsolute()

```
Point VectSharp.GraphicsPath.GetNormalAtAbsolute (
    double length )
```

Gets the normal to the point at the absolute position specified on the [GraphicsPath](#).

Parameters

<i>length</i>	The distance to the point from the start of the GraphicsPath .
---------------	--

Returns

The normal to the point at the specified position.

Definition at line 4733 of file Graphics.cs.

6.14.2.13 GetNormalAtRelative()

```
Point VectSharp.GraphicsPath.GetNormalAtRelative (
    double position )
```

Gets the normal to the point at the relative position specified on the [GraphicsPath](#).

Parameters

<i>position</i>	The position on the GraphicsPath (0 is the start of the path, 1 is the end of the path).
-----------------	--

Returns

The normal to the point at the specified position.

Definition at line 4744 of file Graphics.cs.

6.14.2.14 GetPointAtAbsolute()

```
Point VectSharp.GraphicsPath.GetPointAtAbsolute (
    double length )
```

Gets the point at the absolute position specified on the [GraphicsPath](#).

Parameters

<i>length</i>	The distance to the point from the start of the GraphicsPath .
---------------	--

Returns

The point at the specified position.

Definition at line 4149 of file Graphics.cs.

6.14.2.15 GetPointAtRelative()

```
Point VectSharp.GraphicsPath.GetPointAtRelative (
    double position )
```

Gets the point at the relative position specified on the [GraphicsPath](#).

Parameters

<i>position</i>	The position on the GraphicsPath (0 is the start of the path, 1 is the end of the path).
-----------------	--

Returns

The point at the specified position.

Definition at line 4139 of file Graphics.cs.

6.14.2.16 GetPoints()

```
IEnumerable<List<Point> > VectSharp.GraphicsPath.GetPoints ( )
```

Gets a collection of the end points of all the segments in the [GraphicsPath](#), divided by figure.

Returns

A collection of the end points of all the segments in the [GraphicsPath](#), divided by figure.

Definition at line 4783 of file Graphics.cs.

6.14.2.17 GetTangentAtAbsolute()

```
Point VectSharp.GraphicsPath.GetTangentAtAbsolute (
    double length )
```

Gets the tangent to the point at the absolute position specified on the [GraphicsPath](#).

Parameters

<i>length</i>	The distance to the point from the start of the GraphicsPath .
---------------	--

Returns

The tangent to the point at the specified position.

Definition at line 4446 of file Graphics.cs.

6.14.2.18 GetTangentAtRelative()

```
Point VectSharp.GraphicsPath.GetTangentAtRelative (
    double position )
```

Gets the tangent to the point at the relative position specified on the [GraphicsPath](#).

Parameters

<i>position</i>	The position on the GraphicsPath (0 is the start of the path, 1 is the end of the path).
-----------------	--

Returns

The tangent to the point at the specified position.

Definition at line 4436 of file Graphics.cs.

6.14.2.19 Linearise()

```
GraphicsPath VectSharp.GraphicsPath.Linearise (
    double resolution )
```

Linearises a [GraphicsPath](#), replacing curve segments with series of line segments that approximate them.

Parameters

<i>resolution</i>	The absolute length between successive samples in curve segments.
-------------------	---

Returns

A [GraphicsPath](#) composed only of linear segments that approximates the current [GraphicsPath](#).

Definition at line 4755 of file Graphics.cs.

6.14.2.20 LineTo() [1/2]

```
GraphicsPath VectSharp.GraphicsPath.LineTo (
    double x,
    double y )
```

Move the current point and trace a segment from the previous point.

Parameters

<i>x</i>	The horizontal coordinate of the new point.
<i>y</i>	The vertical coordinate of the new point.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3638 of file Graphics.cs.

6.14.2.21 LineTo() [2/2]

```
GraphicsPath VectSharp.GraphicsPath.LineTo (
    Point p )
```

Move the current point and trace a segment from the previous point.

Parameters

<i>p</i>	The new point.
----------	----------------

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3619 of file Graphics.cs.

6.14.2.22 MeasureLength()

```
double VectSharp.GraphicsPath.MeasureLength ( )
```

Measures the length of the [GraphicsPath](#).

Returns

The length of the [GraphicsPath](#)

Definition at line 4067 of file Graphics.cs.

6.14.2.23 MoveTo() [1/2]

```
GraphicsPath VectSharp.GraphicsPath.MoveTo (
    double x,
    double y )
```

Move the current point without tracing a segment from the previous point.

Parameters

<i>x</i>	The horizontal coordinate of the new point.
<i>y</i>	The vertical coordinate of the new point.

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3608 of file Graphics.cs.

6.14.2.24 MoveTo() [2/2]

```
GraphicsPath VectSharp.GraphicsPath.MoveTo (
    Point p )
```

Move the current point without tracing a segment from the previous point.

Parameters

<i>p</i>	The new point.
----------	----------------

Returns

The [GraphicsPath](#), to allow for chained calls.

Definition at line 3596 of file Graphics.cs.

6.14.2.25 Transform()

```
GraphicsPath VectSharp.GraphicsPath.Transform (
    Func< Point, Point > transformationFunction )
```

Transforms all of the [Points](#) in the [GraphicsPath](#) with an arbitrary transformation function.

Parameters

<i>transformationFunction</i>	An arbitrary transformation function.
-------------------------------	---------------------------------------

Returns

A new [GraphicsPath](#) in which all points have been replaced using the *transformationFunction* .

Definition at line 5900 of file Graphics.cs.

6.14.2.26 Triangulate()

```
IEnumerable<GraphicsPath> VectSharp.GraphicsPath.Triangulate (
    double resolution,
    bool clockwise )
```

Divides a [GraphicsPath](#) into triangles.

Parameters

<i>resolution</i>	The resolution that will be used to linearise curve segments in the GraphicsPath .
<i>clockwise</i>	If this is <code>true</code> , the triangles will have their vertices in a clockwise order, otherwise they will be in anticlockwise order.

Returns

A collection of distinct [GraphicsPaths](#), each representing one triangle.

Definition at line 4911 of file Graphics.cs.

6.14.3 Property Documentation

6.14.3.1 Segments

```
List<Segment> VectSharp.GraphicsPath.Segments = new List<Segment>() [get], [set]
```

The segments that make up the path.

Definition at line 3588 of file Graphics.cs.

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

- VectSharp/Graphics.cs

6.15 VectSharp.IGraphicsContext Interface Reference

This interface should be implemented by classes intended to provide graphics output capability to a [Graphics](#) object.

Public Member Functions

- void [Save](#) ()
Save the current transform state (rotation, translation, scale). This should be implemented as a LIFO stack.
- void [Restore](#) ()
Restore the previous transform state (rotation, translation, scale). This should be implemented as a LIFO stack.
- void [Translate](#) (double x, double y)
Translate the coordinate system origin.
- void [Rotate](#) (double angle)
Rotate the coordinate system around the origin.
- void [Scale](#) (double scaleX, double scaleY)
Scale the coordinate system with respect to the origin.
- void [Transform](#) (double a, double b, double c, double d, double e, double f)
Transform the coordinate system with the specified transformation matrix $\begin{bmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{bmatrix}$.
- void [FillText](#) (string text, double x, double y)
Fill a text string using the current [Font](#) and [TextBaseline](#).
- void [StrokeText](#) (string text, double x, double y)
Stroke the outline of a text string using the current [Font](#) and [TextBaseline](#).
- void [MoveTo](#) (double x, double y)
Change the current point without drawing a line from the previous point. If necessary, start a new figure.
- void [LineTo](#) (double x, double y)
Draw a line from the previous point to the specified point.
- void [Close](#) ()
Close the current figure.
- void [Stroke](#) ()
Stroke the current path using the current [StrokeStyle](#), [LineWidth](#), [LineCap](#), [LineJoin](#) and [LineDash](#).
- void [SetClippingPath](#) ()
Set the current clipping path as the intersection of the previous clipping path and the current path.
- void [SetFillStyle](#) ((int r, int g, int b, double a) style)
Set the current [FillStyle](#).
- void [SetFillStyle](#) (Colour style)
Set the current [FillStyle](#).
- void [SetStrokeStyle](#) ((int r, int g, int b, double a) style)

- Set the current [StrokeStyle](#).
 - void [SetStrokeStyle](#) ([Colour](#) style)
- Set the current [StrokeStyle](#).
 - void [CubicBezierTo](#) (double p1X, double p1Y, double p2X, double p2Y, double p3X, double p3Y)
 - Add to the current figure a cubic Bezier from the current point to a destination point, with two control points.
 - void [Rectangle](#) (double x0, double y0, double width, double height)
 - Add a rectangle figure to the current path.
 - void [Fill](#) ()
 - Fill the current path using the current [FillStyle](#).
 - void [SetLineDash](#) ([LineDash](#) dash)
 - Set the current line dash pattern.
 - void [DrawRasterImage](#) (int sourceX, int sourceY, int sourceWidth, int sourceHeight, double destinationX, double destinationY, double destinationWidth, double destinationHeight, [RasterImage](#) image)
 - Draw a raster image.

Properties

- double [Width](#) [get]
 - Width of the graphic surface.
- double [Height](#) [get]
 - Height of the graphic surface.
- [Font](#) [Font](#) [get, set]
 - The current font.
- [TextBaselines](#) [TextBaseline](#) [get, set]
 - The current text baseline.
- [Colour](#) [FillStyle](#) [get]
 - Current colour used to fill paths.
- [Colour](#) [StrokeStyle](#) [get]
 - Current colour used to stroke paths.
- double [LineWidth](#) [get, set]
 - Current line width used to stroke paths.
- [LineCaps](#) [LineCap](#) [set]
 - Current line cap used to stroke paths.
- [LineJoins](#) [LineJoin](#) [set]
 - Current line join used to stroke paths.
- string [Tag](#) [get, set]
 - The current tag. How this can be used depends on each implementation.

6.15.1 Detailed Description

This interface should be implemented by classes intended to provide graphics output capability to a [Graphics](#) object.

Definition at line 2081 of file Graphics.cs.

6.15.2 Member Function Documentation

6.15.2.1 Close()

```
void VectSharp.IGraphicsContext.Close ( )
```

Close the current figure.

6.15.2.2 CubicBezierTo()

```
void VectSharp.IGraphicsContext.CubicBezierTo (
    double p1X,
    double p1Y,
    double p2X,
    double p2Y,
    double p3X,
    double p3Y )
```

Add to the current figure a cubic Bezier from the current point to a destination point, with two control points.

Parameters

<i>p1X</i>	The horizontal coordinate of the first control point.
<i>p1Y</i>	The vertical coordinate of the first control point.
<i>p2X</i>	The horizontal coordinate of the second control point.
<i>p2Y</i>	The vertical coordinate of the second control point.
<i>p3X</i>	The horizontal coordinate of the destination point.
<i>p3Y</i>	The vertical coordinate of the destination point.

6.15.2.3 DrawRasterImage()

```
void VectSharp.IGraphicsContext.DrawRasterImage (
    int sourceX,
    int sourceY,
    int sourceWidth,
    int sourceHeight,
    double destinationX,
    double destinationY,
    double destinationWidth,
    double destinationHeight,
    RasterImage image )
```

Draw a raster image.

Parameters

<i>sourceX</i>	The horizontal coordinate of the top-left corner of the rectangle delimiting the source area of the image.
<i>sourceY</i>	The vertical coordinate of the top-left corner of the rectangle delimiting the source area of the image.

Parameters

<i>sourceWidth</i>	The width of the rectangle delimiting the source area of the image.
<i>sourceHeight</i>	The height of the rectangle delimiting the source area of the image.
<i>destinationX</i>	The horizontal coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>destinationY</i>	The vertical coordinate of the top-left corner of the rectangle delimiting the destination area of the image.
<i>destinationWidth</i>	The width of the rectangle delimiting the destination area of the image.
<i>destinationHeight</i>	The height of the rectangle delimiting the destination area of the image.
<i>image</i>	The image to draw.

6.15.2.4 Fill()

```
void VectSharp.IGraphicsContext.Fill ( )
```

Fill the current path using the current [FillStyle](#).

6.15.2.5 FillText()

```
void VectSharp.IGraphicsContext.FillText (
    string text,
    double x,
    double y )
```

Fill a text string using the current [Font](#) and [TextBaseline](#).

Parameters

<i>text</i>	The string to draw.
<i>x</i>	The horizontal coordinate of the text origin.
<i>y</i>	The vertical coordinate of the text origin.

6.15.2.6 LineTo()

```
void VectSharp.IGraphicsContext.LineTo (
    double x,
    double y )
```

Draw a line from the previous point to the specified point.

Parameters

<i>x</i>	The horizontal coordinate of the point.
<i>y</i>	The vertical coordinate of the point.

6.15.2.7 MoveTo()

```
void VectSharp.IGraphicsContext.MoveTo (
    double x,
    double y )
```

Change the current point without drawing a line from the previous point. If necessary, start a new figure.

Parameters

<i>x</i>	The horizontal coordinate of the point.
<i>y</i>	The vertical coordinate of the point.

6.15.2.8 Rectangle()

```
void VectSharp.IGraphicsContext.Rectangle (
    double x0,
    double y0,
    double width,
    double height )
```

Add a rectangle figure to the current path.

Parameters

<i>x0</i>	The horizontal coordinate of the top-left corner of the rectangle.
<i>y0</i>	The vertical coordinate of the top-left corner of the rectangle.
<i>width</i>	The width of corner of the rectangle.
<i>height</i>	The height of corner of the rectangle.

6.15.2.9 Restore()

```
void VectSharp.IGraphicsContext.Restore ( )
```

Restore the previous transform state (rotation, translation, scale). This should be implemented as a LIFO stack.

6.15.2.10 Rotate()

```
void VectSharp.IGraphicsContext.Rotate (
    double angle )
```

Rotate the coordinate system around the origin.

Parameters

<i>angle</i>	The angle (in radians) by which to rotate the coordinate system.
--------------	--

6.15.2.11 Save()

```
void VectSharp.IGraphicsContext.Save ( )
```

Save the current transform state (rotation, translation, scale). This should be implemented as a LIFO stack.

6.15.2.12 Scale()

```
void VectSharp.IGraphicsContext.Scale (
    double scaleX,
    double scaleY )
```

Scale the coordinate system with respect to the origin.

Parameters

<i>scaleX</i>	The horizontal scale.
<i>scaleY</i>	The vertical scale.

6.15.2.13 SetClippingPath()

```
void VectSharp.IGraphicsContext.SetClippingPath ( )
```

Set the current clipping path as the intersection of the previous clipping path and the current path.

6.15.2.14 SetFillStyle() [1/2]

```
void VectSharp.IGraphicsContext.SetFillStyle (
    (int r, int g, int b, double a) style )
```

Set the current [FillStyle](#).

Parameters

<i>style</i>	A <code>ValueTuple<Int32, Int32, Int32, Double></code> containing component information for the colour. For r, g, and b, range: [0, 255]; for a, range: [0, 1].
--------------	---

6.15.2.15 SetFillStyle() [2/2]

```
void VectSharp.IGraphicsContext.SetFillStyle (
    Colour style )
```

Set the current [FillStyle](#).

Parameters

<i>style</i>	The new fill style.
--------------	---------------------

6.15.2.16 SetLineDash()

```
void VectSharp.IGraphicsContext.SetLineDash (
    LineDash dash )
```

Set the current line dash pattern.

Parameters

<i>dash</i>	The line dash pattern.
-------------	------------------------

6.15.2.17 SetStrokeStyle() [1/2]

```
void VectSharp.IGraphicsContext.SetStrokeStyle (
    (int r, int g, int b, double a) style )
```

Set the current [StrokeStyle](#).

Parameters

<i>style</i>	A <code>ValueTuple<Int32, Int32, Int32, Double></code> containing component information for the colour. For r, g, and b, range: [0, 255]; for a, range: [0, 1].
--------------	---

6.15.2.18 SetStrokeStyle() [2/2]

```
void VectSharp.IGraphicsContext.SetStrokeStyle (
    Colour style )
```

Set the current [StrokeStyle](#).

Parameters

<i>style</i>	The new stroke style.
--------------	-----------------------

6.15.2.19 Stroke()

```
void VectSharp.IGraphicsContext.Stroke ( )
```

Stroke the current path using the current [StrokeStyle](#), [LineWidth](#), [LineCap](#), [LineJoin](#) and [LineDash](#).

6.15.2.20 StrokeText()

```
void VectSharp.IGraphicsContext.StrokeText (
    string text,
    double x,
    double y )
```

Stroke the outline of a text string using the current [Font](#) and [TextBaseline](#).

Parameters

<i>text</i>	The string to draw.
<i>x</i>	The horizontal coordinate of the text origin.
<i>y</i>	The vertical coordinate of the text origin.

6.15.2.21 Transform()

```
void VectSharp.IGraphicsContext.Transform (
    double a,
    double b,
    double c,
    double d,
    double e,
    double f )
```

Transform the coordinate system with the specified transformation matrix [[a, c, e], [b, d, f], [0, 0, 1]].

Parameters

<i>a</i>	The first element of the first column.
<i>b</i>	The second element of the first column.
<i>c</i>	The first element of the second column.
<i>d</i>	The second element of the second column.
<i>e</i>	The first element of the third column.
<i>f</i>	The second element of the third column.

6.15.2.22 Translate()

```
void VectSharp.IGraphicsContext.Translate (
    double x,
    double y )
```

Translate the coordinate system origin.

Parameters

<i>x</i>	The horizontal translation.
<i>y</i>	The vertical translation.

6.15.3 Property Documentation**6.15.3.1 FillStyle**

`Colour` VectSharp.IGraphicsContext.FillStyle [get]

Current colour used to fill paths.

Definition at line 2192 of file Graphics.cs.

6.15.3.2 Font

`Font` VectSharp.IGraphicsContext.Font [get], [set]

The current font.

Definition at line 2137 of file Graphics.cs.

6.15.3.3 Height

```
double VectSharp.IGraphicsContext.Height [get]
```

Height of the graphic surface.

Definition at line 2091 of file Graphics.cs.

6.15.3.4 LineCap

```
LineCaps VectSharp.IGraphicsContext.LineCap [set]
```

Current line cap used to stroke paths.

Definition at line 2256 of file Graphics.cs.

6.15.3.5 LineJoin

```
LineJoins VectSharp.IGraphicsContext.LineJoin [set]
```

Current line join used to stroke paths.

Definition at line 2261 of file Graphics.cs.

6.15.3.6 LineWidth

```
double VectSharp.IGraphicsContext.LineWidth [get], [set]
```

Current line width used to stroke paths.

Definition at line 2251 of file Graphics.cs.

6.15.3.7 StrokeStyle

```
Colour VectSharp.IGraphicsContext.StrokeStyle [get]
```

Current colour used to stroke paths.

Definition at line 2209 of file Graphics.cs.

6.15.3.8 Tag

```
string VectSharp.IGraphicsContext.Tag [get], [set]
```

The current tag. How this can be used depends on each implementation.

Definition at line 2272 of file Graphics.cs.

6.15.3.9 TextBaseline

```
TextBaselines VectSharp.IGraphicsContext.TextBaseline [get], [set]
```

The current text baseline.

Definition at line 2142 of file Graphics.cs.

6.15.3.10 Width

```
double VectSharp.IGraphicsContext.Width [get]
```

Width of the graphic surface.

Definition at line 2086 of file Graphics.cs.

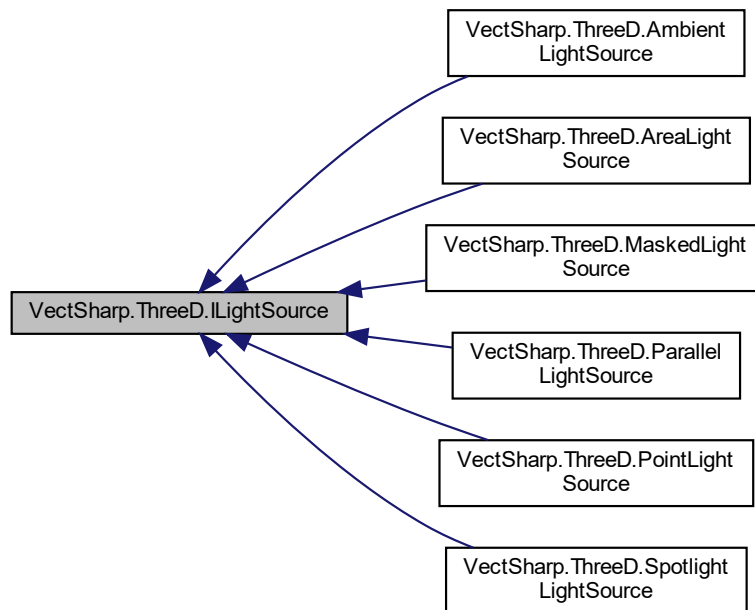
The documentation for this interface was generated from the following file:

- VectSharp/Graphics.cs

6.16 VectSharp.ThreeD.ILightSource Interface Reference

Represents a light source.

Inheritance diagram for VectSharp.ThreeD.ILightSource:



Public Member Functions

- [LightIntensity](#) [GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< Triangle3DElement > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles .

Properties

- bool [CastsShadow](#) [get]
Determines whether the light casts a shadow or not.

6.16.1 Detailed Description

Represents a light source.

Definition at line 48 of file Lights.cs.

6.16.2 Member Function Documentation

6.16.2.1 GetLightAt()

```
LightIntensity VectSharp.ThreeD.ILightSource.GetLightAt (
    Point3D point )
```

Computes the light intensity at the specified point, without taking into account any obstructions.

Parameters

<i>point</i>	The Point3DElement at which the light intensity should be computed.
--------------	---

Returns

Implemented in [VectSharp.ThreeD.AreaLightSource](#), [VectSharp.ThreeD.MaskedLightSource](#), [VectSharp.ThreeD.SpotlightLightSource](#), [VectSharp.ThreeD.PointLightSource](#), [VectSharp.ThreeD.ParallelLightSource](#), and [VectSharp.ThreeD.AmbientLightSource](#).

6.16.2.2 GetObstruction()

```
double VectSharp.ThreeD.ILightSource.GetObstruction (
    Point3D point,
    IEnumerable< Triangle3DElement > shadowingTriangles )
```

Determines the amount of obstruction of the light that results at *point* due to the specified *shadowingTriangles* .

Parameters

<i>point</i>	The Point3D at which the obstruction should be computed.
<i>shadowingTriangles</i>	A collection of Triangle3DElement casting shadows.

Returns

1 if the light is completely obstructed, 0 if the light is completely visible, a value between these if the light is partially obstructed.

Implemented in [VectSharp.ThreeD.AreaLightSource](#), [VectSharp.ThreeD.MaskedLightSource](#), [VectSharp.ThreeD.SpotlightLightSource](#), [VectSharp.ThreeD.PointLightSource](#), [VectSharp.ThreeD.ParallelLightSource](#), and [VectSharp.ThreeD.AmbientLightSource](#).

6.16.3 Property Documentation

6.16.3.1 CastsShadow

```
bool VectSharp.ThreeD.ILightSource.CastsShadow [get]
```

Determines whether the light casts a shadow or not.

Definition at line 60 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.17 VectSharp.MuPDFUtils.ImageURIParser Class Reference

Provides a method to parse an image URI into a page.

Static Public Member Functions

- static Func< string, bool, [Page](#) > [Parser](#) (Func< string, bool, [Page](#) > parseSVG)
Parses an image URI into a page. This is intended to replace the default image URI interpreter in [VectSharp.SVG.Parser.ParseImageURI](#). To do this, use something like:

6.17.1 Detailed Description

Provides a method to parse an image URI into a page.

Definition at line 29 of file ImageURIParser.cs.

6.17.2 Member Function Documentation

6.17.2.1 Parser()

```
static Func<string, bool, Page> VectSharp.MuPDFUtils.ImageURIParser.Parser (
    Func< string, bool, Page > parseSVG ) [static]
```

Parses an image URI into a page. This is intended to replace the default image URI interpreter in [VectSharp.SVG.Parser.ParseImageURI](#). To do this, use something like:

```
VectSharp.SVG.Parser.ParseImageURI = VectSharp.MuPDFUtils.ImageURIParser.Parser (VectShar
```

Parameters

<i>parseSVG</i>	A function to parse an SVG image uri into a page. You should pass VectSharp.SVG.Parser.ParseSVGURI as this argument.
-----------------	--

Returns

A function to parse an image URI into a page.

Definition at line 37 of file ImageURIParser.cs.

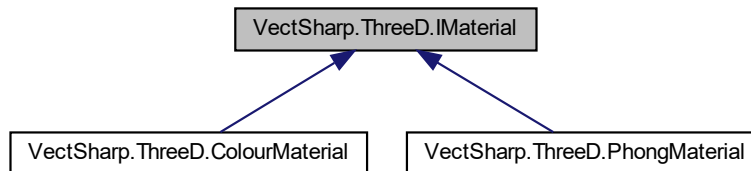
The documentation for this class was generated from the following file:

- VectSharp.MuPDFUtils/ImageURIParser.cs

6.18 VectSharp.ThreeD.IMaterial Interface Reference

Represents a material used to the determine the appearance of Triangle3DElement.

Inheritance diagram for VectSharp.ThreeD.IMaterial:



Public Member Functions

- [Colour](#) [GetColour](#) (Point3D point, NormalizedVector3D surfaceNormal, Camera camera, IList< [ILightSource](#) > lights, IList< double > obstructions)
Obtains the [Colour](#) at the specified point.

6.18.1 Detailed Description

Represents a material used to the determine the appearance of Triangle3DElement.

Definition at line 14 of file Materials.cs.

6.18.2 Member Function Documentation

6.18.2.1 GetColour()

```

Colour VectSharp.ThreeD.IMaterial.GetColour (
    Point3D point,
    NormalizedVector3D surfaceNormal,
    Camera camera,
    IList< ILightSource > lights,
    IList< double > obstructions )
  
```

Obtains the [Colour](#) at the specified point.

Parameters

<i>point</i>	The point whose colour should be determined.
<i>surfaceNormal</i>	The normal to the surface at the specified <i>point</i> .
<i>camera</i>	The camera being used to render the scene.
<i>lights</i>	A list of light sources that are present in the scene.
<i>obstructions</i>	A list of values indicating how obstructed each light source is.

Returns

The [Colour](#) of the specified point.

Implemented in [VectSharp.ThreeD.PhongMaterial](#), and [VectSharp.ThreeD.ColourMaterial](#).

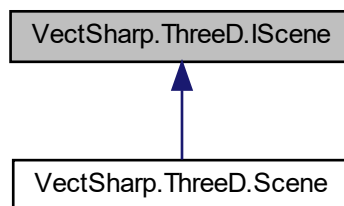
The documentation for this interface was generated from the following file:

- VectSharp.ThreeD/Materials.cs

6.19 VectSharp.ThreeD.IScene Interface Reference

Represents a 3D scene.

Inheritance diagram for VectSharp.ThreeD.IScene:



Public Member Functions

- void [AddElement](#) (Element3D element)
Adds the specified element to the scene.
- void [AddRange](#) (IEnumerable< Element3D > elements)
Adds the specified elements to the scene.
- void [Replace](#) (Func< Element3D, Element3D > replacementFunction)
Replaces each element in the scene with the element returned by the replacementFunction .
- void [Replace](#) (Func< Element3D, IEnumerable< Element3D >> replacementFunction)
Replaces each element in the scene with the element(s) returned by the replacementFunction .

Properties

- IEnumerable< Element3D > [SceneElements](#) [get]
The Element3Ds constituting the scene.
- object [SceneLock](#) [get]
An object used to synchronise multithreaded rendering of the same scene.

6.19.1 Detailed Description

Represents a 3D scene.

Definition at line 9 of file Scene.cs.

6.19.2 Member Function Documentation

6.19.2.1 AddElement()

```
void VectSharp.ThreeD.IScene.AddElement (
    Element3D element )
```

Adds the specified *element* to the scene.

Parameters

<i>element</i>	The Element3D to add.
----------------	-----------------------

Implemented in [VectSharp.ThreeD.Scene](#).

6.19.2.2 AddRange()

```
void VectSharp.ThreeD.IScene.AddRange (
    IEnumerable< Element3D > elements )
```

Adds the specified *elements* to the scene.

Parameters

<i>elements</i>	A collection of Element3Ds to add.
-----------------	------------------------------------

Implemented in [VectSharp.ThreeD.Scene](#).

6.19.2.3 Replace() [1/2]

```
void VectSharp.ThreeD.IScene.Replace (
    Func< Element3D, Element3D > replacementFunction )
```

Replaces each element in the scene with the element returned by the *replacementFunction* .

Parameters

<i>replacementFunction</i>	A function replacing each Element3D in the scene with another Element3D.
----------------------------	--

Implemented in [VectSharp.ThreeD.Scene](#).

6.19.2.4 Replace() [2/2]

```
void VectSharp.ThreeD.IScene.Replace (
    Func< Element3D, IEnumerable< Element3D >> replacementFunction )
```

Replaces each element in the scene with the element(s) returned by the *replacementFunction* .

Parameters

<i>replacementFunction</i>	A function replacing each Element3D in the scene with 0 or more Element3Ds.
----------------------------	---

Implemented in [VectSharp.ThreeD.Scene](#).

6.19.3 Property Documentation

6.19.3.1 SceneElements

```
IEnumerable<Element3D> VectSharp.ThreeD.IScene.SceneElements [get]
```

The Element3Ds constituting the scene.

Definition at line 14 of file Scene.cs.

6.19.3.2 SceneLock

```
object VectSharp.ThreeD.IScene.SceneLock [get]
```

An object used to synchronise multithreaded rendering of the same scene.

Definition at line 43 of file Scene.cs.

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

- VectSharp.ThreeD/Scene.cs

6.20 VectSharp.ThreeD.LightIntensity Struct Reference

Represents the intensity of a light source at a particular point.

Public Member Functions

- [LightIntensity](#) (double intensity, NormalizedVector3D direction)
Creates a new [LightIntensity](#).
- void [Deconstruct](#) (out double intensity, out NormalizedVector3D direction)
Deconstructs the struct.

Public Attributes

- double [Intensity](#)
The intensity of the light.
- NormalizedVector3D [Direction](#)
The direction towards from which the light comes.

6.20.1 Detailed Description

Represents the intensity of a light source at a particular point.

Definition at line 10 of file Lights.cs.

6.20.2 Constructor & Destructor Documentation

6.20.2.1 LightIntensity()

```
VectSharp.ThreeD.LightIntensity.LightIntensity (
    double intensity,
    NormalizedVector3D direction )
```

Creates a new [LightIntensity](#).

Parameters

<i>intensity</i>	The intensity of the light.
<i>direction</i>	The direction from which the light comes.

Definition at line 27 of file Lights.cs.

6.20.3 Member Function Documentation

6.20.3.1 Deconstruct()

```
void VectSharp.ThreeD.LightIntensity.Deconstruct (
    out double intensity,
    out NormalizedVector3D direction )
```

Deconstructs the struct.

Parameters

<i>intensity</i>	This parameter will hold the Intensity of the light.
<i>direction</i>	This parameter will hold the Direction of the light.

Definition at line 38 of file Lights.cs.

6.20.4 Member Data Documentation

6.20.4.1 Direction

```
NormalizedVector3D VectSharp.ThreeD.LightIntensity.Direction
```

The direction towards from which the light comes.

Definition at line 20 of file Lights.cs.

6.20.4.2 Intensity

```
double VectSharp.ThreeD.LightIntensity.Intensity
```

The intensity of the light.

Definition at line 15 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.21 VectSharp.LineDash Struct Reference

Represents instructions on how to paint a dashed line.

Public Member Functions

- [LineDash](#) (double unitsOn, double unitsOff, double phase)
Define a new line dash pattern.

Public Attributes

- double [UnitsOn](#)
Length of the "on" (painted) segment.
- double [UnitsOff](#)
Length of the "off" (not painted) segment.
- double [Phase](#)
Position in the dash pattern at which the line starts.

Static Public Attributes

- static [LineDash SolidLine](#) = new [LineDash](#)(0, 0, 0)
A solid (not dashed) line

6.21.1 Detailed Description

Represents instructions on how to paint a dashed line.

Definition at line 130 of file Graphics.cs.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 LineDash()

```
VectSharp.LineDash.LineDash (
    double unitsOn,
    double unitsOff,
    double phase )
```

Define a new line dash pattern.

Parameters

<i>unitsOn</i>	The length of the "on" (painted) segment.
<i>unitsOff</i>	The length of the "off" (not painted) segment.
<i>phase</i>	The position in the dash pattern at which the line starts.

Definition at line 158 of file Graphics.cs.

6.21.3 Member Data Documentation

6.21.3.1 Phase

```
double VectSharp.LineDash.Phase
```

Position in the dash pattern at which the line starts.

Definition at line 150 of file Graphics.cs.

6.21.3.2 SolidLine

```
LineDash VectSharp.LineDash.SolidLine = new LineDash(0, 0, 0) [static]
```

A solid (not dashed) line

Definition at line 135 of file Graphics.cs.

6.21.3.3 UnitsOff

```
double VectSharp.LineDash.UnitsOff
```

Length of the "off" (not painted) segment.

Definition at line 145 of file Graphics.cs.

6.21.3.4 UnitsOn

```
double VectSharp.LineDash.UnitsOn
```

Length of the "on" (painted) segment.

Definition at line 140 of file Graphics.cs.

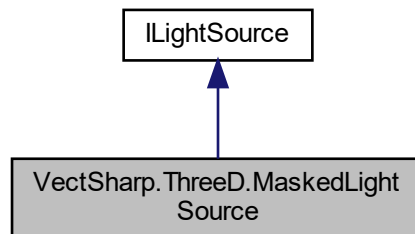
The documentation for this struct was generated from the following file:

- VectSharp/Graphics.cs

6.22 VectSharp.ThreeD.MaskedLightSource Class Reference

Represents a point light source with a stencil in front of it.

Inheritance diagram for VectSharp.ThreeD.MaskedLightSource:



Public Member Functions

- [MaskedLightSource](#) (double intensity, Point3D position, NormalizedVector3D direction, double distance, [GraphicsPath](#) mask, double maskOrientation, double triangulationResolution)
Creates a new [MaskedLightSource](#) by triangulating the specified [GraphicsPath](#).
- [MaskedLightSource](#) (double intensity, Point3D position, NormalizedVector3D direction, double distance, IEnumerable< [GraphicsPath](#) > triangulatedMask, double maskOrientation)
Creates a new [MaskedLightSource](#) using the specified triangulatedMask.
- [LightIntensity](#) [GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< [Triangle3DElement](#) > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles.

Properties

- bool [CastsShadow](#) = true [get, set]
- Point3D [Position](#) [get]
The position of the light source.
- Point3D [Origin](#) [get]
The projection of the [Position](#) on the mask plane along the light's [Direction](#).
- NormalizedVector3D [Direction](#) [get]
The direction of the light.
- double [Distance](#) [get]
The distance between the light source and the mask plane.
- double [Intensity](#) [get, set]
The base intensity of the light.
- double [DistanceAttenuationExponent](#) = 2 [get, set]
An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.
- double [AngleAttenuationExponent](#) = 1 [get, set]
An exponent determining how fast the light attenuates away from the light's axis. Set to 0 to disable angular attenuation.

6.22.1 Detailed Description

Represents a point light source with a stencil in front of it.

Definition at line 368 of file Lights.cs.

6.22.2 Constructor & Destructor Documentation

6.22.2.1 MaskedLightSource() [1/2]

```
VectSharp.ThreeD.MaskedLightSource.MaskedLightSource (
    double intensity,
    Point3D position,
    NormalizedVector3D direction,
    double distance,
    GraphicsPath mask,
    double maskOrientation,
    double triangulationResolution )
```

Creates a new [MaskedLightSource](#) by triangulating the specified [GraphicsPath](#).

Parameters

<i>intensity</i>	The base intensity of the light.
<i>position</i>	The position of the light source.
<i>direction</i>	The direction of the light.
<i>distance</i>	The distance between the light source and the mask plane.
<i>mask</i>	A GraphicsPath representing the transparent part of the mask.
<i>maskOrientation</i>	An angle in radians determining the orientation of the 2D mask in the mask plane.
<i>triangulationResolution</i>	The resolution to use to triangulate the <i>mask</i> .

Definition at line 420 of file Lights.cs.

6.22.2.2 MaskedLightSource() [2/2]

```
VectSharp.ThreeD.MaskedLightSource.MaskedLightSource (
    double intensity,
    Point3D position,
    NormalizedVector3D direction,
    double distance,
    IEnumerable< GraphicsPath > triangulatedMask,
    double maskOrientation )
```

Creates a new [MaskedLightSource](#) using the specified *triangulatedMask* .

Parameters

<i>intensity</i>	The base intensity of the light.
<i>position</i>	The position of the light source.
<i>direction</i>	The direction of the light.
<i>distance</i>	The distance between the light source and the mask plane.
<i>triangulatedMask</i>	A collection of GraphicsPaths representing the transparent part of the mask. Each GraphicsPath should represent a single triangle.
<i>maskOrientation</i>	An angle in radians determining the orientation of the 2D mask in the mask plane.

Definition at line 434 of file Lights.cs.

6.22.3 Property Documentation

6.22.3.1 AngleAttenuationExponent

```
double VectSharp.ThreeD.MaskedLightSource.AngleAttenuationExponent = 1 [get], [set]
```

An exponent determining how fast the light attenuates away from the light's axis. Set to 0 to disable angular attenuation.

Definition at line 408 of file Lights.cs.

6.22.3.2 Direction

```
NormalizedVector3D VectSharp.ThreeD.MaskedLightSource.Direction [get]
```

The direction of the light.

Definition at line 386 of file Lights.cs.

6.22.3.3 Distance

```
double VectSharp.ThreeD.MaskedLightSource.Distance [get]
```

The distance between the light source and the mask plane.

Definition at line 391 of file Lights.cs.

6.22.3.4 DistanceAttenuationExponent

```
double VectSharp.ThreeD.MaskedLightSource.DistanceAttenuationExponent = 2 [get], [set]
```

An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.

Definition at line 403 of file Lights.cs.

6.22.3.5 Intensity

```
double VectSharp.ThreeD.MaskedLightSource.Intensity [get], [set]
```

The base intensity of the light.

Definition at line 398 of file Lights.cs.

6.22.3.6 Origin

```
Point3D VectSharp.ThreeD.MaskedLightSource.Origin [get]
```

The projection of the [Position](#) on the mask plane along the light's [Direction](#).

Definition at line 381 of file Lights.cs.

6.22.3.7 Position

```
Point3D VectSharp.ThreeD.MaskedLightSource.Position [get]
```

The position of the light source.

Definition at line 376 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.23 VectSharp.ThreeD.ObjectFactory Class Reference

A static class containing methods to create complex 3D objects.

Static Public Member Functions

- static List< Element3D > [CreateCube](#) (Point3D center, double size, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a cube.
- static List< Element3D > [CreateCuboid](#) (Point3D center, double sizeX, double sizeY, double sizeZ, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a cuboid.
- static List< Element3D > [CreateRectangle](#) (Point3D point1, Point3D point2, Point3D point3, Point3D point4, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a quadrilater. All the vertices need not be coplanar.
- static List< Element3D > [CreateRectangle](#) (Point3D point1, Point3D point2, Point3D point3, Point3D point4, NormalizedVector3D point1Normal, NormalizedVector3D point2Normal, NormalizedVector3D point3Normal, NormalizedVector3D point4Normal, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a quadrilater, specifying the vertex normals at the four vertices. All the vertices need not be coplanar.
- static List< Element3D > [CreateSphere](#) (Point3D center, double radius, int steps, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a sphere.
- static List< Element3D > [CreateTetrahedron](#) (Point3D center, double radius, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a tetrahedron inscribed in a sphere.
- static List< Element3D > [CreatePolygon](#) ([GraphicsPath](#) polygon2D, double triangulationResolution, Point3D origin, NormalizedVector3D xAxis, NormalizedVector3D yAxis, bool reverseTriangles, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a flat polygon.
- static List< Element3D > [CreatePrism](#) ([GraphicsPath](#) polygonBase2D, double triangulationResolution, Point3D bottomOrigin, Point3D topOrigin, NormalizedVector3D baseXAxis, NormalizedVector3D baseYAxis, IEnumerable< [IMaterial](#) > fill, string tag=null, int zIndex=0)
Creates a prism with the specified base.
- static List< Element3D > [CreateWireframe](#) (IEnumerable< Element3D > object3D, [Colour](#) colour, double thickness=1, [LineCaps](#) lineCap=[LineCaps.Butt](#), [LineDash?](#) lineDash=null, string tag=null, int zIndex=0)
Creates a wireframe from a collection of Element3Ds.
- static List< Element3D > [CreatePoints](#) (IEnumerable< Element3D > object3D, [Colour](#) colour, double diameter=1, string tag=null, int zIndex=0)
Obtains a list of Point3DElement corresponding to the vertices of a list of Element3Ds.

6.23.1 Detailed Description

A static class containing methods to create complex 3D objects.

Definition at line 11 of file ObjectFactory.cs.

6.23.2 Member Function Documentation

6.23.2.1 CreateCube()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateCube (
    Point3D center,
    double size,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]
```

Creates a cube.

Parameters

<i>center</i>	The centre of the cube.
<i>size</i>	The length of each side of the cube.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list of Triangle3DElements that constitute the cube.

Definition at line 22 of file ObjectFactory.cs.

6.23.2.2 CreateCuboid()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateCuboid (
    Point3D center,
    double sizeX,
    double sizeY,
    double sizeZ,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]
```

Creates a cuboid.

Parameters

<i>center</i>	The centre of the cube.
<i>sizeX</i>	The length of the sides of the cube parallel to the x axis.
<i>sizeY</i>	The length of the sides of the cube parallel to the y axis.
<i>sizeZ</i>	The length of the sides of the cube parallel to the z axis.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list of Triangle3DElements that constitute the cuboid.

Definition at line 38 of file ObjectFactory.cs.

6.23.2.3 CreatePoints()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreatePoints (
    IEnumerable< Element3D > object3D,
    Colour colour,
    double diameter = 1,
    string tag = null,
    int zIndex = 0 ) [static]
```

Obtains a list of Point3DElement corresponding to the vertices of a list of Element3Ds.

Parameters

<i>object3D</i>	The collection of Element3Ds. Point3DElements are ignored.
<i>colour</i>	The colour of the Point3DElements returned by this method.
<i>diameter</i>	The diameter of the Point3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Point3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Point3DElements returned by this method.

Returns

A list of Point3DElements corresponding to the vertices of the Element3Ds.

Definition at line 395 of file ObjectFactory.cs.

6.23.2.4 CreatePolygon()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreatePolygon (
    GraphicsPath polygon2D,
    double triangulationResolution,
    Point3D origin,
    NormalizedVector3D xAxis,
    NormalizedVector3D yAxis,
    bool reverseTriangles,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]
```

Creates a flat polygon.

Parameters

<i>polygon2D</i>	A 2D GraphicsPath representing the polygon.
<i>triangulationResolution</i>	The resolution that will be used to linearise curve segments in the GraphicsPath .
<i>origin</i>	A Point3D that will correspond to the origin of the 2D reference system.
<i>xAxis</i>	A NormalizedVector3D that will correspond to the x axis of the 2D reference system. This will be orthonormalised to the <i>yAxis</i> .
<i>yAxis</i>	A NormalizedVector3D that will correspond to the y axis of the 2D reference system.
<i>reverseTriangles</i>	Indicates whether the order of the points (and thus the normals) of all the triangles returned by this method should be reversed.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
Generated by Doxygen	
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list of Triangle3DElements that constitute the polygon.

Definition at line 256 of file ObjectFactory.cs.

6.23.2.5 CreatePrism()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreatePrism (
    GraphicsPath polygonBase2D,
    double triangulationResolution,
    Point3D bottomOrigin,
    Point3D topOrigin,
    NormalizedVector3D baseXAxis,
    NormalizedVector3D baseYAxis,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]
```

Creates a prism with the specified base.

Parameters

<i>polygonBase2D</i>	A 2D GraphicsPath representing the base of the prism.
<i>triangulationResolution</i>	The resolution that will be used to linearise curve segments in the GraphicsPath .
<i>bottomOrigin</i>	A Point3D that will correspond to the origin of the 2D reference system of the bottom base.
<i>topOrigin</i>	A Point3D that will correspond to the origin of the 2D reference system of the top base.
<i>baseXAxis</i>	A NormalizedVector3D that will correspond to the x axis of the 2D reference system of the bases. This will be orthonormalised to the <i>baseYAxis</i> .
<i>baseYAxis</i>	A NormalizedVector3D that will correspond to the y axis of the 2D reference system of the bases.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list of Triangle3DElements that constitute the prism.

Definition at line 297 of file ObjectFactory.cs.

6.23.2.6 CreateRectangle() [1/2]

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateRectangle (
    Point3D point1,
```

```

    Point3D point2,
    Point3D point3,
    Point3D point4,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]

```

Creates a quadrilater. All the vertices need not be coplanar.

Parameters

<i>point1</i>	The first vertex of the quadrilater.
<i>point2</i>	The second vertex of the quadrilater.
<i>point3</i>	The third vertex of the quadrilater.
<i>point4</i>	The fourth vertex of the quadrilater.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list containing two Triangle3DElements representing the quadrilater.

Definition at line 76 of file ObjectFactory.cs.

6.23.2.7 CreateRectangle() [2/2]

```

static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateRectangle (
    Point3D point1,
    Point3D point2,
    Point3D point3,
    Point3D point4,
    NormalizedVector3D point1Normal,
    NormalizedVector3D point2Normal,
    NormalizedVector3D point3Normal,
    NormalizedVector3D point4Normal,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]

```

Creates a quadrilater, specifying the vertex normals at the four vertices. All the vertices need not be coplanar.

Parameters

<i>point1</i>	The first vertex of the quadrilater.
<i>point2</i>	The second vertex of the quadrilater.
<i>point3</i>	The third vertex of the quadrilater.
<i>point4</i>	The fourth vertex of the quadrilater.
<i>point1Normal</i>	The vertex normal at the first vertex of the quadrilater.
<i>point2Normal</i>	The vertex normal at the second vertex of the quadrilater.

Parameters

<i>point3Normal</i>	The vertex normal at the third vertex of the quadrilater.
<i>point4Normal</i>	The vertex normal at the fourth vertex of the quadrilater.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list containing two Triangle3DElements representing the quadrilater.

Definition at line 106 of file ObjectFactory.cs.

6.23.2.8 CreateSphere()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateSphere (
    Point3D center,
    double radius,
    int steps,
    IEnumerable< IMaterial > fill,
    string tag = null,
    int zIndex = 0 ) [static]
```

Creates a sphere.

Parameters

<i>center</i>	The centre of the sphere.
<i>radius</i>	The radius of the sphere.
<i>steps</i>	The number of meridians and parallels to use when generating the sphere.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list of Triangle3DElements that constitute the sphere.

Definition at line 131 of file ObjectFactory.cs.

6.23.2.9 CreateTetrahedron()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateTetrahedron (
    Point3D center,
```

```
double radius,
IEnumerable< IMaterial > fill,
string tag = null,
int zIndex = 0 ) [static]
```

Creates a tetrahedron inscribed in a sphere.

Parameters

<i>center</i>	The centre of the tetrahedron.
<i>radius</i>	The radius of the sphere in which the tetrahedron is inscribed.
<i>fill</i>	A collection of materials that will be applied to the Triangle3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Triangle3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Triangle3DElements returned by this method.

Returns

A list of Triangle3DElements that constitute the sphere.

Definition at line 221 of file ObjectFactory.cs.

6.23.2.10 CreateWireframe()

```
static List<Element3D> VectSharp.ThreeD.ObjectFactory.CreateWireframe (
    IEnumerable< Element3D > object3D,
    Colour colour,
    double thickness = 1,
    LineCaps lineCap = LineCaps.Butt,
    LineDash? lineDash = null,
    string tag = null,
    int zIndex = 0 ) [static]
```

Creates a wireframe from a collection of Element3Ds.

Parameters

<i>object3D</i>	The collection of Element3Ds. Line3DElements and Point3DElements are ignored.
<i>colour</i>	The colour of the Line3DElements returned by this method.
<i>thickness</i>	The thickness of the Line3DElements returned by this method.
<i>lineCap</i>	The line cap of the Line3DElements returned by this method.
<i>lineDash</i>	The line dash of the Line3DElements returned by this method.
<i>tag</i>	A tag that will be applied to the Line3DElements returned by this method.
<i>zIndex</i>	A z-index that will be applied to the Line3DElements returned by this method.

Returns

A list of Line3DElements that constitute the wireframe.

Definition at line 353 of file ObjectFactory.cs.

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

- VectSharp.ThreeD/ObjectFactory.cs

6.24 VectSharp.Page Class Reference

Represents a [Graphics](#) object with a width and height.

Public Member Functions

- [Page](#) (double width, double height)
Create a new page.
- void [Crop](#) ([Point](#) topLeft, [Size](#) size)
Translate and resize the [Page](#) so that it displays the rectangle defined by topLeft and size .

Properties

- double [Width](#) [get, set]
Width of the page.
- double [Height](#) [get, set]
Height of the page.
- [Graphics Graphics](#) [get, set]
[Graphics](#) surface of the page.
- [Colour Background](#) = [Colour.FromRgba](#)(255, 255, 255, 0) [get, set]
Background colour of the page.

6.24.1 Detailed Description

Represents a [Graphics](#) object with a width and height.

Definition at line 47 of file Document.cs.

6.24.2 Constructor & Destructor Documentation

6.24.2.1 Page()

```
VectSharp.Page.Page (
    double width,
    double height )
```

Create a new page.

Parameters

<i>width</i>	The width of the page.
<i>height</i>	The height of the page.

Definition at line 74 of file Document.cs.

6.24.3 Member Function Documentation

6.24.3.1 Crop()

```
void VectSharp.Page.Crop (  
    Point topLeft,  
    Size size )
```

Translate and resize the [Page](#) so that it displays the rectangle defined by *topLeft* and *size* .

Parameters

<i>topLeft</i>	The top left corner of the area to include in the page.
<i>size</i>	The size of the area to include in the page.

Definition at line 88 of file Document.cs.

6.24.4 Property Documentation

6.24.4.1 Background

```
Colour VectSharp.Page.Background = Colour.FromRgba(255, 255, 255, 0) [get], [set]
```

Background colour of the page.

Definition at line 67 of file Document.cs.

6.24.4.2 Graphics

```
Graphics VectSharp.Page.Graphics [get], [set]
```

[Graphics](#) surface of the page.

Definition at line 62 of file Document.cs.

6.24.4.3 Height

```
double VectSharp.Page.Height [get], [set]
```

Height of the page.

Definition at line 57 of file Document.cs.

6.24.4.4 Width

```
double VectSharp.Page.Width [get], [set]
```

Width of the page.

Definition at line 52 of file Document.cs.

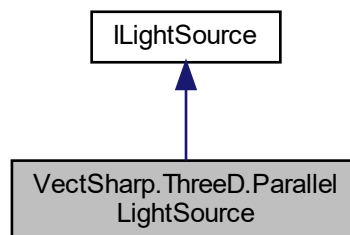
The documentation for this class was generated from the following file:

- VectSharp/Document.cs

6.25 VectSharp.ThreeD.ParallelLightSource Class Reference

Represents a parallel light source.

Inheritance diagram for VectSharp.ThreeD.ParallelLightSource:



Public Member Functions

- [ParallelLightSource](#) (double intensity, NormalizedVector3D direction)
Creates a new [ParallelLightSource](#) instance.
- [LightIntensity GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< Triangle3DElement > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles .

Properties

- double [Intensity](#) [get, set]
The intensity of the light.
- NormalizedVector3D [Direction](#) [get]
The direction along which the light travels.
- NormalizedVector3D [ReverseDirection](#) [get]
The reverse of [Direction](#).
- bool [CastsShadow](#) = true [get, set]

6.25.1 Detailed Description

Represents a parallel light source.

Definition at line 109 of file Lights.cs.

6.25.2 Constructor & Destructor Documentation

6.25.2.1 ParallelLightSource()

```
VectSharp.ThreeD.ParallelLightSource.ParallelLightSource (
    double intensity,
    NormalizedVector3D direction )
```

Creates a new [ParallelLightSource](#) instance.

Parameters

<i>intensity</i>	The intensity of the light.
<i>direction</i>	The direction along which the light travels.

Definition at line 134 of file Lights.cs.

6.25.3 Property Documentation

6.25.3.1 Direction

```
NormalizedVector3D VectSharp.ThreeD.ParallelLightSource.Direction [get]
```

The direction along which the light travels.

Definition at line 119 of file Lights.cs.

6.25.3.2 Intensity

```
double VectSharp.ThreeD.ParallelLightSource.Intensity [get], [set]
```

The intensity of the light.

Definition at line 114 of file Lights.cs.

6.25.3.3 ReverseDirection

```
NormalizedVector3D VectSharp.ThreeD.ParallelLightSource.ReverseDirection [get]
```

The reverse of [Direction](#).

Definition at line 124 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.26 VectSharp.SVG.Parser Class Reference

Contains methods to read an [SVG](#) image file.

Static Public Member Functions

- static [Page ParseSVGURI](#) (string uri, bool ignored=false)
Parses an [SVG](#) image URI.
- static [Page FromString](#) (string svgSource)
Parses [SVG](#) source into a [Page](#) containing the image represented by the code.
- static [Page FromFile](#) (string fileName)
Parses an [SVG](#) image file into a [Page](#) containing the image.
- static [Page FromStream](#) (Stream svgSourceStream)
Parses an stream containing [SVG](#) source code into a [Page](#) containing the image represented by the code.

Static Public Attributes

- static Func< string, bool, [Page](#) > [ParseImageURI](#)
A function that takes as input an image URI and a boolean value indicating whether the image should be interpolated, and returns a [Page](#) object containing the image. By default, this is equal to [ParseSVGURI](#), i.e. it is only able to parse [SVG](#) images. If you wish to enable the parsing of other formats, you should install the "VectSharp.MuPDFUtils" NuGet package and enable the parser in your program by doing something like:

6.26.1 Detailed Description

Contains methods to read an [SVG](#) image file.

Definition at line 32 of file SVGParser.cs.

6.26.2 Member Function Documentation

6.26.2.1 FromFile()

```
static Page VectSharp.SVG.Parser.FromFile (  
    string fileName ) [static]
```

Parses an [SVG](#) image file into a [Page](#) containing the image.

Parameters

<i>fileName</i>	The path to the SVG image file.
-----------------	---

Returns

A [Page](#) containing the image represented by the file.

Definition at line 144 of file SVGParser.cs.

6.26.2.2 FromStream()

```
static Page VectSharp.SVG.Parser.FromStream (  
    Stream svgSourceStream ) [static]
```

Parses a stream containing [SVG](#) source code into a [Page](#) containing the image represented by the code.

Parameters

<i>svgSourceStream</i>	The stream containing SVG source code.
------------------------	--

Returns

A [Page](#) containing the image represented by the *svgSourceStream* .

Definition at line 154 of file SVGParser.cs.

6.26.2.3 FromString()

```
static Page VectSharp.SVG.Parser.FromString (
    string svgSource ) [static]
```

Parses [SVG](#) source into a [Page](#) containing the image represented by the code.

Parameters

<i>svgSource</i>	The SVG source code.
------------------	--------------------------------------

Returns

A [Page](#) containing the image represented by the *svgSource* .

Definition at line 102 of file SVGParser.cs.

6.26.2.4 ParseSVGURI()

```
static Page VectSharp.SVG.Parser.ParseSVGURI (
    string uri,
    bool ignored = false ) [static]
```

Parses an [SVG](#) image URI.

Parameters

<i>uri</i>	The image URI to parse.
<i>ignored</i>	This value is ignored and is only needed for compatibility.

Returns

A [Page](#) containing the parsed [SVG](#) image, or null.

Definition at line 53 of file SVGParser.cs.

6.26.3 Member Data Documentation

6.26.3.1 ParseImageURI

```
Func<string, bool, Page> VectSharp.SVG.Parser.ParseImageURI [static]
```

A function that takes as input an image URI and a boolean value indicating whether the image should be interpolated, and returns a [Page](#) object containing the image. By default, this is equal to [ParseSVGURI](#), i.e. it is only able

to parse [SVG](#) images. If you wish to enable the parsing of other formats, you should install the "VectSharp.MuPDFUtils" NuGet package and enable the parser in your program by doing something like:

```
VectSharp.SVG.Parser.ParseImageURI = VectSharp.MuPDFUtils.ImageURIParser.Parser (VectSharp.SVG.Parser.ParseImageURI)
```

Definition at line 45 of file SVGParser.cs.

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

- VectSharp.SVG/SVGParser.cs

6.27 VectSharp.PDF.PDFContextInterpreter Class Reference

Contains methods to render a [Document](#) as a [PDF](#) document.

Public Types

- enum [TextOptions](#) { [TextOptions.SubsetFont](#), [TextOptions.ConvertIntoPaths](#) }
Defines whether the used fonts should be included in the file.

Static Public Member Functions

- static void [SaveAsPDF](#) (this [Document](#) document, string fileName, [TextOptions](#) textOption=[TextOptions.SubsetFont](#), bool compressStreams=true)
Save the document to a [PDF](#) file.
- static void [SaveAsPDF](#) (this [Document](#) document, Stream stream, [TextOptions](#) textOption=[TextOptions.SubsetFont](#), bool compressStreams=true)
Save the document to a [PDF](#) stream.

6.27.1 Detailed Description

Contains methods to render a [Document](#) as a [PDF](#) document.

Definition at line 573 of file PDFContext.cs.

6.27.2 Member Enumeration Documentation

6.27.2.1 TextOptions

```
enum VectSharp.PDF.PDFContextInterpreter.TextOptions [strong]
```

Defines whether the used fonts should be included in the file.

Enumerator

SubsetFonts	Embeds subsetting font files containing only the glyphs for the characters that have been used.
ConvertIntoPaths	Does not embed any font file and converts all text items into paths.

Definition at line 761 of file PDFContext.cs.

6.27.3 Member Function Documentation

6.27.3.1 SaveAsPDF() [1/2]

```
static void VectSharp.PDF.PDFContextInterpreter.SaveAsPDF (
    this Document document,
    Stream stream,
    TextOptions textOption = TextOptions.SubsetFonts,
    bool compressStreams = true ) [static]
```

Save the document to a PDF stream.

Parameters

<i>document</i>	The Document to save.
<i>stream</i>	The stream to which the PDF data will be written.
<i>textOption</i>	Defines whether the used fonts should be included in the file.
<i>compressStreams</i>	Indicates whether the streams in the PDF file should be compressed.

Definition at line 783 of file PDFContext.cs.

6.27.3.2 SaveAsPDF() [2/2]

```
static void VectSharp.PDF.PDFContextInterpreter.SaveAsPDF (
    this Document document,
    string fileName,
    TextOptions textOption = TextOptions.SubsetFonts,
    bool compressStreams = true ) [static]
```

Save the document to a PDF file.

Parameters

<i>document</i>	The Document to save.
<i>fileName</i>	The full path to the file to save. If it exists, it will be overwritten.
<i>textOption</i>	Defines whether the used fonts should be included in the file.
<i>compressStreams</i>	Indicates whether the streams in the PDF file should be compressed.

Definition at line 750 of file PDFContext.cs.

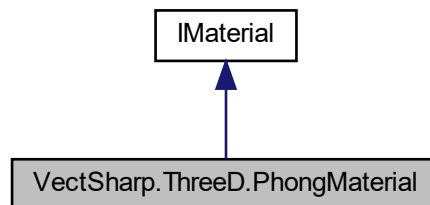
The documentation for this class was generated from the following file:

- VectSharp.PDF/PDFContext.cs

6.28 VectSharp.ThreeD.PhongMaterial Class Reference

Represents a material that uses a Phong reflection model to determine the colour of the material based on the light sources that hit it.

Inheritance diagram for VectSharp.ThreeD.PhongMaterial:



Public Member Functions

- [PhongMaterial](#) ([Colour](#) colour)
Creates a new [PhongMaterial](#) instance.
- [Colour](#) [GetColour](#) (Point3D point, NormalizedVector3D surfaceNormal, Camera camera, IList< [ILightSource](#) > lights, IList< double > obstructions)
Obtains the [Colour](#) at the specified point.

Properties

- [Colour](#) [Colour](#) [get]
The base colour of the material.
- double [AmbientReflectionCoefficient](#) = 1 [get, set]
A coefficient determining how much ambient light is reflected by the material.
- double [DiffuseReflectionCoefficient](#) = 1 [get, set]
A coefficient determining how much directional light is reflected by the material.
- double [SpecularReflectionCoefficient](#) = 1 [get, set]
A coefficient determining the intensity of specular highlights.
- double [SpecularShininess](#) = 1 [get, set]
A coefficient determining the extent of specular highlights.

6.28.1 Detailed Description

Represents a material that uses a Phong reflection model to determine the colour of the material based on the light sources that hit it.

Definition at line 57 of file Materials.cs.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 PhongMaterial()

```
VectSharp.ThreeD.PhongMaterial.PhongMaterial (
    Colour colour )
```

Creates a new [PhongMaterial](#) instance.

Parameters

<i>colour</i>	The base colour of the material.
---------------	----------------------------------

Definition at line 94 of file Materials.cs.

6.28.3 Property Documentation

6.28.3.1 AmbientReflectionCoefficient

```
double VectSharp.ThreeD.PhongMaterial.AmbientReflectionCoefficient = 1 [get], [set]
```

A coefficient determining how much ambient light is reflected by the material.

Definition at line 73 of file Materials.cs.

6.28.3.2 Colour

```
Colour VectSharp.ThreeD.PhongMaterial.Colour [get]
```

The base colour of the material.

Definition at line 62 of file Materials.cs.

6.28.3.3 DiffuseReflectionCoefficient

```
double VectSharp.ThreeD.PhongMaterial.DiffuseReflectionCoefficient = 1 [get], [set]
```

A coefficient determining how much directional light is reflected by the material.

Definition at line 78 of file Materials.cs.

6.28.3.4 SpecularReflectionCoefficient

```
double VectSharp.ThreeD.PhongMaterial.SpecularReflectionCoefficient = 1 [get], [set]
```

A coefficient determining the intensity of specular highlights.

Definition at line 83 of file Materials.cs.

6.28.3.5 SpecularShininess

```
double VectSharp.ThreeD.PhongMaterial.SpecularShininess = 1 [get], [set]
```

A coefficient determining the extent of specular highlights.

Definition at line 88 of file Materials.cs.

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

- VectSharp.ThreeD/Materials.cs

6.29 VectSharp.Point Struct Reference

Represents a point relative to an origin in the top-left corner.

Public Member Functions

- [Point](#) (double x, double y)
Create a new [Point](#).
- double [Modulus](#) ()
Computes the modulus of the vector represented by the [Point](#).
- [Point Normalize](#) ()
Normalises a [Point](#).
- bool [IsEqual](#) ([Point](#) p2, double tolerance)
Checks whether this [Point](#) is equal to another [Point](#), up to a specified tolerance.

Public Attributes

- double [X](#)
Horizontal (x) coordinate, measured to the right of the origin.
- double [Y](#)
Vertical (y) coordinate, measured to the bottom of the origin.

6.29.1 Detailed Description

Represents a point relative to an origin in the top-left corner.

Definition at line 1228 of file Graphics.cs.

6.29.2 Constructor & Destructor Documentation

6.29.2.1 Point()

```
VectSharp.Point.Point (
    double x,
    double y )
```

Create a new [Point](#).

Parameters

<i>x</i>	The horizontal (x) coordinate.
<i>y</i>	The vertical (y) coordinate.

Definition at line 1245 of file Graphics.cs.

6.29.3 Member Function Documentation

6.29.3.1 IsEqual()

```
bool VectSharp.Point.IsEqual (
    Point p2,
    double tolerance )
```

Checks whether this [Point](#) is equal to another [Point](#), up to a specified tolerance.

Parameters

<i>p2</i>	The Point to compare.
<i>tolerance</i>	The tolerance threshold.

Returns

`true` if both coordinates of the [Points](#) are closer than *tolerance* or if their relative difference (i.e. $(a - b) / (a + b) * 2$) is smaller than *tolerance*. `false` otherwise.

Definition at line 1276 of file Graphics.cs.

6.29.3.2 Modulus()

```
double VectSharp.Point.Modulus ( )
```

Computes the modulus of the vector represented by the [Point](#).

Returns

The modulus of the vector represented by the [Point](#).

Definition at line 1255 of file Graphics.cs.

6.29.3.3 Normalize()

```
Point VectSharp.Point.Normalize ( )
```

Normalises a [Point](#).

Returns

The normalised [Point](#).

Definition at line 1264 of file Graphics.cs.

6.29.4 Member Data Documentation

6.29.4.1 X

```
double VectSharp.Point.X
```

Horizontal (x) coordinate, measured to the right of the origin.

Definition at line 1233 of file Graphics.cs.

6.29.4.2 Y

```
double VectSharp.Point.Y
```

Vertical (y) coordinate, measured to the bottom of the origin.

Definition at line 1238 of file Graphics.cs.

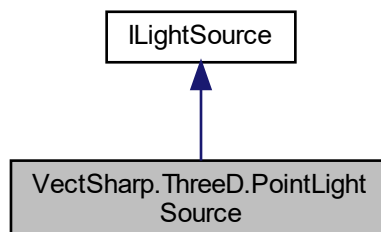
The documentation for this struct was generated from the following file:

- VectSharp/Graphics.cs

6.30 VectSharp.ThreeD.PointLightSource Class Reference

Represents a point light source.

Inheritance diagram for VectSharp.ThreeD.PointLightSource:



Public Member Functions

- [PointLightSource](#) (double intensity, Point3D position)
Creates a new [PointLightSource](#) instance.
- [LightIntensity GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< Triangle3DElement > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles .

Properties

- bool [CastsShadow](#) = true [get, set]
- Point3D [Position](#) [get, set]
The position of the light source.
- double [Intensity](#) [get, set]
The base intensity of the light.
- double [DistanceAttenuationExponent](#) = 2 [get, set]
An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.

6.30.1 Detailed Description

Represents a point light source.

Definition at line 167 of file Lights.cs.

6.30.2 Constructor & Destructor Documentation

6.30.2.1 PointLightSource()

```
VectSharp.ThreeD.PointLightSource.PointLightSource (
    double intensity,
    Point3D position )
```

Creates a new [PointLightSource](#) instance.

Parameters

<i>intensity</i>	The intensity of the light.
<i>position</i>	The position of the light source.

Definition at line 192 of file Lights.cs.

6.30.3 Property Documentation

6.30.3.1 DistanceAttenuationExponent

```
double VectSharp.ThreeD.PointLightSource.DistanceAttenuationExponent = 2 [get], [set]
```

An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.

Definition at line 185 of file Lights.cs.

6.30.3.2 Intensity

```
double VectSharp.ThreeD.PointLightSource.Intensity [get], [set]
```

The base intensity of the light.

Definition at line 180 of file Lights.cs.

6.30.3.3 Position

```
Point3D VectSharp.ThreeD.PointLightSource.Position [get], [set]
```

The position of the light source.

Definition at line 175 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.31 VectSharp.Raster.Raster Class Reference

Contains methods to render a page to a PNG image.

Static Public Member Functions

- static void [SaveAsPNG](#) (this [Page](#) page, string fileName, double scale=1)
Render the page to a PNG file.
- static void [SaveAsPNG](#) (this [Page](#) page, Stream stream, double scale=1)
Render the page to a PNG stream.

6.31.1 Detailed Description

Contains methods to render a page to a PNG image.

Definition at line 27 of file Raster.cs.

6.31.2 Member Function Documentation

6.31.2.1 SaveAsPNG() [1/2]

```
static void VectSharp.Raster.Raster.SaveAsPNG (  
    this Page page,  
    Stream stream,  
    double scale = 1 ) [static]
```

Render the page to a PNG stream.

Parameters

<i>page</i>	The Page to render.
<i>stream</i>	The stream to which the PNG data will be written.
<i>scale</i>	The scale to be used when rasterising the page. This will determine the width and height of the image file.

Definition at line 59 of file Raster.cs.

6.31.2.2 SaveAsPNG() [2/2]

```
static void VectSharp.Raster.Raster.SaveAsPNG (  
    this Page page,  
    string fileName,  
    double scale = 1 ) [static]
```

Render the page to a PNG file.

Parameters

<i>page</i>	The Page to render.
<i>fileName</i>	The full path to the file to save. If it exists, it will be overwritten.
<i>scale</i>	The scale to be used when rasterising the page. This will determine the width and height of the image file.

Definition at line 36 of file Raster.cs.

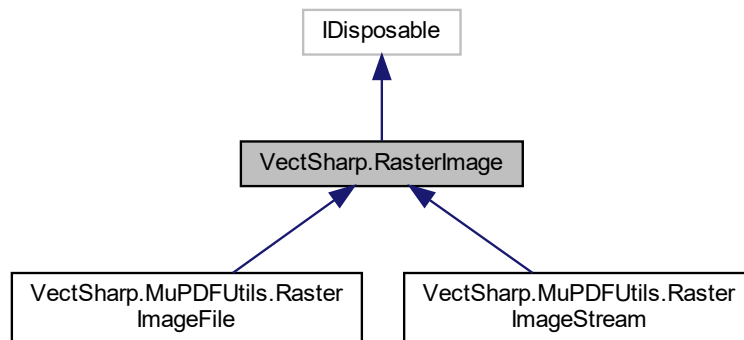
The documentation for this class was generated from the following file:

- VectSharp.Raster/Raster.cs

6.32 VectSharp.RasterImage Class Reference

Represents a raster image, created from raw pixel data. Consider using the derived classes included in the NuGet package "VectSharp.MuPDFUtils" if you need to load a raster image from a file or a Stream.

Inheritance diagram for VectSharp.RasterImage:



Public Member Functions

- [RasterImage](#) (IntPtr pixelData, int width, int height, bool hasAlpha, bool interpolate)
Creates a new [RasterImage](#) instance from the specified pixel data in RGB or RGBA format.
- [RasterImage](#) (ref [DisposableIntPtr](#) pixelData, int width, int height, bool hasAlpha, bool interpolate)
Creates a new [RasterImage](#) instance from the specified pixel data in RGB or RGBA format.
- [RasterImage](#) (byte[] data, int width, int height, [PixelFormat](#) pixelFormat, bool interpolate)
Creates a new [RasterImage](#) instance copying the specified pixel data.
- void [ClearPNGCache](#) ()
Disposes the [PNGStream](#). Also useful if is is necessary to regenerate it, e.g. because the underlying image pixel data has changed.
- void [Dispose](#) ()

Properties

- IntPtr [ImageDataAddress](#) [get]
The memory address of the image pixel data.
- IDisposable [DataHolder](#) [get]
An IDisposable that will be disposed when the image is disposed.
- string [Id](#) [get]
A univocal identifier for this image.
- bool [HasAlpha](#) [get]
Determines whether the image has an alpha channel.
- int [Width](#) [get]
The width in pixels of the image.
- int [Height](#) [get]
The height in pixels of the image.
- bool [Interpolate](#) [get]
Determines whether the image should be interpolated when it is resized.
- MemoryStream [PNGStream](#) [get]
Contains a representation of the image in PNG format. Generated at the first access and cached until the image is disposed.

6.32.1 Detailed Description

Represents a raster image, created from raw pixel data. Consider using the derived classes included in the NuGet package "VectSharp.MuPDFUtils" if you need to load a raster image from a file or a Stream.

Definition at line 98 of file RasterImage.cs.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 RasterImage() [1/3]

```
VectSharp.RasterImage.RasterImage (
    IntPtr pixelData,
    int width,
    int height,
    bool hasAlpha,
    bool interpolate )
```

Creates a new [RasterImage](#) instance from the specified pixel data in RGB or RGBA format.

Parameters

<i>pixelData</i>	The address of the image pixel data in RGB or RGBA format.
<i>width</i>	The width in pixels of the image.
<i>height</i>	The height in pixels of the image.
<i>hasAlpha</i>	true if the image is in RGBA format, false if it is in RGB format.
<i>interpolate</i>	Whether the image should be interpolated when it is resized.

Definition at line 170 of file RasterImage.cs.

6.32.2.2 RasterImage() [2/3]

```
VectSharp.RasterImage.RasterImage (
    ref DisposableIntPtr pixelData,
    int width,
    int height,
    bool hasAlpha,
    bool interpolate )
```

Creates a new [RasterImage](#) instance from the specified pixel data in RGB or RGBA format.

Parameters

<i>pixelData</i>	The address of the image pixel data in RGB or RGBA format wrapped in a DisposableIntPtr . The RasterImage will take ownership of this memory.
------------------	---

Parameters

<i>width</i>	The width in pixels of the image.
<i>height</i>	The height in pixels of the image.
<i>hasAlpha</i>	true if the image is in RGBA format, false if it is in RGB format.
<i>interpolate</i>	Whether the image should be interpolated when it is resized.

Definition at line 188 of file RasterImage.cs.

6.32.2.3 RasterImage() [3/3]

```
VectSharp.RasterImage.RasterImage (
    byte[] data,
    int width,
    int height,
    PixelFormats pixelFormat,
    bool interpolate )
```

Creates a new [RasterImage](#) instance copying the specified pixel data.

Parameters

<i>data</i>	The image pixel data that will be copied.
<i>width</i>	The width in pixels of the image.
<i>height</i>	The height in pixels of the image.
<i>pixelFormat</i>	The format of the pixel data.
<i>interpolate</i>	Whether the image should be interpolated when it is resized.

Definition at line 207 of file RasterImage.cs.

6.32.3 Member Function Documentation

6.32.3.1 ClearPNGCache()

```
void VectSharp.RasterImage.ClearPNGCache ( )
```

Disposes the [PNGStream](#). Also useful if it is necessary to regenerate it, e.g. because the underlying image pixel data has changed.

Definition at line 261 of file RasterImage.cs.

6.32.4 Property Documentation

6.32.4.1 DataHolder

```
IDisposable VectSharp.RasterImage.DataHolder [get]
```

An IDisposable that will be disposed when the image is disposed.

Definition at line 108 of file RasterImage.cs.

6.32.4.2 HasAlpha

```
bool VectSharp.RasterImage.HasAlpha [get]
```

Determines whether the image has an alpha channel.

Definition at line 118 of file RasterImage.cs.

6.32.4.3 Height

```
int VectSharp.RasterImage.Height [get]
```

The height in pixels of the image.

Definition at line 128 of file RasterImage.cs.

6.32.4.4 Id

```
string VectSharp.RasterImage.Id [get]
```

A univocal identifier for this image.

Definition at line 113 of file RasterImage.cs.

6.32.4.5 ImageDataAddress

```
IntPtr VectSharp.RasterImage.ImageDataAddress [get]
```

The memory address of the image pixel data.

Definition at line 103 of file RasterImage.cs.

6.32.4.6 Interpolate

```
bool VectSharp.RasterImage.Interpolate [get]
```

Determines whether the image should be interpolated when it is resized.

Definition at line 133 of file RasterImage.cs.

6.32.4.7 PNGStream

```
MemoryStream VectSharp.RasterImage.PNGStream [get]
```

Contains a representation of the image in PNG format. Generated at the first access and cached until the image is disposed.

Definition at line 140 of file RasterImage.cs.

6.32.4.8 Width

```
int VectSharp.RasterImage.Width [get]
```

The width in pixels of the image.

Definition at line 123 of file RasterImage.cs.

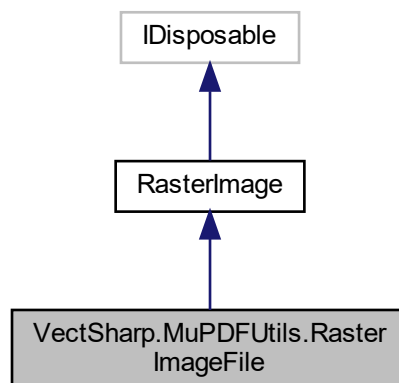
The documentation for this class was generated from the following file:

- VectSharp/RasterImage.cs

6.33 VectSharp.MuPDFUtils.RasterImageFile Class Reference

A [RasterImage](#) created from a file.

Inheritance diagram for VectSharp.MuPDFUtils.RasterImageFile:



Public Member Functions

- [RasterImageFile](#) (string fileName, int pageNumber=0, double scale=1, bool alpha=true, bool interpolate=true)
Creates a new [RasterImage](#) from the specified file.

Additional Inherited Members

6.33.1 Detailed Description

A [RasterImage](#) created from a file.

Definition at line 28 of file RasterImages.cs.

6.33.2 Constructor & Destructor Documentation

6.33.2.1 RasterImageFile()

```
VectSharp.MuPDFUtils.RasterImageFile.RasterImageFile (
    string fileName,
    int pageNumber = 0,
    double scale = 1,
    bool alpha = true,
    bool interpolate = true )
```

Creates a new [RasterImage](#) from the specified file.

Parameters

<i>fileName</i>	The path to the file containing the image.
<i>pageNumber</i>	The number of the page in the file from which the image should be created, starting at 0. Only useful for multi-page formats, such as PDF .
<i>scale</i>	The scale factor at which to render the image.
<i>alpha</i>	A boolean value indicating whether transparency (alpha) data from the image should be preserved or not.
<i>interpolate</i>	A boolean value indicating whether the image should be interpolated when it is resized or not.

Definition at line 38 of file RasterImages.cs.

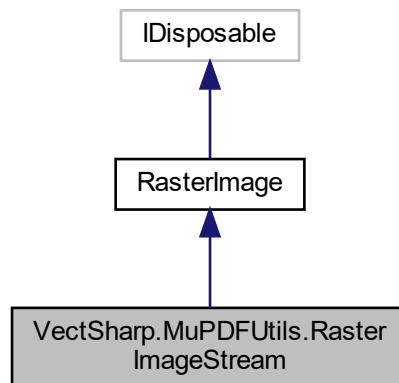
The documentation for this class was generated from the following file:

- VectSharp.MuPDFUtils/RasterImages.cs

6.34 VectSharp.MuPDFUtils.RasterImageStream Class Reference

A [RasterImage](#) created from a stream.

Inheritance diagram for VectSharp.MuPDFUtils.RasterImageStream:



Public Member Functions

- [RasterImageStream](#) (Stream imageStream, InputFileTypes fileType, int pageNumber=0, double scale=1, bool alpha=true, bool interpolate=true)
Creates a new [RasterImage](#) from the specified stream.
- [RasterImageStream](#) (IntPtr imageAddress, long imageLength, InputFileTypes fileType, int pageNumber=0, double scale=1, bool alpha=true, bool interpolate=true)
Creates a new [RasterImage](#) from the specified stream.

Additional Inherited Members

6.34.1 Detailed Description

A [RasterImage](#) created from a stream.

Definition at line 69 of file RasterImages.cs.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 RasterImageStream() [1/2]

```
VectSharp.MuPDFUtils.RasterImageStream.RasterImageStream (
    Stream imageStream,
    InputFileTypes fileType,
    int pageNumber = 0,
    double scale = 1,
    bool alpha = true,
    bool interpolate = true )
```

Creates a new [RasterImage](#) from the specified stream.

Parameters

<i>imageStream</i>	The stream containing the image data.
<i>fileType</i>	The type of the image contained in the stream.
<i>pageNumber</i>	The number of the page in the file from which the image should be created, starting at 0. Only useful for multi-page formats, such as PDF .
<i>scale</i>	The scale factor at which to render the image.
<i>alpha</i>	A boolean value indicating whether transparency (alpha) data from the image should be preserved or not.
<i>interpolate</i>	A boolean value indicating whether the image should be interpolated when it is resized or not.

Definition at line 80 of file RasterImages.cs.

6.34.2.2 RasterImageStream() [2/2]

```
VectSharp.MuPDFUtils.RasterImageStream.RasterImageStream (
    IntPtr imageAddress,
    long imageLength,
    InputFileTypes fileType,
    int pageNumber = 0,
    double scale = 1,
    bool alpha = true,
    bool interpolate = true )
```

Creates a new [RasterImage](#) from the specified stream.

Parameters

<i>imageAddress</i>	A pointer to the address where the image data is contained.
<i>imageLength</i>	The length in bytes of the image data.
<i>fileType</i>	The type of the image contained in the stream.
<i>pageNumber</i>	The number of the page in the file from which the image should be created, starting at 0. Only useful for multi-page formats, such as PDF .
<i>scale</i>	The scale factor at which to render the image.
<i>alpha</i>	A boolean value indicating whether transparency (alpha) data from the image should be preserved or not.
<i>interpolate</i>	A boolean value indicating whether the image should be interpolated when it is resized or not.

Definition at line 148 of file RasterImages.cs.

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

- VectSharp.MuPDFUtils/RasterImages.cs

6.35 VectSharp.Canvas.RenderAction Class Reference

Represents a light-weight rendering action.

Public Types

- enum [ActionTypes](#) { [ActionTypes.Path](#), [ActionTypes.Text](#), [ActionTypes.RasterImage](#) }
Types of rendering actions.

Public Member Functions

- void [BringToFront](#) ()
Brings the render action to the front of the rendering queue. This method can only be invoked after the output has been fully initialised.
- void [SendToBack](#) ()
Brings the render action to the back of the rendering queue. This method can only be invoked after the output has been fully initialised.

Static Public Member Functions

- static [RenderAction PathAction](#) ([Geometry](#) geometry, Pen stroke, IBrush fill, Avalonia.Matrix transform, [Geometry](#) clippingPath, string tag=null)
Creates a new [RenderAction](#) representing a Path.
- static [RenderAction TextAction](#) (FormattedText text, IBrush fill, Avalonia.Matrix transform, [Geometry](#) clippingPath, string tag=null)
Creates a new [RenderAction](#) representing text.
- static [RenderAction ImageAction](#) (string imageId, Avalonia.Rect sourceRect, Avalonia.Rect destinationRect, Avalonia.Matrix transform, [Geometry](#) clippingPath, string tag=null)
Creates a new [RenderAction](#) representing an image.

Properties

- [ActionTypes ActionType](#) [get]
Type of the rendering action.
- [Geometry Geometry](#) [get, set]
Geometry that needs to be rendered (null if the action type is [ActionTypes.Text](#)). If you change this, you need to invalidate the [Parent](#)'s visual.
- [FormattedText Text](#) [get, set]
Text that needs to be rendered (null if the action type is [ActionTypes.Path](#)). If you change this, you need to invalidate the [Parent](#)'s visual.
- [Pen Stroke](#) [get, set]
Rendering stroke (null if the action type is [ActionTypes.Text](#) or if the rendered action only has a [Fill](#)). If you change this, you need to invalidate the [Parent](#)'s visual.
- [IBrush Fill](#) [get, set]
Rendering fill (null if the rendered action only has a [Stroke](#)). If you change this, you need to invalidate the [Parent](#)'s visual.
- string [ImageId](#) [get, set]
Univocal identifier of the image that needs to be drawn.
- Avalonia.Rect [ImageSource](#) [get, set]
The source rectangle of the image.
- Avalonia.Rect [ImageDestination](#) [get, set]
The destination rectangle of the image.
- [Geometry ClippingPath](#) [get, set]
The current clipping path.

- Avalonia.Matrix [InverseTransform](#) = Avalonia.Matrix.Identity [get]
Inverse transformation matrix.
- Avalonia.Matrix [Transform](#) [get, set]
Rendering transformation matrix. If you change this, you need to invalidate the [Parent](#)'s visual.
- string [Tag](#) [get, set]
A tag to access the [RenderAction](#).
- Avalonia.Controls.Canvas [Parent](#) [get]
The container of this [RenderAction](#).

Events

- EventHandler< Avalonia.Input.PointerEventArgs > [PointerEnter](#)
Raised when the pointer enters the area covered by the [RenderAction](#).
- EventHandler< Avalonia.Input.PointerEventArgs > [PointerLeave](#)
Raised when the pointer leaves the area covered by the [RenderAction](#).
- EventHandler< Avalonia.Input.PointerPressedEventArgs > [PointerPressed](#)
Raised when the pointer is pressed while over the area covered by the [RenderAction](#).
- EventHandler< Avalonia.Input.PointerReleasedEventArgs > [PointerReleased](#)
Raised when the pointer is released after a [PointerPressed](#) event.

6.35.1 Detailed Description

Represents a light-weight rendering action.

Definition at line 1013 of file AvaloniaContext.cs.

6.35.2 Member Enumeration Documentation

6.35.2.1 ActionTypes

```
enum VectSharp.Canvas.RenderAction.ActionTypes [strong]
```

Types of rendering actions.

Enumerator

Path	The render action represents a path object.
Text	The render action represents a text object.
RasterImage	The render action represents a raster image.

Definition at line 1018 of file AvaloniaContext.cs.

6.35.3 Member Function Documentation

6.35.3.1 BringToFront()

```
void VectSharp.Canvas.RenderAction.BringToFront ( )
```

Brings the render action to the front of the rendering queue. This method can only be invoked after the output has been fully initialised.

Definition at line 1239 of file AvaloniaContext.cs.

6.35.3.2 ImageAction()

```
static RenderAction VectSharp.Canvas.RenderAction.ImageAction (
    string imageId,
    Avalonia.Rect sourceRect,
    Avalonia.Rect destinationRect,
    Avalonia.Matrix transform,
    Geometry clippingPath,
    string tag = null ) [static]
```

Creates a new [RenderAction](#) representing an image.

Parameters

<i>imageId</i>	The univocal identifier of the image to draw.
<i>sourceRect</i>	The source rectangle of the image.
<i>destinationRect</i>	The destination rectangle of the image.
<i>transform</i>	The transform that will be applied to the image.
<i>clippingPath</i>	The clipping path.
<i>tag</i>	A tag to access the RenderAction . If this is null this RenderAction is not visible in the hit test.

Returns

Definition at line 1222 of file AvaloniaContext.cs.

6.35.3.3 PathAction()

```
static RenderAction VectSharp.Canvas.RenderAction.PathAction (
    Geometry geometry,
    Pen stroke,
```

```

    IBrush fill,
    Avalonia.Matrix transform,
    Geometry clippingPath,
    string tag = null ) [static]

```

Creates a new [RenderAction](#) representing a Path.

Parameters

<i>geometry</i>	The geometry to be rendered.
<i>stroke</i>	The stroke of the path (can be null).
<i>fill</i>	The fill of the path (can be null).
<i>transform</i>	The transform that will be applied to the path.
<i>clippingPath</i>	The clipping path.
<i>tag</i>	A tag to access the RenderAction . If this is null this RenderAction is not visible in the hit test.

Returns

A new [RenderAction](#) representing a Path.

Definition at line 1175 of file AvaloniaContext.cs.

6.35.3.4 SendToBack()

```
void VectSharp.Canvas.RenderAction.SendToBack ( )
```

Brings the render action to the back of the rendering queue. This method can only be invoked after the output has been fully initialised.

Definition at line 1247 of file AvaloniaContext.cs.

6.35.3.5 TextAction()

```

static RenderAction VectSharp.Canvas.RenderAction.TextAction (
    FormattedText text,
    IBrush fill,
    Avalonia.Matrix transform,
    Geometry clippingPath,
    string tag = null ) [static]

```

Creates a new [RenderAction](#) representing text.

Parameters

<i>text</i>	The text to be rendered.
<i>fill</i>	The fill of the text (can be null).
<i>transform</i>	The transform that will be applied to the text.
<i>clippingPath</i>	The clipping path.
<i>tag</i>	A tag to access the RenderAction . If this is null this RenderAction is not visible in the hit test.

Returns

Definition at line 1198 of file AvaloniaContext.cs.

6.35.4 Property Documentation

6.35.4.1 ActionType

`ActionTypes VectSharp.Canvas.RenderAction.ActionType [get]`

Type of the rendering action.

Definition at line 1039 of file AvaloniaContext.cs.

6.35.4.2 ClippingPath

`Geometry VectSharp.Canvas.RenderAction.ClippingPath [get], [set]`

The current clipping path.

Definition at line 1079 of file AvaloniaContext.cs.

6.35.4.3 Fill

`IBrush VectSharp.Canvas.RenderAction.Fill [get], [set]`

Rendering fill (null if the rendered action only has a [Stroke](#)). If you change this, you need to invalidate the [Parent's](#) visual.

Definition at line 1059 of file AvaloniaContext.cs.

6.35.4.4 Geometry

`Geometry VectSharp.Canvas.RenderAction.Geometry [get], [set]`

Geometry that needs to be rendered (null if the action type is [ActionTypes.Text](#)). If you change this, you need to invalidate the [Parent's](#) visual.

Definition at line 1044 of file AvaloniaContext.cs.

6.35.4.5 ImageDestination

```
Avalonia.? Rect VectSharp.Canvas.RenderAction.ImageDestination [get], [set]
```

The destination rectangle of the image.

Definition at line 1074 of file AvaloniaContext.cs.

6.35.4.6 ImageId

```
string VectSharp.Canvas.RenderAction.ImageId [get], [set]
```

Univocal identifier of the image that needs to be drawn.

Definition at line 1064 of file AvaloniaContext.cs.

6.35.4.7 ImageSource

```
Avalonia.? Rect VectSharp.Canvas.RenderAction.ImageSource [get], [set]
```

The source rectangle of the image.

Definition at line 1069 of file AvaloniaContext.cs.

6.35.4.8 InverseTransform

```
Avalonia.Matrix VectSharp.Canvas.RenderAction.InverseTransform = Avalonia.Matrix.Identity  
[get]
```

Inverse transformation matrix.

Definition at line 1086 of file AvaloniaContext.cs.

6.35.4.9 Parent

```
Avalonia.Controls.Canvas VectSharp.Canvas.RenderAction.Parent [get]
```

The container of this [RenderAction](#).

Definition at line 1111 of file AvaloniaContext.cs.

6.35.4.10 Stroke

```
Pen VectSharp.Canvas.RenderAction.Stroke [get], [set]
```

Rendering stroke (null if the action type is [ActionTypes.Text](#) or if the rendered action only has a [Fill](#)). If you change this, you need to invalidate the [Parent](#)'s visual.

Definition at line 1054 of file AvaloniaContext.cs.

6.35.4.11 Tag

```
string VectSharp.Canvas.RenderAction.Tag [get], [set]
```

A tag to access the [RenderAction](#).

Definition at line 1104 of file AvaloniaContext.cs.

6.35.4.12 Text

```
FormattedText VectSharp.Canvas.RenderAction.Text [get], [set]
```

Text that needs to be rendered (null if the action type is [ActionTypes.Path](#)). If you change this, you need to invalidate the [Parent](#)'s visual.

Definition at line 1049 of file AvaloniaContext.cs.

6.35.4.13 Transform

```
Avalonia.Matrix VectSharp.Canvas.RenderAction.Transform [get], [set]
```

Rendering transformation matrix. If you change this, you need to invalidate the [Parent](#)'s visual.

Definition at line 1091 of file AvaloniaContext.cs.

6.35.5 Event Documentation

6.35.5.1 PointerEnter

```
EventHandler<Avalonia.Input.PointerEventArgs> VectSharp.Canvas.RenderAction.PointerEnter
```

Raised when the pointer enters the area covered by the [RenderAction](#).

Definition at line 1122 of file AvaloniaContext.cs.

6.35.5.2 PointerLeave

```
EventHandler<Avalonia.Input.PointerEventArgs> VectSharp.Canvas.RenderAction.PointerLeave
```

Raised when the pointer leaves the area covered by the [RenderAction](#).

Definition at line 1127 of file AvaloniaContext.cs.

6.35.5.3 PointerPressed

```
EventHandler<Avalonia.Input.PointerPressedEventArgs> VectSharp.Canvas.RenderAction.Pointer←  
Pressed
```

Raised when the pointer is pressed while over the area covered by the [RenderAction](#).

Definition at line 1132 of file AvaloniaContext.cs.

6.35.5.4 PointerReleased

```
EventHandler<Avalonia.Input.PointerReleasedEventArgs> VectSharp.Canvas.RenderAction.Pointer←  
Released
```

Raised when the pointer is released after a [PointerPressed](#) event.

Definition at line 1137 of file AvaloniaContext.cs.

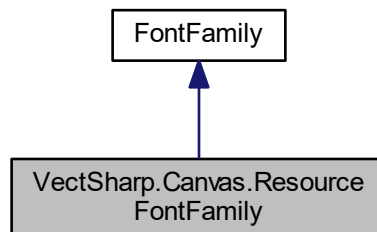
The documentation for this class was generated from the following file:

- VectSharp.Canvas/AvaloniaContext.cs

6.36 VectSharp.Canvas.ResourceFontFamily Class Reference

Represents a [FontFamily](#) created from a resource stream.

Inheritance diagram for VectSharp.Canvas.ResourceFontFamily:



Public Member Functions

- [ResourceFontFamily](#) (System.IO.Stream resourceStream, string resourceName)

Create a new [ResourceFontFamily](#) from the specified *resourceStream* containing a TTF file, passing the specified *resourceName* to the `Avalonia.Media.FontFamily.Parse(string, Uri)` method.

Additional Inherited Members

6.36.1 Detailed Description

Represents a [FontFamily](#) created from a resource stream.

Definition at line 31 of file `AvaloniaContext.cs`.

6.36.2 Constructor & Destructor Documentation

6.36.2.1 ResourceFontFamily()

```
VectSharp.Canvas.ResourceFontFamily.ResourceFontFamily (
    System.IO.Stream resourceStream,
    string resourceName )
```

Create a new [ResourceFontFamily](#) from the specified *resourceStream* containing a TTF file, passing the specified *resourceName* to the `Avalonia.Media.FontFamily.Parse(string, Uri)` method.

Parameters

<i>resourceStream</i>	A resource stream containing a TTF file.
<i>resourceName</i>	The name of the embedded resource, which will be parsed using <code>Avalonia.Media.FontFamily.Parse(string, Uri)</code> .

Definition at line 40 of file `AvaloniaContext.cs`.

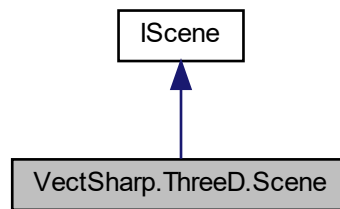
The documentation for this class was generated from the following file:

- `VectSharp.Canvas/AvaloniaContext.cs`

6.37 VectSharp.ThreeD.Scene Class Reference

Represents a 3D scene.

Inheritance diagram for VectSharp.ThreeD.Scene:



Public Member Functions

- [Scene](#) ()
Creates a new [Scene](#).
- void [AddElement](#) (Element3D element)
Adds the specified element to the scene.
- void [AddRange](#) (IEnumerable< Element3D > elements)
Adds the specified elements to the scene.
- void [Replace](#) (Func< Element3D, Element3D > replacementFunction)
Replaces each element in the scene with the element returned by the replacementFunction .
- void [Replace](#) (Func< Element3D, IEnumerable< Element3D >> replacementFunction)
Replaces each element in the scene with the element(s) returned by the replacementFunction .

Public Attributes

- IEnumerable< Element3D > [SceneElements](#) => sceneElements

Properties

- object [SceneLock](#) [get]

6.37.1 Detailed Description

Represents a 3D scene.

Definition at line 49 of file Scene.cs.

6.37.2 Constructor & Destructor Documentation

6.37.2.1 Scene()

VectSharp.ThreeD.Scene.Scene ()

Creates a new [Scene](#).

Definition at line 62 of file Scene.cs.

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

- VectSharp.ThreeD/Scene.cs

6.38 VectSharp.Segment Class Reference

Represents a segment as part of a [GraphicsPath](#).

Public Member Functions

- abstract [Segment Clone](#) ()
Creates a copy of the [Segment](#).
- abstract double [Measure](#) ([Point](#) previousPoint)
Computes the length of the [Segment](#).
- abstract [Point GetPointAt](#) ([Point](#) previousPoint, double position)
Gets the point on the [Segment](#) at the specified (relative) position .
- abstract [Point GetTangentAt](#) ([Point](#) previousPoint, double position)
Gets the tangent to the [Segment](#) at the specified (relative) position .
- abstract IEnumerable< [Segment](#) > [Linearise](#) ([Point?](#) previousPoint, double resolution)
Transform the segment into a series of linear segments. Segments that are already linear are not changed.
- abstract IEnumerable< [Point](#) > [GetLinearisationTangents](#) ([Point?](#) previousPoint, double resolution)
Gets the tangent at the points at which the segment would be linearised.
- abstract IEnumerable< [Segment](#) > [Transform](#) (Func< [Point](#), [Point](#) > transformationFunction)
Applies an arbitrary transformation to all of the points of the [Segment](#).

Properties

- abstract [SegmentType Type](#) [get]
The type of the [Segment](#).
- [Point\[\] Points](#) [get]
The points used to define the [Segment](#).
- virtual [Point Point](#) [get]
The end point of the [Segment](#).

6.38.1 Detailed Description

Represents a segment as part of a [GraphicsPath](#).

Definition at line 1343 of file Graphics.cs.

6.38.2 Member Function Documentation

6.38.2.1 Clone()

```
abstract Segment VectSharp.Segment.Clone ( ) [pure virtual]
```

Creates a copy of the [Segment](#).

Returns

A copy of the [Segment](#).

6.38.2.2 GetLinearisationTangents()

```
abstract IEnumerable<Point> VectSharp.Segment.GetLinearisationTangents (
    Point? previousPoint,
    double resolution ) [pure virtual]
```

Gets the tangent at the points at which the segment would be linearised.

Parameters

<i>previousPoint</i>	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
<i>resolution</i>	The absolute length between successive samples in curve segments.

Returns

A collection of tangents at the points in which the segment would be linearised.

6.38.2.3 GetPointAt()

```
abstract Point VectSharp.Segment.GetPointAt (
    Point previousPoint,
    double position ) [pure virtual]
```

Gets the point on the [Segment](#) at the specified (relative) *position*).

Parameters

<i>previousPoint</i>	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
<i>position</i>	The relative position on the Segment (0 is the start of the Segment , 1 is the end of the Segment).

Returns

The point at the specified position.

6.38.2.4 GetTangentAt()

```
abstract Point VectSharp.Segment.GetTangentAt (
    Point previousPoint,
    double position ) [pure virtual]
```

Gets the tangent to the [Segment](#) at the specified (relative) *position* .

Parameters

<i>previousPoint</i>	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
<i>position</i>	The relative position on the Segment (0 is the start of the Segment , 1 is the end of the Segment).

Returns

The tangent to the point at the specified position.

6.38.2.5 Linearise()

```
abstract IEnumerable<Segment> VectSharp.Segment.Linearise (
    Point? previousPoint,
    double resolution ) [pure virtual]
```

Transform the segment into a series of linear segments. Segments that are already linear are not changed.

Parameters

<i>previousPoint</i>	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
<i>resolution</i>	The absolute length between successive samples in curve segments.

Returns

A collection of linear segments that approximate the current segment.

6.38.2.6 Measure()

```
abstract double VectSharp.Segment.Measure (
    Point previousPoint ) [pure virtual]
```

Computes the length of the [Segment](#).

Parameters

<i>previousPoint</i>	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
----------------------	---

Returns

The length of the segment.

6.38.2.7 Transform()

```
abstract IEnumerable<Segment> VectSharp.Segment.Transform (
    Func< Point, Point > transformationFunction ) [pure virtual]
```

Applies an arbitrary transformation to all of the points of the [Segment](#).

Parameters

<i>transformationFunction</i>	An arbitrary transformation function.
-------------------------------	---------------------------------------

Returns

A collection of [Segments](#) that have been transformed according to the *transformationFunction* .

6.38.3 Property Documentation**6.38.3.1 Point**

```
virtual Point VectSharp.Segment.Point [get]
```

The end point of the [Segment](#).

Definition at line 1359 of file Graphics.cs.

6.38.3.2 Points

```
Point [ ] VectSharp.Segment.Points [get]
```

The points used to define the [Segment](#).

Definition at line 1354 of file Graphics.cs.

6.38.3.3 Type

```
abstract SegmentType VectSharp.Segment.Type [get]
```

The type of the [Segment](#).

Definition at line 1349 of file Graphics.cs.

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

- VectSharp/Graphics.cs

6.39 VectSharp.Size Struct Reference

Represents the size of an object.

Public Member Functions

- [Size](#) (double width, double height)
Create a new [Size](#).

Public Attributes

- double [Width](#)
Width of the object.
- double [Height](#)
Height of the object.

6.39.1 Detailed Description

Represents the size of an object.

Definition at line 1285 of file Graphics.cs.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 Size()

```
VectSharp.Size.Size (  
    double width,  
    double height )
```

Create a new [Size](#).

Parameters

<i>width</i>	The width of the object.
<i>height</i>	The height of the object.

Definition at line 1302 of file Graphics.cs.

6.39.3 Member Data Documentation

6.39.3.1 Height

```
double VectSharp.Size.Height
```

Height of the object.

Definition at line 1295 of file Graphics.cs.

6.39.3.2 Width

```
double VectSharp.Size.Width
```

Width of the object.

Definition at line 1290 of file Graphics.cs.

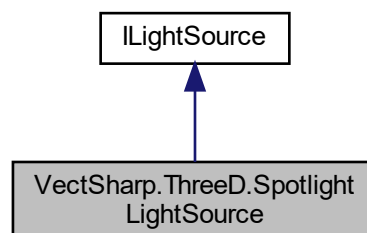
The documentation for this struct was generated from the following file:

- VectSharp/Graphics.cs

6.40 VectSharp.ThreeD.SpotlightLightSource Class Reference

Represents a conic spotlight.

Inheritance diagram for VectSharp.ThreeD.SpotlightLightSource:



Public Member Functions

- [SpotlightLightSource](#) (double intensity, Point3D position, NormalizedVector3D direction, double beamWidthAngle, double cutoffAngle)
Creates a new [SpotlightLightSource](#) instance.
- [LightIntensity GetLightAt](#) (Point3D point)
Computes the light intensity at the specified point, without taking into account any obstructions.
- double [GetObstruction](#) (Point3D point, IEnumerable< Triangle3DElement > shadowingTriangles)
Determines the amount of obstruction of the light that results at point due to the specified shadowingTriangles .

Properties

- bool [CastsShadow](#) = true [get, set]
- Point3D [Position](#) [get, set]
The position of the light source.
- NormalizedVector3D [Direction](#) [get, set]
The direction of the cone axis.
- double [Intensity](#) [get, set]
The base intensity of the light.
- double [BeamWidthAngle](#) [get, set]
The angular size of the light cone, in radians.
- double [CutoffAngle](#) [get, set]
The angular size of the cutoff cone, in radians.
- double [DistanceAttenuationExponent](#) = 2 [get, set]
An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.
- double [AngleAttenuationExponent](#) = 1 [get, set]
An exponent determining how fast the light attenuates between the main light cone and the cutoff cone.

6.40.1 Detailed Description

Represents a conic spotlight.

Definition at line 239 of file Lights.cs.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 SpotlightLightSource()

```
VectSharp.ThreeD.SpotlightLightSource.SpotlightLightSource (
    double intensity,
    Point3D position,
    NormalizedVector3D direction,
    double beamWidthAngle,
    double cutoffAngle )
```

Creates a new [SpotlightLightSource](#) instance.

Parameters

<i>intensity</i>	The intensity of the light.
<i>position</i>	The position of the light source.
<i>direction</i>	The direction of the cone's axis.
<i>beamWidthAngle</i>	The angular size of the light cone, in radians.
<i>cutoffAngle</i>	The angular size of the cutoff cone, in radians.

Definition at line 287 of file Lights.cs.

6.40.3 Property Documentation

6.40.3.1 AngleAttenuationExponent

```
double VectSharp.ThreeD.SpotlightLightSource.AngleAttenuationExponent = 1 [get], [set]
```

An exponent determining how fast the light attenuates between the main light cone and the cutoff cone.

Definition at line 277 of file Lights.cs.

6.40.3.2 BeamWidthAngle

```
double VectSharp.ThreeD.SpotlightLightSource.BeamWidthAngle [get], [set]
```

The angular size of the light cone, in radians.

Definition at line 262 of file Lights.cs.

6.40.3.3 CutoffAngle

```
double VectSharp.ThreeD.SpotlightLightSource.CutoffAngle [get], [set]
```

The angular size of the cutoff cone, in radians.

Definition at line 267 of file Lights.cs.

6.40.3.4 Direction

```
NormalizedVector3D VectSharp.ThreeD.SpotlightLightSource.Direction [get], [set]
```

The direction of the cone axis.

Definition at line 252 of file Lights.cs.

6.40.3.5 DistanceAttenuationExponent

```
double VectSharp.ThreeD.SpotlightLightSource.DistanceAttenuationExponent = 2 [get], [set]
```

An exponent determining how fast the light attenuates with increasing distance. Set to 0 to disable distance attenuation.

Definition at line 272 of file Lights.cs.

6.40.3.6 Intensity

```
double VectSharp.ThreeD.SpotlightLightSource.Intensity [get], [set]
```

The base intensity of the light.

Definition at line 257 of file Lights.cs.

6.40.3.7 Position

```
Point3D VectSharp.ThreeD.SpotlightLightSource.Position [get], [set]
```

The position of the light source.

Definition at line 247 of file Lights.cs.

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

- VectSharp.ThreeD/Lights.cs

6.41 VectSharp.SVG.SVGContextInterpreter Class Reference

Contains methods to render a [Page](#) as an [SVG](#) file.

Public Types

- enum [TextOptions](#) { [TextOptions.EmbedFonts](#), [TextOptions.SubsetFonts](#), [TextOptions.ConvertIntoPaths](#), [TextOptions.DoNotEmbed](#) }

Defines whether the used fonts should be included in the file.

Static Public Member Functions

- static void [SaveAsSVG](#) (this [Page](#) page, string fileName, [TextOptions](#) textOption=[TextOptions.SubsetFonts](#))
Render the page to an [SVG](#) file.
- static void [SaveAsSVG](#) (this [Page](#) page, Stream stream, [TextOptions](#) textOption=[TextOptions.SubsetFonts](#))
Render the page to an [SVG](#) stream.

6.41.1 Detailed Description

Contains methods to render a [Page](#) as an [SVG](#) file.

Definition at line 848 of file SVGContext.cs.

6.41.2 Member Enumeration Documentation

6.41.2.1 TextOptions

```
enum VectSharp.SVG.SVGContextInterpreter.TextOptions [strong]
```

Defines whether the used fonts should be included in the file.

Enumerator

EmbedFonts	Embeds the full font files.
SubsetFonts	Embeds subsetted font files containing only the glyphs for the characters that have been used.
ConvertIntoPaths	Does not embed any font file and converts all text items into paths.
DoNotEmbed	Does not embed any font file, but still encodes text items as such.

Definition at line 868 of file SVGContext.cs.

6.41.3 Member Function Documentation

6.41.3.1 SaveAsSVG() [1/2]

```
static void VectSharp.SVG.SVGContextInterpreter.SaveAsSVG (
    this Page page,
    Stream stream,
    TextOptions textOption = TextOptions.SubsetFont ) [static]
```

Render the page to an SVG stream.

Parameters

<i>page</i>	The Page to render.
<i>stream</i>	The stream to which the SVG data will be written.
<i>textOption</i>	Defines whether the used fonts should be included in the file.

Definition at line 897 of file SVGContext.cs.

6.41.3.2 SaveAsSVG() [2/2]

```
static void VectSharp.SVG.SVGContextInterpreter.SaveAsSVG (
    this Page page,
    string fileName,
    TextOptions textOption = TextOptions.SubsetFont ) [static]
```

Render the page to an SVG file.

Parameters

<i>page</i>	The Page to render.
<i>fileName</i>	The full path to the file to save. If it exists, it will be overwritten.
<i>textOption</i>	Defines whether the used fonts should be included in the file.

Definition at line 857 of file SVGContext.cs.

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

- VectSharp.SVG/SVGContext.cs

6.42 VectSharp.TrueTypeFile Class Reference

Represents a font file in TrueType format. Reference: <http://stevehanov.ca/blog/?id=143>, <https://developer.apple.com/fonts/TrueType-Reference-Manual/>, <https://docs.microsoft.com/en-us/typography/opentype/spec/>

Classes

- struct [Bearings](#)
Represents the left- and right-side bearings of a glyph.
- struct [TrueTypePoint](#)
Represents a point in a TrueType path description.
- struct [VerticalMetrics](#)
Represents the maximum height above and depth below the baseline of a glyph.

Public Member Functions

- void [Destroy](#) ()
Remove this TrueType file from the cache, clear the tables and release the [FontStream](#). Only call this when the actual file that was used to create this object needs to be changed!
- [TrueTypeFile SubsetFont](#) (string charactersToInclude, bool consolidateAt32=false, Dictionary< char, char > outputEncoding=null)
Create a subset of the TrueType file, containing only the glyphs for the specified characters.
- string [GetFontFamilyName](#) ()
Obtains the font family name from the TrueType file.
- string [GetFontName](#) ()
Obtains the PostScript font name from the TrueType file.
- ushort [GetFirstCharIndex](#) ()
Returns the index of the first character glyph represented by the font.
- ushort [GetLastCharIndex](#) ()
Returns the index of the last character glyph represented by the font.
- bool [IsItalic](#) ()
Determines whether the typeface is Italic or Oblique or not.
- bool [IsOblique](#) ()
Determines whether the typeface is Oblique or not.
- bool [IsBold](#) ()
Determines whether the typeface is Bold or not.
- bool [IsFixedPitch](#) ()
Determines whether the typeface is fixed-pitch (aka monospaces) or not.
- bool [IsSerif](#) ()
Determines whether the typeface is serifed or not.
- bool [IsScript](#) ()
Determines whether the typeface is a script typeface or not.
- int [GetGlyphIndex](#) (char glyph)
Determines the index of the glyph corresponding to a certain character.
- [TrueTypePoint](#)[][] [GetGlyphPath](#) (int glyphIndex, double size)
Get the path that describes the shape of a glyph.
- [TrueTypePoint](#)[][] [GetGlyphPath](#) (char glyph, double size)
Get the path that describes the shape of a glyph.
- double [Get1000EmGlyphWidth](#) (char glyph)
Computes the advance width of a glyph, in thousandths of em unit.
- double [Get1000EmGlyphWidth](#) (int glyphIndex)
Computes the advance width of a glyph, in thousandths of em unit.
- double [Get1000EmAscent](#) ()
Computes the font ascent, in thousandths of em unit.
- double [Get1000EmDescent](#) ()

- Computes the font descent, in thousandths of em unit.*

 - double [Get1000EmYMax](#) ()

Computes the maximum height over the baseline of the font, in thousandths of em unit.

 - double [Get1000EmYMin](#) ()

Computes the maximum depth below the baseline of the font, in thousandths of em unit.

 - double [Get1000EmXMax](#) ()

Computes the maximum distance to the right of the glyph origin of the font, in thousandths of em unit.

 - double [Get1000EmXMin](#) ()

Computes the maximum distance to the left of the glyph origin of the font, in thousandths of em unit.

 - [Bearings Get1000EmGlyphBearings](#) (char glyph)

Computes the left- and right- side bearings of a glyph, in thousandths of em unit.

 - [VerticalMetrics Get1000EmGlyphVerticalMetrics](#) (char glyph)

Computes the vertical metrics of a glyph, in thousandths of em unit.

Properties

- Stream [FontStream](#) [get]
- A stream pointing to the TrueType file source (either on disk or in memory). Never dispose this stream directly; if you really need to, call [Destroy](#) instead.*

6.42.1 Detailed Description

Represents a font file in TrueType format. Reference: <http://stevehanov.ca/blog/?id=143>, <https://developer.apple.com/fonts/TrueType-Reference-Manual/>, <https://docs.microsoft.com/en-us/typography/opentype/spec/>

Definition at line 30 of file TrueType.cs.

6.42.2 Member Function Documentation

6.42.2.1 Destroy()

```
void VectSharp.TrueTypeFile.Destroy ( )
```

Remove this TrueType file from the cache, clear the tables and release the [FontStream](#). Only call this when the actual file that was used to create this object needs to be changed!

Definition at line 52 of file TrueType.cs.

6.42.2.2 Get1000EmAscent()

```
double VectSharp.TrueTypeFile.Get1000EmAscent ( )
```

Computes the font ascent, in thousandths of em unit.

Returns

The font ascent in thousandths of em unit.

Definition at line 2061 of file TrueType.cs.

6.42.2.3 Get1000EmDescent()

```
double VectSharp.TrueTypeFile.Get1000EmDescent ( )
```

Computes the font descent, in thousandths of em unit.

Returns

The font descent in thousandths of em unit.

Definition at line 2071 of file TrueType.cs.

6.42.2.4 Get1000EmGlyphBearings()

```
Bearings VectSharp.TrueTypeFile.Get1000EmGlyphBearings (
    char glyph )
```

Computes the left- and right- side bearings of a glyph, in thousandths of em unit.

Parameters

<i>glyph</i>	The glyph whose bearings are to be computed.
--------------	--

Returns

The left- and right- side bearings of the glyph in thousandths of em unit

Definition at line 2153 of file TrueType.cs.

6.42.2.5 Get1000EmGlyphVerticalMetrics()

```
VerticalMetrics VectSharp.TrueTypeFile.Get1000EmGlyphVerticalMetrics (
    char glyph )
```

Computes the vertical metrics of a glyph, in thousandths of em unit.

Parameters

<i>glyph</i>	The glyph whose vertical metrics are to be computed.
--------------	--

Returns

The vertical metrics of a glyph, in thousandths of em unit.

Definition at line 2201 of file TrueType.cs.

6.42.2.6 Get1000EmGlyphWidth() [1/2]

```
double VectSharp.TrueTypeFile.Get1000EmGlyphWidth (
    char glyph )
```

Computes the advance width of a glyph, in thousandths of em unit.

Parameters

<i>glyph</i>	The glyph whose advance width is to be computed.
--------------	--

Returns

The advance width of the glyph in thousandths of em unit.

Definition at line 2032 of file TrueType.cs.

6.42.2.7 Get1000EmGlyphWidth() [2/2]

```
double VectSharp.TrueTypeFile.Get1000EmGlyphWidth (
    int glyphIndex )
```

Computes the advance width of a glyph, in thousandths of em unit.

Parameters

<i>glyphIndex</i>	The index of the glyph whose advance width is to be computed.
-------------------	---

Returns

The advance width of the glyph in thousandths of em unit.

Definition at line 2050 of file TrueType.cs.

6.42.2.8 Get1000EmXMax()

```
double VectSharp.TrueTypeFile.Get1000EmXMax ( )
```

Computes the maximum distance to the right of the glyph origin of the font, in thousandths of em unit.

Returns

The maximum distance to the right of the glyph origin of the font in thousandths of em unit.

Definition at line 2098 of file TrueType.cs.

6.42.2.9 Get1000EmXMin()

```
double VectSharp.TrueTypeFile.Get1000EmXMin ( )
```

Computes the maximum distance to the left of the glyph origin of the font, in thousandths of em unit.

Returns

The maximum distance to the left of the glyph origin of the font in thousandths of em unit.

Definition at line 2107 of file TrueType.cs.

6.42.2.10 Get1000EmYMax()

```
double VectSharp.TrueTypeFile.Get1000EmYMax ( )
```

Computes the maximum height over the baseline of the font, in thousandths of em unit.

Returns

The maximum height over the baseline of the font in thousandths of em unit.

Definition at line 2080 of file TrueType.cs.

6.42.2.11 Get1000EmYMin()

```
double VectSharp.TrueTypeFile.Get1000EmYMin ( )
```

Computes the maximum depth below the baseline of the font, in thousandths of em unit.

Returns

The maximum depth below the baseline of the font in thousandths of em unit.

Definition at line 2089 of file TrueType.cs.

6.42.2.12 GetFirstCharIndex()

```
ushort VectSharp.TrueTypeFile.GetFirstCharIndex ( )
```

Returns the index of the first character glyph represented by the font.

Returns

The index of the first character glyph represented by the font.

Definition at line 1870 of file TrueType.cs.

6.42.2.13 GetFontFamilyName()

```
string VectSharp.TrueTypeFile.GetFontFamilyName ( )
```

Obtains the font family name from the TrueType file.

Returns

The font family name, if available; `null` otherwise.

Definition at line 1823 of file TrueType.cs.

6.42.2.14 GetFontName()

```
string VectSharp.TrueTypeFile.GetFontName ( )
```

Obtains the PostScript font name from the TrueType file.

Returns

The PostScript font name, if available; `null` otherwise.

Definition at line 1851 of file TrueType.cs.

6.42.2.15 GetGlyphIndex()

```
int VectSharp.TrueTypeFile.GetGlyphIndex (
    char glyph )
```

Determines the index of the glyph corresponding to a certain character.

Parameters

<i>glyph</i>	The character whose glyph is sought.
--------------	--------------------------------------

Returns

The index of the glyph in the TrueType file.

Definition at line 1960 of file TrueType.cs.

6.42.2.16 GetGlyphPath() [1/2]

```
TrueTypePoint [][] VectSharp.TrueTypeFile.GetGlyphPath (
    char glyph,
    double size )
```

Get the path that describes the shape of a glyph.

Parameters

<i>glyph</i>	The glyph whose path is sought.
<i>size</i>	The font size to be used for the font coordinates.

Returns

An array of contours, each of which is itself an array of TrueType points.

Definition at line 2022 of file TrueType.cs.

6.42.2.17 GetGlyphPath() [2/2]

```
TrueTypePoint [][] VectSharp.TrueTypeFile.GetGlyphPath (
    int glyphIndex,
    double size )
```

Get the path that describes the shape of a glyph.

Parameters

<i>glyphIndex</i>	The index of the glyph whose path is sought.
<i>size</i>	The font size to be used for the font coordinates.

Returns

An array of contours, each of which is itself an array of TrueType points.

Definition at line 2011 of file TrueType.cs.

6.42.2.18 GetLastCharIndex()

```
ushort VectSharp.TrueTypeFile.GetLastCharIndex ( )
```

Returns the index of the last character glyph represented by the font.

Returns

The index of the last character glyph represented by the font.

Definition at line 1881 of file TrueType.cs.

6.42.2.19 IsBold()

```
bool VectSharp.TrueTypeFile.IsBold ( )
```

Determines whether the typeface is Bold or not.

Returns

A bool indicating whether the typeface is Bold or not

Definition at line 1915 of file TrueType.cs.

6.42.2.20 IsFixedPitch()

```
bool VectSharp.TrueTypeFile.IsFixedPitch ( )
```

Determines whether the typeface is fixed-pitch (aka monospaces) or not.

Returns

A bool indicating whether the typeface is fixed-pitch (aka monospaces) or not.

Definition at line 1926 of file TrueType.cs.

6.42.2.21 IsItalic()

```
bool VectSharp.TrueTypeFile.IsItalic ( )
```

Determines whether the typeface is Italic or Oblique or not.

Returns

A bool indicating whether the typeface is Italic or Oblique or not.

Definition at line 1893 of file TrueType.cs.

6.42.2.22 IsOblique()

```
bool VectSharp.TrueTypeFile.IsOblique ( )
```

Determines whether the typeface is Oblique or not.

Returns

A bool indicating whether the typeface is Oblique or not.

Definition at line 1904 of file TrueType.cs.

6.42.2.23 IsScript()

```
bool VectSharp.TrueTypeFile.IsScript ( )
```

Determines whether the typeface is a script typeface or not.

Returns

A bool indicating whether the typeface is a script typeface or not.

Definition at line 1948 of file TrueType.cs.

6.42.2.24 IsSerif()

```
bool VectSharp.TrueTypeFile.IsSerif ( )
```

Determines whether the typeface is serified or not.

Returns

A bool indicating whether the typeface is serified or not.

Definition at line 1937 of file TrueType.cs.

6.42.2.25 SubsetFont()

```
TrueTypeFile VectSharp.TrueTypeFile.SubsetFont (
    string charactersToInclude,
    bool consolidateAt32 = false,
    Dictionary< char, char > outputEncoding = null )
```

Create a subset of the TrueType file, containing only the glyphs for the specified characters.

Parameters

<i>charactersToInclude</i>	A string containing the characters for which the glyphs should be included.
<i>consolidateAt32</i>	If true, the character map is rearranged so that the included glyphs start at the unicode U+0032 control point.
<i>outputEncoding</i>	If <i>consolidateAt32</i> is true, entries will be added to this dictionary mapping the original characters to the new map (that starts at U+0033).

Returns

Definition at line 544 of file TrueType.cs.

6.42.3 Property Documentation

6.42.3.1 FontStream

```
Stream VectSharp.TrueTypeFile.FontStream [get]
```

A stream pointing to the TrueType file source (either on disk or in memory). Never dispose this stream directly; if you really need to, call [Destroy](#) instead.

Definition at line 46 of file TrueType.cs.

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

- VectSharp/TrueType.cs

6.43 VectSharp.TrueTypeFile.TrueTypePoint Struct Reference

Represents a point in a TrueType path description.

Public Attributes

- double [X](#)
The horizontal coordinate of the point.
- double [Y](#)
The vertical coordinate of the point.
- bool [IsOnCurve](#)
Whether the point is a point on the curve, or a control point of a quadratic Bezier curve.

6.43.1 Detailed Description

Represents a point in a TrueType path description.

Definition at line 1337 of file TrueType.cs.

6.43.2 Member Data Documentation

6.43.2.1 IsOnCurve

```
bool VectSharp.TrueTypeFile.TrueTypePoint.IsOnCurve
```

Whether the point is a point on the curve, or a control point of a quadratic Bezier curve.

Definition at line 1352 of file TrueType.cs.

6.43.2.2 X

```
double VectSharp.TrueTypeFile.TrueTypePoint.X
```

The horizontal coordinate of the point.

Definition at line 1342 of file TrueType.cs.

6.43.2.3 Y

```
double VectSharp.TrueTypeFile.TrueTypePoint.Y
```

The vertical coordinate of the point.

Definition at line 1347 of file TrueType.cs.

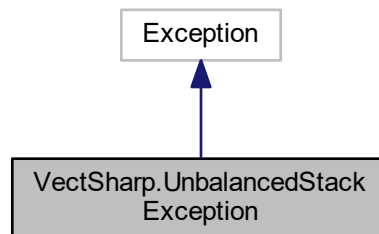
The documentation for this struct was generated from the following file:

- VectSharp/TrueType.cs

6.44 VectSharp.UnbalancedStackException Class Reference

The exception that is thrown when an unbalanced graphics state stack occurs.

Inheritance diagram for VectSharp.UnbalancedStackException:



6.44.1 Detailed Description

The exception that is thrown when an unbalanced graphics state stack occurs.

Definition at line 2313 of file Graphics.cs.

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

- VectSharp/Graphics.cs

6.45 VectSharp.TrueTypeFile.VerticalMetrics Struct Reference

Represents the maximum height above and depth below the baseline of a glyph.

Public Attributes

- int [YMin](#)
The maximum depth below the baseline of the glyph.
- int [YMax](#)
The maximum height above the baseline of the glyph.

6.45.1 Detailed Description

Represents the maximum height above and depth below the baseline of a glyph.

Definition at line 2170 of file TrueType.cs.

6.45.2 Member Data Documentation

6.45.2.1 YMax

```
int VectSharp.TrueTypeFile.VerticalMetrics.YMax
```

The maximum height above the baseline of the glyph.

Definition at line 2180 of file TrueType.cs.

6.45.2.2 YMin

```
int VectSharp.TrueTypeFile.VerticalMetrics.YMin
```

The maximum depth below the baseline of the glyph.

Definition at line 2175 of file TrueType.cs.

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

- VectSharp/TrueType.cs

Index

A

- VectSharp.Colour, [39](#)
- ActionType
 - VectSharp.Canvas.RenderAction, [186](#)
- ActionTypes
 - VectSharp.Canvas.RenderAction, [183](#)
- AddElement
 - VectSharp.ThreeD.IScene, [139](#)
- AddRange
 - VectSharp.ThreeD.IScene, [139](#)
- AddSmoothSpline
 - VectSharp.GraphicsPath, [111](#)
- AddText
 - VectSharp.GraphicsPath, [112](#)
- AddTextOnPath
 - VectSharp.GraphicsPath, [113](#)
- AliceBlue
 - VectSharp.Colours, [49](#)
- AlwaysConvert
 - VectSharp.Canvas.AvaloniaContextInterpreter, [25](#)
- AmbientLightSource
 - VectSharp.ThreeD.AmbientLightSource, [20](#)
- AmbientReflectionCoefficient
 - VectSharp.ThreeD.PhongMaterial, [166](#)
- AngleAttenuationExponent
 - VectSharp.ThreeD.MaskedLightSource, [147](#)
 - VectSharp.ThreeD.SpotlightLightSource, [199](#)
- AntiqueWhite
 - VectSharp.Colours, [49](#)
- Aqua
 - VectSharp.Colours, [49](#)
- Aquamarine
 - VectSharp.Colours, [49](#)
- Arc
 - VectSharp, [15](#)
 - VectSharp.GraphicsPath, [113](#), [114](#)
- AreaLightSource
 - VectSharp.ThreeD.AreaLightSource, [22](#)
- Ascent
 - VectSharp.Font, [85](#)
- Azure
 - VectSharp.Colours, [49](#)

B

- VectSharp.Colour, [39](#)
- Background
 - VectSharp.Page, [157](#)
- Baseline
 - VectSharp, [16](#)
- BeamWidthAngle

- VectSharp.ThreeD.SpotlightLightSource, [199](#)
- Beige
 - VectSharp.Colours, [49](#)
- Bevel
 - VectSharp, [15](#)
- BGR
 - VectSharp, [15](#)
- BGRA
 - VectSharp, [15](#)
- Bisque
 - VectSharp.Colours, [50](#)
- Black
 - VectSharp.Colours, [50](#)
- BlanchedAlmond
 - VectSharp.Colours, [50](#)
- Blue
 - VectSharp.Colours, [50](#)
- BlueViolet
 - VectSharp.Colours, [50](#)
- Bottom
 - VectSharp, [16](#)
 - VectSharp.Font.DetailedFontMetrics, [79](#)
- BringToFront
 - VectSharp.Canvas.RenderAction, [184](#)
- Brown
 - VectSharp.Colours, [51](#)
- BurlyWood
 - VectSharp.Colours, [51](#)
- Butt
 - VectSharp, [14](#)
- CadetBlue
 - VectSharp.Colours, [51](#)
- CastsShadow
 - VectSharp.ThreeD.ILightSource, [135](#)
- Center
 - VectSharp, [16](#)
 - VectSharp.ThreeD.AreaLightSource, [22](#)
- Chartreuse
 - VectSharp.Colours, [51](#)
- Chocolate
 - VectSharp.Colours, [51](#)
- ClearPNGCache
 - VectSharp.RasterImage, [176](#)
- ClippingPath
 - VectSharp.Canvas.RenderAction, [186](#)
- Clone
 - VectSharp.Segment, [193](#)
- Close
 - VectSharp, [15](#)

- VectSharp.GraphicsPath, [114](#)
- VectSharp.IGraphicsContext, [124](#)
- Colour
 - VectSharp.ThreeD.ColourMaterial, [42](#)
 - VectSharp.ThreeD.PhongMaterial, [166](#)
- ColourMaterial
 - VectSharp.ThreeD.ColourMaterial, [42](#)
- ConvertIfNecessary
 - VectSharp.Canvas.AvaloniaContextInterpreter, [25](#)
- ConvertIntoPaths
 - VectSharp.PDF.PDFContextInterpreter, [164](#)
 - VectSharp.SVG.SVGContextInterpreter, [201](#)
- CopyToIGraphicsContext
 - VectSharp.Graphics, [93](#)
- Coral
 - VectSharp.Colours, [52](#)
- CornflowerBlue
 - VectSharp.Colours, [52](#)
- Cornsilk
 - VectSharp.Colours, [52](#)
- Courier
 - VectSharp.FontFamily, [88](#)
- CourierBold
 - VectSharp.FontFamily, [88](#)
- CourierBoldOblique
 - VectSharp.FontFamily, [88](#)
- CourierOblique
 - VectSharp.FontFamily, [88](#)
- CreateCube
 - VectSharp.ThreeD.ObjectFactory, [149](#)
- CreateCuboid
 - VectSharp.ThreeD.ObjectFactory, [150](#)
- CreatePoints
 - VectSharp.ThreeD.ObjectFactory, [150](#)
- CreatePolygon
 - VectSharp.ThreeD.ObjectFactory, [151](#)
- CreatePrism
 - VectSharp.ThreeD.ObjectFactory, [152](#)
- CreateRectangle
 - VectSharp.ThreeD.ObjectFactory, [152](#), [153](#)
- CreateSphere
 - VectSharp.ThreeD.ObjectFactory, [154](#)
- CreateTetrahedron
 - VectSharp.ThreeD.ObjectFactory, [154](#)
- CreateWireframe
 - VectSharp.ThreeD.ObjectFactory, [155](#)
- Crimson
 - VectSharp.Colours, [52](#)
- Crop
 - VectSharp.Page, [157](#)
- CubicBezier
 - VectSharp, [15](#)
- CubicBezierTo
 - VectSharp.GraphicsPath, [115](#)
 - VectSharp.IGraphicsContext, [125](#)
- CutoffAngle
 - VectSharp.ThreeD.SpotlightLightSource, [199](#)
- Cyan
 - VectSharp.Colours, [52](#)
- DarkBlue
 - VectSharp.Colours, [53](#)
- DarkCyan
 - VectSharp.Colours, [53](#)
- DarkGoldenRod
 - VectSharp.Colours, [53](#)
- DarkGray
 - VectSharp.Colours, [53](#)
- DarkGreen
 - VectSharp.Colours, [53](#)
- DarkGrey
 - VectSharp.Colours, [54](#)
- DarkKhaki
 - VectSharp.Colours, [54](#)
- DarkMagenta
 - VectSharp.Colours, [54](#)
- DarkOliveGreen
 - VectSharp.Colours, [54](#)
- DarkOrange
 - VectSharp.Colours, [54](#)
- DarkOrchid
 - VectSharp.Colours, [55](#)
- DarkRed
 - VectSharp.Colours, [55](#)
- DarkSalmon
 - VectSharp.Colours, [55](#)
- DarkSeaGreen
 - VectSharp.Colours, [55](#)
- DarkSlateBlue
 - VectSharp.Colours, [55](#)
- DarkSlateGray
 - VectSharp.Colours, [56](#)
- DarkSlateGrey
 - VectSharp.Colours, [56](#)
- DarkTurquoise
 - VectSharp.Colours, [56](#)
- DarkViolet
 - VectSharp.Colours, [56](#)
- DataHolder
 - VectSharp.RasterImage, [176](#)
- Deconstruct
 - VectSharp.ThreeD.LightIntensity, [142](#)
- DeepPink
 - VectSharp.Colours, [56](#)
- DeepSkyBlue
 - VectSharp.Colours, [57](#)
- Descent
 - VectSharp.Font, [85](#)
- Destroy
 - VectSharp.TrueTypeFile, [204](#)
- DiffuseReflectionCoefficient
 - VectSharp.ThreeD.PhongMaterial, [166](#)
- DimGray
 - VectSharp.Colours, [57](#)
- DimGrey
 - VectSharp.Colours, [57](#)
- Direction

- VectSharp.ThreeD.AreaLightSource, 22
- VectSharp.ThreeD.LightIntensity, 142
- VectSharp.ThreeD.MaskedLightSource, 147
- VectSharp.ThreeD.ParallelLightSource, 159
- VectSharp.ThreeD.SpotlightLightSource, 199
- DisposableIntPtr
 - VectSharp.DisposableIntPtr, 81
- Distance
 - VectSharp.ThreeD.MaskedLightSource, 147
- DistanceAttenuationExponent
 - VectSharp.ThreeD.AreaLightSource, 23
 - VectSharp.ThreeD.MaskedLightSource, 147
 - VectSharp.ThreeD.PointLightSource, 171
 - VectSharp.ThreeD.SpotlightLightSource, 200
- Document
 - VectSharp.Document, 82
- DodgerBlue
 - VectSharp.Colours, 57
- DoNotEmbed
 - VectSharp.SVG.SVGContextInterpreter, 201
- DrawGraphics
 - VectSharp.Graphics, 94
- DrawRasterImage
 - VectSharp.Graphics, 94, 95, 97
 - VectSharp.IGraphicsContext, 125
- EllipticalArc
 - VectSharp.GraphicsPath, 116
- EmbedFonts
 - VectSharp.SVG.SVGContextInterpreter, 201
- FileName
 - VectSharp.FontFamily, 90
- Fill
 - VectSharp.Canvas.RenderAction, 186
 - VectSharp.IGraphicsContext, 126
- FillPath
 - VectSharp.Graphics, 98
- FillRectangle
 - VectSharp.Graphics, 98, 99
- FillStyle
 - VectSharp.IGraphicsContext, 131
- FillText
 - VectSharp.Graphics, 99
 - VectSharp.IGraphicsContext, 126
- FillTextOnPath
 - VectSharp.Graphics, 100
- FireBrick
 - VectSharp.Colours, 57
- FloralWhite
 - VectSharp.Colours, 58
- Font
 - VectSharp.Font, 83
 - VectSharp.IGraphicsContext, 131
- FontFamily
 - VectSharp.Font, 85
 - VectSharp.FontFamily, 88, 89
- FontSize
 - VectSharp.Font, 85
- FontStream
 - VectSharp.TrueTypeFile, 212
- ForestGreen
 - VectSharp.Colours, 58
- FromCSSString
 - VectSharp.Colour, 31
- FromFile
 - VectSharp.SVG.Parser, 161
- FromHSL
 - VectSharp.Colour, 31
- FromLab
 - VectSharp.Colour, 32
- FromRgb
 - VectSharp.Colour, 32, 33
- FromRgba
 - VectSharp.Colour, 33–36
- FromStream
 - VectSharp.SVG.Parser, 161
- FromString
 - VectSharp.SVG.Parser, 161
- FromXYZ
 - VectSharp.Colour, 36
- Fuchsia
 - VectSharp.Colours, 58
- G
 - VectSharp.Colour, 39
- Gainsboro
 - VectSharp.Colours, 58
- Geometry
 - VectSharp.Canvas.RenderAction, 186
- Get1000EmAscent
 - VectSharp.TrueTypeFile, 204
- Get1000EmDescent
 - VectSharp.TrueTypeFile, 205
- Get1000EmGlyphBearings
 - VectSharp.TrueTypeFile, 205
- Get1000EmGlyphVerticalMetrics
 - VectSharp.TrueTypeFile, 205
- Get1000EmGlyphWidth
 - VectSharp.TrueTypeFile, 206
- Get1000EmXMax
 - VectSharp.TrueTypeFile, 207
- Get1000EmXMin
 - VectSharp.TrueTypeFile, 207
- Get1000EmYMax
 - VectSharp.TrueTypeFile, 207
- Get1000EmYMin
 - VectSharp.TrueTypeFile, 207
- GetColour
 - VectSharp.ThreeD.IMaterial, 137
- GetFirstCharIndex
 - VectSharp.TrueTypeFile, 208
- GetFontFamilyName
 - VectSharp.TrueTypeFile, 208
- GetFontName
 - VectSharp.TrueTypeFile, 208
- GetGlyphIndex
 - VectSharp.TrueTypeFile, 208

- GetGlyphPath
 - VectSharp.TrueTypeFile, [209](#)
- GetLastCharIndex
 - VectSharp.TrueTypeFile, [210](#)
- GetLightAt
 - VectSharp.ThreeD.ILightSource, [134](#)
- GetLinearisationPointsNormals
 - VectSharp.GraphicsPath, [116](#)
- GetLinearisationTangents
 - VectSharp.Segment, [193](#)
- GetNormalAtAbsolute
 - VectSharp.GraphicsPath, [117](#)
- GetNormalAtRelative
 - VectSharp.GraphicsPath, [117](#)
- GetObstruction
 - VectSharp.ThreeD.ILightSource, [135](#)
- GetPointAt
 - VectSharp.Segment, [193](#)
- GetPointAtAbsolute
 - VectSharp.GraphicsPath, [117](#)
- GetPointAtRelative
 - VectSharp.GraphicsPath, [118](#)
- GetPoints
 - VectSharp.GraphicsPath, [118](#)
- GetTangentAt
 - VectSharp.Segment, [194](#)
- GetTangentAtAbsolute
 - VectSharp.GraphicsPath, [118](#)
- GetTangentAtRelative
 - VectSharp.GraphicsPath, [119](#)
- GhostWhite
 - VectSharp.Colours, [58](#)
- Gold
 - VectSharp.Colours, [59](#)
- GoldenRod
 - VectSharp.Colours, [59](#)
- Graphics
 - VectSharp.Page, [157](#)
- Gray
 - VectSharp.Colours, [59](#)
- Green
 - VectSharp.Colours, [59](#)
- GreenYellow
 - VectSharp.Colours, [59](#)
- Grey
 - VectSharp.Colours, [60](#)
- H
 - VectSharp.Colour, [40](#)
- HasAlpha
 - VectSharp.RasterImage, [177](#)
- Height
 - VectSharp.Font.DetailedFontMetrics, [79](#)
 - VectSharp.IGraphicsContext, [131](#)
 - VectSharp.Page, [157](#)
 - VectSharp.RasterImage, [177](#)
 - VectSharp.Size, [197](#)
- Helvetica
 - VectSharp.FontFamily, [88](#)
- HelveticaBold
 - VectSharp.FontFamily, [88](#)
- HelveticaBoldOblique
 - VectSharp.FontFamily, [88](#)
- HelveticaOblique
 - VectSharp.FontFamily, [88](#)
- HoneyDew
 - VectSharp.Colours, [60](#)
- HotPink
 - VectSharp.Colours, [60](#)
- Id
 - VectSharp.RasterImage, [177](#)
- Ignore
 - VectSharp, [16](#)
- ImageAction
 - VectSharp.Canvas.RenderAction, [184](#)
- ImageDataAddress
 - VectSharp.RasterImage, [177](#)
- ImageDestination
 - VectSharp.Canvas.RenderAction, [186](#)
- ImageId
 - VectSharp.Canvas.RenderAction, [187](#)
- ImageSource
 - VectSharp.Canvas.RenderAction, [187](#)
- IndianRed
 - VectSharp.Colours, [60](#)
- Indigo
 - VectSharp.Colours, [60](#)
- Intensity
 - VectSharp.ThreeD.AmbientLightSource, [20](#)
 - VectSharp.ThreeD.AreaLightSource, [23](#)
 - VectSharp.ThreeD.LightIntensity, [142](#)
 - VectSharp.ThreeD.MaskedLightSource, [148](#)
 - VectSharp.ThreeD.ParallelLightSource, [159](#)
 - VectSharp.ThreeD.PointLightSource, [171](#)
 - VectSharp.ThreeD.SpotlightLightSource, [200](#)
- InternalPointer
 - VectSharp.DisposableIntPtr, [81](#)
- Interpolate
 - VectSharp.RasterImage, [177](#)
- InverseTransform
 - VectSharp.Canvas.RenderAction, [187](#)
- IsBold
 - VectSharp.FontFamily, [90](#)
 - VectSharp.TrueTypeFile, [210](#)
- IsEqual
 - VectSharp.Point, [168](#)
- IsFixedPitch
 - VectSharp.TrueTypeFile, [210](#)
- IsItalic
 - VectSharp.FontFamily, [90](#)
 - VectSharp.TrueTypeFile, [210](#)
- IsOblique
 - VectSharp.FontFamily, [90](#)
 - VectSharp.TrueTypeFile, [211](#)
- IsOnCurve
 - VectSharp.TrueTypeFile.TrueTypePoint, [213](#)
- IsScript

- VectSharp.TrueTypeFile, 211
- IsSerif
 - VectSharp.TrueTypeFile, 211
- IsStandardFamily
 - VectSharp.FontFamily, 90
- Ivory
 - VectSharp.Colours, 61
- Khaki
 - VectSharp.Colours, 61
- L
 - VectSharp.Colour, 40
- Lavender
 - VectSharp.Colours, 61
- LavenderBlush
 - VectSharp.Colours, 61
- LawnGreen
 - VectSharp.Colours, 61
- Left
 - VectSharp, 16
- LeftSideBearing
 - VectSharp.Font.DetailedFontMetrics, 79
 - VectSharp.TrueTypeFile.Bearings, 28
- LemonChiffon
 - VectSharp.Colours, 62
- LightBlue
 - VectSharp.Colours, 62
- LightCoral
 - VectSharp.Colours, 62
- LightCyan
 - VectSharp.Colours, 62
- LightGoldenRodYellow
 - VectSharp.Colours, 62
- LightGray
 - VectSharp.Colours, 63
- LightGreen
 - VectSharp.Colours, 63
- LightGrey
 - VectSharp.Colours, 63
- LightIntensity
 - VectSharp.ThreeD.LightIntensity, 141
- LightPink
 - VectSharp.Colours, 63
- LightSalmon
 - VectSharp.Colours, 63
- LightSeaGreen
 - VectSharp.Colours, 64
- LightSkyBlue
 - VectSharp.Colours, 64
- LightSlateGray
 - VectSharp.Colours, 64
- LightSlateGrey
 - VectSharp.Colours, 64
- LightSteelBlue
 - VectSharp.Colours, 64
- LightYellow
 - VectSharp.Colours, 65
- Lime
 - VectSharp.Colours, 65
- LimeGreen
 - VectSharp.Colours, 65
- Line
 - VectSharp, 15
- Linearise
 - VectSharp.Graphics, 101
 - VectSharp.GraphicsPath, 119
 - VectSharp.Segment, 194
- LineCap
 - VectSharp.IGraphicsContext, 132
- LineCaps
 - VectSharp, 14
- LineDash
 - VectSharp.LineDash, 143
- LineJoin
 - VectSharp.IGraphicsContext, 132
- LineJoins
 - VectSharp, 14
- Linen
 - VectSharp.Colours, 65
- LineTo
 - VectSharp.GraphicsPath, 120
 - VectSharp.IGraphicsContext, 126
- LineWidth
 - VectSharp.IGraphicsContext, 132
- Magenta
 - VectSharp.Colours, 65
- Maroon
 - VectSharp.Colours, 66
- MaskedLightSource
 - VectSharp.ThreeD.MaskedLightSource, 146
- Measure
 - VectSharp.Segment, 194
- MeasureLength
 - VectSharp.GraphicsPath, 120
- MeasureText
 - VectSharp.Font, 84
 - VectSharp.Graphics, 101
- MeasureTextAdvanced
 - VectSharp.Font, 84
- MediumAquaMarine
 - VectSharp.Colours, 66
- MediumBlue
 - VectSharp.Colours, 66
- MediumOrchid
 - VectSharp.Colours, 66
- MediumPurple
 - VectSharp.Colours, 66
- MediumSeaGreen
 - VectSharp.Colours, 67
- MediumSlateBlue
 - VectSharp.Colours, 67
- MediumSpringGreen
 - VectSharp.Colours, 67
- MediumTurquoise
 - VectSharp.Colours, 67
- MediumVioletRed

- VectSharp.Colours, 67
- Middle
 - VectSharp, 16
- MidnightBlue
 - VectSharp.Colours, 68
- MintCream
 - VectSharp.Colours, 68
- MistyRose
 - VectSharp.Colours, 68
- Miter
 - VectSharp, 15
- Moccasin
 - VectSharp.Colours, 68
- Modulus
 - VectSharp.Point, 169
- Move
 - VectSharp, 15
- MoveTo
 - VectSharp.GraphicsPath, 121
 - VectSharp.IGraphicsContext, 127
- NavajoWhite
 - VectSharp.Colours, 68
- Navy
 - VectSharp.Colours, 69
- NeverConvert
 - VectSharp.Canvas.AvaloniaContextInterpreter, 25
- Normalize
 - VectSharp.Point, 169
- OldLace
 - VectSharp.Colours, 69
- Olive
 - VectSharp.Colours, 69
- OliveDrab
 - VectSharp.Colours, 69
- Orange
 - VectSharp.Colours, 69
- OrangeRed
 - VectSharp.Colours, 70
- Orchid
 - VectSharp.Colours, 70
- Origin
 - VectSharp.ThreeD.MaskedLightSource, 148
- Page
 - VectSharp.Page, 156
- Pages
 - VectSharp.Document, 82
- PaintToCanvas
 - VectSharp.Canvas.AvaloniaContextInterpreter, 25–27
- PaleGoldenRod
 - VectSharp.Colours, 70
- PaleGreen
 - VectSharp.Colours, 70
- PaleTurquoise
 - VectSharp.Colours, 70
- PaleVioletRed
 - VectSharp.Colours, 71
- PapayaWhip
 - VectSharp.Colours, 71
- ParallelLightSource
 - VectSharp.ThreeD.ParallelLightSource, 159
- Parent
 - VectSharp.Canvas.RenderAction, 187
- ParselImageURI
 - VectSharp.SVG.Parser, 162
- Parser
 - VectSharp.MuPDFUtils.ImageURIParser, 136
- ParseSVGURI
 - VectSharp.SVG.Parser, 162
- Path
 - VectSharp.Canvas.RenderAction, 183
- PathAction
 - VectSharp.Canvas.RenderAction, 184
- PeachPuff
 - VectSharp.Colours, 71
- PenumbraAttenuationExponent
 - VectSharp.ThreeD.AreaLightSource, 23
- PenumbraRadius
 - VectSharp.ThreeD.AreaLightSource, 23
- Peru
 - VectSharp.Colours, 71
- Phase
 - VectSharp.LineDash, 144
- PhongMaterial
 - VectSharp.ThreeD.PhongMaterial, 166
- Pink
 - VectSharp.Colours, 71
- PixelFormats
 - VectSharp, 15
- Plum
 - VectSharp.Colours, 72
- PNGStream
 - VectSharp.RasterImage, 178
- Point
 - VectSharp.Point, 168
 - VectSharp.Segment, 195
- PointerEnter
 - VectSharp.Canvas.RenderAction, 188
- PointerLeave
 - VectSharp.Canvas.RenderAction, 188
- PointerPressed
 - VectSharp.Canvas.RenderAction, 189
- PointerReleased
 - VectSharp.Canvas.RenderAction, 189
- PointLightSource
 - VectSharp.ThreeD.PointLightSource, 171
- Points
 - VectSharp.Segment, 195
- Position
 - VectSharp.ThreeD.MaskedLightSource, 148
 - VectSharp.ThreeD.PointLightSource, 172
 - VectSharp.ThreeD.SpotlightLightSource, 200
- PowderBlue
 - VectSharp.Colours, 72

- Purple
 - VectSharp.Colours, 72
- R
 - VectSharp.Colour, 40
- Radius
 - VectSharp.ThreeD.AreaLightSource, 23
- RasterImage
 - VectSharp.Canvas.RenderAction, 183
 - VectSharp.RasterImage, 175, 176
- RasterImageFile
 - VectSharp.MuPDFUtils.RasterImageFile, 179
- RasterImageStream
 - VectSharp.MuPDFUtils.RasterImageStream, 180, 181
- RebeccaPurple
 - VectSharp.Colours, 72
- Rectangle
 - VectSharp.IGraphicsContext, 127
- Red
 - VectSharp.Colours, 72
- Replace
 - VectSharp.ThreeD.IScene, 139, 140
- ResourceFontFamily
 - VectSharp.Canvas.ResourceFontFamily, 190
- Restore
 - VectSharp.Graphics, 101
 - VectSharp.IGraphicsContext, 127
- ReverseDirection
 - VectSharp.ThreeD.ParallelLightSource, 160
- RGB
 - VectSharp, 15
- RGBA
 - VectSharp, 15
- Right
 - VectSharp, 16
- RightSideBearing
 - VectSharp.Font.DetailedFontMetrics, 79
 - VectSharp.TrueTypeFile.Bearings, 28
- RosyBrown
 - VectSharp.Colours, 73
- Rotate
 - VectSharp.Graphics, 102
 - VectSharp.IGraphicsContext, 127
- RotateAt
 - VectSharp.Graphics, 102
- Round
 - VectSharp, 14, 15
- RoyalBlue
 - VectSharp.Colours, 73
- SaddleBrown
 - VectSharp.Colours, 73
- Salmon
 - VectSharp.Colours, 73
- SandyBrown
 - VectSharp.Colours, 73
- Save
 - VectSharp.Graphics, 102
 - VectSharp.IGraphicsContext, 128
- SaveAsPDF
 - VectSharp.PDF.PDFContextInterpreter, 164
- SaveAsPNG
 - VectSharp.Raster.Raster, 172, 173
- SaveAsSVG
 - VectSharp.SVG.SVGContextInterpreter, 201, 202
- Scale
 - VectSharp.Graphics, 102
 - VectSharp.IGraphicsContext, 128
- Scene
 - VectSharp.ThreeD.Scene, 191
- SceneElements
 - VectSharp.ThreeD.IScene, 140
- SceneLock
 - VectSharp.ThreeD.IScene, 140
- SeaGreen
 - VectSharp.Colours, 74
- SeaShell
 - VectSharp.Colours, 74
- Segments
 - VectSharp.GraphicsPath, 122
- SegmentType
 - VectSharp, 15
- SendToBack
 - VectSharp.Canvas.RenderAction, 185
- SetClippingPath
 - VectSharp.Graphics, 103, 104
 - VectSharp.IGraphicsContext, 128
- SetFillStyle
 - VectSharp.IGraphicsContext, 128, 129
- SetLineDash
 - VectSharp.IGraphicsContext, 129
- SetStrokeStyle
 - VectSharp.IGraphicsContext, 129
- ShadowSamplingPointCount
 - VectSharp.ThreeD.AreaLightSource, 24
- Sienna
 - VectSharp.Colours, 74
- SilentlyFix
 - VectSharp, 16
- Silver
 - VectSharp.Colours, 74
- Size
 - VectSharp.Size, 196
- SkyBlue
 - VectSharp.Colours, 74
- SlateBlue
 - VectSharp.Colours, 75
- SlateGray
 - VectSharp.Colours, 75
- SlateGrey
 - VectSharp.Colours, 75
- Snow
 - VectSharp.Colours, 75
- SolidLine
 - VectSharp.LineDash, 144
- SourceDistance

- VectSharp.ThreeD.AreaLightSource, 24
- SpecularReflectionCoefficient
 - VectSharp.ThreeD.PhongMaterial, 167
- SpecularShininess
 - VectSharp.ThreeD.PhongMaterial, 167
- SpotlightLightSource
 - VectSharp.ThreeD.SpotlightLightSource, 198
- SpringGreen
 - VectSharp.Colours, 75
- Square
 - VectSharp, 14
- StandardFamilies
 - VectSharp.FontFamily, 89
- StandardFontFamilies
 - VectSharp.FontFamily, 87
- StandardFontFamilyResources
 - VectSharp.FontFamily, 89
- SteelBlue
 - VectSharp.Colours, 76
- Stroke
 - VectSharp.Canvas.RenderAction, 187
 - VectSharp.IGraphicsContext, 130
- StrokePath
 - VectSharp.Graphics, 104
- StrokeRectangle
 - VectSharp.Graphics, 104, 105
- StrokeStyle
 - VectSharp.IGraphicsContext, 132
- StrokeText
 - VectSharp.Graphics, 106
 - VectSharp.IGraphicsContext, 130
- StrokeTextOnPath
 - VectSharp.Graphics, 107
- SubsetFont
 - VectSharp.TrueTypeFile, 211
- SubsetFonts
 - VectSharp.PDF.PDFContextInterpreter, 164
 - VectSharp.SVG.SVGContextInterpreter, 201
- Symbol
 - VectSharp.FontFamily, 88
- Tag
 - VectSharp.Canvas.RenderAction, 188
 - VectSharp.IGraphicsContext, 132
- Tan
 - VectSharp.Colours, 76
- Teal
 - VectSharp.Colours, 76
- Text
 - VectSharp.Canvas.RenderAction, 183, 188
- TextAction
 - VectSharp.Canvas.RenderAction, 185
- TextAnchors
 - VectSharp, 16
- TextBaseline
 - VectSharp.IGraphicsContext, 133
- TextBaselines
 - VectSharp, 16
- TextOptions
 - VectSharp.Canvas.AvaloniaContextInterpreter, 25
 - VectSharp.PDF.PDFContextInterpreter, 163
 - VectSharp.SVG.SVGContextInterpreter, 201
- Thistle
 - VectSharp.Colours, 76
- Throw
 - VectSharp, 16
- TimesBold
 - VectSharp.FontFamily, 88
- TimesBoldItalic
 - VectSharp.FontFamily, 88
- TimesItalic
 - VectSharp.FontFamily, 88
- TimesRoman
 - VectSharp.FontFamily, 88
- ToCSSString
 - VectSharp.Colour, 37
- Tomato
 - VectSharp.Colours, 76
- Top
 - VectSharp, 16
 - VectSharp.Font.DetailedFontMetrics, 79
- Transform
 - VectSharp.Canvas.RenderAction, 188
 - VectSharp.Graphics, 108
 - VectSharp.GraphicsPath, 122
 - VectSharp.IGraphicsContext, 130
 - VectSharp.Segment, 195
- Translate
 - VectSharp.Graphics, 109
 - VectSharp.IGraphicsContext, 131
- Triangulate
 - VectSharp.GraphicsPath, 122
- TrueTypeFile
 - VectSharp.FontFamily, 91
- Turquoise
 - VectSharp.Colours, 77
- Type
 - VectSharp.Segment, 195
- UnbalancedStackAction
 - VectSharp.Graphics, 109
- UnbalancedStackActions
 - VectSharp, 16
- UnitsOff
 - VectSharp.LineDash, 144
- UnitsOn
 - VectSharp.LineDash, 144
- VectSharp, 13
 - Arc, 15
 - Baseline, 16
 - Bevel, 15
 - BGR, 15
 - BGRA, 15
 - Bottom, 16
 - Butt, 14
 - Center, 16
 - Close, 15

- CubicBezier, 15
- Ignore, 16
- Left, 16
- Line, 15
- LineCaps, 14
- LineJoins, 14
- Middle, 16
- Miter, 15
- Move, 15
- PixelFormats, 15
- RGB, 15
- RGBA, 15
- Right, 16
- Round, 14, 15
- SegmentType, 15
- SilentlyFix, 16
- Square, 14
- TextAnchors, 16
- TextBaselines, 16
- Throw, 16
- Top, 16
- UnbalancedStackActions, 16
- VectSharp.Canvas, 17
- VectSharp.Canvas.AvaloniaContextInterpreter, 24
 - AlwaysConvert, 25
 - ConvertIfNecessary, 25
 - NeverConvert, 25
 - PaintToCanvas, 25–27
 - TextOptions, 25
- VectSharp.Canvas.RenderAction, 181
 - ActionType, 186
 - ActionTypes, 183
 - BringToFront, 184
 - ClippingPath, 186
 - Fill, 186
 - Geometry, 186
 - ImageAction, 184
 - ImageDestination, 186
 - ImageId, 187
 - ImageSource, 187
 - InverseTransform, 187
 - Parent, 187
 - Path, 183
 - PathAction, 184
 - PointerEnter, 188
 - PointerLeave, 188
 - PointerPressed, 189
 - PointerReleased, 189
 - RasterImage, 183
 - SendToBack, 185
 - Stroke, 187
 - Tag, 188
 - Text, 183, 188
 - TextAction, 185
 - Transform, 188
- VectSharp.Canvas.ResourceFontFamily, 189
 - ResourceFontFamily, 190
- VectSharp.Colour, 29
 - A, 39
 - B, 39
 - FromCSSString, 31
 - FromHSL, 31
 - FromLab, 32
 - FromRgb, 32, 33
 - FromRgba, 33–36
 - FromXYZ, 36
 - G, 39
 - H, 40
 - L, 40
 - R, 40
 - ToCSSString, 37
 - WithAlpha, 37–39
 - X, 40
- VectSharp.Colours, 42
 - AliceBlue, 49
 - AntiqueWhite, 49
 - Aqua, 49
 - Aquamarine, 49
 - Azure, 49
 - Beige, 49
 - Bisque, 50
 - Black, 50
 - BlanchedAlmond, 50
 - Blue, 50
 - BlueViolet, 50
 - Brown, 51
 - BurlyWood, 51
 - CadetBlue, 51
 - Chartreuse, 51
 - Chocolate, 51
 - Coral, 52
 - CornflowerBlue, 52
 - Cornsilk, 52
 - Crimson, 52
 - Cyan, 52
 - DarkBlue, 53
 - DarkCyan, 53
 - DarkGoldenRod, 53
 - DarkGray, 53
 - DarkGreen, 53
 - DarkGrey, 54
 - DarkKhaki, 54
 - DarkMagenta, 54
 - DarkOliveGreen, 54
 - DarkOrange, 54
 - DarkOrchid, 55
 - DarkRed, 55
 - DarkSalmon, 55
 - DarkSeaGreen, 55
 - DarkSlateBlue, 55
 - DarkSlateGray, 56
 - DarkSlateGrey, 56
 - DarkTurquoise, 56
 - DarkViolet, 56
 - DeepPink, 56
 - DeepSkyBlue, 57

- DimGray, [57](#)
- DimGrey, [57](#)
- DodgerBlue, [57](#)
- FireBrick, [57](#)
- FloralWhite, [58](#)
- ForestGreen, [58](#)
- Fuchsia, [58](#)
- Gainsboro, [58](#)
- GhostWhite, [58](#)
- Gold, [59](#)
- GoldenRod, [59](#)
- Gray, [59](#)
- Green, [59](#)
- GreenYellow, [59](#)
- Grey, [60](#)
- HoneyDew, [60](#)
- HotPink, [60](#)
- IndianRed, [60](#)
- Indigo, [60](#)
- Ivory, [61](#)
- Khaki, [61](#)
- Lavender, [61](#)
- LavenderBlush, [61](#)
- LawnGreen, [61](#)
- LemonChiffon, [62](#)
- LightBlue, [62](#)
- LightCoral, [62](#)
- LightCyan, [62](#)
- LightGoldenRodYellow, [62](#)
- LightGray, [63](#)
- LightGreen, [63](#)
- LightGrey, [63](#)
- LightPink, [63](#)
- LightSalmon, [63](#)
- LightSeaGreen, [64](#)
- LightSkyBlue, [64](#)
- LightSlateGray, [64](#)
- LightSlateGrey, [64](#)
- LightSteelBlue, [64](#)
- LightYellow, [65](#)
- Lime, [65](#)
- LimeGreen, [65](#)
- Linen, [65](#)
- Magenta, [65](#)
- Maroon, [66](#)
- MediumAquaMarine, [66](#)
- MediumBlue, [66](#)
- MediumOrchid, [66](#)
- MediumPurple, [66](#)
- MediumSeaGreen, [67](#)
- MediumSlateBlue, [67](#)
- MediumSpringGreen, [67](#)
- MediumTurquoise, [67](#)
- MediumVioletRed, [67](#)
- MidnightBlue, [68](#)
- MintCream, [68](#)
- MistyRose, [68](#)
- Moccasin, [68](#)
- NavajoWhite, [68](#)
- Navy, [69](#)
- OldLace, [69](#)
- Olive, [69](#)
- OliveDrab, [69](#)
- Orange, [69](#)
- OrangeRed, [70](#)
- Orchid, [70](#)
- PaleGoldenRod, [70](#)
- PaleGreen, [70](#)
- PaleTurquoise, [70](#)
- PaleVioletRed, [71](#)
- PapayaWhip, [71](#)
- PeachPuff, [71](#)
- Peru, [71](#)
- Pink, [71](#)
- Plum, [72](#)
- PowderBlue, [72](#)
- Purple, [72](#)
- RebeccaPurple, [72](#)
- Red, [72](#)
- RosyBrown, [73](#)
- RoyalBlue, [73](#)
- SaddleBrown, [73](#)
- Salmon, [73](#)
- SandyBrown, [73](#)
- SeaGreen, [74](#)
- SeaShell, [74](#)
- Sienna, [74](#)
- Silver, [74](#)
- SkyBlue, [74](#)
- SlateBlue, [75](#)
- SlateGray, [75](#)
- SlateGrey, [75](#)
- Snow, [75](#)
- SpringGreen, [75](#)
- SteelBlue, [76](#)
- Tan, [76](#)
- Teal, [76](#)
- Thistle, [76](#)
- Tomato, [76](#)
- Turquoise, [77](#)
- Violet, [77](#)
- Wheat, [77](#)
- White, [77](#)
- WhiteSmoke, [77](#)
- Yellow, [78](#)
- YellowGreen, [78](#)
- VectSharp.DisposableIntPtr, [80](#)
 - DisposableIntPtr, [81](#)
 - IntPtr, [81](#)
- VectSharp.Document, [82](#)
 - Document, [82](#)
 - Pages, [82](#)
- VectSharp.Font, [83](#)
 - Ascent, [85](#)
 - Descent, [85](#)
 - Font, [83](#)

- FontFamily, [85](#)
- FontSize, [85](#)
- MeasureText, [84](#)
- MeasureTextAdvanced, [84](#)
- YMax, [85](#)
- YMin, [86](#)
- VectSharp.Font.DetailedFontMetrics, [78](#)
 - Bottom, [79](#)
 - Height, [79](#)
 - LeftSideBearing, [79](#)
 - RightSideBearing, [79](#)
 - Top, [79](#)
 - Width, [80](#)
- VectSharp.FontFamily, [86](#)
 - Courier, [88](#)
 - CourierBold, [88](#)
 - CourierBoldOblique, [88](#)
 - CourierOblique, [88](#)
 - FileName, [90](#)
 - FontFamily, [88](#), [89](#)
 - Helvetica, [88](#)
 - HelveticaBold, [88](#)
 - HelveticaBoldOblique, [88](#)
 - HelveticaOblique, [88](#)
 - IsBold, [90](#)
 - IsItalic, [90](#)
 - IsOblique, [90](#)
 - IsStandardFamily, [90](#)
 - StandardFamilies, [89](#)
 - StandardFontFamilies, [87](#)
 - StandardFontFamilyResources, [89](#)
 - Symbol, [88](#)
 - TimesBold, [88](#)
 - TimesBoldItalic, [88](#)
 - TimesItalic, [88](#)
 - TimesRoman, [88](#)
 - TrueTypeFile, [91](#)
 - ZapfDingbats, [88](#)
- VectSharp.Graphics, [91](#)
 - CopyToIGraphicsContext, [93](#)
 - DrawGraphics, [94](#)
 - DrawRasterImage, [94](#), [95](#), [97](#)
 - FillPath, [98](#)
 - FillRectangle, [98](#), [99](#)
 - FillText, [99](#)
 - FillTextOnPath, [100](#)
 - Linearise, [101](#)
 - MeasureText, [101](#)
 - Restore, [101](#)
 - Rotate, [102](#)
 - RotateAt, [102](#)
 - Save, [102](#)
 - Scale, [102](#)
 - SetClippingPath, [103](#), [104](#)
 - StrokePath, [104](#)
 - StrokeRectangle, [104](#), [105](#)
 - StrokeText, [106](#)
 - StrokeTextOnPath, [107](#)
 - Transform, [108](#)
 - Translate, [109](#)
 - UnbalancedStackAction, [109](#)
- VectSharp.GraphicsPath, [110](#)
 - AddSmoothSpline, [111](#)
 - AddText, [112](#)
 - AddTextOnPath, [113](#)
 - Arc, [113](#), [114](#)
 - Close, [114](#)
 - CubicBezierTo, [115](#)
 - EllipticalArc, [116](#)
 - GetLinearisationPointsNormals, [116](#)
 - GetNormalAtAbsolute, [117](#)
 - GetNormalAtRelative, [117](#)
 - GetPointAtAbsolute, [117](#)
 - GetPointAtRelative, [118](#)
 - GetPoints, [118](#)
 - GetTangentAtAbsolute, [118](#)
 - GetTangentAtRelative, [119](#)
 - Linearise, [119](#)
 - LineTo, [120](#)
 - MeasureLength, [120](#)
 - MoveTo, [121](#)
 - Segments, [122](#)
 - Transform, [122](#)
 - Triangulate, [122](#)
- VectSharp.IGraphicsContext, [123](#)
 - Close, [124](#)
 - CubicBezierTo, [125](#)
 - DrawRasterImage, [125](#)
 - Fill, [126](#)
 - FillStyle, [131](#)
 - FillText, [126](#)
 - Font, [131](#)
 - Height, [131](#)
 - LineCap, [132](#)
 - LineJoin, [132](#)
 - LineTo, [126](#)
 - LineWidth, [132](#)
 - MoveTo, [127](#)
 - Rectangle, [127](#)
 - Restore, [127](#)
 - Rotate, [127](#)
 - Save, [128](#)
 - Scale, [128](#)
 - SetClippingPath, [128](#)
 - SetFillStyle, [128](#), [129](#)
 - SetLineDash, [129](#)
 - SetStrokeStyle, [129](#)
 - Stroke, [130](#)
 - StrokeStyle, [132](#)
 - StrokeText, [130](#)
 - Tag, [132](#)
 - TextBaseline, [133](#)
 - Transform, [130](#)
 - Translate, [131](#)
 - Width, [133](#)
- VectSharp.LineDash, [143](#)

- LineDash, [143](#)
 - Phase, [144](#)
 - SolidLine, [144](#)
 - UnitsOff, [144](#)
 - UnitsOn, [144](#)
- VectSharp.MuPDFUtils, [17](#)
- VectSharp.MuPDFUtils.ImageURIParser, [136](#)
 - Parser, [136](#)
- VectSharp.MuPDFUtils.RasterImageFile, [178](#)
 - RasterImageFile, [179](#)
- VectSharp.MuPDFUtils.RasterImageStream, [179](#)
 - RasterImageStream, [180](#), [181](#)
- VectSharp.Page, [156](#)
 - Background, [157](#)
 - Crop, [157](#)
 - Graphics, [157](#)
 - Height, [157](#)
 - Page, [156](#)
 - Width, [158](#)
- VectSharp.PDF, [17](#)
- VectSharp.PDF.PDFContextInterpreter, [163](#)
 - ConvertIntoPaths, [164](#)
 - SaveAsPDF, [164](#)
 - SubsetFonts, [164](#)
 - TextOptions, [163](#)
- VectSharp.Point, [167](#)
 - IsEqual, [168](#)
 - Modulus, [169](#)
 - Normalize, [169](#)
 - Point, [168](#)
 - X, [169](#)
 - Y, [170](#)
- VectSharp.Raster, [17](#)
- VectSharp.Raster.Raster, [172](#)
 - SaveAsPNG, [172](#), [173](#)
- VectSharp.RasterImage, [173](#)
 - ClearPNGCache, [176](#)
 - DataHolder, [176](#)
 - HasAlpha, [177](#)
 - Height, [177](#)
 - Id, [177](#)
 - ImageDataAddress, [177](#)
 - Interpolate, [177](#)
 - PNGStream, [178](#)
 - RasterImage, [175](#), [176](#)
 - Width, [178](#)
- VectSharp.Segment, [192](#)
 - Clone, [193](#)
 - GetLinearisationTangents, [193](#)
 - GetPointAt, [193](#)
 - GetTangentAt, [194](#)
 - Linearise, [194](#)
 - Measure, [194](#)
 - Point, [195](#)
 - Points, [195](#)
 - Transform, [195](#)
 - Type, [195](#)
- VectSharp.Size, [196](#)
 - Height, [197](#)
 - Size, [196](#)
 - Width, [197](#)
- VectSharp.SVG, [17](#)
- VectSharp.SVG.Parser, [160](#)
 - FromFile, [161](#)
 - FromStream, [161](#)
 - FromString, [161](#)
 - ParselImageURI, [162](#)
 - ParseSVGURI, [162](#)
- VectSharp.SVG.SVGContextInterpreter, [200](#)
 - ConvertIntoPaths, [201](#)
 - DoNotEmbed, [201](#)
 - EmbedFonts, [201](#)
 - SaveAsSVG, [201](#), [202](#)
 - SubsetFonts, [201](#)
 - TextOptions, [201](#)
- VectSharp.ThreeD, [18](#)
- VectSharp.ThreeD.AmbientLightSource, [19](#)
 - AmbientLightSource, [20](#)
 - Intensity, [20](#)
- VectSharp.ThreeD.AreaLightSource, [21](#)
 - AreaLightSource, [22](#)
 - Center, [22](#)
 - Direction, [22](#)
 - DistanceAttenuationExponent, [23](#)
 - Intensity, [23](#)
 - PenumbraAttenuationExponent, [23](#)
 - PenumbraRadius, [23](#)
 - Radius, [23](#)
 - ShadowSamplingPointCount, [24](#)
 - SourceDistance, [24](#)
- VectSharp.ThreeD.ColourMaterial, [41](#)
 - Colour, [42](#)
 - ColourMaterial, [42](#)
- VectSharp.ThreeD.ILightSource, [133](#)
 - CastsShadow, [135](#)
 - GetLightAt, [134](#)
 - GetObstruction, [135](#)
- VectSharp.ThreeD.IMaterial, [137](#)
 - GetColour, [137](#)
- VectSharp.ThreeD.IScene, [138](#)
 - AddElement, [139](#)
 - AddRange, [139](#)
 - Replace, [139](#), [140](#)
 - SceneElements, [140](#)
 - SceneLock, [140](#)
- VectSharp.ThreeD.LightIntensity, [141](#)
 - Deconstruct, [142](#)
 - Direction, [142](#)
 - Intensity, [142](#)
 - LightIntensity, [141](#)
- VectSharp.ThreeD.MaskedLightSource, [145](#)
 - AngleAttenuationExponent, [147](#)
 - Direction, [147](#)
 - Distance, [147](#)
 - DistanceAttenuationExponent, [147](#)
 - Intensity, [148](#)

- MaskedLightSource, 146
- Origin, 148
- Position, 148
- VectSharp.ThreeD.ObjectFactory, 148
 - CreateCube, 149
 - CreateCuboid, 150
 - CreatePoints, 150
 - CreatePolygon, 151
 - CreatePrism, 152
 - CreateRectangle, 152, 153
 - CreateSphere, 154
 - CreateTetrahedron, 154
 - CreateWireframe, 155
- VectSharp.ThreeD.ParallelLightSource, 158
 - Direction, 159
 - Intensity, 159
 - ParallelLightSource, 159
 - ReverseDirection, 160
- VectSharp.ThreeD.PhongMaterial, 165
 - AmbientReflectionCoefficient, 166
 - Colour, 166
 - DiffuseReflectionCoefficient, 166
 - PhongMaterial, 166
 - SpecularReflectionCoefficient, 167
 - SpecularShininess, 167
- VectSharp.ThreeD.PointLightSource, 170
 - DistanceAttenuationExponent, 171
 - Intensity, 171
 - PointLightSource, 171
 - Position, 172
- VectSharp.ThreeD.Scene, 190
 - Scene, 191
- VectSharp.ThreeD.SpotlightLightSource, 197
 - AngleAttenuationExponent, 199
 - BeamWidthAngle, 199
 - CutoffAngle, 199
 - Direction, 199
 - DistanceAttenuationExponent, 200
 - Intensity, 200
 - Position, 200
 - SpotlightLightSource, 198
- VectSharp.TrueTypeFile, 202
 - Destroy, 204
 - FontStream, 212
 - Get1000EmAscent, 204
 - Get1000EmDescent, 205
 - Get1000EmGlyphBearings, 205
 - Get1000EmGlyphVerticalMetrics, 205
 - Get1000EmGlyphWidth, 206
 - Get1000EmXMax, 207
 - Get1000EmXMin, 207
 - Get1000EmYMax, 207
 - Get1000EmYMin, 207
 - GetFirstCharIndex, 208
 - GetFontFamilyName, 208
 - GetFontName, 208
 - GetGlyphIndex, 208
 - GetGlyphPath, 209
 - GetLastCharIndex, 210
 - IsBold, 210
 - IsFixedPitch, 210
 - IsItalic, 210
 - IsOblique, 211
 - IsScript, 211
 - IsSerif, 211
 - SubsetFont, 211
- VectSharp.TrueTypeFile.Bearings, 28
 - LeftSideBearing, 28
 - RightSideBearing, 28
- VectSharp.TrueTypeFile.TrueTypePoint, 212
 - IsOnCurve, 213
 - X, 213
 - Y, 213
- VectSharp.TrueTypeFile.VerticalMetrics, 214
 - YMax, 215
 - YMin, 215
- VectSharp.UnbalancedStackException, 214
- Violet
 - VectSharp.Colours, 77
- Wheat
 - VectSharp.Colours, 77
- White
 - VectSharp.Colours, 77
- WhiteSmoke
 - VectSharp.Colours, 77
- Width
 - VectSharp.Font.DetailedFontMetrics, 80
 - VectSharp.IGraphicsContext, 133
 - VectSharp.Page, 158
 - VectSharp.RasterImage, 178
 - VectSharp.Size, 197
- WithAlpha
 - VectSharp.Colour, 37–39
- X
 - VectSharp.Colour, 40
 - VectSharp.Point, 169
 - VectSharp.TrueTypeFile.TrueTypePoint, 213
- Y
 - VectSharp.Point, 170
 - VectSharp.TrueTypeFile.TrueTypePoint, 213
- Yellow
 - VectSharp.Colours, 78
- YellowGreen
 - VectSharp.Colours, 78
- YMax
 - VectSharp.Font, 85
 - VectSharp.TrueTypeFile.VerticalMetrics, 215
- YMin
 - VectSharp.Font, 86
 - VectSharp.TrueTypeFile.VerticalMetrics, 215
- ZapfDingbats
 - VectSharp.FontFamily, 88