

VectSharp

1.4.2

Generated by Doxygen 1.8.18



<b>1 VectSharp: a light library for C# vector graphics</b>	<b>1</b>
1.1 Introduction	1
1.2 Installing VectSharp	1
1.3 Usage	1
1.4 Creating new output layers	2
1.5 Compiling VectSharp from source	2
1.5.1 Windows	3
1.5.2 macOS and Linux	3
<b>2 Namespace Index</b>	<b>5</b>
2.1 Packages	5
<b>3 Hierarchical Index</b>	<b>7</b>
3.1 Class Hierarchy	7
<b>4 Class Index</b>	<b>9</b>
4.1 Class List	9
<b>5 Namespace Documentation</b>	<b>11</b>
5.1 VectSharp Namespace Reference	11
5.1.1 Enumeration Type Documentation	12
5.1.1.1 LineCaps	12
5.1.1.2 LineJoins	12
5.1.1.3 SegmentType	13
5.1.1.4 TextAnchors	13
5.1.1.5 TextBaselines	13
5.2 VectSharp.Canvas Namespace Reference	14
5.3 VectSharp.PDF Namespace Reference	14
5.4 VectSharp.Raster Namespace Reference	14
5.5 VectSharp.SVG Namespace Reference	14
<b>6 Class Documentation</b>	<b>15</b>
6.1 VectSharp.Canvas.AvaloniaContextInterpreter Class Reference	15
6.1.1 Detailed Description	15
6.1.2 Member Enumeration Documentation	16
6.1.2.1 TextOptions	16
6.1.3 Member Function Documentation	17
6.1.3.1 PaintToCanvas() [1/4]	17
6.1.3.2 PaintToCanvas() [2/4]	18
6.1.3.3 PaintToCanvas() [3/4]	18
6.1.3.4 PaintToCanvas() [4/4]	19
6.2 VectSharp.TrueTypeFile.Bearings Struct Reference	19
6.2.1 Detailed Description	19
6.2.2 Member Data Documentation	20

6.2.2.1 LeftSideBearing	20
6.2.2.2 RightSideBearing	20
6.3 VectSharp.Colour Struct Reference	20
6.3.1 Detailed Description	22
6.3.2 Member Function Documentation	22
6.3.2.1 FromCSSString()	22
6.3.2.2 FromRgb() [1/3]	22
6.3.2.3 FromRgb() [2/3]	23
6.3.2.4 FromRgb() [3/3]	23
6.3.2.5 FromRgba() [1/6]	24
6.3.2.6 FromRgba() [2/6]	24
6.3.2.7 FromRgba() [3/6]	25
6.3.2.8 FromRgba() [4/6]	25
6.3.2.9 FromRgba() [5/6]	26
6.3.2.10 FromRgba() [6/6]	26
6.3.2.11 ToCSSString()	27
6.3.2.12 WithAlpha() [1/4]	27
6.3.2.13 WithAlpha() [2/4]	27
6.3.2.14 WithAlpha() [3/4]	28
6.3.2.15 WithAlpha() [4/4]	28
6.3.3 Member Data Documentation	29
6.3.3.1 A	29
6.3.3.2 B	29
6.3.3.3 G	29
6.3.3.4 R	29
6.4 VectSharp.Colours Class Reference	30
6.4.1 Detailed Description	36
6.4.2 Member Data Documentation	36
6.4.2.1 AliceBlue	36
6.4.2.2 AntiqueWhite	36
6.4.2.3 Aqua	36
6.4.2.4 Aquamarine	36
6.4.2.5 Azure	37
6.4.2.6 Beige	37
6.4.2.7 Bisque	37
6.4.2.8 Black	37
6.4.2.9 BlanchedAlmond	37
6.4.2.10 Blue	38
6.4.2.11 BlueViolet	38
6.4.2.12 Brown	38
6.4.2.13 BurlyWood	38
6.4.2.14 CadetBlue	38

---

6.4.2.15 Chartreuse . . . . .	39
6.4.2.16 Chocolate . . . . .	39
6.4.2.17 Coral . . . . .	39
6.4.2.18 CornflowerBlue . . . . .	39
6.4.2.19 Cornsilk . . . . .	39
6.4.2.20 Crimson . . . . .	40
6.4.2.21 Cyan . . . . .	40
6.4.2.22 DarkBlue . . . . .	40
6.4.2.23 DarkCyan . . . . .	40
6.4.2.24 DarkGoldenRod . . . . .	40
6.4.2.25 DarkGray . . . . .	41
6.4.2.26 DarkGreen . . . . .	41
6.4.2.27 DarkGrey . . . . .	41
6.4.2.28 DarkKhaki . . . . .	41
6.4.2.29 DarkMagenta . . . . .	41
6.4.2.30 DarkOliveGreen . . . . .	42
6.4.2.31 DarkOrange . . . . .	42
6.4.2.32 DarkOrchid . . . . .	42
6.4.2.33 DarkRed . . . . .	42
6.4.2.34 DarkSalmon . . . . .	42
6.4.2.35 DarkSeaGreen . . . . .	43
6.4.2.36 DarkSlateBlue . . . . .	43
6.4.2.37 DarkSlateGray . . . . .	43
6.4.2.38 DarkSlateGrey . . . . .	43
6.4.2.39 DarkTurquoise . . . . .	43
6.4.2.40 DarkViolet . . . . .	44
6.4.2.41 DeepPink . . . . .	44
6.4.2.42 DeepSkyBlue . . . . .	44
6.4.2.43 DimGray . . . . .	44
6.4.2.44 DimGrey . . . . .	44
6.4.2.45 DodgerBlue . . . . .	45
6.4.2.46 FireBrick . . . . .	45
6.4.2.47 FloralWhite . . . . .	45
6.4.2.48 ForestGreen . . . . .	45
6.4.2.49 Fuchsia . . . . .	45
6.4.2.50 Gainsboro . . . . .	46
6.4.2.51 GhostWhite . . . . .	46
6.4.2.52 Gold . . . . .	46
6.4.2.53 GoldenRod . . . . .	46
6.4.2.54 Gray . . . . .	46
6.4.2.55 Green . . . . .	47
6.4.2.56 GreenYellow . . . . .	47

6.4.2.57 Grey	47
6.4.2.58 HoneyDew	47
6.4.2.59 HotPink	47
6.4.2.60 IndianRed	48
6.4.2.61 Indigo	48
6.4.2.62 Ivory	48
6.4.2.63 Khaki	48
6.4.2.64 Lavender	48
6.4.2.65 LavenderBlush	49
6.4.2.66 LawnGreen	49
6.4.2.67 LemonChiffon	49
6.4.2.68 LightBlue	49
6.4.2.69 LightCoral	49
6.4.2.70 LightCyan	50
6.4.2.71 LightGoldenRodYellow	50
6.4.2.72 LightGray	50
6.4.2.73 LightGreen	50
6.4.2.74 LightGrey	50
6.4.2.75 LightPink	51
6.4.2.76 LightSalmon	51
6.4.2.77 LightSeaGreen	51
6.4.2.78 LightSkyBlue	51
6.4.2.79 LightSlateGray	51
6.4.2.80 LightSlateGrey	52
6.4.2.81 LightSteelBlue	52
6.4.2.82 LightYellow	52
6.4.2.83 Lime	52
6.4.2.84 LimeGreen	52
6.4.2.85 Linen	53
6.4.2.86 Magenta	53
6.4.2.87 Maroon	53
6.4.2.88 MediumAquaMarine	53
6.4.2.89 MediumBlue	53
6.4.2.90 MediumOrchid	54
6.4.2.91 MediumPurple	54
6.4.2.92 MediumSeaGreen	54
6.4.2.93 MediumSlateBlue	54
6.4.2.94 MediumSpringGreen	54
6.4.2.95 MediumTurquoise	55
6.4.2.96 MediumVioletRed	55
6.4.2.97 MidnightBlue	55
6.4.2.98 MintCream	55

6.4.2.99 MistyRose . . . . .	55
6.4.2.100 Moccasin . . . . .	56
6.4.2.101 NavajoWhite . . . . .	56
6.4.2.102 Navy . . . . .	56
6.4.2.103 OldLace . . . . .	56
6.4.2.104 Olive . . . . .	56
6.4.2.105 OliveDrab . . . . .	57
6.4.2.106 Orange . . . . .	57
6.4.2.107 OrangeRed . . . . .	57
6.4.2.108 Orchid . . . . .	57
6.4.2.109 PaleGoldenRod . . . . .	57
6.4.2.110 PaleGreen . . . . .	58
6.4.2.111 PaleTurquoise . . . . .	58
6.4.2.112 PaleVioletRed . . . . .	58
6.4.2.113 PapayaWhip . . . . .	58
6.4.2.114 PeachPuff . . . . .	58
6.4.2.115 Peru . . . . .	59
6.4.2.116 Pink . . . . .	59
6.4.2.117 Plum . . . . .	59
6.4.2.118 PowderBlue . . . . .	59
6.4.2.119 Purple . . . . .	59
6.4.2.120 RebeccaPurple . . . . .	60
6.4.2.121 Red . . . . .	60
6.4.2.122 RosyBrown . . . . .	60
6.4.2.123 RoyalBlue . . . . .	60
6.4.2.124 SaddleBrown . . . . .	60
6.4.2.125 Salmon . . . . .	61
6.4.2.126 SandyBrown . . . . .	61
6.4.2.127 SeaGreen . . . . .	61
6.4.2.128 SeaShell . . . . .	61
6.4.2.129 Sienna . . . . .	61
6.4.2.130 Silver . . . . .	62
6.4.2.131 SkyBlue . . . . .	62
6.4.2.132 SlateBlue . . . . .	62
6.4.2.133 SlateGray . . . . .	62
6.4.2.134 SlateGrey . . . . .	62
6.4.2.135 Snow . . . . .	63
6.4.2.136 SpringGreen . . . . .	63
6.4.2.137 SteelBlue . . . . .	63
6.4.2.138 Tan . . . . .	63
6.4.2.139 Teal . . . . .	63
6.4.2.140 Thistle . . . . .	64

6.4.2.141 Tomato	64
6.4.2.142 Turquoise	64
6.4.2.143 Violet	64
6.4.2.144 Wheat	64
6.4.2.145 White	65
6.4.2.146 WhiteSmoke	65
6.4.2.147 Yellow	65
6.4.2.148 YellowGreen	65
6.5 VectSharp.Font.DetailedFontMetrics Class Reference	65
6.5.1 Detailed Description	66
6.5.2 Property Documentation	66
6.5.2.1 Bottom	66
6.5.2.2 Height	66
6.5.2.3 LeftSideBearing	67
6.5.2.4 RightSideBearing	67
6.5.2.5 Top	67
6.5.2.6 Width	67
6.6 VectSharp.Document Class Reference	67
6.6.1 Detailed Description	68
6.6.2 Constructor & Destructor Documentation	68
6.6.2.1 Document()	68
6.6.3 Member Data Documentation	68
6.6.3.1 Pages	68
6.7 VectSharp.Font Class Reference	69
6.7.1 Detailed Description	69
6.7.2 Constructor & Destructor Documentation	69
6.7.2.1 Font()	69
6.7.3 Member Function Documentation	70
6.7.3.1 MeasureText()	70
6.7.3.2 MeasureTextAdvanced()	70
6.7.4 Property Documentation	71
6.7.4.1 Ascent	71
6.7.4.2 Descent	71
6.7.4.3 FontFamily	71
6.7.4.4 FontSize	71
6.7.4.5 YMax	72
6.7.4.6 YMin	72
6.8 VectSharp.FontFamily Class Reference	72
6.8.1 Detailed Description	73
6.8.2 Member Enumeration Documentation	73
6.8.2.1 StandardFontFamilies	74
6.8.3 Constructor & Destructor Documentation	74



6.8.3.1 <a href="#">FontFamily()</a> [1/3]	74
6.8.3.2 <a href="#">FontFamily()</a> [2/3]	75
6.8.3.3 <a href="#">FontFamily()</a> [3/3]	75
6.8.4 Member Data Documentation	75
6.8.4.1 <a href="#">StandardFamilies</a>	75
6.8.4.2 <a href="#">StandardFontFamilyResources</a>	76
6.8.5 Property Documentation	76
6.8.5.1 <a href="#">FileName</a>	76
6.8.5.2 <a href="#">IsBold</a>	76
6.8.5.3 <a href="#">IsItalic</a>	76
6.8.5.4 <a href="#">IsOblique</a>	77
6.8.5.5 <a href="#">IsStandardFamily</a>	77
6.8.5.6 <a href="#">TrueTypeFile</a>	77
6.9 VectSharp.Graphics Class Reference	77
6.9.1 Detailed Description	79
6.9.2 Member Function Documentation	79
6.9.2.1 <a href="#">CopyToGraphicsContext()</a>	79
6.9.2.2 <a href="#">DrawGraphics()</a> [1/2]	79
6.9.2.3 <a href="#">DrawGraphics()</a> [2/2]	80
6.9.2.4 <a href="#">FillPath()</a>	80
6.9.2.5 <a href="#">FillRectangle()</a> [1/2]	80
6.9.2.6 <a href="#">FillRectangle()</a> [2/2]	81
6.9.2.7 <a href="#">FillText()</a> [1/2]	81
6.9.2.8 <a href="#">FillText()</a> [2/2]	82
6.9.2.9 <a href="#">FillTextOnPath()</a>	82
6.9.2.10 <a href="#">MeasureText()</a>	83
6.9.2.11 <a href="#">Restore()</a>	84
6.9.2.12 <a href="#">Rotate()</a>	84
6.9.2.13 <a href="#">RotateAt()</a>	84
6.9.2.14 <a href="#">Save()</a>	84
6.9.2.15 <a href="#">Scale()</a>	85
6.9.2.16 <a href="#">StrokePath()</a>	85
6.9.2.17 <a href="#">StrokeRectangle()</a> [1/2]	85
6.9.2.18 <a href="#">StrokeRectangle()</a> [2/2]	86
6.9.2.19 <a href="#">StrokeText()</a> [1/2]	87
6.9.2.20 <a href="#">StrokeText()</a> [2/2]	87
6.9.2.21 <a href="#">StrokeTextOnPath()</a>	88
6.9.2.22 <a href="#">Transform()</a>	89
6.9.2.23 <a href="#">Translate()</a> [1/2]	89
6.9.2.24 <a href="#">Translate()</a> [2/2]	89
6.10 VectSharp.GraphicsPath Class Reference	91
6.10.1 Detailed Description	92

6.10.2 Member Function Documentation	92
6.10.2.1 AddSmoothSpline()	92
6.10.2.2 AddText() [1/2]	93
6.10.2.3 AddText() [2/2]	93
6.10.2.4 AddTextOnPath()	94
6.10.2.5 Arc() [1/2]	94
6.10.2.6 Arc() [2/2]	95
6.10.2.7 Close()	95
6.10.2.8 CubicBezierTo() [1/2]	96
6.10.2.9 CubicBezierTo() [2/2]	96
6.10.2.10 EllipticalArc()	97
6.10.2.11 GetNormalAtAbsolute()	97
6.10.2.12 GetNormalAtRelative()	98
6.10.2.13 GetPointAtAbsolute()	98
6.10.2.14 GetPointAtRelative()	98
6.10.2.15 GetTangentAtAbsolute()	99
6.10.2.16 GetTangentAtRelative()	99
6.10.2.17 LineTo() [1/2]	99
6.10.2.18 LineTo() [2/2]	101
6.10.2.19 MeasureLength()	101
6.10.2.20 MoveTo() [1/2]	101
6.10.2.21 MoveTo() [2/2]	102
6.10.3 Property Documentation	102
6.10.3.1 Segments	102
6.11 VectSharp.IGraphicsContext Interface Reference	103
6.11.1 Detailed Description	104
6.11.2 Member Function Documentation	104
6.11.2.1 Close()	104
6.11.2.2 CubicBezierTo()	104
6.11.2.3 Fill()	105
6.11.2.4 FillText()	105
6.11.2.5 LineTo()	105
6.11.2.6 MoveTo()	106
6.11.2.7 Rectangle()	106
6.11.2.8 Restore()	106
6.11.2.9 Rotate()	106
6.11.2.10 Save()	107
6.11.2.11 Scale()	107
6.11.2.12 SetFillStyle() [1/2]	107
6.11.2.13 SetFillStyle() [2/2]	107
6.11.2.14 SetLineDash()	109
6.11.2.15 SetStrokeStyle() [1/2]	109

6.11.2.16 SetStrokeStyle() [2/2]	109
6.11.2.17 Stroke()	110
6.11.2.18 StrokeText()	110
6.11.2.19 Transform()	110
6.11.2.20 Translate()	111
6.11.3 Property Documentation	111
6.11.3.1 FillStyle	111
6.11.3.2 Font	111
6.11.3.3 Height	111
6.11.3.4 LineCap	112
6.11.3.5 LineJoin	112
6.11.3.6 LineWidth	112
6.11.3.7 StrokeStyle	112
6.11.3.8 Tag	112
6.11.3.9 TextBaseline	113
6.11.3.10 Width	113
6.12 VectSharp.LineDash Struct Reference	113
6.12.1 Detailed Description	114
6.12.2 Constructor & Destructor Documentation	114
6.12.2.1 LineDash()	114
6.12.3 Member Data Documentation	114
6.12.3.1 Phase	114
6.12.3.2 SolidLine	114
6.12.3.3 UnitsOff	115
6.12.3.4 UnitsOn	115
6.13 VectSharp.Page Class Reference	115
6.13.1 Detailed Description	115
6.13.2 Constructor & Destructor Documentation	116
6.13.2.1 Page()	116
6.13.3 Member Function Documentation	116
6.13.3.1 Crop()	116
6.13.4 Property Documentation	116
6.13.4.1 Background	116
6.13.4.2 Graphics	117
6.13.4.3 Height	117
6.13.4.4 Width	117
6.14 VectSharp.SVG.Parser Class Reference	117
6.14.1 Detailed Description	118
6.14.2 Member Function Documentation	118
6.14.2.1 FromFile()	118
6.14.2.2 FromStream()	118
6.14.2.3 FromString()	119

6.15 VectSharp.PDF.PDFContextInterpreter Class Reference	119
6.15.1 Detailed Description	119
6.15.2 Member Enumeration Documentation	119
6.15.2.1 TextOptions	119
6.15.3 Member Function Documentation	120
6.15.3.1 SaveAsPDF() [1/2]	120
6.15.3.2 SaveAsPDF() [2/2]	120
6.16 VectSharp.Point Struct Reference	121
6.16.1 Detailed Description	121
6.16.2 Constructor & Destructor Documentation	121
6.16.2.1 Point()	121
6.16.3 Member Function Documentation	122
6.16.3.1 Modulus()	122
6.16.3.2 Normalize()	122
6.16.4 Member Data Documentation	122
6.16.4.1 X	122
6.16.4.2 Y	123
6.17 VectSharp.Raster.RasterContextInterpreter Class Reference	123
6.17.1 Detailed Description	123
6.17.2 Member Function Documentation	123
6.17.2.1 SaveAsPNG() [1/2]	123
6.17.2.2 SaveAsPNG() [2/2]	124
6.18 VectSharp.Canvas.RenderAction Class Reference	124
6.18.1 Detailed Description	125
6.18.2 Member Enumeration Documentation	126
6.18.2.1 ActionTypes	126
6.18.3 Member Function Documentation	126
6.18.3.1 BringToFront()	126
6.18.3.2 PathAction()	126
6.18.3.3 SendToBack()	127
6.18.3.4 TextAction()	127
6.18.4 Property Documentation	127
6.18.4.1 ActionType	128
6.18.4.2 Fill	128
6.18.4.3 Geometry	128
6.18.4.4 InverseTransform	128
6.18.4.5 Parent	128
6.18.4.6 Stroke	129
6.18.4.7 Tag	129
6.18.4.8 Text	129
6.18.4.9 Transform	129
6.18.5 Event Documentation	129

6.18.5.1 PointerEnter . . . . .	129
6.18.5.2 PointerLeave . . . . .	130
6.18.5.3 PointerPressed . . . . .	130
6.18.5.4 PointerReleased . . . . .	130
6.19 VectSharp.Segment Class Reference . . . . .	130
6.19.1 Detailed Description . . . . .	131
6.19.2 Member Function Documentation . . . . .	131
6.19.2.1 Clone() . . . . .	131
6.19.2.2 GetPointAt() . . . . .	131
6.19.2.3 GetTangentAt() . . . . .	132
6.19.2.4 Measure() . . . . .	132
6.19.3 Property Documentation . . . . .	132
6.19.3.1 Point . . . . .	132
6.19.3.2 Points . . . . .	133
6.19.3.3 Type . . . . .	133
6.20 VectSharp.Size Struct Reference . . . . .	133
6.20.1 Detailed Description . . . . .	134
6.20.2 Constructor & Destructor Documentation . . . . .	134
6.20.2.1 Size() . . . . .	134
6.20.3 Member Data Documentation . . . . .	134
6.20.3.1 Height . . . . .	134
6.20.3.2 Width . . . . .	134
6.21 VectSharp.SVG.SVGContextInterpreter Class Reference . . . . .	135
6.21.1 Detailed Description . . . . .	135
6.21.2 Member Enumeration Documentation . . . . .	135
6.21.2.1 TextOptions . . . . .	135
6.21.3 Member Function Documentation . . . . .	136
6.21.3.1 SaveAsSVG() [1/2] . . . . .	136
6.21.3.2 SaveAsSVG() [2/2] . . . . .	136
6.22 VectSharp.TrueTypeFile Class Reference . . . . .	136
6.22.1 Detailed Description . . . . .	138
6.22.2 Member Function Documentation . . . . .	138
6.22.2.1 Destroy() . . . . .	138
6.22.2.2 Get1000EmAscent() . . . . .	139
6.22.2.3 Get1000EmDescent() . . . . .	139
6.22.2.4 Get1000EmGlyphBearings() . . . . .	139
6.22.2.5 Get1000EmGlyphVerticalMetrics() . . . . .	140
6.22.2.6 Get1000EmGlyphWidth() [1/2] . . . . .	140
6.22.2.7 Get1000EmGlyphWidth() [2/2] . . . . .	140
6.22.2.8 Get1000EmXMax() . . . . .	141
6.22.2.9 Get1000EmXMin() . . . . .	141
6.22.2.10 Get1000EmYMax() . . . . .	141

6.22.2.11 Get1000EmYMin()	142
6.22.2.12 GetFirstCharIndex()	142
6.22.2.13 GetFontFamilyName()	142
6.22.2.14 GetFontName()	142
6.22.2.15 GetGlyphIndex()	142
6.22.2.16 GetGlyphPath() [1/2]	143
6.22.2.17 GetGlyphPath() [2/2]	143
6.22.2.18 GetLastCharIndex()	144
6.22.2.19 IsBold()	144
6.22.2.20 IsFixedPitch()	144
6.22.2.21 IsItalic()	145
6.22.2.22 IsOblique()	145
6.22.2.23 IsScript()	145
6.22.2.24 IsSerif()	145
6.22.2.25 SubsetFont()	145
6.22.3 Property Documentation	146
6.22.3.1 FontStream	146
6.23 VectSharp.TrueTypeFile.TrueTypePoint Struct Reference	146
6.23.1 Detailed Description	147
6.23.2 Member Data Documentation	147
6.23.2.1 IsOnCurve	147
6.23.2.2 X	147
6.23.2.3 Y	147
6.24 VectSharp.TrueTypeFile.VerticalMetrics Struct Reference	147
6.24.1 Detailed Description	148
6.24.2 Member Data Documentation	148
6.24.2.1 YMax	148
6.24.2.2 YMin	148
<b>Index</b>	<b>149</b>

# 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** 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.

### 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`.

### 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](#).

## 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 [.NET Core 3.0 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.



## Chapter 2

# Namespace Index

### 2.1 Packages

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

<a href="#">VectSharp</a> . . . . .	11
<a href="#">VectSharp.Canvas</a> . . . . .	14
<a href="#">VectSharp.PDF</a> . . . . .	14
<a href="#">VectSharp.Raster</a> . . . . .	14
<a href="#">VectSharp.SVG</a> . . . . .	14



## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

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

VectSharp.Canvas.AvaloniaContextInterpreter . . . . .	15
VectSharp.TrueTypeFile.Bearings . . . . .	19
VectSharp.Colours . . . . .	30
VectSharp.Font.DetailedFontMetrics . . . . .	65
VectSharp.Document . . . . .	67
VectSharp.Font . . . . .	69
VectSharp.FontFamily . . . . .	72
VectSharp.Graphics . . . . .	77
VectSharp.GraphicsPath . . . . .	91
IEquatable	
VectSharp.Colour . . . . .	20
VectSharp.IGraphicsContext . . . . .	103
VectSharp.LineDash . . . . .	113
VectSharp.Page . . . . .	115
VectSharp.SVG.Parser . . . . .	117
VectSharp.PDF.PDFContextInterpreter . . . . .	119
VectSharp.Point . . . . .	121
VectSharp.Raster.RasterContextInterpreter . . . . .	123
VectSharp.Canvas.RenderAction . . . . .	124
VectSharp.Segment . . . . .	130
VectSharp.Size . . . . .	133
VectSharp.SVG.SVGContextInterpreter . . . . .	135
VectSharp.TrueTypeFile . . . . .	136
VectSharp.TrueTypeFile.TrueTypePoint . . . . .	146
VectSharp.TrueTypeFile.VerticalMetrics . . . . .	147



## Chapter 4

# Class Index

### 4.1 Class List

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

<a href="#">VectSharp.Canvas.AvaloniaContextInterpreter</a>	
Contains methods to render a <a href="#">Page</a> to an Avalonia.Controls.Canvas . . . . .	15
<a href="#">VectSharp.TrueTypeFile.Bearings</a>	
Represents the left- and right-side bearings of a glyph . . . . .	19
<a href="#">VectSharp.Colour</a>	
Represents an RGB colour . . . . .	20
<a href="#">VectSharp.Colours</a>	
Standard colours . . . . .	30
<a href="#">VectSharp.Font.DetailedFontMetrics</a>	
Represents detailed information about the metrics of a text string when drawn with a certain font	65
<a href="#">VectSharp.Document</a>	
Represents a collection of pages . . . . .	67
<a href="#">VectSharp.Font</a>	
Represents a typeface with a specific size . . . . .	69
<a href="#">VectSharp.FontFamily</a>	
Represents a typeface . . . . .	72
<a href="#">VectSharp.Graphics</a>	
Represents an abstract drawing surface . . . . .	77
<a href="#">VectSharp.GraphicsPath</a>	
Represents a graphics path that can be filled or stroked . . . . .	91
<a href="#">VectSharp.IGraphicsContext</a>	
This interface should be implemented by classes intended to provide graphics output capability to a <a href="#">Graphics</a> object . . . . .	103
<a href="#">VectSharp.LineDash</a>	
Represents instructions on how to paint a dashed line . . . . .	113
<a href="#">VectSharp.Page</a>	
Represents a <a href="#">Graphics</a> object with a width and height . . . . .	115
<a href="#">VectSharp.SVG.Parser</a>	
Contains methods to read an <a href="#">SVG</a> image file . . . . .	117
<a href="#">VectSharp.PDF.PDFContextInterpreter</a>	
Contains methods to render a <a href="#">Document</a> as a <a href="#">PDF</a> document . . . . .	119
<a href="#">VectSharp.Point</a>	
Represents a point relative to an origin in the top-left corner . . . . .	121
<a href="#">VectSharp.Raster.RasterContextInterpreter</a>	
Contains methods to render a <a href="#">Page</a> as a raster image . . . . .	123

<a href="#">VectSharp.Canvas.RenderAction</a>	
Represents a light-weight rendering action . . . . .	124
<a href="#">VectSharp.Segment</a>	
Represents a segment as part of a <a href="#">GraphicsPath</a> . . . . .	130
<a href="#">VectSharp.Size</a>	
Represents the size of an object . . . . .	133
<a href="#">VectSharp.SVG.SVGContextInterpreter</a>	
Contains methods to render a <a href="#">Page</a> as an <a href="#">SVG</a> file . . . . .	135
<a href="#">VectSharp.TrueTypeFile</a>	
Represents a font file in TrueType format. Reference: <a href="http://stevehanov.ca/blog/?id=143">http://stevehanov.ca/blog/?id=143</a> , <a href="https://developer.apple.com/fonts/TrueType-Reference-Manual/">https://developer.apple.com/fonts/TrueType-Reference-Manual/</a> , <a href="https://docs.microsoft.com/en-us/typography/opentype/spec/">https://docs.microsoft.com/en-us/typography/opentype/spec/</a>	136
<a href="#">VectSharp.TrueTypeFile.TrueTypePoint</a>	
Represents a point in a TrueType path description . . . . .	146
<a href="#">VectSharp.TrueTypeFile.VerticalMetrics</a>	
Represents the maximum heighth above and depth below the baseline of a glyph . . . . .	147



## Chapter 5

# Namespace Documentation

### 5.1 VectSharp Namespace Reference

#### Classes

- struct [Colour](#)  
*Represents an RGB colour.*
- class [Colours](#)  
*Standard colours.*
- 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 [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/>*

## 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.*

### 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 83 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 104 of file Graphics.cs.

### 5.1.1.3 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 1029 of file Graphics.cs.

### 5.1.1.4 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 62 of file Graphics.cs.

### 5.1.1.5 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 36 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.*

## 5.3 VectSharp.PDF Namespace Reference

### Classes

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

## 5.4 VectSharp.Raster Namespace Reference

### Classes

- class [RasterContextInterpreter](#)  
*Contains methods to render a [Page](#) as a raster image.*

## 5.5 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.*

## Chapter 6

# Class Documentation

### 6.1 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.*

#### 6.1.1 Detailed Description

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

Definition at line 1472 of file AvaloniaContext.cs.

## 6.1.2 Member Enumeration Documentation

### 6.1.2.1 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 1477 of file AvaloniaContext.cs.

### 6.1.3 Member Function Documentation

#### 6.1.3.1 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 <a href="#">Page</a> 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 <a href="#">RenderAction</a> 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 1540 of file AvaloniaContext.cs.

### 6.1.3.2 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 <a href="#">Page</a> 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 1516 of file AvaloniaContext.cs.

### 6.1.3.3 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 <a href="#">Page</a> 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 1563 of file AvaloniaContext.cs.

#### 6.1.3.4 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 <a href="#">Page</a> 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 1501 of file AvaloniaContext.cs.

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

- VectSharp.Canvas/AvaloniaContext.cs

## 6.2 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.*

### 6.2.1 Detailed Description

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

Definition at line 2115 of file TrueType.cs.

## 6.2.2 Member Data Documentation

### 6.2.2.1 LeftSideBearing

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

The left-side bearing of the glyph.

Definition at line 2120 of file TrueType.cs.

### 6.2.2.2 RightSideBearing

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

The right-side bearing of the glyph.

Definition at line 2125 of file TrueType.cs.

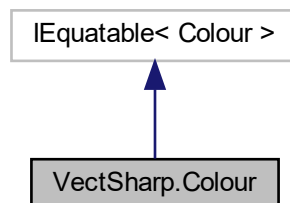
The documentation for this struct was generated from the following file:

- VectSharp/TrueType.cs

## 6.3 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 .

## 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 .

## 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].

### 6.3.1 Detailed Description

Represents an RGB colour.

Definition at line 164 of file Graphics.cs.

### 6.3.2 Member Function Documentation

#### 6.3.2.1 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 364 of file Graphics.cs.

#### 6.3.2.2 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 213 of file Graphics.cs.

**6.3.2.3 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 201 of file Graphics.cs.

**6.3.2.4 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 225 of file Graphics.cs.

### 6.3.2.5 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 299 of file Graphics.cs.

### 6.3.2.6 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 251 of file Graphics.cs.

### 6.3.2.7 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 264 of file Graphics.cs.

### 6.3.2.8 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 238 of file Graphics.cs.

### 6.3.2.9 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 289 of file Graphics.cs.

### 6.3.2.10 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 276 of file Graphics.cs.



### 6.3.2.11 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 347 of file Graphics.cs.

### 6.3.2.12 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 <a href="#">Colour</a> .
--------------	---

#### Returns

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

Definition at line 500 of file Graphics.cs.

### 6.3.2.13 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 <a href="#">Colour</a> from which the RGB components will be taken.
<i>alpha</i>	The alpha component of the new <a href="#">Colour</a> .

## Returns

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

Definition at line 480 of file Graphics.cs.

**6.3.2.14 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 <a href="#">Colour</a> from which the RGB components will be taken.
<i>alpha</i>	The alpha component of the new <a href="#">Colour</a> .

## Returns

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

Definition at line 469 of file Graphics.cs.

**6.3.2.15 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 <a href="#">Colour</a> .
--------------	---

## Returns

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

Definition at line 490 of file Graphics.cs.

### 6.3.3 Member Data Documentation

#### 6.3.3.1 A

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

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

Definition at line 184 of file Graphics.cs.

#### 6.3.3.2 B

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

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

Definition at line 179 of file Graphics.cs.

#### 6.3.3.3 G

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

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

Definition at line 174 of file Graphics.cs.

#### 6.3.3.4 R

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

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

Definition at line 169 of file Graphics.cs.

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

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

## 6.4 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*

### 6.4.1 Detailed Description

Standard colours.

Definition at line 182 of file StandardColours.cs.

### 6.4.2 Member Data Documentation

#### 6.4.2.1 AliceBlue

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

AliceBlue #F0F8FF

Definition at line 599 of file StandardColours.cs.

#### 6.4.2.2 AntiqueWhite

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

AntiqueWhite #FAEBD7

Definition at line 639 of file StandardColours.cs.

#### 6.4.2.3 Aqua

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

Aqua #00FFFF

Definition at line 243 of file StandardColours.cs.

#### 6.4.2.4 Aquamarine

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

Aquamarine #7FFFD4

Definition at line 375 of file StandardColours.cs.

#### 6.4.2.5 Azure

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

Azure #F0FFFF

Definition at line 607 of file StandardColours.cs.

#### 6.4.2.6 Beige

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

Beige #F5F5DC

Definition at line 619 of file StandardColours.cs.

#### 6.4.2.7 Bisque

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

Bisque #FFE4C4

Definition at line 723 of file StandardColours.cs.

#### 6.4.2.8 Black

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

Black #000000

Definition at line 187 of file StandardColours.cs.

#### 6.4.2.9 BlanchedAlmond

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

BlanchedAlmond #FFEBCD

Definition at line 731 of file StandardColours.cs.

#### 6.4.2.10 Blue

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

Blue #0000FF

Definition at line 203 of file StandardColours.cs.

#### 6.4.2.11 BlueViolet

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

BlueViolet #8A2BE2

Definition at line 407 of file StandardColours.cs.

#### 6.4.2.12 Brown

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

Brown #A52A2A

Definition at line 455 of file StandardColours.cs.

#### 6.4.2.13 BurlyWood

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

BurlyWood #DEB887

Definition at line 567 of file StandardColours.cs.

#### 6.4.2.14 CadetBlue

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

CadetBlue #5F9EA0

Definition at line 315 of file StandardColours.cs.

#### 6.4.2.15 Chartreuse

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

Chartreuse #7FFF00

Definition at line 371 of file StandardColours.cs.

#### 6.4.2.16 Chocolate

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

Chocolate #D2691E

Definition at line 523 of file StandardColours.cs.

#### 6.4.2.17 Coral

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

Coral #FF7F50

Definition at line 683 of file StandardColours.cs.

#### 6.4.2.18 CornflowerBlue

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

CornflowerBlue #6495ED

Definition at line 319 of file StandardColours.cs.

#### 6.4.2.19 Cornsilk

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

Cornsilk #FFF8DC

Definition at line 747 of file StandardColours.cs.

#### 6.4.2.20 Crimson

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

Crimson #DC143C

Definition at line 555 of file StandardColours.cs.

#### 6.4.2.21 Cyan

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

Cyan #00FFFF

Definition at line 247 of file StandardColours.cs.

#### 6.4.2.22 DarkBlue

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

DarkBlue #00008B

Definition at line 195 of file StandardColours.cs.

#### 6.4.2.23 DarkCyan

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

DarkCyan #008B8B

Definition at line 219 of file StandardColours.cs.

#### 6.4.2.24 DarkGoldenRod

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

DarkGoldenRod #B8860B

Definition at line 491 of file StandardColours.cs.

#### 6.4.2.25 DarkGray

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

DarkGray #A9A9A9

Definition at line 459 of file StandardColours.cs.

#### 6.4.2.26 DarkGreen

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

DarkGreen #006400

Definition at line 207 of file StandardColours.cs.

#### 6.4.2.27 DarkGrey

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

DarkGrey #A9A9A9

Definition at line 463 of file StandardColours.cs.

#### 6.4.2.28 DarkKhaki

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

DarkKhaki #BDB76B

Definition at line 503 of file StandardColours.cs.

#### 6.4.2.29 DarkMagenta

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

DarkMagenta #8B008B

Definition at line 415 of file StandardColours.cs.

#### 6.4.2.30 DarkOliveGreen

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

DarkOliveGreen #556B2F

Definition at line 311 of file StandardColours.cs.

#### 6.4.2.31 DarkOrange

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

DarkOrange #FF8C00

Definition at line 687 of file StandardColours.cs.

#### 6.4.2.32 DarkOrchid

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

DarkOrchid #9932CC

Definition at line 443 of file StandardColours.cs.

#### 6.4.2.33 DarkRed

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

DarkRed #8B0000

Definition at line 411 of file StandardColours.cs.

#### 6.4.2.34 DarkSalmon

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

DarkSalmon #E9967A

Definition at line 579 of file StandardColours.cs.



#### 6.4.2.35 DarkSeaGreen

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

DarkSeaGreen #8FBC8F

Definition at line 423 of file StandardColours.cs.

#### 6.4.2.36 DarkSlateBlue

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

DarkSlateBlue #483D8B

Definition at line 299 of file StandardColours.cs.

#### 6.4.2.37 DarkSlateGray

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

DarkSlateGray #2F4F4F

Definition at line 271 of file StandardColours.cs.

#### 6.4.2.38 DarkSlateGrey

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

DarkSlateGrey #2F4F4F

Definition at line 275 of file StandardColours.cs.

#### 6.4.2.39 DarkTurquoise

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

DarkTurquoise #00CED1

Definition at line 227 of file StandardColours.cs.

#### 6.4.2.40 DarkViolet

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

DarkViolet #9400D3

Definition at line 435 of file StandardColours.cs.

#### 6.4.2.41 DeepPink

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

DeepPink #FF1493

Definition at line 667 of file StandardColours.cs.

#### 6.4.2.42 DeepSkyBlue

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

DeepSkyBlue #00BFFF

Definition at line 223 of file StandardColours.cs.

#### 6.4.2.43 DimGray

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

DimGray #696969

Definition at line 331 of file StandardColours.cs.

#### 6.4.2.44 DimGrey

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

DimGrey #696969

Definition at line 335 of file StandardColours.cs.

#### 6.4.2.45 DodgerBlue

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

DodgerBlue #1E90FF

Definition at line 255 of file StandardColours.cs.

#### 6.4.2.46 FireBrick

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

FireBrick #B22222

Definition at line 487 of file StandardColours.cs.

#### 6.4.2.47 FloralWhite

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

FloralWhite #FFFAF0

Definition at line 755 of file StandardColours.cs.

#### 6.4.2.48 ForestGreen

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

ForestGreen #228B22

Definition at line 263 of file StandardColours.cs.

#### 6.4.2.49 Fuchsia

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

Fuchsia #FF00FF

Definition at line 659 of file StandardColours.cs.

#### 6.4.2.50 Gainsboro

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

Gainsboro #DCDCDC

Definition at line 559 of file StandardColours.cs.

#### 6.4.2.51 GhostWhite

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

GhostWhite #F8F8FF

Definition at line 631 of file StandardColours.cs.

#### 6.4.2.52 Gold

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

Gold #FFD700

Definition at line 707 of file StandardColours.cs.

#### 6.4.2.53 GoldenRod

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

GoldenRod #DAA520

Definition at line 547 of file StandardColours.cs.

#### 6.4.2.54 Gray

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

Gray #808080

Definition at line 391 of file StandardColours.cs.

#### 6.4.2.55 Green

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

Green #008000

Definition at line 211 of file StandardColours.cs.

#### 6.4.2.56 GreenYellow

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

GreenYellow #ADFF2F

Definition at line 471 of file StandardColours.cs.

#### 6.4.2.57 Grey

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

Grey #808080

Definition at line 395 of file StandardColours.cs.

#### 6.4.2.58 HoneyDew

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

HoneyDew #F0FFF0

Definition at line 603 of file StandardColours.cs.

#### 6.4.2.59 HotPink

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

HotPink #FF69B4

Definition at line 679 of file StandardColours.cs.

#### 6.4.2.60 IndianRed

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

IndianRed #CD5C5C

Definition at line 515 of file StandardColours.cs.

#### 6.4.2.61 Indigo

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

Indigo #4B0082

Definition at line 307 of file StandardColours.cs.

#### 6.4.2.62 Ivory

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

Ivory #FFFFFF0

Definition at line 771 of file StandardColours.cs.

#### 6.4.2.63 Khaki

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

Khaki #F0E68C

Definition at line 595 of file StandardColours.cs.

#### 6.4.2.64 Lavender

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

Lavender #E6E6FA

Definition at line 575 of file StandardColours.cs.

#### 6.4.2.65 LavenderBlush

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

LavenderBlush #FFF0F5

Definition at line 739 of file StandardColours.cs.

#### 6.4.2.66 LawnGreen

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

LawnGreen #7CFC00

Definition at line 367 of file StandardColours.cs.

#### 6.4.2.67 LemonChiffon

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

LemonChiffon #FFFACD

Definition at line 751 of file StandardColours.cs.

#### 6.4.2.68 LightBlue

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

LightBlue #ADD8E6

Definition at line 467 of file StandardColours.cs.

#### 6.4.2.69 LightCoral

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

LightCoral #F08080

Definition at line 591 of file StandardColours.cs.

#### 6.4.2.70 LightCyan

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

LightCyan #E0FFFF

Definition at line 571 of file StandardColours.cs.

#### 6.4.2.71 LightGoldenRodYellow

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

LightGoldenRodYellow #FAFAD2

Definition at line 647 of file StandardColours.cs.

#### 6.4.2.72 LightGray

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

LightGray #D3D3D3

Definition at line 531 of file StandardColours.cs.

#### 6.4.2.73 LightGreen

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

LightGreen #90EE90

Definition at line 427 of file StandardColours.cs.

#### 6.4.2.74 LightGrey

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

LightGrey #D3D3D3

Definition at line 535 of file StandardColours.cs.



#### 6.4.2.75 LightPink

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

LightPink #FFB6C1

Definition at line 699 of file StandardColours.cs.

#### 6.4.2.76 LightSalmon

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

LightSalmon #FFA07A

Definition at line 691 of file StandardColours.cs.

#### 6.4.2.77 LightSeaGreen

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

LightSeaGreen #20B2AA

Definition at line 259 of file StandardColours.cs.

#### 6.4.2.78 LightSkyBlue

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

LightSkyBlue #87CEFA

Definition at line 403 of file StandardColours.cs.

#### 6.4.2.79 LightSlateGray

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

LightSlateGray #778899

Definition at line 355 of file StandardColours.cs.

#### 6.4.2.80 LightSlateGrey

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

LightSlateGrey #778899

Definition at line 359 of file StandardColours.cs.

#### 6.4.2.81 LightSteelBlue

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

LightSteelBlue #B0C4DE

Definition at line 479 of file StandardColours.cs.

#### 6.4.2.82 LightYellow

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

LightYellow #FFFFE0

Definition at line 767 of file StandardColours.cs.

#### 6.4.2.83 Lime

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

Lime #00FF00

Definition at line 235 of file StandardColours.cs.

#### 6.4.2.84 LimeGreen

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

LimeGreen #32CD32

Definition at line 279 of file StandardColours.cs.

#### 6.4.2.85 Linen

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

Linen #FAF0E6

Definition at line 643 of file StandardColours.cs.

#### 6.4.2.86 Magenta

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

Magenta #FF00FF

Definition at line 663 of file StandardColours.cs.

#### 6.4.2.87 Maroon

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

Maroon #800000

Definition at line 379 of file StandardColours.cs.

#### 6.4.2.88 MediumAquaMarine

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

MediumAquaMarine #66CDAA

Definition at line 327 of file StandardColours.cs.

#### 6.4.2.89 MediumBlue

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

MediumBlue #0000CD

Definition at line 199 of file StandardColours.cs.

#### 6.4.2.90 MediumOrchid

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

MediumOrchid #BA55D3

Definition at line 495 of file StandardColours.cs.

#### 6.4.2.91 MediumPurple

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

MediumPurple #9370DB

Definition at line 431 of file StandardColours.cs.

#### 6.4.2.92 MediumSeaGreen

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

MediumSeaGreen #3CB371

Definition at line 283 of file StandardColours.cs.

#### 6.4.2.93 MediumSlateBlue

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

MediumSlateBlue #7B68EE

Definition at line 363 of file StandardColours.cs.

#### 6.4.2.94 MediumSpringGreen

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

MediumSpringGreen #00FA9A

Definition at line 231 of file StandardColours.cs.

#### 6.4.2.95 MediumTurquoise

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

MediumTurquoise #48D1CC

Definition at line 303 of file StandardColours.cs.

#### 6.4.2.96 MediumVioletRed

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

MediumVioletRed #C71585

Definition at line 511 of file StandardColours.cs.

#### 6.4.2.97 MidnightBlue

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

MidnightBlue #191970

Definition at line 251 of file StandardColours.cs.

#### 6.4.2.98 MintCream

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

MintCream #F5FFFA

Definition at line 627 of file StandardColours.cs.

#### 6.4.2.99 MistyRose

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

MistyRose #FFE4E1

Definition at line 727 of file StandardColours.cs.

#### 6.4.2.100 Moccasin

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

Moccasin #FFE4B5

Definition at line 719 of file StandardColours.cs.

#### 6.4.2.101 NavajoWhite

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

NavajoWhite #FFDEAD

Definition at line 715 of file StandardColours.cs.

#### 6.4.2.102 Navy

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

Navy #000080

Definition at line 191 of file StandardColours.cs.

#### 6.4.2.103 OldLace

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

OldLace #FDF5E6

Definition at line 651 of file StandardColours.cs.

#### 6.4.2.104 Olive

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

Olive #808000

Definition at line 387 of file StandardColours.cs.

#### 6.4.2.105 OliveDrab

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

OliveDrab #6B8E23

Definition at line 343 of file StandardColours.cs.

#### 6.4.2.106 Orange

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

Orange #FFA500

Definition at line 695 of file StandardColours.cs.

#### 6.4.2.107 OrangeRed

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

OrangeRed #FF4500

Definition at line 671 of file StandardColours.cs.

#### 6.4.2.108 Orchid

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

Orchid #DA70D6

Definition at line 543 of file StandardColours.cs.

#### 6.4.2.109 PaleGoldenRod

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

PaleGoldenRod #EEE8AA

Definition at line 587 of file StandardColours.cs.

#### 6.4.2.110 PaleGreen

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

PaleGreen #98FB98

Definition at line 439 of file StandardColours.cs.

#### 6.4.2.111 PaleTurquoise

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

PaleTurquoise #AFEEEE

Definition at line 475 of file StandardColours.cs.

#### 6.4.2.112 PaleVioletRed

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

PaleVioletRed #DB7093

Definition at line 551 of file StandardColours.cs.

#### 6.4.2.113 PapayaWhip

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

PapayaWhip #FFEFD5

Definition at line 735 of file StandardColours.cs.

#### 6.4.2.114 PeachPuff

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

PeachPuff #FFDAB9

Definition at line 711 of file StandardColours.cs.



#### 6.4.2.115 Peru

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

Peru #CD853F

Definition at line 519 of file StandardColours.cs.

#### 6.4.2.116 Pink

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

Pink #FFC0CB

Definition at line 703 of file StandardColours.cs.

#### 6.4.2.117 Plum

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

Plum #DDA0DD

Definition at line 563 of file StandardColours.cs.

#### 6.4.2.118 PowderBlue

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

PowderBlue #B0E0E6

Definition at line 483 of file StandardColours.cs.

#### 6.4.2.119 Purple

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

Purple #800080

Definition at line 383 of file StandardColours.cs.

#### 6.4.2.120 RebeccaPurple

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

RebeccaPurple #663399

Definition at line 323 of file StandardColours.cs.

#### 6.4.2.121 Red

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

Red #FF0000

Definition at line 655 of file StandardColours.cs.

#### 6.4.2.122 RosyBrown

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

RosyBrown #BC8F8F

Definition at line 499 of file StandardColours.cs.

#### 6.4.2.123 RoyalBlue

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

RoyalBlue #4169E1

Definition at line 291 of file StandardColours.cs.

#### 6.4.2.124 SaddleBrown

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

SaddleBrown #8B4513

Definition at line 419 of file StandardColours.cs.

#### 6.4.2.125 Salmon

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

Salmon #FA8072

Definition at line 635 of file StandardColours.cs.

#### 6.4.2.126 SandyBrown

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

SandyBrown #F4A460

Definition at line 611 of file StandardColours.cs.

#### 6.4.2.127 SeaGreen

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

SeaGreen #2E8B57

Definition at line 267 of file StandardColours.cs.

#### 6.4.2.128 SeaShell

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

SeaShell #FFF5EE

Definition at line 743 of file StandardColours.cs.

#### 6.4.2.129 Sienna

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

Sienna #A0522D

Definition at line 451 of file StandardColours.cs.

#### 6.4.2.130 Silver

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

Silver #C0C0C0

Definition at line 507 of file StandardColours.cs.

#### 6.4.2.131 SkyBlue

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

SkyBlue #87CEEB

Definition at line 399 of file StandardColours.cs.

#### 6.4.2.132 SlateBlue

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

SlateBlue #6A5ACD

Definition at line 339 of file StandardColours.cs.

#### 6.4.2.133 SlateGray

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

SlateGray #708090

Definition at line 347 of file StandardColours.cs.

#### 6.4.2.134 SlateGrey

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

SlateGrey #708090

Definition at line 351 of file StandardColours.cs.

#### 6.4.2.135 Snow

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

Snow #FFFAFA

Definition at line 759 of file StandardColours.cs.

#### 6.4.2.136 SpringGreen

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

SpringGreen #00FF7F

Definition at line 239 of file StandardColours.cs.

#### 6.4.2.137 SteelBlue

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

SteelBlue #4682B4

Definition at line 295 of file StandardColours.cs.

#### 6.4.2.138 Tan

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

Tan #D2B48C

Definition at line 527 of file StandardColours.cs.

#### 6.4.2.139 Teal

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

Teal #008080

Definition at line 215 of file StandardColours.cs.

#### 6.4.2.140 Thistle

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

Thistle #D8BFD8

Definition at line 539 of file StandardColours.cs.

#### 6.4.2.141 Tomato

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

Tomato #FF6347

Definition at line 675 of file StandardColours.cs.

#### 6.4.2.142 Turquoise

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

Turquoise #40E0D0

Definition at line 287 of file StandardColours.cs.

#### 6.4.2.143 Violet

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

Violet #EE82EE

Definition at line 583 of file StandardColours.cs.

#### 6.4.2.144 Wheat

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

Wheat #F5DEB3

Definition at line 615 of file StandardColours.cs.

#### 6.4.2.145 White

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

White #FFFFFF

Definition at line 775 of file StandardColours.cs.

#### 6.4.2.146 WhiteSmoke

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

WhiteSmoke #F5F5F5

Definition at line 623 of file StandardColours.cs.

#### 6.4.2.147 Yellow

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

Yellow #FFFF00

Definition at line 763 of file StandardColours.cs.

#### 6.4.2.148 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

## 6.5 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  $\geq 0$ .*
- double `Bottom` [get]  
*Depth of the deepest glyph in the string below the baseline. Always  $\leq 0$ .*

### 6.5.1 Detailed Description

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

Definition at line 514 of file Graphics.cs.

### 6.5.2 Property Documentation

#### 6.5.2.1 Bottom

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

Depth of the deepest glyph in the string below the baseline. Always  $\leq 0$ .

Definition at line 544 of file Graphics.cs.

#### 6.5.2.2 Height

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

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

Definition at line 524 of file Graphics.cs.



### 6.5.2.3 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 529 of file Graphics.cs.

### 6.5.2.4 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 534 of file Graphics.cs.

### 6.5.2.5 Top

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

Height of the tallest glyph in the string over the baseline. Always  $\geq 0$ .

Definition at line 539 of file Graphics.cs.

### 6.5.2.6 Width

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

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

Definition at line 519 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.6 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.*

### 6.6.1 Detailed Description

Represents a collection of pages.

Definition at line 27 of file Document.cs.

### 6.6.2 Constructor & Destructor Documentation

#### 6.6.2.1 Document()

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

Create a new document.

Definition at line 38 of file Document.cs.

### 6.6.3 Member Data Documentation

#### 6.6.3.1 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

## 6.7 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  $\geq 0$ .*
- double [Descent](#) [get]  
*Maximum depth below the baseline of the usual glyphs in the font (there may be glyphs deeper than this). Always  $\leq 0$ .*
- double [YMax](#) [get]  
*Absolute maximum height over the baseline of the glyphs in the font. Always  $\geq 0$ .*
- double [YMin](#) [get]  
*Absolute maximum depth below the baseline of the glyphs in the font. Always  $\leq 0$ .*

#### 6.7.1 Detailed Description

Represents a typeface with a specific size.

Definition at line 509 of file Graphics.cs.

#### 6.7.2 Constructor & Destructor Documentation

##### 6.7.2.1 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 <a href="#">FontFamily</a> .
<i>fontSize</i>	The font size, in graphics units.

Definition at line 572 of file Graphics.cs.

## 6.7.3 Member Function Documentation

### 6.7.3.1 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 655 of file Graphics.cs.

### 6.7.3.2 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 688 of file Graphics.cs.

## 6.7.4 Property Documentation

### 6.7.4.1 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  $\geq 0$ .

Definition at line 581 of file Graphics.cs.

### 6.7.4.2 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  $\leq 0$ .

Definition at line 599 of file Graphics.cs.

### 6.7.4.3 FontFamily

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

Font typeface.

Definition at line 565 of file Graphics.cs.

### 6.7.4.4 FontSize

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

Font size, in graphics units.

Definition at line 560 of file Graphics.cs.

#### 6.7.4.5 YMax

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

Absolute maximum height over the baseline of the glyphs in the font. Always  $\geq 0$ .

Definition at line 617 of file Graphics.cs.

#### 6.7.4.6 YMin

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

Absolute maximum depth below the baseline of the glyphs in the font. Always  $\leq 0$ .

Definition at line 635 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.8 VectSharp.FontFamily Class Reference

Represents a typeface.

### 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-BoldItalic", "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.*

### 6.8.1 Detailed Description

Represents a typeface.

Definition at line 724 of file Graphics.cs.

### 6.8.2 Member Enumeration Documentation

## Enumerator

---

### 6.8.2.1 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 763 of file Graphics.cs.

## 6.8.3 Constructor & Destructor Documentation

### 6.8.3.1 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 866 of file Graphics.cs.



### 6.8.3.2 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 915 of file Graphics.cs.

### 6.8.3.3 FontFamily() [3/3]

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

Create a new standard [FontFamily](#).

#### Parameters

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

Definition at line 931 of file Graphics.cs.

## 6.8.4 Member Data Documentation

### 6.8.4.1 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 742 of file Graphics.cs.

### 6.8.4.2 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 747 of file Graphics.cs.

## 6.8.5 Property Documentation

### 6.8.5.1 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 839 of file Graphics.cs.

### 6.8.5.2 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 850 of file Graphics.cs.

### 6.8.5.3 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 855 of file Graphics.cs.

#### 6.8.5.4 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 860 of file Graphics.cs.

#### 6.8.5.5 IsStandardFamily

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

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

Definition at line 758 of file Graphics.cs.

#### 6.8.5.6 TrueTypeFile

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

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

See also

[VectSharp.TrueTypeFile](#)

.

Definition at line 845 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.9 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 **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 **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 [CopyToGraphicsContext](#) ([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.

## 6.9.1 Detailed Description

Represents an abstract drawing surface.

Definition at line 1864 of file Graphics.cs.

## 6.9.2 Member Function Documentation

### 6.9.2.1 CopyToIGraphicsContext()

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

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

Parameters

<i>destinationContext</i>	The <a href="#">IGraphicsContext</a> on which the graphics are to be copied.
---------------------------	--

Definition at line 2327 of file Graphics.cs.

### 6.9.2.2 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 <a href="#">Graphics</a> object to draw on the current <a href="#">Graphics</a> object.

Definition at line 2526 of file Graphics.cs.

### 6.9.2.3 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 <a href="#">Graphics</a> object to draw on the current <a href="#">Graphics</a> object.

Definition at line 2510 of file Graphics.cs.

### 6.9.2.4 FillPath()

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

Fill a [GraphicsPath](#).

## Parameters

<i>path</i>	The <a href="#">GraphicsPath</a> to fill.
<i>fillColour</i>	The <a href="#">Colour</a> with which to fill the <a href="#">GraphicsPath</a> .
<i>tag</i>	A tag to identify the filled path.

Definition at line 1874 of file Graphics.cs.

### 6.9.2.5 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 1982 of file Graphics.cs.

#### 6.9.2.6 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 1968 of file Graphics.cs.

#### 6.9.2.7 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 2045 of file Graphics.cs.

### 6.9.2.8 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 2030 of file Graphics.cs.

### 6.9.2.9 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 <a href="#">GraphicsPath</a> 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 2098 of file Graphics.cs.

#### 6.9.2.10 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 2302 of file Graphics.cs.

#### 6.9.2.11 Restore()

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

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

Definition at line 2318 of file Graphics.cs.

#### 6.9.2.12 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 1899 of file Graphics.cs.

#### 6.9.2.13 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 1909 of file Graphics.cs.

#### 6.9.2.14 Save()

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

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

Definition at line 2310 of file Graphics.cs.

### 6.9.2.15 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 1956 of file Graphics.cs.

### 6.9.2.16 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 <a href="#">GraphicsPath</a> to stroke.
<i>strokeColour</i>	The <a href="#">Colour</a> with which to stroke the <a href="#">GraphicsPath</a> .
<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 1890 of file Graphics.cs.

### 6.9.2.17 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 2016 of file Graphics.cs.

#### 6.9.2.18 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.
<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 1998 of file Graphics.cs.

### 6.9.2.19 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 2082 of file Graphics.cs.

### 6.9.2.20 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 2063 of file Graphics.cs.

## 6.9.2.21 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 <a href="#">GraphicsPath</a> 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.
<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 2204 of file Graphics.cs.

### 6.9.2.22 Transform()

```
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 1926 of file Graphics.cs.

### 6.9.2.23 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 1937 of file Graphics.cs.

### 6.9.2.24 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 1946 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.10 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](#).*

## Properties

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

### 6.10.1 Detailed Description

Represents a graphics path that can be filled or stroked.

Definition at line 2676 of file Graphics.cs.

### 6.10.2 Member Function Documentation

#### 6.10.2.1 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 3127 of file Graphics.cs.

**6.10.2.2 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 2925 of file Graphics.cs.

**6.10.2.3 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 2938 of file Graphics.cs.

**6.10.2.4 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 <a href="#">GraphicsPath</a> 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 3015 of file Graphics.cs.

**6.10.2.5 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 2766 of file Graphics.cs.

## 6.10.2.6 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 2746 of file Graphics.cs.

## 6.10.2.7 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 2910 of file Graphics.cs.

### 6.10.2.8 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 2900 of file Graphics.cs.

### 6.10.2.9 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 2879 of file Graphics.cs.

### 6.10.2.10 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 counterclockwise arc is drawn.
<i>endPoint</i>	The end point of the arc.

#### Returns

Definition at line 2782 of file Graphics.cs.

### 6.10.2.11 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 <a href="#">GraphicsPath</a> .
---------------	--

#### Returns

The normal to the point at the specified position.

Definition at line 3826 of file Graphics.cs.

### 6.10.2.12 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 <a href="#">GraphicsPath</a> (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 3837 of file Graphics.cs.

### 6.10.2.13 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 <a href="#">GraphicsPath</a> .
---------------	--

#### Returns

The point at the specified position.

Definition at line 3242 of file Graphics.cs.

### 6.10.2.14 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 <a href="#">GraphicsPath</a> (0 is the start of the path, 1 is the end of the path).
-----------------	--



**Returns**

The point at the specified position.

Definition at line 3232 of file Graphics.cs.

**6.10.2.15 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 <a href="#">GraphicsPath</a> .
---------------	--

**Returns**

The tangent to the point at the specified position.

Definition at line 3539 of file Graphics.cs.

**6.10.2.16 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 <a href="#">GraphicsPath</a> (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 3529 of file Graphics.cs.

**6.10.2.17 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 2731 of file Graphics.cs.

**6.10.2.18 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 2712 of file Graphics.cs.

**6.10.2.19 MeasureLength()**

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

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

## Returns

The length of the [GraphicsPath](#)

Definition at line 3160 of file Graphics.cs.

**6.10.2.20 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 2701 of file Graphics.cs.

**6.10.2.21 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 2689 of file Graphics.cs.

**6.10.3 Property Documentation****6.10.3.1 Segments**

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

The segments that make up the path.

Definition at line 2681 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.11 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 [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.*

## 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.*

### 6.11.1 Detailed Description

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

Definition at line 1671 of file Graphics.cs.

### 6.11.2 Member Function Documentation

#### 6.11.2.1 Close()

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

Close the current figure.

#### 6.11.2.2 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.

**6.11.2.3 Fill()**

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

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

**6.11.2.4 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.

**6.11.2.5 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.

### 6.11.2.6 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.

### 6.11.2.7 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.

### 6.11.2.8 Restore()

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

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

### 6.11.2.9 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.
--------------	--

**6.11.2.10 Save()**

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

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

**6.11.2.11 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.

**6.11.2.12 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&lt;Int32, Int32, Int32, Double&gt;</code> containing component information for the colour. For r, g, and b, range: [0, 255]; for a, range: [0, 1].
--------------	---

**6.11.2.13 SetFillStyle() [2/2]**

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

Set the current [FillStyle](#).

## Parameters

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

**6.11.2.14 SetLineDash()**

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

Set the current line dash pattern.

## Parameters

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

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

**6.11.2.16 SetStrokeStyle() [2/2]**

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

Set the current [StrokeStyle](#).

## Parameters

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

### 6.11.2.17 Stroke()

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

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

### 6.11.2.18 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.

### 6.11.2.19 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.

### 6.11.2.20 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.

## 6.11.3 Property Documentation

### 6.11.3.1 FillStyle

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

Current colour used to fill paths.

Definition at line 1778 of file Graphics.cs.

### 6.11.3.2 Font

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

The current font.

Definition at line 1727 of file Graphics.cs.

### 6.11.3.3 Height

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

Height of the graphic surface.

Definition at line 1681 of file Graphics.cs.

#### 6.11.3.4 LineCap

`LineCaps` VectSharp.IGraphicsContext.LineCap [set]

Current line cap used to stroke paths.

Definition at line 1842 of file Graphics.cs.

#### 6.11.3.5 LineJoin

`LineJoins` VectSharp.IGraphicsContext.LineJoin [set]

Current line join used to stroke paths.

Definition at line 1847 of file Graphics.cs.

#### 6.11.3.6 LineWidth

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

Current line width used to stroke paths.

Definition at line 1837 of file Graphics.cs.

#### 6.11.3.7 StrokeStyle

`Colour` VectSharp.IGraphicsContext.StrokeStyle [get]

Current colour used to stroke paths.

Definition at line 1795 of file Graphics.cs.

#### 6.11.3.8 Tag

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

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

Definition at line 1858 of file Graphics.cs.

### 6.11.3.9 TextBaseline

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

The current text baseline.

Definition at line 1733 of file Graphics.cs.

### 6.11.3.10 Width

`double` VectSharp.IGraphicsContext.Width [get]

Width of the graphic surface.

Definition at line 1676 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.12 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*

### 6.12.1 Detailed Description

Represents instructions on how to paint a dashed line.

Definition at line 125 of file Graphics.cs.

### 6.12.2 Constructor & Destructor Documentation

#### 6.12.2.1 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 153 of file Graphics.cs.

### 6.12.3 Member Data Documentation

#### 6.12.3.1 Phase

```
double VectSharp.LineDash.Phase
```

Position in the dash pattern at which the line starts.

Definition at line 145 of file Graphics.cs.

#### 6.12.3.2 SolidLine

```
LineDash VectSharp.LineDash.SolidLine = new LineDash(0, 0, 0) [static]
```

A solid (not dashed) line

Definition at line 130 of file Graphics.cs.



### 6.12.3.3 UnitsOff

```
double VectSharp.LineDash.UnitsOff
```

Length of the "off" (not painted) segment.

Definition at line 140 of file Graphics.cs.

### 6.12.3.4 UnitsOn

```
double VectSharp.LineDash.UnitsOn
```

Length of the "on" (painted) segment.

Definition at line 135 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.13 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.*

### 6.13.1 Detailed Description

Represents a [Graphics](#) object with a width and height.

Definition at line 47 of file Document.cs.

## 6.13.2 Constructor & Destructor Documentation

### 6.13.2.1 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.

## 6.13.3 Member Function Documentation

### 6.13.3.1 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.

## 6.13.4 Property Documentation

### 6.13.4.1 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.

#### 6.13.4.2 Graphics

[Graphics](#) VectSharp.Page.Graphics [get], [set]

[Graphics](#) surface of the page.

Definition at line 62 of file Document.cs.

#### 6.13.4.3 Height

double VectSharp.Page.Height [get], [set]

Height of the page.

Definition at line 57 of file Document.cs.

#### 6.13.4.4 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

## 6.14 VectSharp.SVG.Parser Class Reference

Contains methods to read an [SVG](#) image file.

### Static Public Member Functions

- 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.*

### 6.14.1 Detailed Description

Contains methods to read an [SVG](#) image file.

Definition at line 31 of file SVGParser.cs.

### 6.14.2 Member Function Documentation

#### 6.14.2.1 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 <a href="#">SVG</a> image file.
-----------------	---

##### Returns

A [Page](#) containing the image represented by the file.

Definition at line 69 of file SVGParser.cs.

#### 6.14.2.2 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 <a href="#">SVG</a> source code.
------------------------	--

##### Returns

A [Page](#) containing the image represented by the *svgSourceStream* .

Definition at line 79 of file SVGParser.cs.

### 6.14.2.3 FromString()

```
static Page VectSharp.SVG.Parser.FromString (
    string svgSource ) [static]
```

Parses [SVG](#) source into a [Page](#) containing the image represented by the code.

#### Parameters

<code>svgSource</code>	The <a href="#">SVG</a> source code.
------------------------	--------------------------------------

#### Returns

A [Page](#) containing the image represented by the `svgSource` .

Definition at line 38 of file SVGParser.cs.

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

- VectSharp.SVG/SVGParser.cs

## 6.15 VectSharp.PDF.PDFContextInterpreter Class Reference

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

### Public Types

- enum [TextOptions](#) { [TextOptions.SubsetFont](#)s, [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](#)s, bool compressStreams=true)  
*Save the document to a [PDF](#) file.*
- static void [SaveAsPDF](#) (this [Document](#) document, Stream stream, [TextOptions](#) textOption=[TextOptions.SubsetFont](#)s, bool compressStreams=true)  
*Save the document to a [PDF](#) stream.*

### 6.15.1 Detailed Description

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

Definition at line 509 of file PDFContext.cs.

### 6.15.2 Member Enumeration Documentation

#### 6.15.2.1 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 677 of file PDFContext.cs.

### 6.15.3 Member Function Documentation

#### 6.15.3.1 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 699 of file PDFContext.cs.

#### 6.15.3.2 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 666 of file PDFContext.cs.

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

- VectSharp.PDF/PDFContext.cs

## 6.16 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](#).*

### 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.*

### 6.16.1 Detailed Description

Represents a point relative to an origin in the top-left corner.

Definition at line 956 of file Graphics.cs.

### 6.16.2 Constructor & Destructor Documentation

#### 6.16.2.1 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 973 of file Graphics.cs.

## 6.16.3 Member Function Documentation

### 6.16.3.1 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 983 of file Graphics.cs.

### 6.16.3.2 Normalize()

```
Point VectSharp.Point.Normalize ( )
```

Normalises a [Point](#).

**Returns**

The normalised [Point](#).

Definition at line 992 of file Graphics.cs.

## 6.16.4 Member Data Documentation

### 6.16.4.1 X

```
double VectSharp.Point.X
```

Horizontal (x) coordinate, measured to the right of the origin.

Definition at line 961 of file Graphics.cs.



#### 6.16.4.2 Y

```
double VectSharp.Point.Y
```

Vertical (y) coordinate, measured to the bottom of the origin.

Definition at line 966 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.17 VectSharp.Raster.RasterContextInterpreter Class Reference

Contains methods to render a [Page](#) as a raster 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.*

### 6.17.1 Detailed Description

Contains methods to render a [Page](#) as a raster image.

Definition at line 993 of file RasterContext.cs.

### 6.17.2 Member Function Documentation

#### 6.17.2.1 SaveAsPNG() [1/2]

```
static void VectSharp.Raster.RasterContextInterpreter.SaveAsPNG (
    this Page page,
    Stream stream,
    double scale = 1 ) [static]
```

Render the page to a PNG stream.

#### Parameters

<i>page</i>	The <a href="#">Page</a> 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 1016 of file RasterContext.cs.

### 6.17.2.2 SaveAsPNG() [2/2]

```
static void VectSharp.Raster.RasterContextInterpreter.SaveAsPNG (
    this Page page,
    string fileName,
    double scale = 1 ) [static]
```

Render the page to a PNG file.

#### Parameters

<i>page</i>	The <a href="#">Page</a> 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 1002 of file RasterContext.cs.

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

- VectSharp.Raster/RasterContext.cs

## 6.18 VectSharp.Canvas.RenderAction Class Reference

Represents a light-weight rendering action.

### Public Types

- enum [ActionTypes](#) { [ActionTypes.Path](#), [ActionTypes.Text](#) }  
*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, string tag=null)  
*Creates a new [RenderAction](#) representing a Path.*
- static [RenderAction TextAction](#) (FormattedText text, IBrush fill, Avalonia.Matrix transform, string tag=null)  
*Creates a new [RenderAction](#) representing text.*

## 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.*
- 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.*

### 6.18.1 Detailed Description

Represents a light-weight rendering action.

Definition at line 777 of file AvaloniaContext.cs.

## 6.18.2 Member Enumeration Documentation

### 6.18.2.1 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.

Definition at line 782 of file AvaloniaContext.cs.

## 6.18.3 Member Function Documentation

### 6.18.3.1 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 951 of file AvaloniaContext.cs.

### 6.18.3.2 PathAction()

```
static RenderAction VectSharp.Canvas.RenderAction.PathAction (
    Geometry geometry,
    Pen stroke,
    IBrush fill,
    Avalonia.Matrix transform,
    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>tag</i>	A tag to access the <code>RenderAction</code> . If this is null this <code>RenderAction</code> is not visible in the hit test.

### Returns

A new [RenderAction](#) representing a Path.

Definition at line 914 of file AvaloniaContext.cs.

#### 6.18.3.3 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 959 of file AvaloniaContext.cs.

#### 6.18.3.4 TextAction()

```
static RenderAction VectSharp.Canvas.RenderAction.TextAction (
    FormattedText text,
    IBrush fill,
    Avalonia.Matrix transform,
    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>tag</i>	A tag to access the <a href="#">RenderAction</a> . If this is null this <a href="#">RenderAction</a> is not visible in the hit test.

### Returns

Definition at line 935 of file AvaloniaContext.cs.

## 6.18.4 Property Documentation

#### 6.18.4.1 ActionType

`ActionTypes VectSharp.Canvas.RenderAction.ActionType [get]`

Type of the rendering action.

Definition at line 798 of file AvaloniaContext.cs.

#### 6.18.4.2 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 818 of file AvaloniaContext.cs.

#### 6.18.4.3 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 803 of file AvaloniaContext.cs.

#### 6.18.4.4 InverseTransform

`Avalonia.Matrix VectSharp.Canvas.RenderAction.InverseTransform = Avalonia.Matrix.Identity [get]`

Inverse transformation matrix.

Definition at line 826 of file AvaloniaContext.cs.

#### 6.18.4.5 Parent

`Avalonia.Controls.Canvas VectSharp.Canvas.RenderAction.Parent [get]`

The container of this [RenderAction](#).

Definition at line 851 of file AvaloniaContext.cs.

#### 6.18.4.6 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 813 of file AvaloniaContext.cs.

#### 6.18.4.7 Tag

```
string VectSharp.Canvas.RenderAction.Tag [get], [set]
```

A tag to access the [RenderAction](#).

Definition at line 844 of file AvaloniaContext.cs.

#### 6.18.4.8 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 808 of file AvaloniaContext.cs.

#### 6.18.4.9 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 831 of file AvaloniaContext.cs.

### 6.18.5 Event Documentation

#### 6.18.5.1 PointerEnter

```
EventHandler<Avalonia.Input.PointerEventArgs> VectSharp.Canvas.RenderAction.PointerEnter
```

Raised when the pointer enters the area covered by the [RenderAction](#).

Definition at line 862 of file AvaloniaContext.cs.

### 6.18.5.2 PointerLeave

```
EventHandler<Avalonia.Input.PointerEventArgs> VectSharp.Canvas.RenderAction.PointerLeave
```

Raised when the pointer leaves the area covered by the [RenderAction](#).

Definition at line 867 of file AvaloniaContext.cs.

### 6.18.5.3 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 872 of file AvaloniaContext.cs.

### 6.18.5.4 PointerReleased

```
EventHandler<Avalonia.Input.PointerReleasedEventArgs> VectSharp.Canvas.RenderAction.Pointer↵  
Released
```

Raised when the pointer is released after a [PointerPressed](#) event.

Definition at line 877 of file AvaloniaContext.cs.

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

- VectSharp.Canvas/AvaloniaContext.cs

## 6.19 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 ).*



## 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](#).*

### 6.19.1 Detailed Description

Represents a segment as part of a [GraphicsPath](#).

Definition at line 1060 of file Graphics.cs.

### 6.19.2 Member Function Documentation

#### 6.19.2.1 Clone()

```
abstract Segment VectSharp.Segment.Clone ( ) [pure virtual]
```

Creates a copy of the [Segment](#).

##### Returns

A copy of the [Segment](#).

#### 6.19.2.2 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 <a href="#">Segment</a> starts (i.e. the endpoint of the previous <a href="#">Segment</a> ).
<i>position</i>	The relative position on the <a href="#">Segment</a> (0 is the start of the <a href="#">Segment</a> , 1 is the end of the <a href="#">Segment</a> ).

**Returns**

The point at the specified position.

**6.19.2.3 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 <a href="#">Segment</a> starts (i.e. the endpoint of the previous <a href="#">Segment</a> ).
<i>position</i>	The relative position on the <a href="#">Segment</a> (0 is the start of the <a href="#">Segment</a> , 1 is the end of the <a href="#">Segment</a> ).

**Returns**

The tangent to the point at the specified position.

**6.19.2.4 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 <a href="#">Segment</a> starts (i.e. the endpoint of the previous <a href="#">Segment</a> ).
----------------------	---

**Returns**

The length of the segment.

**6.19.3 Property Documentation****6.19.3.1 Point**

```
virtual Point VectSharp.Segment.Point [get]
```

The end point of the [Segment](#).

Definition at line 1076 of file Graphics.cs.

### 6.19.3.2 Points

```
Point [ ] VectSharp.Segment.Points [get]
```

The points used to define the [Segment](#).

Definition at line 1071 of file Graphics.cs.

### 6.19.3.3 Type

```
abstract SegmentType VectSharp.Segment.Type [get]
```

The type of the [Segment](#).

Definition at line 1066 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.20 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.*

### 6.20.1 Detailed Description

Represents the size of an object.

Definition at line 1002 of file Graphics.cs.

### 6.20.2 Constructor & Destructor Documentation

#### 6.20.2.1 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 1019 of file Graphics.cs.

### 6.20.3 Member Data Documentation

#### 6.20.3.1 Height

```
double VectSharp.Size.Height
```

Height of the object.

Definition at line 1012 of file Graphics.cs.

#### 6.20.3.2 Width

```
double VectSharp.Size.Width
```

Width of the object.

Definition at line 1007 of file Graphics.cs.

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

- VectSharp/Graphics.cs

## 6.21 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.*

#### 6.21.1 Detailed Description

Contains methods to render a [Page](#) as an [SVG](#) file.

Definition at line 649 of file SVGContext.cs.

#### 6.21.2 Member Enumeration Documentation

##### 6.21.2.1 TextOptions

```
enum VectSharp.SVG.SVGContextInterpreter.TextOptions [strong]
```

Defines whether the used fonts should be included in the file.

##### Enumerator

<a href="#">EmbedFonts</a>	Embeds the full font files.
<a href="#">SubsetFonts</a>	Embeds subsetting font files containing only the glyphs for the characters that have been used.
<a href="#">ConvertIntoPaths</a>	Does not embed any font file and converts all text items into paths.
<a href="#">DoNotEmbed</a>	Does not embed any font file, but still encodes text items as such.

Definition at line 669 of file SVGContext.cs.

## 6.21.3 Member Function Documentation

### 6.21.3.1 SaveAsSVG() [1/2]

```
static void VectSharp.SVG.SVGContextInterpreter.SaveAsSVG (
    this Page page,
    Stream stream,
    TextOptions textOption = TextOptions.SubsetFonts ) [static]
```

Render the page to an SVG stream.

#### Parameters

<i>page</i>	The <a href="#">Page</a> to render.
<i>stream</i>	The stream to which the <a href="#">SVG</a> data will be written.
<i>textOption</i>	Defines whether the used fonts should be included in the file.

Definition at line 698 of file SVGContext.cs.

### 6.21.3.2 SaveAsSVG() [2/2]

```
static void VectSharp.SVG.SVGContextInterpreter.SaveAsSVG (
    this Page page,
    string fileName,
    TextOptions textOption = TextOptions.SubsetFonts ) [static]
```

Render the page to an SVG file.

#### Parameters

<i>page</i>	The <a href="#">Page</a> 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 658 of file SVGContext.cs.

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

- VectSharp.SVG/SVGContext.cs

## 6.22 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.*

### 6.22.1 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.

### 6.22.2 Member Function Documentation

#### 6.22.2.1 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.



### 6.22.2.2 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.

### 6.22.2.3 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.

### 6.22.2.4 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.

### 6.22.2.5 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.

### 6.22.2.6 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.

### 6.22.2.7 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.

**6.22.2.8 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.

**6.22.2.9 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.

**6.22.2.10 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.

#### 6.22.2.11 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.

#### 6.22.2.12 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.

#### 6.22.2.13 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.

#### 6.22.2.14 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.

#### 6.22.2.15 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.

**6.22.2.16 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.

**6.22.2.17 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.

**6.22.2.18 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.

**6.22.2.19 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.

**6.22.2.20 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.

#### 6.22.2.21 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.

#### 6.22.2.22 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.

#### 6.22.2.23 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.

#### 6.22.2.24 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.

#### 6.22.2.25 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.

**6.22.3 Property Documentation****6.22.3.1 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

**6.23 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.*



### 6.23.1 Detailed Description

Represents a point in a TrueType path description.

Definition at line 1337 of file TrueType.cs.

### 6.23.2 Member Data Documentation

#### 6.23.2.1 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.

#### 6.23.2.2 X

```
double VectSharp.TrueTypeFile.TrueTypePoint.X
```

The horizontal coordinate of the point.

Definition at line 1342 of file TrueType.cs.

#### 6.23.2.3 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

## 6.24 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.*

### 6.24.1 Detailed Description

Represents the maximum height above and depth below the baseline of a glyph.

Definition at line 2170 of file `TrueType.cs`.

### 6.24.2 Member Data Documentation

#### 6.24.2.1 YMax

```
int VectSharp.TrueTypeFile.VerticalMetrics.YMax
```

The maximum height above the baseline of the glyph.

Definition at line 2180 of file `TrueType.cs`.

#### 6.24.2.2 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, [29](#)
- ActionType
  - VectSharp.Canvas.RenderAction, [127](#)
- ActionTypes
  - VectSharp.Canvas.RenderAction, [126](#)
- AddSmoothSpline
  - VectSharp.GraphicsPath, [92](#)
- AddText
  - VectSharp.GraphicsPath, [93](#)
- AddTextOnPath
  - VectSharp.GraphicsPath, [94](#)
- AliceBlue
  - VectSharp.Colours, [36](#)
- AlwaysConvert
  - VectSharp.Canvas.AvaloniaContextInterpreter, [17](#)
- AntiqueWhite
  - VectSharp.Colours, [36](#)
- Aqua
  - VectSharp.Colours, [36](#)
- Aquamarine
  - VectSharp.Colours, [36](#)
- Arc
  - VectSharp, [13](#)
  - VectSharp.GraphicsPath, [94](#), [95](#)
- Ascent
  - VectSharp.Font, [71](#)
- Azure
  - VectSharp.Colours, [36](#)

## B

- VectSharp.Colour, [29](#)
- Background
  - VectSharp.Page, [116](#)
- Baseline
  - VectSharp, [13](#)
- Beige
  - VectSharp.Colours, [37](#)
- Bevel
  - VectSharp, [12](#)
- Bisque
  - VectSharp.Colours, [37](#)
- Black
  - VectSharp.Colours, [37](#)
- BlanchedAlmond
  - VectSharp.Colours, [37](#)
- Blue
  - VectSharp.Colours, [37](#)
- BlueViolet
  - VectSharp.Colours, [38](#)

## Bottom

- VectSharp, [13](#)
  - VectSharp.Font.DetailedFontMetrics, [66](#)
- BringToFront
  - VectSharp.Canvas.RenderAction, [126](#)
- Brown
  - VectSharp.Colours, [38](#)
- BurlyWood
  - VectSharp.Colours, [38](#)
- Butt
  - VectSharp, [12](#)

## CadetBlue

- VectSharp.Colours, [38](#)

## Center

- VectSharp, [13](#)

## Chartreuse

- VectSharp.Colours, [38](#)

## Chocolate

- VectSharp.Colours, [39](#)

## Clone

- VectSharp.Segment, [131](#)

## Close

- VectSharp, [13](#)
  - VectSharp.GraphicsPath, [95](#)
  - VectSharp.IGraphicsContext, [104](#)

## ConvertIfNecessary

- VectSharp.Canvas.AvaloniaContextInterpreter, [17](#)

## ConvertIntoPaths

- VectSharp.PDF.PDFContextInterpreter, [120](#)
  - VectSharp.SVG.SVGContextInterpreter, [135](#)

## CopyToIGraphicsContext

- VectSharp.Graphics, [79](#)

## Coral

- VectSharp.Colours, [39](#)

## CornflowerBlue

- VectSharp.Colours, [39](#)

## Cornsilk

- VectSharp.Colours, [39](#)

## Courier

- VectSharp.FontFamily, [74](#)

## CourierBold

- VectSharp.FontFamily, [74](#)

## CourierBoldOblique

- VectSharp.FontFamily, [74](#)

## CourierOblique

- VectSharp.FontFamily, [74](#)

## Crimson

- VectSharp.Colours, [39](#)

## Crop

- VectSharp.Page, 116
- CubicBezier
  - VectSharp, 13
- CubicBezierTo
  - VectSharp.GraphicsPath, 95, 96
  - VectSharp.IGraphicsContext, 104
- Cyan
  - VectSharp.Colours, 40
- DarkBlue
  - VectSharp.Colours, 40
- DarkCyan
  - VectSharp.Colours, 40
- DarkGoldenRod
  - VectSharp.Colours, 40
- DarkGray
  - VectSharp.Colours, 40
- DarkGreen
  - VectSharp.Colours, 41
- DarkGrey
  - VectSharp.Colours, 41
- DarkKhaki
  - VectSharp.Colours, 41
- DarkMagenta
  - VectSharp.Colours, 41
- DarkOliveGreen
  - VectSharp.Colours, 41
- DarkOrange
  - VectSharp.Colours, 42
- DarkOrchid
  - VectSharp.Colours, 42
- DarkRed
  - VectSharp.Colours, 42
- DarkSalmon
  - VectSharp.Colours, 42
- DarkSeaGreen
  - VectSharp.Colours, 42
- DarkSlateBlue
  - VectSharp.Colours, 43
- DarkSlateGray
  - VectSharp.Colours, 43
- DarkSlateGrey
  - VectSharp.Colours, 43
- DarkTurquoise
  - VectSharp.Colours, 43
- DarkViolet
  - VectSharp.Colours, 43
- DeepPink
  - VectSharp.Colours, 44
- DeepSkyBlue
  - VectSharp.Colours, 44
- Descent
  - VectSharp.Font, 71
- Destroy
  - VectSharp.TrueTypeFile, 138
- DimGray
  - VectSharp.Colours, 44
- DimGrey
  - VectSharp.Colours, 44
- Document
  - VectSharp.Document, 68
- DodgerBlue
  - VectSharp.Colours, 44
- DoNotEmbed
  - VectSharp.SVG.SVGContextInterpreter, 135
- DrawGraphics
  - VectSharp.Graphics, 79, 80
- EllipticalArc
  - VectSharp.GraphicsPath, 97
- EmbedFonts
  - VectSharp.SVG.SVGContextInterpreter, 135
- FileName
  - VectSharp.FontFamily, 76
- Fill
  - VectSharp.Canvas.RenderAction, 128
  - VectSharp.IGraphicsContext, 105
- FillPath
  - VectSharp.Graphics, 80
- FillRectangle
  - VectSharp.Graphics, 80, 81
- FillStyle
  - VectSharp.IGraphicsContext, 111
- FillText
  - VectSharp.Graphics, 81, 82
  - VectSharp.IGraphicsContext, 105
- FillTextOnPath
  - VectSharp.Graphics, 82
- FireBrick
  - VectSharp.Colours, 45
- FloralWhite
  - VectSharp.Colours, 45
- Font
  - VectSharp.Font, 69
  - VectSharp.IGraphicsContext, 111
- FontFamily
  - VectSharp.Font, 71
  - VectSharp.FontFamily, 74, 75
- FontSize
  - VectSharp.Font, 71
- FontStream
  - VectSharp.TrueTypeFile, 146
- ForestGreen
  - VectSharp.Colours, 45
- FromCSSString
  - VectSharp.Colour, 22
- FromFile
  - VectSharp.SVG.Parser, 118
- FromRgb
  - VectSharp.Colour, 22, 23
- FromRgba
  - VectSharp.Colour, 24–26
- FromStream
  - VectSharp.SVG.Parser, 118
- FromString
  - VectSharp.SVG.Parser, 118
- Fuchsia

- VectSharp.Colours, [45](#)
- G
  - VectSharp.Colour, [29](#)
- Gainsboro
  - VectSharp.Colours, [45](#)
- Geometry
  - VectSharp.Canvas.RenderAction, [128](#)
- Get1000EmAscent
  - VectSharp.TrueTypeFile, [138](#)
- Get1000EmDescent
  - VectSharp.TrueTypeFile, [139](#)
- Get1000EmGlyphBearings
  - VectSharp.TrueTypeFile, [139](#)
- Get1000EmGlyphVerticalMetrics
  - VectSharp.TrueTypeFile, [139](#)
- Get1000EmGlyphWidth
  - VectSharp.TrueTypeFile, [140](#)
- Get1000EmXMax
  - VectSharp.TrueTypeFile, [141](#)
- Get1000EmXMin
  - VectSharp.TrueTypeFile, [141](#)
- Get1000EmYMax
  - VectSharp.TrueTypeFile, [141](#)
- Get1000EmYMin
  - VectSharp.TrueTypeFile, [141](#)
- GetFirstCharIndex
  - VectSharp.TrueTypeFile, [142](#)
- GetFontFamilyName
  - VectSharp.TrueTypeFile, [142](#)
- GetFontName
  - VectSharp.TrueTypeFile, [142](#)
- GetGlyphIndex
  - VectSharp.TrueTypeFile, [142](#)
- GetGlyphPath
  - VectSharp.TrueTypeFile, [143](#)
- GetLastCharIndex
  - VectSharp.TrueTypeFile, [144](#)
- GetNormalAtAbsolute
  - VectSharp.GraphicsPath, [97](#)
- GetNormalAtRelative
  - VectSharp.GraphicsPath, [97](#)
- GetPointAt
  - VectSharp.Segment, [131](#)
- GetPointAtAbsolute
  - VectSharp.GraphicsPath, [98](#)
- GetPointAtRelative
  - VectSharp.GraphicsPath, [98](#)
- GetTangentAt
  - VectSharp.Segment, [132](#)
- GetTangentAtAbsolute
  - VectSharp.GraphicsPath, [99](#)
- GetTangentAtRelative
  - VectSharp.GraphicsPath, [99](#)
- GhostWhite
  - VectSharp.Colours, [46](#)
- Gold
  - VectSharp.Colours, [46](#)
- GoldenRod
  - VectSharp.Colours, [46](#)
- VectSharp.Colours, [46](#)
- Graphics
  - VectSharp.Page, [117](#)
- Gray
  - VectSharp.Colours, [46](#)
- Green
  - VectSharp.Colours, [46](#)
- GreenYellow
  - VectSharp.Colours, [47](#)
- Grey
  - VectSharp.Colours, [47](#)
- Height
  - VectSharp.Font.DetailedFontMetrics, [66](#)
  - VectSharp.IGraphicsContext, [111](#)
  - VectSharp.Page, [117](#)
  - VectSharp.Size, [134](#)
- Helvetica
  - VectSharp.FontFamily, [74](#)
- HelveticaBold
  - VectSharp.FontFamily, [74](#)
- HelveticaBoldOblique
  - VectSharp.FontFamily, [74](#)
- HelveticaOblique
  - VectSharp.FontFamily, [74](#)
- HoneyDew
  - VectSharp.Colours, [47](#)
- HotPink
  - VectSharp.Colours, [47](#)
- IndianRed
  - VectSharp.Colours, [47](#)
- Indigo
  - VectSharp.Colours, [48](#)
- InverseTransform
  - VectSharp.Canvas.RenderAction, [128](#)
- IsBold
  - VectSharp.FontFamily, [76](#)
  - VectSharp.TrueTypeFile, [144](#)
- IsFixedPitch
  - VectSharp.TrueTypeFile, [144](#)
- IsItalic
  - VectSharp.FontFamily, [76](#)
  - VectSharp.TrueTypeFile, [144](#)
- IsOblique
  - VectSharp.FontFamily, [76](#)
  - VectSharp.TrueTypeFile, [145](#)
- IsOnCurve
  - VectSharp.TrueTypeFile.TrueTypePoint, [147](#)
- IsScript
  - VectSharp.TrueTypeFile, [145](#)
- IsSerif
  - VectSharp.TrueTypeFile, [145](#)
- IsStandardFamily
  - VectSharp.FontFamily, [77](#)
- Ivory
  - VectSharp.Colours, [48](#)
- Khaki

- VectSharp.Colours, 48
- Lavender
  - VectSharp.Colours, 48
- LavenderBlush
  - VectSharp.Colours, 48
- LawnGreen
  - VectSharp.Colours, 49
- Left
  - VectSharp, 13
- LeftSideBearing
  - VectSharp.Font.DetailedFontMetrics, 66
  - VectSharp.TrueTypeFile.Bearings, 20
- LemonChiffon
  - VectSharp.Colours, 49
- LightBlue
  - VectSharp.Colours, 49
- LightCoral
  - VectSharp.Colours, 49
- LightCyan
  - VectSharp.Colours, 49
- LightGoldenRodYellow
  - VectSharp.Colours, 50
- LightGray
  - VectSharp.Colours, 50
- LightGreen
  - VectSharp.Colours, 50
- LightGrey
  - VectSharp.Colours, 50
- LightPink
  - VectSharp.Colours, 50
- LightSalmon
  - VectSharp.Colours, 51
- LightSeaGreen
  - VectSharp.Colours, 51
- LightSkyBlue
  - VectSharp.Colours, 51
- LightSlateGray
  - VectSharp.Colours, 51
- LightSlateGrey
  - VectSharp.Colours, 51
- LightSteelBlue
  - VectSharp.Colours, 52
- LightYellow
  - VectSharp.Colours, 52
- Lime
  - VectSharp.Colours, 52
- LimeGreen
  - VectSharp.Colours, 52
- Line
  - VectSharp, 13
- LineCap
  - VectSharp.IGraphicsContext, 111
- LineCaps
  - VectSharp, 12
- LineDash
  - VectSharp.LineDash, 114
- LineJoin
  - VectSharp.IGraphicsContext, 112
- LineJoins
  - VectSharp, 12
- Linen
  - VectSharp.Colours, 52
- LineTo
  - VectSharp.GraphicsPath, 99, 101
  - VectSharp.IGraphicsContext, 105
- LineWidth
  - VectSharp.IGraphicsContext, 112
- Magenta
  - VectSharp.Colours, 53
- Maroon
  - VectSharp.Colours, 53
- Measure
  - VectSharp.Segment, 132
- MeasureLength
  - VectSharp.GraphicsPath, 101
- MeasureText
  - VectSharp.Font, 70
  - VectSharp.Graphics, 83
- MeasureTextAdvanced
  - VectSharp.Font, 70
- MediumAquaMarine
  - VectSharp.Colours, 53
- MediumBlue
  - VectSharp.Colours, 53
- MediumOrchid
  - VectSharp.Colours, 53
- MediumPurple
  - VectSharp.Colours, 54
- MediumSeaGreen
  - VectSharp.Colours, 54
- MediumSlateBlue
  - VectSharp.Colours, 54
- MediumSpringGreen
  - VectSharp.Colours, 54
- MediumTurquoise
  - VectSharp.Colours, 54
- MediumVioletRed
  - VectSharp.Colours, 55
- Middle
  - VectSharp, 13
- MidnightBlue
  - VectSharp.Colours, 55
- MintCream
  - VectSharp.Colours, 55
- MistyRose
  - VectSharp.Colours, 55
- Miter
  - VectSharp, 12
- Moccasin
  - VectSharp.Colours, 55
- Modulus
  - VectSharp.Point, 122
- Move
  - VectSharp, 13
- MoveTo
  - VectSharp.GraphicsPath, 101, 102

- VectSharp.IGraphicsContext, 106
- NavajoWhite
  - VectSharp.Colours, 56
- Navy
  - VectSharp.Colours, 56
- NeverConvert
  - VectSharp.Canvas.AvaloniaContextInterpreter, 17
- Normalize
  - VectSharp.Point, 122
- OldLace
  - VectSharp.Colours, 56
- Olive
  - VectSharp.Colours, 56
- OliveDrab
  - VectSharp.Colours, 56
- Orange
  - VectSharp.Colours, 57
- OrangeRed
  - VectSharp.Colours, 57
- Orchid
  - VectSharp.Colours, 57
- Page
  - VectSharp.Page, 116
- Pages
  - VectSharp.Document, 68
- PaintToCanvas
  - VectSharp.Canvas.AvaloniaContextInterpreter, 17–19
- PaleGoldenRod
  - VectSharp.Colours, 57
- PaleGreen
  - VectSharp.Colours, 57
- PaleTurquoise
  - VectSharp.Colours, 58
- PaleVioletRed
  - VectSharp.Colours, 58
- PapayaWhip
  - VectSharp.Colours, 58
- Parent
  - VectSharp.Canvas.RenderAction, 128
- Path
  - VectSharp.Canvas.RenderAction, 126
- PathAction
  - VectSharp.Canvas.RenderAction, 126
- PeachPuff
  - VectSharp.Colours, 58
- Peru
  - VectSharp.Colours, 58
- Phase
  - VectSharp.LineDash, 114
- Pink
  - VectSharp.Colours, 59
- Plum
  - VectSharp.Colours, 59
- Point
  - VectSharp.Point, 121
- VectSharp.Segment, 132
- PointerEnter
  - VectSharp.Canvas.RenderAction, 129
- PointerLeave
  - VectSharp.Canvas.RenderAction, 129
- PointerPressed
  - VectSharp.Canvas.RenderAction, 130
- PointerReleased
  - VectSharp.Canvas.RenderAction, 130
- Points
  - VectSharp.Segment, 133
- PowderBlue
  - VectSharp.Colours, 59
- Purple
  - VectSharp.Colours, 59
- R
  - VectSharp.Colour, 29
- RebeccaPurple
  - VectSharp.Colours, 59
- Rectangle
  - VectSharp.IGraphicsContext, 106
- Red
  - VectSharp.Colours, 60
- Restore
  - VectSharp.Graphics, 83
  - VectSharp.IGraphicsContext, 106
- Right
  - VectSharp, 13
- RightSideBearing
  - VectSharp.Font.DetailedFontMetrics, 67
  - VectSharp.TrueTypeFile.Bearings, 20
- RosyBrown
  - VectSharp.Colours, 60
- Rotate
  - VectSharp.Graphics, 84
  - VectSharp.IGraphicsContext, 106
- RotateAt
  - VectSharp.Graphics, 84
- Round
  - VectSharp, 12
- RoyalBlue
  - VectSharp.Colours, 60
- SaddleBrown
  - VectSharp.Colours, 60
- Salmon
  - VectSharp.Colours, 60
- SandyBrown
  - VectSharp.Colours, 61
- Save
  - VectSharp.Graphics, 84
  - VectSharp.IGraphicsContext, 107
- SaveAsPDF
  - VectSharp.PDF.PDFContextInterpreter, 120
- SaveAsPNG
  - VectSharp.Raster.RasterContextInterpreter, 123, 124
- SaveAsSVG

- VectSharp.SVG.SVGContextInterpreter, 136
- Scale
  - VectSharp.Graphics, 84
  - VectSharp.IGraphicsContext, 107
- SeaGreen
  - VectSharp.Colours, 61
- SeaShell
  - VectSharp.Colours, 61
- Segments
  - VectSharp.GraphicsPath, 102
- SegmentType
  - VectSharp, 13
- SendToBack
  - VectSharp.Canvas.RenderAction, 127
- SetFillStyle
  - VectSharp.IGraphicsContext, 107
- SetLineDash
  - VectSharp.IGraphicsContext, 109
- SetStrokeStyle
  - VectSharp.IGraphicsContext, 109
- Sienna
  - VectSharp.Colours, 61
- Silver
  - VectSharp.Colours, 61
- Size
  - VectSharp.Size, 134
- SkyBlue
  - VectSharp.Colours, 62
- SlateBlue
  - VectSharp.Colours, 62
- SlateGray
  - VectSharp.Colours, 62
- SlateGrey
  - VectSharp.Colours, 62
- Snow
  - VectSharp.Colours, 62
- SolidLine
  - VectSharp.LineDash, 114
- SpringGreen
  - VectSharp.Colours, 63
- Square
  - VectSharp, 12
- StandardFamilies
  - VectSharp.FontFamily, 75
- StandardFontFamilies
  - VectSharp.FontFamily, 73
- StandardFontFamilyResources
  - VectSharp.FontFamily, 75
- SteelBlue
  - VectSharp.Colours, 63
- Stroke
  - VectSharp.Canvas.RenderAction, 128
  - VectSharp.IGraphicsContext, 109
- StrokePath
  - VectSharp.Graphics, 85
- StrokeRectangle
  - VectSharp.Graphics, 85, 86
- StrokeStyle
  - VectSharp.IGraphicsContext, 112
- StrokeText
  - VectSharp.Graphics, 87
  - VectSharp.IGraphicsContext, 110
- StrokeTextOnPath
  - VectSharp.Graphics, 88
- SubsetFont
  - VectSharp.TrueTypeFile, 145
- SubsetFonts
  - VectSharp.PDF.PDFContextInterpreter, 120
  - VectSharp.SVG.SVGContextInterpreter, 135
- Symbol
  - VectSharp.FontFamily, 74
- Tag
  - VectSharp.Canvas.RenderAction, 129
  - VectSharp.IGraphicsContext, 112
- Tan
  - VectSharp.Colours, 63
- Teal
  - VectSharp.Colours, 63
- Text
  - VectSharp.Canvas.RenderAction, 126, 129
- TextAction
  - VectSharp.Canvas.RenderAction, 127
- TextAnchors
  - VectSharp, 13
- TextBaseline
  - VectSharp.IGraphicsContext, 112
- TextBaselines
  - VectSharp, 13
- TextOptions
  - VectSharp.Canvas.AvaloniaContextInterpreter, 16
  - VectSharp.PDF.PDFContextInterpreter, 119
  - VectSharp.SVG.SVGContextInterpreter, 135
- Thistle
  - VectSharp.Colours, 63
- TimesBold
  - VectSharp.FontFamily, 74
- TimesBoldItalic
  - VectSharp.FontFamily, 74
- TimesItalic
  - VectSharp.FontFamily, 74
- TimesRoman
  - VectSharp.FontFamily, 74
- ToCSSString
  - VectSharp.Colour, 26
- Tomato
  - VectSharp.Colours, 64
- Top
  - VectSharp, 13
  - VectSharp.Font.DetailedFontMetrics, 67
- Transform
  - VectSharp.Canvas.RenderAction, 129
  - VectSharp.Graphics, 89
  - VectSharp.IGraphicsContext, 110
- Translate
  - VectSharp.Graphics, 89
  - VectSharp.IGraphicsContext, 110



- TrueTypeFile
  - VectSharp.FontFamily, 77
- Turquoise
  - VectSharp.Colours, 64
- Type
  - VectSharp.Segment, 133
- UnitsOff
  - VectSharp.LineDash, 114
- UnitsOn
  - VectSharp.LineDash, 115
- VectSharp, 11
  - Arc, 13
  - Baseline, 13
  - Bevel, 12
  - Bottom, 13
  - Butt, 12
  - Center, 13
  - Close, 13
  - CubicBezier, 13
  - Left, 13
  - Line, 13
  - LineCaps, 12
  - LineJoins, 12
  - Middle, 13
  - Miter, 12
  - Move, 13
  - Right, 13
  - Round, 12
  - SegmentType, 13
  - Square, 12
  - TextAnchors, 13
  - TextBaselines, 13
  - Top, 13
- VectSharp.Canvas, 14
- VectSharp.Canvas.AvaloniaContextInterpreter, 15
  - AlwaysConvert, 17
  - ConvertIfNecessary, 17
  - NeverConvert, 17
  - PaintToCanvas, 17–19
  - TextOptions, 16
- VectSharp.Canvas.RenderAction, 124
  - ActionType, 127
  - ActionTypes, 126
  - BringToFront, 126
  - Fill, 128
  - Geometry, 128
  - InverseTransform, 128
  - Parent, 128
  - Path, 126
  - PathAction, 126
  - PointerEnter, 129
  - PointerLeave, 129
  - PointerPressed, 130
  - PointerReleased, 130
  - SendToBack, 127
  - Stroke, 128
  - Tag, 129
  - Text, 126, 129
  - TextAction, 127
  - Transform, 129
- VectSharp.Colour, 20
  - A, 29
  - B, 29
  - FromCSSString, 22
  - FromRgb, 22, 23
  - FromRgba, 24–26
  - G, 29
  - R, 29
  - ToCSSString, 26
  - WithAlpha, 27, 28
- VectSharp.Colours, 30
  - AliceBlue, 36
  - AntiqueWhite, 36
  - Aqua, 36
  - Aquamarine, 36
  - Azure, 36
  - Beige, 37
  - Bisque, 37
  - Black, 37
  - BlanchedAlmond, 37
  - Blue, 37
  - BlueViolet, 38
  - Brown, 38
  - BurlyWood, 38
  - CadetBlue, 38
  - Chartreuse, 38
  - Chocolate, 39
  - Coral, 39
  - CornflowerBlue, 39
  - Cornsilk, 39
  - Crimson, 39
  - Cyan, 40
  - DarkBlue, 40
  - DarkCyan, 40
  - DarkGoldenRod, 40
  - DarkGray, 40
  - DarkGreen, 41
  - DarkGrey, 41
  - DarkKhaki, 41
  - DarkMagenta, 41
  - DarkOliveGreen, 41
  - DarkOrange, 42
  - DarkOrchid, 42
  - DarkRed, 42
  - DarkSalmon, 42
  - DarkSeaGreen, 42
  - DarkSlateBlue, 43
  - DarkSlateGray, 43
  - DarkSlateGrey, 43
  - DarkTurquoise, 43
  - DarkViolet, 43
  - DeepPink, 44
  - DeepSkyBlue, 44
  - DimGray, 44
  - DimGrey, 44

- DodgerBlue, [44](#)
- FireBrick, [45](#)
- FloralWhite, [45](#)
- ForestGreen, [45](#)
- Fuchsia, [45](#)
- Gainsboro, [45](#)
- GhostWhite, [46](#)
- Gold, [46](#)
- GoldenRod, [46](#)
- Gray, [46](#)
- Green, [46](#)
- GreenYellow, [47](#)
- Grey, [47](#)
- HoneyDew, [47](#)
- HotPink, [47](#)
- IndianRed, [47](#)
- Indigo, [48](#)
- Ivory, [48](#)
- Khaki, [48](#)
- Lavender, [48](#)
- LavenderBlush, [48](#)
- LawnGreen, [49](#)
- LemonChiffon, [49](#)
- LightBlue, [49](#)
- LightCoral, [49](#)
- LightCyan, [49](#)
- LightGoldenRodYellow, [50](#)
- LightGray, [50](#)
- LightGreen, [50](#)
- LightGrey, [50](#)
- LightPink, [50](#)
- LightSalmon, [51](#)
- LightSeaGreen, [51](#)
- LightSkyBlue, [51](#)
- LightSlateGray, [51](#)
- LightSlateGrey, [51](#)
- LightSteelBlue, [52](#)
- LightYellow, [52](#)
- Lime, [52](#)
- LimeGreen, [52](#)
- Linen, [52](#)
- Magenta, [53](#)
- Maroon, [53](#)
- MediumAquaMarine, [53](#)
- MediumBlue, [53](#)
- MediumOrchid, [53](#)
- MediumPurple, [54](#)
- MediumSeaGreen, [54](#)
- MediumSlateBlue, [54](#)
- MediumSpringGreen, [54](#)
- MediumTurquoise, [54](#)
- MediumVioletRed, [55](#)
- MidnightBlue, [55](#)
- MintCream, [55](#)
- MistyRose, [55](#)
- Moccasin, [55](#)
- NavajoWhite, [56](#)
- Navy, [56](#)
- OldLace, [56](#)
- Olive, [56](#)
- OliveDrab, [56](#)
- Orange, [57](#)
- OrangeRed, [57](#)
- Orchid, [57](#)
- PaleGoldenRod, [57](#)
- PaleGreen, [57](#)
- PaleTurquoise, [58](#)
- PaleVioletRed, [58](#)
- PapayaWhip, [58](#)
- PeachPuff, [58](#)
- Peru, [58](#)
- Pink, [59](#)
- Plum, [59](#)
- PowderBlue, [59](#)
- Purple, [59](#)
- RebeccaPurple, [59](#)
- Red, [60](#)
- RosyBrown, [60](#)
- RoyalBlue, [60](#)
- SaddleBrown, [60](#)
- Salmon, [60](#)
- SandyBrown, [61](#)
- SeaGreen, [61](#)
- SeaShell, [61](#)
- Sienna, [61](#)
- Silver, [61](#)
- SkyBlue, [62](#)
- SlateBlue, [62](#)
- SlateGray, [62](#)
- SlateGrey, [62](#)
- Snow, [62](#)
- SpringGreen, [63](#)
- SteelBlue, [63](#)
- Tan, [63](#)
- Teal, [63](#)
- Thistle, [63](#)
- Tomato, [64](#)
- Turquoise, [64](#)
- Violet, [64](#)
- Wheat, [64](#)
- White, [64](#)
- WhiteSmoke, [65](#)
- Yellow, [65](#)
- YellowGreen, [65](#)
- VectSharp.Document, [67](#)
- Document, [68](#)
- Pages, [68](#)
- VectSharp.Font, [69](#)
- Ascent, [71](#)
- Descent, [71](#)
- Font, [69](#)
- FontFamily, [71](#)
- FontSize, [71](#)
- MeasureText, [70](#)
- MeasureTextAdvanced, [70](#)
- YMax, [71](#)

- YMin, [72](#)
- VectSharp.Font.DetailedFontMetrics, [65](#)
  - Bottom, [66](#)
  - Height, [66](#)
  - LeftSideBearing, [66](#)
  - RightSideBearing, [67](#)
  - Top, [67](#)
  - Width, [67](#)
- VectSharp.FontFamily, [72](#)
  - Courier, [74](#)
  - CourierBold, [74](#)
  - CourierBoldOblique, [74](#)
  - CourierOblique, [74](#)
  - FileName, [76](#)
  - FontFamily, [74](#), [75](#)
  - Helvetica, [74](#)
  - HelveticaBold, [74](#)
  - HelveticaBoldOblique, [74](#)
  - HelveticaOblique, [74](#)
  - IsBold, [76](#)
  - IsItalic, [76](#)
  - IsOblique, [76](#)
  - IsStandardFamily, [77](#)
  - StandardFamilies, [75](#)
  - StandardFontFamilies, [73](#)
  - StandardFontFamilyResources, [75](#)
  - Symbol, [74](#)
  - TimesBold, [74](#)
  - TimesBoldItalic, [74](#)
  - TimesItalic, [74](#)
  - TimesRoman, [74](#)
  - TrueTypeFile, [77](#)
  - ZapfDingbats, [74](#)
- VectSharp.Graphics, [77](#)
  - CopyToIGraphicsContext, [79](#)
  - DrawGraphics, [79](#), [80](#)
  - FillPath, [80](#)
  - FillRectangle, [80](#), [81](#)
  - FillText, [81](#), [82](#)
  - FillTextOnPath, [82](#)
  - MeasureText, [83](#)
  - Restore, [83](#)
  - Rotate, [84](#)
  - RotateAt, [84](#)
  - Save, [84](#)
  - Scale, [84](#)
  - StrokePath, [85](#)
  - StrokeRectangle, [85](#), [86](#)
  - StrokeText, [87](#)
  - StrokeTextOnPath, [88](#)
  - Transform, [89](#)
  - Translate, [89](#)
- VectSharp.GraphicsPath, [91](#)
  - AddSmoothSpline, [92](#)
  - AddText, [93](#)
  - AddTextOnPath, [94](#)
  - Arc, [94](#), [95](#)
  - Close, [95](#)
  - CubicBezierTo, [95](#), [96](#)
  - EllipticalArc, [97](#)
  - GetNormalAtAbsolute, [97](#)
  - GetNormalAtRelative, [97](#)
  - GetPointAtAbsolute, [98](#)
  - GetPointAtRelative, [98](#)
  - GetTangentAtAbsolute, [99](#)
  - GetTangentAtRelative, [99](#)
  - LineTo, [99](#), [101](#)
  - MeasureLength, [101](#)
  - MoveTo, [101](#), [102](#)
  - Segments, [102](#)
- VectSharp.IGraphicsContext, [103](#)
  - Close, [104](#)
  - CubicBezierTo, [104](#)
  - Fill, [105](#)
  - FillStyle, [111](#)
  - FillText, [105](#)
  - Font, [111](#)
  - Height, [111](#)
  - LineCap, [111](#)
  - LineJoin, [112](#)
  - LineTo, [105](#)
  - LineWidth, [112](#)
  - MoveTo, [106](#)
  - Rectangle, [106](#)
  - Restore, [106](#)
  - Rotate, [106](#)
  - Save, [107](#)
  - Scale, [107](#)
  - SetFillStyle, [107](#)
  - SetLineDash, [109](#)
  - SetStrokeStyle, [109](#)
  - Stroke, [109](#)
  - StrokeStyle, [112](#)
  - StrokeText, [110](#)
  - Tag, [112](#)
  - TextBaseline, [112](#)
  - Transform, [110](#)
  - Translate, [110](#)
  - Width, [113](#)
- VectSharp.LineDash, [113](#)
  - LineDash, [114](#)
  - Phase, [114](#)
  - SolidLine, [114](#)
  - UnitsOff, [114](#)
  - UnitsOn, [115](#)
- VectSharp.Page, [115](#)
  - Background, [116](#)
  - Crop, [116](#)
  - Graphics, [117](#)
  - Height, [117](#)
  - Page, [116](#)
  - Width, [117](#)
- VectSharp.PDF, [14](#)
- VectSharp.PDF.PDFContextInterpreter, [119](#)
  - ConvertIntoPaths, [120](#)
  - SaveAsPDF, [120](#)

- SubsetFonts, [120](#)
- TextOptions, [119](#)
- VectSharp.Point, [121](#)
  - Modulus, [122](#)
  - Normalize, [122](#)
  - Point, [121](#)
  - X, [122](#)
  - Y, [122](#)
- VectSharp.Raster, [14](#)
- VectSharp.Raster.RasterContextInterpreter, [123](#)
  - SaveAsPNG, [123](#), [124](#)
- VectSharp.Segment, [130](#)
  - Clone, [131](#)
  - GetPointAt, [131](#)
  - GetTangentAt, [132](#)
  - Measure, [132](#)
  - Point, [132](#)
  - Points, [133](#)
  - Type, [133](#)
- VectSharp.Size, [133](#)
  - Height, [134](#)
  - Size, [134](#)
  - Width, [134](#)
- VectSharp.SVG, [14](#)
- VectSharp.SVG.Parser, [117](#)
  - FromFile, [118](#)
  - FromStream, [118](#)
  - FromString, [118](#)
- VectSharp.SVG.SVGContextInterpreter, [135](#)
  - ConvertIntoPaths, [135](#)
  - DoNotEmbed, [135](#)
  - EmbedFonts, [135](#)
  - SaveAsSVG, [136](#)
  - SubsetFonts, [135](#)
  - TextOptions, [135](#)
- VectSharp.TrueTypeFile, [136](#)
  - Destroy, [138](#)
  - FontStream, [146](#)
  - Get1000EmAscent, [138](#)
  - Get1000EmDescent, [139](#)
  - Get1000EmGlyphBearings, [139](#)
  - Get1000EmGlyphVerticalMetrics, [139](#)
  - Get1000EmGlyphWidth, [140](#)
  - Get1000EmXMax, [141](#)
  - Get1000EmXMin, [141](#)
  - Get1000EmYMax, [141](#)
  - Get1000EmYMin, [141](#)
  - GetFirstCharIndex, [142](#)
  - GetFontFamilyName, [142](#)
  - GetFontName, [142](#)
  - GetGlyphIndex, [142](#)
  - GetGlyphPath, [143](#)
  - GetLastCharIndex, [144](#)
  - IsBold, [144](#)
  - IsFixedPitch, [144](#)
  - IsItalic, [144](#)
  - IsOblique, [145](#)
  - IsScript, [145](#)
  - IsSerif, [145](#)
  - SubsetFont, [145](#)
- VectSharp.TrueTypeFile.Bearings, [19](#)
  - LeftSideBearing, [20](#)
  - RightSideBearing, [20](#)
- VectSharp.TrueTypeFile.TrueTypePoint, [146](#)
  - IsOnCurve, [147](#)
  - X, [147](#)
  - Y, [147](#)
- VectSharp.TrueTypeFile.VerticalMetrics, [147](#)
  - YMax, [148](#)
  - YMin, [148](#)
- Violet
  - VectSharp.Colours, [64](#)
- Wheat
  - VectSharp.Colours, [64](#)
- White
  - VectSharp.Colours, [64](#)
- WhiteSmoke
  - VectSharp.Colours, [65](#)
- Width
  - VectSharp.Font.DetailedFontMetrics, [67](#)
  - VectSharp.IGraphicsContext, [113](#)
  - VectSharp.Page, [117](#)
  - VectSharp.Size, [134](#)
- WithAlpha
  - VectSharp.Colour, [27](#), [28](#)
- X
  - VectSharp.Point, [122](#)
  - VectSharp.TrueTypeFile.TrueTypePoint, [147](#)
- Y
  - VectSharp.Point, [122](#)
  - VectSharp.TrueTypeFile.TrueTypePoint, [147](#)
- Yellow
  - VectSharp.Colours, [65](#)
- YellowGreen
  - VectSharp.Colours, [65](#)
- YMax
  - VectSharp.Font, [71](#)
  - VectSharp.TrueTypeFile.VerticalMetrics, [148](#)
- YMin
  - VectSharp.Font, [72](#)
  - VectSharp.TrueTypeFile.VerticalMetrics, [148](#)
- ZapfDingbats
  - VectSharp.FontFamily, [74](#)