VectSharp

1.4.2

Generated by Doxygen 1.8.18

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 `Page`s, drawing to the `Page`s' `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 `Page`s.
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):

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

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 66 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 87 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 1012 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 45 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 19 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 1455 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 1460 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

page	The Page to render.
graphicsAsControls	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).
taggedActions	A Dictionary<String, Delegate> containing the Actions that will be performed on items with the corresponding tag. If *graphicsAsControls* is true, the delegates should be voids that accept one parameter of type TextBlock or Path (depending on the tagged item), otherwise, they should accept one parameter of type RenderAction and return an IEnumerable<RenderAction> of the actions that will actually be performed.
removeTaggedActionsAfterExecution	Whether the Actions should be removed from *taggedActions* after their execution. Set to false if the same Action should be performed on multiple items with the same tag.
textOption	Defines whether text items should be converted into paths when drawing.

Returns

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

Definition at line 1523 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

page	The Page to render.
graphicsAsControls	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).
textOption	Defines whether text items should be converted into paths when drawing.

Returns

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

Definition at line 1499 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

page	The Page to render.
taggedActions	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).
removeTaggedActionsAfterExecution	Whether the Actions should be removed from *taggedActions* after their execution. Set to false if the same Action should be performed on multiple items with the same tag.
textOption	Defines whether text items should be converted into paths when drawing.

Returns

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

Definition at line 1546 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

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

Returns

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

Definition at line 1484 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 2098 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 2103 of file TrueType.cs.

6.2.2.2 RightSideBearing

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

The right-side bearing of the glyph.

Definition at line 2108 of file TrueType.cs.

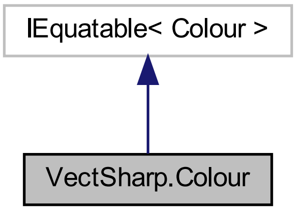
The documentation for this struct was generated from the following file:

- 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 147 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

cssString	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 347 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

r	The red component of the colour. Range: [0, 255].
g	The green component of the colour. Range: [0, 255].
b	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 196 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

r	The red component of the colour. Range: [0, 1].
g	The green component of the colour. Range: [0, 1].
b	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 184 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

r	The red component of the colour. Range: [0, 255].
g	The green component of the colour. Range: [0, 255].
b	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 208 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

colour	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 282 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

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

Returns

A ColourColour struct with the specified components.

Definition at line 234 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

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

Returns

A Colour struct with the specified components.

Definition at line 247 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

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

Returns

A Colour struct with the specified components.

Definition at line 221 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

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

Returns

A Colour struct with the specified components.

Definition at line 272 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

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

Returns

A Colour struct with the specified components.

Definition at line 259 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

includeAlpha	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 330 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

alpha	The alpha component of the new Colour.

Returns

A Colour struct with the same RGB components as the current Colour and the specified *alpha* .

Definition at line 483 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

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

Returns

A Colour struct with the same RGB components as the *original* Colour and the specified *alpha* .

Definition at line 463 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

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

Returns

A Colour struct with the same RGB components as the *original* Colour and the specified *alpha* .

Definition at line 452 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

alpha	The alpha component of the new Colour.

Returns

A Colour struct with the same RGB components as the current Colour and the specified *alpha* .

Definition at line 473 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 167 of file Graphics.cs.

6.3.3.2 B

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

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

Definition at line 162 of file Graphics.cs.

6.3.3.3 G

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

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

Definition at line 157 of file Graphics.cs.

6.3.3.4 R

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

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

Definition at line 152 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 BurlyWood = Colour.FromRgb(222, 184, 135)

  BurlyWood #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 165 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 582 of file StandardColours.cs.

6.4.2.2 AntiqueWhite

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

AntiqueWhite #FAEBD7

Definition at line 622 of file StandardColours.cs.

6.4.2.3 Aqua

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

Aqua #00FFFF

Definition at line 226 of file StandardColours.cs.

6.4.2.4 Aquamarine

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

Aquamarine #7FFFD4

Definition at line 358 of file StandardColours.cs.

6.4.2.5 Azure

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

Azure #F0FFFF

Definition at line 590 of file StandardColours.cs.

6.4.2.6 Beige

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

Beige #F5F5DC

Definition at line 602 of file StandardColours.cs.

6.4.2.7 Bisque

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

Bisque #FFE4C4

Definition at line 706 of file StandardColours.cs.

6.4.2.8 Black

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

Black #000000

Definition at line 170 of file StandardColours.cs.

6.4.2.9 BlanchedAlmond

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

BlanchedAlmond #FFEBCD

Definition at line 714 of file StandardColours.cs.

6.4.2.10 Blue

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

Blue #0000FF

Definition at line 186 of file StandardColours.cs.

6.4.2.11 BlueViolet

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

BlueViolet #8A2BE2

Definition at line 390 of file StandardColours.cs.

6.4.2.12 Brown

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

Brown #A52A2A

Definition at line 438 of file StandardColours.cs.

6.4.2.13 BurlyWood

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

BurlyWood #DEB887

Definition at line 550 of file StandardColours.cs.

6.4.2.14 CadetBlue

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

CadetBlue #5F9EA0

Definition at line 298 of file StandardColours.cs.

6.4.2.15 Chartreuse

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

Chartreuse #7FFF00

Definition at line 354 of file StandardColours.cs.

6.4.2.16 Chocolate

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

Chocolate #D2691E

Definition at line 506 of file StandardColours.cs.

6.4.2.17 Coral

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

Coral #FF7F50

Definition at line 666 of file StandardColours.cs.

6.4.2.18 CornflowerBlue

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

CornflowerBlue #6495ED

Definition at line 302 of file StandardColours.cs.

6.4.2.19 Cornsilk

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

Cornsilk #FFF8DC

Definition at line 730 of file StandardColours.cs.

6.4.2.20 Crimson

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

Crimson #DC143C

Definition at line 538 of file StandardColours.cs.

6.4.2.21 Cyan

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

Cyan #00FFFF

Definition at line 230 of file StandardColours.cs.

6.4.2.22 DarkBlue

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

DarkBlue #00008B

Definition at line 178 of file StandardColours.cs.

6.4.2.23 DarkCyan

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

DarkCyan #008B8B

Definition at line 202 of file StandardColours.cs.

6.4.2.24 DarkGoldenRod

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

DarkGoldenRod #B8860B

Definition at line 474 of file StandardColours.cs.

6.4.2.25 DarkGray

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

DarkGray #A9A9A9

Definition at line 442 of file StandardColours.cs.

6.4.2.26 DarkGreen

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

DarkGreen #006400

Definition at line 190 of file StandardColours.cs.

6.4.2.27 DarkGrey

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

DarkGrey #A9A9A9

Definition at line 446 of file StandardColours.cs.

6.4.2.28 DarkKhaki

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

DarkKhaki #BDB76B

Definition at line 486 of file StandardColours.cs.

6.4.2.29 DarkMagenta

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

DarkMagenta #8B008B

Definition at line 398 of file StandardColours.cs.

6.4.2.30 DarkOliveGreen

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

DarkOliveGreen #556B2F

Definition at line 294 of file StandardColours.cs.

6.4.2.31 DarkOrange

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

DarkOrange #FF8C00

Definition at line 670 of file StandardColours.cs.

6.4.2.32 DarkOrchid

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

DarkOrchid #9932CC

Definition at line 426 of file StandardColours.cs.

6.4.2.33 DarkRed

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

DarkRed #8B0000

Definition at line 394 of file StandardColours.cs.

6.4.2.34 DarkSalmon

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

DarkSalmon #E9967A

Definition at line 562 of file StandardColours.cs.

6.4.2.35 DarkSeaGreen

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

DarkSeaGreen #8FBC8F

Definition at line 406 of file StandardColours.cs.

6.4.2.36 DarkSlateBlue

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

DarkSlateBlue #483D8B

Definition at line 282 of file StandardColours.cs.

6.4.2.37 DarkSlateGray

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

DarkSlateGray #2F4F4F

Definition at line 254 of file StandardColours.cs.

6.4.2.38 DarkSlateGrey

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

DarkSlateGrey #2F4F4F

Definition at line 258 of file StandardColours.cs.

6.4.2.39 DarkTurquoise

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

DarkTurquoise #00CED1

Definition at line 210 of file StandardColours.cs.

6.4.2.40 DarkViolet

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

DarkViolet #9400D3

Definition at line 418 of file StandardColours.cs.

6.4.2.41 DeepPink

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

DeepPink #FF1493

Definition at line 650 of file StandardColours.cs.

6.4.2.42 DeepSkyBlue

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

DeepSkyBlue #00BFFF

Definition at line 206 of file StandardColours.cs.

6.4.2.43 DimGray

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

DimGray #696969

Definition at line 314 of file StandardColours.cs.

6.4.2.44 DimGrey

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

DimGrey #696969

Definition at line 318 of file StandardColours.cs.

6.4.2.45 DodgerBlue

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

DodgerBlue #1E90FF

Definition at line 238 of file StandardColours.cs.

6.4.2.46 FireBrick

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

FireBrick #B22222

Definition at line 470 of file StandardColours.cs.

6.4.2.47 FloralWhite

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

FloralWhite #FFFAF0

Definition at line 738 of file StandardColours.cs.

6.4.2.48 ForestGreen

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

ForestGreen #228B22

Definition at line 246 of file StandardColours.cs.

6.4.2.49 Fuchsia

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

Fuchsia #FF00FF

Definition at line 642 of file StandardColours.cs.

6.4.2.50 Gainsboro

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

Gainsboro #DCDCDC

Definition at line 542 of file StandardColours.cs.

6.4.2.51 GhostWhite

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

GhostWhite #F8F8FF

Definition at line 614 of file StandardColours.cs.

6.4.2.52 Gold

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

Gold #FFD700

Definition at line 690 of file StandardColours.cs.

6.4.2.53 GoldenRod

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

GoldenRod #DAA520

Definition at line 530 of file StandardColours.cs.

6.4.2.54 Gray

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

Gray #808080

Definition at line 374 of file StandardColours.cs.

6.4.2.55 Green

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

Green #008000

Definition at line 194 of file StandardColours.cs.

6.4.2.56 GreenYellow

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

GreenYellow #ADFF2F

Definition at line 454 of file StandardColours.cs.

6.4.2.57 Grey

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

Grey #808080

Definition at line 378 of file StandardColours.cs.

6.4.2.58 HoneyDew

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

HoneyDew #F0FFF0

Definition at line 586 of file StandardColours.cs.

6.4.2.59 HotPink

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

HotPink #FF69B4

Definition at line 662 of file StandardColours.cs.

6.4.2.60 IndianRed

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

IndianRed #CD5C5C

Definition at line 498 of file StandardColours.cs.

6.4.2.61 Indigo

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

Indigo #4B0082

Definition at line 290 of file StandardColours.cs.

6.4.2.62 Ivory

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

Ivory #FFFFF0

Definition at line 754 of file StandardColours.cs.

6.4.2.63 Khaki

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

Khaki #F0E68C

Definition at line 578 of file StandardColours.cs.

6.4.2.64 Lavender

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

Lavender #E6E6FA

Definition at line 558 of file StandardColours.cs.

6.4.2.65 LavenderBlush

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

LavenderBlush #FFF0F5

Definition at line 722 of file StandardColours.cs.

6.4.2.66 LawnGreen

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

LawnGreen #7CFC00

Definition at line 350 of file StandardColours.cs.

6.4.2.67 LemonChiffon

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

LemonChiffon #FFFACD

Definition at line 734 of file StandardColours.cs.

6.4.2.68 LightBlue

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

LightBlue #ADD8E6

Definition at line 450 of file StandardColours.cs.

6.4.2.69 LightCoral

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

LightCoral #F08080

Definition at line 574 of file StandardColours.cs.

6.4.2.70 LightCyan

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

LightCyan #E0FFFF

Definition at line 554 of file StandardColours.cs.

6.4.2.71 LightGoldenRodYellow

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

LightGoldenRodYellow #FAFAD2

Definition at line 630 of file StandardColours.cs.

6.4.2.72 LightGray

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

LightGray #D3D3D3

Definition at line 514 of file StandardColours.cs.

6.4.2.73 LightGreen

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

LightGreen #90EE90

Definition at line 410 of file StandardColours.cs.

6.4.2.74 LightGrey

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

LightGrey #D3D3D3

Definition at line 518 of file StandardColours.cs.

6.4.2.75 LightPink

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

LightPink #FFB6C1

Definition at line 682 of file StandardColours.cs.

6.4.2.76 LightSalmon

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

LightSalmon #FFA07A

Definition at line 674 of file StandardColours.cs.

6.4.2.77 LightSeaGreen

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

LightSeaGreen #20B2AA

Definition at line 242 of file StandardColours.cs.

6.4.2.78 LightSkyBlue

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

LightSkyBlue #87CEFA

Definition at line 386 of file StandardColours.cs.

6.4.2.79 LightSlateGray

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

LightSlateGray #778899

Definition at line 338 of file StandardColours.cs.

6.4.2.80 LightSlateGrey

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

LightSlateGrey #778899

Definition at line 342 of file StandardColours.cs.

6.4.2.81 LightSteelBlue

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

LightSteelBlue #B0C4DE

Definition at line 462 of file StandardColours.cs.

6.4.2.82 LightYellow

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

LightYellow #FFFFE0

Definition at line 750 of file StandardColours.cs.

6.4.2.83 Lime

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

Lime #00FF00

Definition at line 218 of file StandardColours.cs.

6.4.2.84 LimeGreen

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

LimeGreen #32CD32

Definition at line 262 of file StandardColours.cs.

6.4.2.85 Linen

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

Linen #FAF0E6

Definition at line 626 of file StandardColours.cs.

6.4.2.86 Magenta

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

Magenta #FF00FF

Definition at line 646 of file StandardColours.cs.

6.4.2.87 Maroon

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

Maroon #800000

Definition at line 362 of file StandardColours.cs.

6.4.2.88 MediumAquaMarine

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

MediumAquaMarine #66CDAA

Definition at line 310 of file StandardColours.cs.

6.4.2.89 MediumBlue

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

MediumBlue #0000CD

Definition at line 182 of file StandardColours.cs.

6.4.2.90 MediumOrchid

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

MediumOrchid #BA55D3

Definition at line 478 of file StandardColours.cs.

6.4.2.91 MediumPurple

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

MediumPurple #9370DB

Definition at line 414 of file StandardColours.cs.

6.4.2.92 MediumSeaGreen

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

MediumSeaGreen #3CB371

Definition at line 266 of file StandardColours.cs.

6.4.2.93 MediumSlateBlue

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

MediumSlateBlue #7B68EE

Definition at line 346 of file StandardColours.cs.

6.4.2.94 MediumSpringGreen

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

MediumSpringGreen #00FA9A

Definition at line 214 of file StandardColours.cs.

6.4.2.95 MediumTurquoise

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

MediumTurquoise #48D1CC

Definition at line 286 of file StandardColours.cs.

6.4.2.96 MediumVioletRed

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

MediumVioletRed #C71585

Definition at line 494 of file StandardColours.cs.

6.4.2.97 MidnightBlue

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

MidnightBlue #191970

Definition at line 234 of file StandardColours.cs.

6.4.2.98 MintCream

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

MintCream #F5FFFA

Definition at line 610 of file StandardColours.cs.

6.4.2.99 MistyRose

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

MistyRose #FFE4E1

Definition at line 710 of file StandardColours.cs.

6.4.2.100 Moccasin

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

Moccasin #FFE4B5

Definition at line 702 of file StandardColours.cs.

6.4.2.101 NavajoWhite

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

NavajoWhite #FFDEAD

Definition at line 698 of file StandardColours.cs.

6.4.2.102 Navy

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

Navy #000080

Definition at line 174 of file StandardColours.cs.

6.4.2.103 OldLace

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

OldLace #FDF5E6

Definition at line 634 of file StandardColours.cs.

6.4.2.104 Olive

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

Olive #808000

Definition at line 370 of file StandardColours.cs.

6.4.2.105 OliveDrab

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

OliveDrab #6B8E23

Definition at line 326 of file StandardColours.cs.

6.4.2.106 Orange

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

Orange #FFA500

Definition at line 678 of file StandardColours.cs.

6.4.2.107 OrangeRed

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

OrangeRed #FF4500

Definition at line 654 of file StandardColours.cs.

6.4.2.108 Orchid

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

Orchid #DA70D6

Definition at line 526 of file StandardColours.cs.

6.4.2.109 PaleGoldenRod

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

PaleGoldenRod #EEE8AA

Definition at line 570 of file StandardColours.cs.

6.4.2.110 PaleGreen

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

PaleGreen #98FB98

Definition at line 422 of file StandardColours.cs.

6.4.2.111 PaleTurquoise

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

PaleTurquoise #AFEEEE

Definition at line 458 of file StandardColours.cs.

6.4.2.112 PaleVioletRed

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

PaleVioletRed #DB7093

Definition at line 534 of file StandardColours.cs.

6.4.2.113 PapayaWhip

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

PapayaWhip #FFEFD5

Definition at line 718 of file StandardColours.cs.

6.4.2.114 PeachPuff

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

PeachPuff #FFDAB9

Definition at line 694 of file StandardColours.cs.

6.4.2.115 Peru

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

Peru #CD853F

Definition at line 502 of file StandardColours.cs.

6.4.2.116 Pink

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

Pink #FFC0CB

Definition at line 686 of file StandardColours.cs.

6.4.2.117 Plum

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

Plum #DDA0DD

Definition at line 546 of file StandardColours.cs.

6.4.2.118 PowderBlue

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

PowderBlue #B0E0E6

Definition at line 466 of file StandardColours.cs.

6.4.2.119 Purple

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

Purple #800080

Definition at line 366 of file StandardColours.cs.

6.4.2.120 RebeccaPurple

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

RebeccaPurple #663399

Definition at line 306 of file StandardColours.cs.

6.4.2.121 Red

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

Red #FF0000

Definition at line 638 of file StandardColours.cs.

6.4.2.122 RosyBrown

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

RosyBrown #BC8F8F

Definition at line 482 of file StandardColours.cs.

6.4.2.123 RoyalBlue

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

RoyalBlue #4169E1

Definition at line 274 of file StandardColours.cs.

6.4.2.124 SaddleBrown

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

SaddleBrown #8B4513

Definition at line 402 of file StandardColours.cs.

6.4.2.125 Salmon

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

Salmon #FA8072

Definition at line 618 of file StandardColours.cs.

6.4.2.126 SandyBrown

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

SandyBrown #F4A460

Definition at line 594 of file StandardColours.cs.

6.4.2.127 SeaGreen

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

SeaGreen #2E8B57

Definition at line 250 of file StandardColours.cs.

6.4.2.128 SeaShell

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

SeaShell #FFF5EE

Definition at line 726 of file StandardColours.cs.

6.4.2.129 Sienna

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

Sienna #A0522D

Definition at line 434 of file StandardColours.cs.

6.4.2.130 Silver

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

Silver #C0C0C0

Definition at line 490 of file StandardColours.cs.

6.4.2.131 SkyBlue

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

SkyBlue #87CEEB

Definition at line 382 of file StandardColours.cs.

6.4.2.132 SlateBlue

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

SlateBlue #6A5ACD

Definition at line 322 of file StandardColours.cs.

6.4.2.133 SlateGray

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

SlateGray #708090

Definition at line 330 of file StandardColours.cs.

6.4.2.134 SlateGrey

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

SlateGrey #708090

Definition at line 334 of file StandardColours.cs.

6.4.2.135 Snow

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

Snow #FFFAFA

Definition at line 742 of file StandardColours.cs.

6.4.2.136 SpringGreen

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

SpringGreen #00FF7F

Definition at line 222 of file StandardColours.cs.

6.4.2.137 SteelBlue

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

SteelBlue #4682B4

Definition at line 278 of file StandardColours.cs.

6.4.2.138 Tan

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

Tan #D2B48C

Definition at line 510 of file StandardColours.cs.

6.4.2.139 Teal

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

Teal #008080

Definition at line 198 of file StandardColours.cs.

6.4.2.140 Thistle

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

Thistle #D8BFD8

Definition at line 522 of file StandardColours.cs.

6.4.2.141 Tomato

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

Tomato #FF6347

Definition at line 658 of file StandardColours.cs.

6.4.2.142 Turquoise

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

Turquoise #40E0D0

Definition at line 270 of file StandardColours.cs.

6.4.2.143 Violet

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

Violet #EE82EE

Definition at line 566 of file StandardColours.cs.

6.4.2.144 Wheat

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

Wheat #F5DEB3

Definition at line 598 of file StandardColours.cs.

6.4.2.145 White

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

White #FFFFFF

Definition at line 758 of file StandardColours.cs.

6.4.2.146 WhiteSmoke

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

WhiteSmoke #F5F5F5

Definition at line 606 of file StandardColours.cs.

6.4.2.147 Yellow

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

Yellow #FFFF00

Definition at line 746 of file StandardColours.cs.

6.4.2.148 YellowGreen

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

YellowGreen #9ACD32

Definition at line 430 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 >= 0.
- double Bottom `[get]`

  Depth of the deepest glyph in the string below the baseline. Always <= 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 497 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 <= 0.

Definition at line 527 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 507 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 512 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 517 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 >= 0.

Definition at line 522 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 502 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 10 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 21 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 15 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 >= 0.
- double Descent `[get]`

  Maximum depth below the baseline of the usual glyphs in the font (there may be glyphs deeper than this). Always <= 0.
- double YMax `[get]`

  Absolute maximum height over the baseline of the glyphs in the font. Always >= 0.
- double YMin `[get]`

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

6.7.1 Detailed Description

Represents a typeface with a specific size.

Definition at line 492 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

fontFamily	Base typeface. See FontFamily.
fontSize	The font size, in graphics units.

Definition at line 555 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

text	The string to measure.

Returns

A Size object representing the width and height of the text.

Definition at line 638 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

text	The string to measure.

Returns

A DetailedFontMetrics object representing the metrics of the text.

Definition at line 671 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 >= 0.

Definition at line 564 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 <= 0.

Definition at line 582 of file Graphics.cs.

6.7.4.3 FontFamily

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

Font typeface.

Definition at line 548 of file Graphics.cs.

6.7.4.4 FontSize

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

Font size, in graphics units.

Definition at line 543 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 >= 0.

Definition at line 600 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 <= 0.

Definition at line 618 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-Bold↩ Italic", "Helvetica", "Helvetica-Bold", "Helvetica-Oblique", "Helvetica-BoldOblique", "Courier", "Courier-Bold", "Courier-Oblique", "Courier-BoldOblique", "Symbol", "ZapfDingbats" }

  The names of the 14 standard families that are guaranteed to be displayed correctly.
- static string[ ] StandardFontFamilyResources

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

Properties

- bool IsStandardFamily `[get]`

  Whether this is one of the 14 standard font families or not.
- string FileName `[get]`

  Full path to the TrueType font file for this font family (or, if this is a standard font family, name of the font family).
- TrueTypeFile TrueTypeFile `[get]`

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

  See also

  VectSharp.TrueTypeFile

  .
- bool IsBold `[get]`

  Whether this font is bold or not. This is set based on the information included in the OS/2 table of the TrueType file.
- bool IsItalic `[get]`

  Whether this font is italic or oblique or not. This is set based on the information included in the OS/2 table of the TrueType file.
- bool IsOblique `[get]`

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

6.8.1 Detailed Description

Represents a typeface.

Definition at line 707 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 746 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

fileName	The full path to the TrueType font file for this font family or the name of a standard font family.

Definition at line 849 of file Graphics.cs.

6.8.3.2 FontFamily() [2/3]

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

Create a new FontFamily.

Parameters

ttfStream	A stream containing a file in TTF format.

Definition at line 898 of file Graphics.cs.

6.8.3.3 FontFamily() [3/3]

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

Create a new standard FontFamily.

Parameters

standardFontFamily	The standard font family.

Definition at line 914 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 725 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 730 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 822 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 833 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 838 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 843 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 741 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 828 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 line↩ Cap=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 [[a, c, e], [b, d, f], [0, 0, 1]].
- 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 line↩ Cap=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? line↩ Dash=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 text↩ Baseline=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? line↩ Dash=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 line↩ Join=LineJoins.Miter, LineDash? lineDash=null, string tag=null)

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

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

  Stroke a text string along a GraphicsPath.
- Size MeasureText (string text, Font font)

  Measure a text string. See also

See also

Font.MeasureText(string), Font.MeasureTextAdvanced(string)

and .

- void Save ()

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

  Restore the previous transform state (rotation, translation scale).
- void CopyToIGraphicsContext (IGraphicsContext destinationContext)

  Copy the current graphics to an instance of a class implementing IGraphicsContext.
- void DrawGraphics (Point origin, Graphics graphics)

  Draws a Graphics object on the current Graphics object.
- void DrawGraphics (double originX, double originY, Graphics graphics)

  Draws a Graphics object on the current Graphics object.

6.9.1 Detailed Description

Represents an abstract drawing surface.

Definition at line 1847 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

destinationContext	The IGraphicsContext on which the graphics are to be copied.

Definition at line 2310 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

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

Definition at line 2509 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

origin	The point at which to place the origin of *graphics* .
graphics	The Graphics object to draw on the current Graphics object.

Definition at line 2493 of file Graphics.cs.

6.9.2.4 FillPath()

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

Fill a GraphicsPath.

Parameters

path	The GraphicsPath to fill.
fillColour	The Colour with which to fill the GraphicsPath.
tag	A tag to identify the filled path.

Definition at line 1857 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

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

Definition at line 1965 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

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

Definition at line 1951 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

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

Definition at line 2028 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

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

Definition at line 2013 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

path	The GraphicsPath along which the text will flow.
text	The string to draw.
font	The font with which to draw the text.
fillColour	The colour to use to fill the text.
reference	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).
anchor	The anchor in the text string that will correspond to the point specified by the *reference* .
textBaseline	The text baseline (determines which the position of the text in relation to the *path* .
tag	A tag to identify the filled text.

Definition at line 2081 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

text	The string to measure.
font	The font to use to measure the string.

Returns

Definition at line 2285 of file Graphics.cs.

6.9.2.11 Restore()

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

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

Definition at line 2301 of file Graphics.cs.

6.9.2.12 Rotate()

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

Rotate the coordinate system around the origin.

Parameters

angle	The angle (in radians) by which to rotate the coordinate system.

Definition at line 1882 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

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

Definition at line 1892 of file Graphics.cs.

6.9.2.14 Save()

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

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

Definition at line 2293 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

scaleX	The horizontal scale.
scaleY	The vertical scale.

Definition at line 1939 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

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

Definition at line 1873 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

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

Definition at line 1999 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

topLeft	The top-left corner of the rectangle.
size	The size of the rectangle.
strokeColour	The colour with which to stroke the rectangle.
lineWidth	The width of the line with which the rectangle is stroked.
lineCap	The line cap to use to stroke the rectangle.
lineJoin	The line join to use to stroke the rectangle.
lineDash	The line dash to use to stroke the rectangle.
tag	A tag to identify the filled rectangle.

Definition at line 1981 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

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

Definition at line 2065 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

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

Definition at line 2046 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

path	The GraphicsPath along which the text will flow.
text	The string to draw.
font	The font with which to draw the text.
strokeColour	The colour with which to stroke the text.
lineWidth	The width of the line with which the text is stroked.
lineCap	The line cap to use to stroke the text.
lineJoin	The line join to use to stroke the text.
lineDash	The line dash to use to stroke the text.
reference	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).
anchor	The anchor in the text string that will correspond to the point specified by the *reference* .
textBaseline	The text baseline (determines which the position of the text in relation to the *path* .
tag	A tag to identify the stroked text.

Definition at line 2187 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

a	The first element of the first column.
b	The second element of the first column.
c	The first element of the second column.
d	The second element of the second column.
e	The first element of the third column.
f	The second element of the third column.

Definition at line 1909 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

x	The horizontal translation.
y	The vertical translation.

Definition at line 1920 of file Graphics.cs.

6.9.2.24 Translate() [2/2]

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

Translate the coordinate system origin.

Parameters

delta	The new origin point.

Definition at line 1929 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 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 .
- 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 textBaseline=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 2659 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

points	The points through which the spline should pass.

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 3110 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

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

///

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2908 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

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

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2921 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

path	The GraphicsPath along which the text will flow.
text	The string to draw.
font	The font with which to draw the text.
reference	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).
anchor	The anchor in the text string that will correspond to the point specified by the *reference* .
textBaseline	The text baseline (determines which the position of the text in relation to the *path* .

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2998 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

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

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2749 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

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

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2729 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 2893 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

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

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2883 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

control1	The first control point.
control2	The second control point.
endPoint	The destination point.

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2862 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

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

Returns

Definition at line 2765 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

length	The distance to the point from the start of the GraphicsPath.

Returns

The normal to the point at the specified position.

Definition at line 3809 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

position	The position on the GraphicsPath (0 is the start of the path, 1 is the end of the path).

Returns

The normal to the point at the specified position.

Definition at line 3820 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

length	The distance to the point from the start of the GraphicsPath.

Returns

The point at the specified position.

Definition at line 3225 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

position	The position on the GraphicsPath (0 is the start of the path, 1 is the end of the path).

Returns

The point at the specified position.

Definition at line 3215 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

length	The distance to the point from the start of the GraphicsPath.

Returns

The tangent to the point at the specified position.

Definition at line 3522 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

position	The position on the GraphicsPath (0 is the start of the path, 1 is the end of the path).

Returns

The tangent to the point at the specified position.

Definition at line 3512 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

x	The horizontal coordinate of the new point.
y	The vertical coordinate of the new point.

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2714 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

p	The new point.

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2695 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 3143 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

x	The horizontal coordinate of the new point.
y	The vertical coordinate of the new point.

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2684 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

p	The new point.

Returns

The GraphicsPath, to allow for chained calls.

Definition at line 2672 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 2664 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 [[a, c, e], [b, d, f], [0, 0, 1]].
- 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 1654 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

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

text	The string to draw.
x	The horizontal coordinate of the text origin.
y	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

x	The horizontal coordinate of the point.
y	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

x	The horizontal coordinate of the point.
y	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

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

angle	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

scaleX	The horizontal scale.
scaleY	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

style	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].

6.11.2.13 SetFillStyle() [2/2]

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

Set the current FillStyle.

Parameters

style	The new fill style.

6.11.2.14 SetLineDash()

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

Set the current line dash pattern.

Parameters

dash	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

style	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

style	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

text	The string to draw.
x	The horizontal coordinate of the text origin.
y	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

a	The first element of the first column.
b	The second element of the first column.
c	The first element of the second column.
d	The second element of the second column.
e	The first element of the third column.
f	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

x	The horizontal translation.
y	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 1761 of file Graphics.cs.

6.11.3.2 Font

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

The current font.

Definition at line 1710 of file Graphics.cs.

6.11.3.3 Height

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

Height of the graphic surface.

Definition at line 1664 of file Graphics.cs.

6.11.3.4 LineCap

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

Current line cap used to stroke paths.

Definition at line 1825 of file Graphics.cs.

6.11.3.5 LineJoin

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

Current line join used to stroke paths.

Definition at line 1830 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 1820 of file Graphics.cs.

6.11.3.7 StrokeStyle

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

Current colour used to stroke paths.

Definition at line 1778 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 1841 of file Graphics.cs.

6.11.3.9 TextBaseline

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

The current text baseline.

Definition at line 1716 of file Graphics.cs.

6.11.3.10 Width

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

Width of the graphic surface.

Definition at line 1659 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 108 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

unitsOn	The length of the "on" (painted) segment.
unitsOff	The length of the "off" (not painted) segment.
phase	The position in the dash pattern at which the line starts.

Definition at line 136 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 128 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 113 of file Graphics.cs.

6.12.3.3 UnitsOff

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

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

Definition at line 123 of file Graphics.cs.

6.12.3.4 UnitsOn

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

Length of the "on" (painted) segment.

Definition at line 118 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 30 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

width	The width of the page.
height	The height of the page.

Definition at line 57 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

topLeft	The top left corner of the area to include in the page.
size	The size of the area to include in the page.

Definition at line 71 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 50 of file Document.cs.

6.13.4.2 Graphics

```
Graphics VectSharp.Page.Graphics  [get], [set]
```

Graphics surface of the page.

Definition at line 45 of file Document.cs.

6.13.4.3 Height

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

Height of the page.

Definition at line 40 of file Document.cs.

6.13.4.4 Width

```
double VectSharp.Page.Width  [get], [set]
```

Width of the page.

Definition at line 35 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 17 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

fileName	The path to the SVG image file.

Returns

A Page containing the image represented by the file.

Definition at line 55 of file SVGParser.cs.

6.14.2.2 FromStream()

```
static Page VectSharp.SVG.Parser.FromStream (
             Stream svgSourceStream )  [static]
```

Parses an stream containing SVG source code into a Page containing the image represented by the code.

Parameters

svgSourceStream	The stream containing SVG source code.

Returns

A Page containing the image represented by the *svgSourceStream* .

Definition at line 65 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

svgSource	The SVG source code.

Returns

A Page containing the image represented by the *svgSource* .

Definition at line 24 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.SubsetFonts, 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.SubsetFonts, bool compressStreams=true)

  Save the document to a PDF file.
- static void SaveAsPDF (this Document document, Stream stream, TextOptions textOption=TextOptions.SubsetFonts, 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 492 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 subsetted font files containing only the glyphs for the characters that have been used.
ConvertIntoPaths	Does not embed any font file and converts all text items into paths.

Definition at line 660 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

document	The Document to save.
stream	The stream to which the PDF data will be written.
textOption	Defines whether the used fonts should be included in the file.
compressStreams	Indicates whether the streams in the PDF file should be compressed.

Definition at line 682 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

document	The Document to save.
fileName	The full path to the file to save. If it exists, it will be overwritten.
textOption	Defines whether the used fonts should be included in the file.
compressStreams	Indicates whether the streams in the PDF file should be compressed.

Definition at line 649 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 939 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

x	The horizontal (x) coordinate.
y	The vertical (y) coordinate.

Definition at line 956 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 966 of file Graphics.cs.

6.16.3.2 Normalize()

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

Normalises a Point.

Returns

The normalised Point.

Definition at line 975 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 944 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 949 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 976 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

page	The Page to render.
stream	The stream to which the PNG data will be written.
scale	The scale to be used when rasterising the page. This will determine the width and height of the image file.

Definition at line 999 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

page	The Page to render.
fileName	The full path to the file to save. If it exists, it will be overwritten.
scale	The scale to be used when rasterising the page. This will determine the width and height of the image file.

Definition at line 985 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 760 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 765 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 934 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

geometry	The geometry to be rendered.
stroke	The stroke of the path (can be null).
fill	The fill of the path (can be null).
transform	The transform that will be applied to the path.
tag	A tag to access the RenderAction. If this is null this RenderAction is not visible in the hit test.

Returns

A new RenderAction representing a Path.

Definition at line 897 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 942 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

text	The text to be rendered.
fill	The fill of the text (can be null).
transform	The transform that will be applied to the text.
tag	A tag to access the RenderAction. If this is null this RenderAction is not visible in the hit test.

Returns

Definition at line 918 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 781 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 801 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 786 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 809 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 834 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 796 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 827 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 791 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 814 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 845 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 850 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 855 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 860 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 1043 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

previousPoint	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
position	The relative position on the Segment (0 is the start of the Segment, 1 is the end of the Segment).

Returns

The point at the specified position.

6.19.2.3 GetTangentAt()

```
abstract Point VectSharp.Segment.GetTangentAt (
             Point previousPoint,
             double position )  [pure virtual]
```

Gets the tangent to the Segment at the specified (relative) *position*).

Parameters

previousPoint	The point from which the Segment starts (i.e. the endpoint of the previous Segment).
position	The relative position on the Segment (0 is the start of the Segment, 1 is the end of the Segment).

Returns

The tangent to the point at the specified position.

6.19.2.4 Measure()

```
abstract double VectSharp.Segment.Measure (
             Point previousPoint )  [pure virtual]
```

Computes the length of the Segment.

Parameters

previousPoint	The point from which the Segment starts (i.e. the endpoint of the previous Segment).

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 1059 of file Graphics.cs.

6.19.3.2 Points

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

The points used to define the Segment.

Definition at line 1054 of file Graphics.cs.

6.19.3.3 Type

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

The type of the Segment.

Definition at line 1049 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 985 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

width	The width of the object.
height	The height of the object.

Definition at line 1002 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 995 of file Graphics.cs.

6.20.3.2 Width

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

Width of the object.

Definition at line 990 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 632 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

EmbedFonts	Embeds the full font files.
SubsetFonts	Embeds subsetted font files containing only the glyphs for the characters that have been used.
ConvertIntoPaths	Does not embed any font file and converts all text items into paths.
DoNotEmbed	Does not embed any font file, but still encodes text items as such.

Definition at line 652 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

page	The Page to render.
stream	The stream to which the SVG data will be written.
textOption	Defines whether the used fonts should be included in the file.

Definition at line 681 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

page	The Page to render.
fileName	The full path to the file to save. If it exists, it will be overwritten.
textOption	Defines whether the used fonts should be included in the file.

Definition at line 641 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 heigth 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 13 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 35 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 2044 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 2054 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

glyph	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 2136 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

glyph	The glyph whose vertical metrics are to be computed.

Returns

The vertical metrics of a glyph, in thousandths of em unit.

Definition at line 2184 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

glyph	The glyph whose advance width is to be computed.

Returns

The advance width of the glyph in thousandths of em unit.

Definition at line 2015 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

glyphIndex	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 2033 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 2081 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 2090 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 2063 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 2072 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 1853 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 1806 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 1834 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

glyph	The character whose glyph is sought.

Returns

The index of the glyph in the TrueType file.

Definition at line 1943 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

glyph	The glyph whose path is sought.
size	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 2005 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

glyphIndex	The index of the glyph whose path is sought.
size	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 1994 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 1864 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 1898 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 1909 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 1876 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 1887 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 1931 of file TrueType.cs.

6.22.2.24 IsSerif()

```
bool VectSharp.TrueTypeFile.IsSerif ( )
```

Determines whether the typeface is serifed or not.

Returns

A bool indicating whether the typeface is serifed or not.

Definition at line 1920 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

charactersToInclude	A string containing the characters for which the glyphs should be included.
consolidateAt32	If true, the character map is rearranged so that the included glyphs start at the unicode U+0032 control point.
outputEncoding	If *consolidateAt32* 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 527 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 29 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 1320 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 1335 of file TrueType.cs.

6.23.2.2 X

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

The horizontal coordinate of the point.

Definition at line 1325 of file TrueType.cs.

6.23.2.3 Y

```
double VectSharp.TrueTypeFile.TrueTypePoint.Y
```

The vertical coordinate of the point.

Definition at line 1330 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 heigth 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 heigth above and depth below the baseline of a glyph.

Definition at line 2153 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 2163 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 2158 of file TrueType.cs.

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

- VectSharp/TrueType.cs

Index