Graphics Class

.NET Framework (current version)

Note

The .NET API Reference documentation has a new home. Visit the .NET API Browser on docs.microsoft.com to see the new experience.

Encapsulates a GDI+ drawing surface. This class cannot be inherited.

Namespace: System.Drawing
Assembly: System.Drawing (in System.Drawing.dll)

Inheritance Hierarchy

System.Object
  System.MarshalByRefObject
    System.Drawing.Graphics

Syntax

C#

```
public sealed class Graphics : MarshalByRefObject, IDisposable,
        IDeviceContext
```

Properties

	Name	Description
	Clip	Gets or sets a Region that limits the drawing region of this Graphics.
	ClipBounds	Gets a RectangleF structure that bounds the clipping region of this Graphics.
	CompositingMode	Gets a value that specifies how composited images are drawn to this Graphics.
	CompositingQuality	Gets or sets the rendering quality of composited images drawn to this Graphics.

	DpiX	Gets the horizontal resolution of this Graphics.
	DpiY	Gets the vertical resolution of this Graphics.
	InterpolationMode	Gets or sets the interpolation mode associated with this Graphics.
	IsClipEmpty	Gets a value indicating whether the clipping region of this Graphics is empty.
	IsVisibleClipEmpty	Gets a value indicating whether the visible clipping region of this Graphics is empty.
	PageScale	Gets or sets the scaling between world units and page units for this Graphics.
	PageUnit	Gets or sets the unit of measure used for page coordinates in this Graphics.
	PixelOffsetMode	Gets or set a value specifying how pixels are offset during rendering of this Graphics.
	RenderingOrigin	Gets or sets the rendering origin of this Graphics for dithering and for hatch brushes.
	SmoothingMode	Gets or sets the rendering quality for this Graphics.
	TextContrast	Gets or sets the gamma correction value for rendering text.
	TextRenderingHint	Gets or sets the rendering mode for text associated with this Graphics.
	Transform	Gets or sets a copy of the geometric world transformation for this Graphics.
	VisibleClipBounds	Gets the bounding rectangle of the visible clipping region of this Graphics.

Methods

	Name	Description
	AddMetafileComment(Byte[])	Adds a comment to the current Metafile.
	BeginContainer()	Saves a graphics container with the current state of this Graphics and opens and uses a new graphics container.

◆	BeginContainer(Rectangle, Rectangle, GraphicsUnit)	Saves a graphics container with the current state of this Graphics and opens and uses a new graphics container with the specified scale transformation.
◆	BeginContainer(RectangleF, RectangleF, GraphicsUnit)	Saves a graphics container with the current state of this Graphics and opens and uses a new graphics container with the specified scale transformation.
◆	Clear(Color)	Clears the entire drawing surface and fills it with the specified background color.
◆	CopyFromScreen(Int32, Int32, Int32, Int32, Size)	Performs a bit-block transfer of the color data, corresponding to a rectangle of pixels, from the screen to the drawing surface of the Graphics.
◆	CopyFromScreen(Int32, Int32, Int32, Int32, Size, CopyPixelOperation)	Performs a bit-block transfer of the color data, corresponding to a rectangle of pixels, from the screen to the drawing surface of the Graphics.
◆	CopyFromScreen(Point, Point, Size)	Performs a bit-block transfer of color data, corresponding to a rectangle of pixels, from the screen to the drawing surface of the Graphics.
◆	CopyFromScreen(Point, Point, Size, CopyPixelOperation)	Performs a bit-block transfer of color data, corresponding to a rectangle of pixels, from the screen to the drawing surface of the Graphics.
◆	CreateObjRef(Type)	Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object.(Inherited from MarshalByRefObject.)
◆	Dispose()	Releases all resources used by this Graphics.
◆	DrawArc(Pen, Int32, Int32, Int32, Int32, Int32, Int32)	Draws an arc representing a portion of an ellipse specified by a pair of coordinates, a width, and a height.
◆	DrawArc(Pen, Rectangle, Single, Single)	Draws an arc representing a portion of an ellipse specified by a Rectangle structure.
◆	DrawArc(Pen, RectangleF, Single, Single)	Draws an arc representing a portion of an ellipse specified by a RectangleF structure.
◆	DrawArc(Pen, Single, Single, Single, Single, Single, Single)	Draws an arc representing a portion of an ellipse specified by a pair of coordinates, a width, and a height.
◆	DrawBezier(Pen, Point, Point, Point, Point)	Draws a Bézier spline defined by four Point structures.
◆	DrawBezier(Pen, PointF, PointF,	Draws a Bézier spline defined by four PointF

	PointF, PointF)	structures.
◆	DrawBezier(Pen, Single, Single, Single, Single, Single, Single, Single, Single)	Draws a Bézier spline defined by four ordered pairs of coordinates that represent points.
◆	DrawBeziers(Pen, Point[])	Draws a series of Bézier splines from an array of Point structures.
◆	DrawBeziers(Pen, PointF[])	Draws a series of Bézier splines from an array of PointF structures.
◆	DrawClosedCurve(Pen, Point[])	Draws a closed cardinal spline defined by an array of Point structures.
◆	DrawClosedCurve(Pen, Point[], Single, FillMode)	Draws a closed cardinal spline defined by an array of Point structures using a specified tension.
◆	DrawClosedCurve(Pen, PointF[])	Draws a closed cardinal spline defined by an array of PointF structures.
◆	DrawClosedCurve(Pen, PointF[], Single, FillMode)	Draws a closed cardinal spline defined by an array of PointF structures using a specified tension.
◆	DrawCurve(Pen, Point[])	Draws a cardinal spline through a specified array of Point structures.
◆	DrawCurve(Pen, Point[], Int32, Int32, Single)	Draws a cardinal spline through a specified array of Point structures using a specified tension.
◆	DrawCurve(Pen, Point[], Single)	Draws a cardinal spline through a specified array of Point structures using a specified tension.
◆	DrawCurve(Pen, PointF[])	Draws a cardinal spline through a specified array of PointF structures.
◆	DrawCurve(Pen, PointF[], Int32, Int32)	Draws a cardinal spline through a specified array of PointF structures. The drawing begins offset from the beginning of the array.
◆	DrawCurve(Pen, PointF[], Int32, Int32, Single)	Draws a cardinal spline through a specified array of PointF structures using a specified tension. The drawing begins offset from the beginning of the array.
◆	DrawCurve(Pen, PointF[], Single)	Draws a cardinal spline through a specified array of PointF structures using a specified tension.
◆	DrawEllipse(Pen, Int32, Int32, Int32, Int32)	Draws an ellipse defined by a bounding rectangle specified by coordinates for the upper-left corner of the rectangle, a height, and a width.

◆	DrawEllipse(Pen, Rectangle)	Draws an ellipse specified by a bounding Rectangle structure.
◆	DrawEllipse(Pen, RectangleF)	Draws an ellipse defined by a bounding RectangleF.
◆	DrawEllipse(Pen, Single, Single, Single, Single)	Draws an ellipse defined by a bounding rectangle specified by a pair of coordinates, a height, and a width.
◆	DrawIcon(Icon, Int32, Int32)	Draws the image represented by the specified Icon at the specified coordinates.
◆	DrawIcon(Icon, Rectangle)	Draws the image represented by the specified Icon within the area specified by a Rectangle structure.
◆	DrawIconUnstretched(Icon, Rectangle)	Draws the image represented by the specified Icon without scaling the image.
◆	DrawImage(Image, Int32, Int32)	Draws the specified image, using its original physical size, at the location specified by a coordinate pair.
◆	DrawImage(Image, Int32, Int32, Int32, Int32)	Draws the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Int32, Int32, Rectangle, GraphicsUnit)	Draws a portion of an image at a specified location.
◆	DrawImage(Image, Point)	Draws the specified Image, using its original physical size, at the specified location.
◆	DrawImage(Image, Point[])	Draws the specified Image at the specified location and with the specified shape and size.
◆	DrawImage(Image, Point[], Rectangle, GraphicsUnit)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Point[], Rectangle, GraphicsUnit, ImageAttributes)	Draws the specified portion of the specified Image at the specified location.
◆	DrawImage(Image, Point[], Rectangle, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Point[], Rectangle, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort, Int32)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, PointF)	Draws the specified Image, using its original physical size, at the specified location.
◆	DrawImage(Image, PointF[])	Draws the specified Image at the specified location and with the specified shape and size.

◆	DrawImage(Image, PointF[], RectangleF, GraphicsUnit)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, PointF[], RectangleF, GraphicsUnit, ImageAttributes)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, PointF[], RectangleF, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, PointF[], RectangleF, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort, Int32)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle)	Draws the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Int32, Int32, Int32, Int32, GraphicsUnit)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Int32, Int32, Int32, Int32, GraphicsUnit, ImageAttributes)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Int32, Int32, Int32, Int32, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Int32, Int32, Int32, Int32, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort, IntPtr)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Rectangle, GraphicsUnit)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Single, Single, Single, Single, GraphicsUnit)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Single, Single, Single, Single, GraphicsUnit, ImageAttributes)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Single, Single, Single, Single, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Rectangle, Single,	Draws the specified portion of the specified Image at

	Single, Single, Single, GraphicsUnit, ImageAttributes, Graphics.DrawImageAbort, IntPtr)	the specified location and with the specified size.
◆	DrawImage(Image, RectangleF)	Draws the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, RectangleF, RectangleF, GraphicsUnit)	Draws the specified portion of the specified Image at the specified location and with the specified size.
◆	DrawImage(Image, Single, Single)	Draws the specified Image, using its original physical size, at the specified location.
◆	DrawImage(Image, Single, Single, RectangleF, GraphicsUnit)	Draws a portion of an image at a specified location.
◆	DrawImage(Image, Single, Single, Single, Single)	Draws the specified Image at the specified location and with the specified size.
◆	DrawImageUnscaled(Image, Int32, Int32)	Draws the specified image using its original physical size at the location specified by a coordinate pair.
◆	DrawImageUnscaled(Image, Int32, Int32, Int32, Int32)	Draws a specified image using its original physical size at a specified location.
◆	DrawImageUnscaled(Image, Point)	Draws a specified image using its original physical size at a specified location.
◆	DrawImageUnscaled(Image, Rectangle)	Draws a specified image using its original physical size at a specified location.
◆	DrawImageUnscaledAndClipped(Image, Rectangle)	Draws the specified image without scaling and clips it, if necessary, to fit in the specified rectangle.
◆	DrawLine(Pen, Int32, Int32, Int32, Int32)	Draws a line connecting the two points specified by the coordinate pairs.
◆	DrawLine(Pen, Point, Point)	Draws a line connecting two Point structures.
◆	DrawLine(Pen, PointF, PointF)	Draws a line connecting two PointF structures.
◆	DrawLine(Pen, Single, Single, Single, Single)	Draws a line connecting the two points specified by the coordinate pairs.
◆	DrawLines(Pen, Point[])	Draws a series of line segments that connect an array of Point structures.
◆	DrawLines(Pen, PointF[])	Draws a series of line segments that connect an array of PointF structures.
◆	DrawPath(Pen, GraphicsPath)	Draws a GraphicsPath.

◆	DrawPie(Pen, Int32, Int32, Int32, Int32, Int32, Int32)	Draws a pie shape defined by an ellipse specified by a coordinate pair, a width, a height, and two radial lines.
◆	DrawPie(Pen, Rectangle, Single, Single)	Draws a pie shape defined by an ellipse specified by a Rectangle structure and two radial lines.
◆	DrawPie(Pen, RectangleF, Single, Single)	Draws a pie shape defined by an ellipse specified by a RectangleF structure and two radial lines.
◆	DrawPie(Pen, Single, Single, Single, Single, Single, Single)	Draws a pie shape defined by an ellipse specified by a coordinate pair, a width, a height, and two radial lines.
◆	DrawPolygon(Pen, Point[])	Draws a polygon defined by an array of Point structures.
◆	DrawPolygon(Pen, PointF[])	Draws a polygon defined by an array of PointF structures.
◆	DrawRectangle(Pen, Int32, Int32, Int32, Int32)	Draws a rectangle specified by a coordinate pair, a width, and a height.
◆	DrawRectangle(Pen, Rectangle)	Draws a rectangle specified by a Rectangle structure.
◆	DrawRectangle(Pen, Single, Single, Single, Single)	Draws a rectangle specified by a coordinate pair, a width, and a height.
◆	DrawRectangles(Pen, Rectangle[])	Draws a series of rectangles specified by Rectangle structures.
◆	DrawRectangles(Pen, RectangleF[])	Draws a series of rectangles specified by RectangleF structures.
◆	DrawString(String, Font, Brush, PointF)	Draws the specified text string at the specified location with the specified Brush and Font objects.
◆	DrawString(String, Font, Brush, PointF, StringFormat)	Draws the specified text string at the specified location with the specified Brush and Font objects using the formatting attributes of the specified StringFormat.
◆	DrawString(String, Font, Brush, RectangleF)	Draws the specified text string in the specified rectangle with the specified Brush and Font objects.
◆	DrawString(String, Font, Brush, RectangleF, StringFormat)	Draws the specified text string in the specified rectangle with the specified Brush and Font objects using the formatting attributes of the specified StringFormat.
◆	DrawString(String, Font, Brush,	Draws the specified text string at the specified

	Single, Single)	location with the specified Brush and Font objects.
	DrawString(String, Font, Brush, Single, Single, StringFormat)	Draws the specified text string at the specified location with the specified Brush and Font objects using the formatting attributes of the specified StringFormat.
	EndContainer(GraphicsContainer)	Closes the current graphics container and restores the state of this Graphics to the state saved by a call to the BeginContainer method.
	EnumerateMetafile(Metafile, Point, Graphics.EnumerateMetafileProc)	Sends the records in the specified Metafile, one at a time, to a callback method for display at a specified point.
	EnumerateMetafile(Metafile, Point, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in the specified Metafile, one at a time, to a callback method for display at a specified point.
	EnumerateMetafile(Metafile, Point, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in the specified Metafile, one at a time, to a callback method for display at a specified point using specified image attributes.
	EnumerateMetafile(Metafile, Point, Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display at a specified point.
	EnumerateMetafile(Metafile, Point, Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display at a specified point.
	EnumerateMetafile(Metafile, Point, Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display at a specified point using specified image attributes.
	EnumerateMetafile(Metafile, Point[], Graphics.EnumerateMetafileProc)	Sends the records in the specified Metafile, one at a time, to a callback method for display in a specified parallelogram.
	EnumerateMetafile(Metafile, Point[], Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in the specified Metafile, one at a time, to a callback method for display in a specified parallelogram.
	EnumerateMetafile(Metafile, Point[], Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in the specified Metafile, one at a time, to a callback method for display in a specified parallelogram using specified image attributes.
	EnumerateMetafile(Metafile, Point[], Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified parallelogram.

◆	EnumerateMetafile(Metafile, Point[], Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified parallelogram.
◆	EnumerateMetafile(Metafile, Point[], Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified parallelogram using specified image attributes.
◆	EnumerateMetafile(Metafile, PointF, Graphics.EnumerateMetafileProc)	Sends the records in the specified Metafile, one at a time, to a callback method for display at a specified point.
◆	EnumerateMetafile(Metafile, PointF, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in the specified Metafile, one at a time, to a callback method for display at a specified point.
◆	EnumerateMetafile(Metafile, PointF, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in the specified Metafile, one at a time, to a callback method for display at a specified point using specified image attributes.
◆	EnumerateMetafile(Metafile, PointF, RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display at a specified point.
◆	EnumerateMetafile(Metafile, PointF, RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display at a specified point.
◆	EnumerateMetafile(Metafile, PointF, RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display at a specified point using specified image attributes.
◆	EnumerateMetafile(Metafile, PointF[], Graphics.EnumerateMetafileProc)	Sends the records in the specified Metafile, one at a time, to a callback method for display in a specified parallelogram.
◆	EnumerateMetafile(Metafile, PointF[], Graphics.EnumerateMetafileProc, IntPtr)	Sends the records in the specified Metafile, one at a time, to a callback method for display in a specified parallelogram.
◆	EnumerateMetafile(Metafile, PointF[], Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in the specified Metafile, one at a time, to a callback method for display in a specified parallelogram using specified image attributes.
◆	EnumerateMetafile(Metafile, PointF[], RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified parallelogram.
◆	EnumerateMetafile(Metafile, PointF[], RectangleF, GraphicsUnit,	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for

	Graphics.EnumerateMetafileProc, IntPtr)	display in a specified parallelogram.
◆	EnumerateMetafile(Metafile, PointF[], RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records in a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified parallelogram using specified image attributes.
◆	EnumerateMetafile(Metafile, Rectangle, Graphics.EnumerateMetafileProc)	Sends the records of the specified Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, Rectangle, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records of the specified Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, Rectangle, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records of the specified Metafile, one at a time, to a callback method for display in a specified rectangle using specified image attributes.
◆	EnumerateMetafile(Metafile, Rectangle, Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc)	Sends the records of a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, Rectangle, Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records of a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, Rectangle, Rectangle, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records of a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified rectangle using specified image attributes.
◆	EnumerateMetafile(Metafile, RectangleF, Graphics.EnumerateMetafileProc)	Sends the records of the specified Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, RectangleF, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records of the specified Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, RectangleF, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records of the specified Metafile, one at a time, to a callback method for display in a specified rectangle using specified image attributes.
◆	EnumerateMetafile(Metafile, RectangleF, RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc)	Sends the records of a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified rectangle.
◆	EnumerateMetafile(Metafile, RectangleF, RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr)	Sends the records of a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified rectangle.

	EnumerateMetafile(Metafile, RectangleF, RectangleF, GraphicsUnit, Graphics.EnumerateMetafileProc, IntPtr, ImageAttributes)	Sends the records of a selected rectangle from a Metafile, one at a time, to a callback method for display in a specified rectangle using specified image attributes.
	Equals(Object)	Determines whether the specified object is equal to the current object.(Inherited from Object.)
	ExcludeClip(Rectangle)	Updates the clip region of this Graphics to exclude the area specified by a Rectangle structure.
	ExcludeClip(Region)	Updates the clip region of this Graphics to exclude the area specified by a Region.
	FillClosedCurve(Brush, Point[])	Fills the interior of a closed cardinal spline curve defined by an array of Point structures.
	FillClosedCurve(Brush, Point[], FillMode)	Fills the interior of a closed cardinal spline curve defined by an array of Point structures using the specified fill mode.
	FillClosedCurve(Brush, Point[], FillMode, Single)	Fills the interior of a closed cardinal spline curve defined by an array of Point structures using the specified fill mode and tension.
	FillClosedCurve(Brush, PointF[])	Fills the interior of a closed cardinal spline curve defined by an array of PointF structures.
	FillClosedCurve(Brush, PointF[], FillMode)	Fills the interior of a closed cardinal spline curve defined by an array of PointF structures using the specified fill mode.
	FillClosedCurve(Brush, PointF[], FillMode, Single)	Fills the interior of a closed cardinal spline curve defined by an array of PointF structures using the specified fill mode and tension.
	FillEllipse(Brush, Int32, Int32, Int32, Int32)	Fills the interior of an ellipse defined by a bounding rectangle specified by a pair of coordinates, a width, and a height.
	FillEllipse(Brush, Rectangle)	Fills the interior of an ellipse defined by a bounding rectangle specified by a Rectangle structure.
	FillEllipse(Brush, RectangleF)	Fills the interior of an ellipse defined by a bounding rectangle specified by a RectangleF structure.
	FillEllipse(Brush, Single, Single, Single, Single)	Fills the interior of an ellipse defined by a bounding rectangle specified by a pair of coordinates, a width, and a height.
	FillPath(Brush, GraphicsPath)	Fills the interior of a GraphicsPath.

◆	FillPie(Brush, Int32, Int32, Int32, Int32, Int32, Int32)	Fills the interior of a pie section defined by an ellipse specified by a pair of coordinates, a width, a height, and two radial lines.
◆	FillPie(Brush, Rectangle, Single, Single)	Fills the interior of a pie section defined by an ellipse specified by a RectangleF structure and two radial lines.
◆	FillPie(Brush, Single, Single, Single, Single, Single, Single)	Fills the interior of a pie section defined by an ellipse specified by a pair of coordinates, a width, a height, and two radial lines.
◆	FillPolygon(Brush, Point[])	Fills the interior of a polygon defined by an array of points specified by Point structures.
◆	FillPolygon(Brush, Point[], FillMode)	Fills the interior of a polygon defined by an array of points specified by Point structures using the specified fill mode.
◆	FillPolygon(Brush, PointF[])	Fills the interior of a polygon defined by an array of points specified by PointF structures.
◆	FillPolygon(Brush, PointF[], FillMode)	Fills the interior of a polygon defined by an array of points specified by PointF structures using the specified fill mode.
◆	FillRectangle(Brush, Int32, Int32, Int32, Int32)	Fills the interior of a rectangle specified by a pair of coordinates, a width, and a height.
◆	FillRectangle(Brush, Rectangle)	Fills the interior of a rectangle specified by a Rectangle structure.
◆	FillRectangle(Brush, RectangleF)	Fills the interior of a rectangle specified by a RectangleF structure.
◆	FillRectangle(Brush, Single, Single, Single, Single)	Fills the interior of a rectangle specified by a pair of coordinates, a width, and a height.
◆	FillRectangles(Brush, Rectangle[])	Fills the interiors of a series of rectangles specified by Rectangle structures.
◆	FillRectangles(Brush, RectangleF[])	Fills the interiors of a series of rectangles specified by RectangleF structures.
◆	FillRegion(Brush, Region)	Fills the interior of a Region.
🔑◆	Finalize()	Allows an object to try to free resources and perform other cleanup operations before it is reclaimed by garbage collection.(Overrides Object.Finalize().)
◆	Flush()	Forces execution of all pending graphics operations

		and returns immediately without waiting for the operations to finish.
◆	Flush(FlushIntention)	Forces execution of all pending graphics operations with the method waiting or not waiting, as specified, to return before the operations finish.
◆ S	FromHdc(IntPtr)	Creates a new Graphics from the specified handle to a device context.
◆ S	FromHdc(IntPtr, IntPtr)	Creates a new Graphics from the specified handle to a device context and handle to a device.
◆ S	FromHdcInternal(IntPtr)	This API supports the product infrastructure and is not intended to be used directly from your code. Returns a Graphics for the specified device context.
◆ S	FromHwnd(IntPtr)	Creates a new Graphics from the specified handle to a window.
◆ S	FromHwndInternal(IntPtr)	This API supports the product infrastructure and is not intended to be used directly from your code. Creates a new Graphics for the specified windows handle.
◆ S	FromImage(Image)	Creates a new Graphics from the specified Image.
◆	GetContextInfo()	This API supports the product infrastructure and is not intended to be used directly from your code. Gets the cumulative graphics context.
◆ S	GetHalftonePalette()	Gets a handle to the current Windows halftone palette.
◆	GetHashCode()	Serves as the default hash function. (Inherited from Object.)
◆	GetHdc()	Gets the handle to the device context associated with this Graphics.
◆	GetLifetimeService()	Retrieves the current lifetime service object that controls the lifetime policy for this instance.(Inherited from MarshalByRefObject.)
◆	GetNearestColor(Color)	Gets the nearest color to the specified Color structure.
◆	GetType()	Gets the Type of the current instance.(Inherited from Object.)
◆	InitializeLifetimeService()	Obtains a lifetime service object to control the

		lifetime policy for this instance.(Inherited from MarshalByRefObject.)
◆	IntersectClip(Rectangle)	Updates the clip region of this Graphics to the intersection of the current clip region and the specified Rectangle structure.
◆	IntersectClip(RectangleF)	Updates the clip region of this Graphics to the intersection of the current clip region and the specified RectangleF structure.
◆	IntersectClip(Region)	Updates the clip region of this Graphics to the intersection of the current clip region and the specified Region.
◆	IsVisible(Int32, Int32)	Indicates whether the point specified by a pair of coordinates is contained within the visible clip region of this Graphics.
◆	IsVisible(Int32, Int32, Int32, Int32)	Indicates whether the rectangle specified by a pair of coordinates, a width, and a height is contained within the visible clip region of this Graphics.
◆	IsVisible(Point)	Indicates whether the specified Point structure is contained within the visible clip region of this Graphics.
◆	IsVisible(PointF)	Indicates whether the specified PointF structure is contained within the visible clip region of this Graphics.
◆	IsVisible(Rectangle)	Indicates whether the rectangle specified by a Rectangle structure is contained within the visible clip region of this Graphics.
◆	IsVisible(RectangleF)	Indicates whether the rectangle specified by a RectangleF structure is contained within the visible clip region of this Graphics.
◆	IsVisible(Single, Single)	Indicates whether the point specified by a pair of coordinates is contained within the visible clip region of this Graphics.
◆	IsVisible(Single, Single, Single, Single)	Indicates whether the rectangle specified by a pair of coordinates, a width, and a height is contained within the visible clip region of this Graphics.
◆	MeasureCharacterRanges(String, Font, RectangleF, StringFormat)	Gets an array of Region objects, each of which bounds a range of character positions within the specified string.
◆	MeasureString(String, Font)	Measures the specified string when drawn with the

		specified Font.
◆	MeasureString(String, Font, Int32)	Measures the specified string when drawn with the specified Font.
◆	MeasureString(String, Font, Int32, StringFormat)	Measures the specified string when drawn with the specified Font and formatted with the specified StringFormat.
◆	MeasureString(String, Font, PointF, StringFormat)	Measures the specified string when drawn with the specified Font and formatted with the specified StringFormat.
◆	MeasureString(String, Font, SizeF)	Measures the specified string when drawn with the specified Font within the specified layout area.
◆	MeasureString(String, Font, SizeF, StringFormat)	Measures the specified string when drawn with the specified Font and formatted with the specified StringFormat.
◆	MeasureString(String, Font, SizeF, StringFormat, Int32, Int32)	Measures the specified string when drawn with the specified Font and formatted with the specified StringFormat.
◆	MultiplyTransform(Matrix)	Multiplies the world transformation of this Graphics and specified the Matrix.
◆	MultiplyTransform(Matrix, MatrixOrder)	Multiplies the world transformation of this Graphics and specified the Matrix in the specified order.
◆	ReleaseHdc()	Releases a device context handle obtained by a previous call to the GetHdc method of this Graphics.
◆	ReleaseHdc(IntPtr)	Releases a device context handle obtained by a previous call to the GetHdc method of this Graphics.
◆	ReleaseHdcInternal(IntPtr)	This API supports the product infrastructure and is not intended to be used directly from your code. Releases a handle to a device context.
◆	ResetClip()	Resets the clip region of this Graphics to an infinite region.
◆	ResetTransform()	Resets the world transformation matrix of this Graphics to the identity matrix.
◆	Restore(GraphicsState)	Restores the state of this Graphics to the state represented by a GraphicsState.
◆	RotateTransform(Single)	Applies the specified rotation to the transformation matrix of this Graphics.

◆	RotateTransform(Single, MatrixOrder)	Applies the specified rotation to the transformation matrix of this Graphics in the specified order.
◆	Save()	Saves the current state of this Graphics and identifies the saved state with a GraphicsState.
◆	ScaleTransform(Single, Single)	Applies the specified scaling operation to the transformation matrix of this Graphics by prepending it to the object's transformation matrix.
◆	ScaleTransform(Single, Single, MatrixOrder)	Applies the specified scaling operation to the transformation matrix of this Graphics in the specified order.
◆	SetClip(Graphics)	Sets the clipping region of this Graphics to the **Clip** property of the specified Graphics.
◆	SetClip(Graphics, CombineMode)	Sets the clipping region of this Graphics to the result of the specified combining operation of the current clip region and the Clip property of the specified Graphics.
◆	SetClip(GraphicsPath)	Sets the clipping region of this Graphics to the specified GraphicsPath.
◆	SetClip(GraphicsPath, CombineMode)	Sets the clipping region of this Graphics to the result of the specified operation combining the current clip region and the specified GraphicsPath.
◆	SetClip(Rectangle)	Sets the clipping region of this Graphics to the rectangle specified by a Rectangle structure.
◆	SetClip(Rectangle, CombineMode)	Sets the clipping region of this Graphics to the result of the specified operation combining the current clip region and the rectangle specified by a Rectangle structure.
◆	SetClip(RectangleF)	Sets the clipping region of this Graphics to the rectangle specified by a RectangleF structure.
◆	SetClip(RectangleF, CombineMode)	Sets the clipping region of this Graphics to the result of the specified operation combining the current clip region and the rectangle specified by a RectangleF structure.
◆	SetClip(Region, CombineMode)	Sets the clipping region of this Graphics to the result of the specified operation combining the current clip region and the specified Region.
◆	ToString()	Returns a string that represents the current object. (Inherited from Object.)

◆	TransformPoints(CoordinateSpace, CoordinateSpace, Point[])	Transforms an array of points from one coordinate space to another using the current world and page transformations of this Graphics.
◆	TransformPoints(CoordinateSpace, CoordinateSpace, PointF[])	Transforms an array of points from one coordinate space to another using the current world and page transformations of this Graphics.
◆	TranslateClip(Int32, Int32)	Translates the clipping region of this Graphics by specified amounts in the horizontal and vertical directions.
◆	TranslateClip(Single, Single)	Translates the clipping region of this Graphics by specified amounts in the horizontal and vertical directions.
◆	TranslateTransform(Single, Single)	Changes the origin of the coordinate system by prepending the specified translation to the transformation matrix of this Graphics.
◆	TranslateTransform(Single, Single, MatrixOrder)	Changes the origin of the coordinate system by applying the specified translation to the transformation matrix of this Graphics in the specified order.

Remarks

The Graphics class provides methods for drawing objects to the display device. A Graphics is associated with a specific device context.

You can obtain a Graphics object by calling the Control.CreateGraphics method on an object that inherits from System.Windows.Forms.Control, or by handling a control's Control.Paint event and accessing the Graphics property of the System.Windows.Forms.PaintEventArgs class. You can also create a Graphics object from an image by using the FromImage method. For more information about creating a Graphics object, see How to: Create Graphics Objects for Drawing.

You can draw many different shapes and lines by using a Graphics object. For more information about how to draw lines and shapes, see the specific **Draw***GraphicalElement* method for the line or shape you want to draw. These methods include DrawLine, DrawArc, DrawClosedCurve, DrawPolygon, and DrawRectangle. For more information about how to draw lines and shapes, see Using a Pen to Draw Lines and Shapes and Using a Brush to Fill Shapes.

You can also draw images and icons by using the DrawImage and DrawIcon methods, respectively. To perform a bit-block transfer of color data from the screen to the drawing surface of the Graphics object, see CopyFromScreen. For more information about how to draw images with a Graphics object, see Working with Images, Bitmaps, Icons, and Metafiles.

In addition, you can manipulate the coordinate system used by the Graphics object. For more information on the coordinate system and how to manipulate it, see Coordinate Systems and Transformations.

Examples

The following code example is designed for use with Windows Forms and requires a PaintEventArgs object. The PaintEventArgs object is named *e* and is a parameter of the Paint event handler. The code performs the following actions:

- Creates an image from a JPEG file. The file is named SampImag.jpg and is located in the folder of the example.
- Creates a point at which to draw the upper-left corner of the image.
- Draws the unscaled image to the screen by using a Graphics object.

C#

```
private void DrawImagePointF(PaintEventArgs e)
{

    // Create image.
    Image newImage = Image.FromFile("SampImag.jpg");

    // Create point for upper-left corner of image.
    PointF ulCorner = new PointF(100.0F, 100.0F);

    // Draw image to screen.
    e.Graphics.DrawImage(newImage, ulCorner);
}
```

Version Information

.NET Framework
Available since 1.1

Thread Safety

Any public static (**Shared** in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

See Also

System.Drawing Namespace
How to: Create Graphics Objects for Drawing
Getting Started with Graphics Programming
Managing the State of a Graphics Object

Return to top