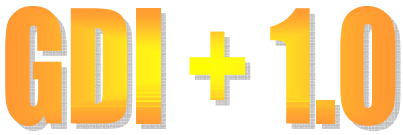


by Stephen Rodriguez

# LinearGradientBrush Class

## General.

Our GDI+ LinearGradientBrush class is housed within the "gdiPlus_ LinearGradientBrush Class.pbi" source file.

The class contains elements of the following c/c++ classes :

**LinearGradientBrush**

This is not a *helper-class* (containing just simple macros etc.) but a full OOP class.

## Note.

A **LinearGradientBrush** object has two parallel boundaries: a starting boundary and an ending boundary. A color is associated with each of these two boundaries. Each boundary is a straight line that passes through a specified point — the starting boundary passes through the starting point; the ending boundary passes through the ending point — and is perpendicular to the direction of the linear gradient brush. The direction of the linear gradient brush follows the line that is defined by the starting and ending points. This line, the "directional line," may be horizontal, vertical, or diagonal. All points that lie on a line that is parallel to the boundaries are the same color. When you fill an area with a linear gradient brush, the color changes gradually from one line to the next as you move along the directional line from the starting boundary to the ending boundary. By default, the change in color is proportional to the change in distance; that is, a line 30 percent of the distance between the starting boundary and the ending boundary has a color that is 30 percent of the distance between the starting boundary color and the ending boundary color. The color pattern is repeated outside of the starting and ending boundaries.

## Interface / base-class.

Our LinearGradientBrush class exposes a single **interface** with the name gdiPlus_LinearGradientBrush.

This interface extends our gdiPlus_BrushClass and gdiPlus_BaseClass interfaces which, between them, expose the following methods :

***Destroy()***

***GetGdiPlusHandle.i()***
Returns the Brush handle used natively by GDI+.

***GetType.i()***
Returns a BrushType constant, e.g. *#BrushTypeLinearGradient*.

## '*gdiPlus_LinearGradientBrush*' constructors.

The following all return (if successful) instances of our gdiPlus_LinearGradientBrush class whose methods are listed in the following section. These correspond to constructors from the appropriate c/c++ wrapper class and so the reader is advised to look on the appropriate MSDN pages for detailed descriptions.

***gdiPlus_CreateLinearGradientBrushFromPointF(*point1.PointF, *point2.PointF, color1, color2, wrapMode=#WrapModeTile)***

***gdiPlus_CreateLinearGradientBrushFromPointI(*point1.Point, *point2.Point, color1, color2, wrapMode=#WrapModeTile)***

***gdiPlus_CreateLinearGradientBrushFromRectF(*rc.RectF, color1, color2, mode=#LinearGradientModeHorizontal, wrapMode=#WrapModeTile)***

***gdiPlus_CreateLinearGradientBrushFromRectI(*rc.RectI, color1, color2, mode=#LinearGradientModeHorizontal, wrapMode=#WrapModeTile)***

## '*gdiPlus_LinearGradientBrush*' methods.

Unless specified otherwise, the following all return a gdiPlus status code (beginning with *#Ok*).

All of these methods correspond to methods from the appropriate c/c++ wrapper class and so the reader is advised to look on the appropriate MSDN pages for detailed descriptions.

***Clone.i()***
Returns, if successful, a new gdiPlus_LinearGradientBrush object containing an exact copy of the original object.

***GetBlend.i(*blends, *positions, count)***
*blends and *positions should both point to an array of floats.

***GetBlendCount.i()***
Returns the number of blend factors currently set for this brush.

***GetGammaCorrection.i()***
Returns *#True* if gamma correction is enabled.

***GetInterpolationColorCount.i()***
Returns the number of colors currently set to be interpolated for this linear gradient brush.

***GetInterpolationColors.i(*blendColors, *positions, count)***
*blendColors should point to an array of ARGB values.
*positions should point to an array of floats.

***GetLinearColors.i(*colors)***
*colors must point to an array large enough for 2 ARGB values.

***GetRectF.i(*rc.RectF)***

***GetRectI.i(*rc.RectI)***

***GetTransform.i(matrix.gdiPlus_Matrix)***
Pass a gdiPlus_Matrix object as a parameter which will be modified to reflect the brushes transformation matrix.

***GetWrapMode.i()***
Returns a WrapMode constant, e.g. *#WrapModeTile*.

***MultiplyTransform.i(matrix.gdiPlus_Matrix, matrixOrder=#MatrixOrderPrepend)***
Updates this brush's transformation matrix with the product of itself and another matrix.

***ResetTransform.i()***
Resets the transformation matrix of this linear gradient brush to the identity matrix.

***RotateTransform.i(angle.f, matrixOrder=#MatrixOrderPrepend)***
Updates this brush's current transformation matrix with the product of itself and a rotation matrix.

***ScaleTransform.i(sx.f, sy.f, matrixOrder=#MatrixOrderPrepend)***
Updates this brush's current transformation matrix with the product of itself and a scaling matrix.

***SetBlend.i(*blends, *positions, count)***
*blends and *positions should both point to an array of floats with values between 0 and 1.

***SetBlendBellShape.i(focus.f, scale.f)***
Both parameters should be between 0 and 1.

***SetBlendTriangularShape.i(focus.f, scale.f)***
Both parameters should be between 0 and 1.

***SetGammaCorrection.i(blnUseGammaCorrection)***
Set the parameter to *#True* or *#False*.

***SetInterpolationColors.i(*blendColors, *positions, count)***
*blendColors should point to an array of ARGB values.
*positions should point to an array of floats with values between 0 and 1.

***SetLinearColors.i(color1, color2)***

***SetTransform.i(matrix.gdiPlus_Matrix)***
Sets the transformation matrix of this linear gradient brush.

***SetWrapMode.i(wrapMode)***
wrapMode = a WrapMode constant, e.g. *#WrapModeTile*.

***TranslateTransform.i(dx.f, dy.f, matrixOrder=#MatrixOrderPrepend)***
Updates this brush's current transformation matrix with the product of itself and a translation matrix.