TransparentBlt

-
Alyce

For more details on using TransparentBlt, see Janet Terra's TransparentBlt article.

TransparentBlt | Syntax | Remarks | The Long Color Value | Demo One | Demo Two *Some text below is copied from the Microsoft Developers Network Library.*

For an eBook or printed book on using the API with Liberty BASIC, see:
APIs for Liberty BASIC

TransparentBlt

The TransparentBlt function performs a bit-block transfer of the color data corresponding to a rectangle of pixels from the specified source device context into a destination device context. Like StretchBlt, it can perform a stretching or shrinking of the image. In addition, it makes transparent the color specified. This allows us to place an image directly on any background without a surrounding rectangle, much like sprites.

There is one other important difference between StretchBlt and TransparentBlt. StretchBlt is part of the Windows DLL GDI32.DLL. This DLL needn't be opened in Liberty BASIC. It can be addressed directly as #gdi32. The TransparentBlt function is part of MSIMG32.DLL. We must open this DLL in order to use it.

```
open msimg32.dll for dll as #msimg
```

We must close it when we no longer need it.

```
close #msimg
```

Syntax

```
    open "Msimg32.dll" for DLL as #msimg
    CallDll #msimg, "TransparentBlt",_
    hdc as ulong,_      'The destination DC = graphicbox
    xDest as long,_     'x location on destination
    yDest as long,_     'y location on destination
    wDest as long,_     'width to stretch to
    hDest as long,_     'height to stretch to
    hMemDC as ulong,_  'The source DC = memory
    xSrc as long,_     'x location in source
    ySrc as long,_     'y location in source
    wSrc as long,_     'width to take from source
    hSrc as long,_     'height to take from source
    lColor as ulong,_  'RGB/long color value to make transparent
    result as long  'nonzero if successful
    close #msimg
```

Remarks

From MSDN:
If the source and destination rectangles are not the same size, the source bitmap is stretched to match the destination rectangle. When the SetStretchBltMode function is used, the iStretchMode modes of BLACKONWHITE and WHITEONBLACK are converted to COLORONCOLOR for the TransparentBlt function.

The destination device context specifies the transformation type for the destination coordinates. The source device context specifies the transformation type for the source coordinates.

TransparentBlt does not mirror a bitmap if either the width or height, of either the source or destination, is negative.

The Long Color Value

TransparentBlt allows us to specify the color that is to be transparent when the image is transferred. This needs to be in the form of a single, long value. We get this value from the red, green and blue components with the following formula:

LongColor = (blue*256*256) + (green*256) + red

If the blue value is 200, green is 100 and red is 10, it looks like this:

LongColor = (200*256*256) + (100*256) + 10

Demo One

The following program demonstrates the transparency of the image when it is transferred. The graphicbox is filled with black and a word is printed on it. The code captures a bitmap from the graphicbox, then fills it with yellow. When the original image is transferred back to the graphicbox, the black is transparent, so the word appears properly on the yellow color.

```
nomainwin
winWide=700:winHigh=500
WindowWidth=winWide+50:WindowHeight=winHigh+50
UpperLeftX=1:UpperLeftY=1

graphicbox #1.g, 0,0,winWide,winHigh
open "GDI Demo" for window as #1
    #1 "trapclose [quit]"
    #1.g "down; fill black; backcolor black; color red"
    #1.g "font arial 20"
    #1.g "\\\Sample"
    'capture a bitmap in memory with native commands
    #1.g "getbmp display 0 0 160 190"
    'get handle for memory bmp
    handleBmp = hbmp("display")
    #1.g "fill yellow;flush"

    h=hwnd(#1.g)  'graphicbox handle

    'get device context for window:
    calldll #user32, "GetDC",_
    h as ulong,_ 'graphicbox handle
    hdc as ulong 'returns handle to device context

    calldll #gdi32, "CreateCompatibleDC",_
    hdc as ulong,_  'graphicbox DC
    hMemDC as ulong 'memory DC

    CallDLL #gdi32,"SelectObject",_
    hMemDC as uLong,_     'memory DC
    handleBmp as uLong,_  'handle of bmp
    oldBmp as uLong       'returns previously selected bitmap

    open "Msimg32.dll" for DLL as #msimg
    CallDll #msimg, "TransparentBlt",_
    hdc as ulong,_     'The destination DC = graphicbox
    30 as long,_       'x location on destination
    0 as long,_        'y location on destination
    250 as long,_      'width to stretch to
    300 as long,_      'height to stretch to
    hMemDC as ulong,_  'The source DC = memory
    0 as long,_        'x location in source
    0 as long,_        'y location in source
    100 as long,_      'width to take from source
    90 as long,_       'height to take from source
    0 as ulong,_       'RGB/long color value to make transparent
    result as long  'nonzero if successful
    close #msimg
wait

[quit]
    calldll #gdi32, "DeleteDC",_
    hMemDC as ulong,_   'DC to delete
    re as long          'nonzero=success

    calldll #user32, "ReleaseDC",_
    h as ulong,_    'window handle
    hdc as ulong,_  'device context
    ret as long

    close #1:end
```

Demo Two

The second demo is identical to the first, with one important difference. We fill with an RGB color and specify that color as the transparent color in TransparentBlt.

```
nomainwin
winWide=700:winHigh=500
WindowWidth=winWide+50:WindowHeight=winHigh+50
UpperLeftX=1:UpperLeftY=1

graphicbox #1.g, 0,0,winWide,winHigh
open "GDI Demo" for window as #1
    #1 "trapclose [quit]"
    #1.g "down; fill 10 100 200; backcolor 10 100 200; color red"
    #1.g "font arial 20"
    #1.g "\\\Sample"
    'capture a bitmap in memory with native commands
    #1.g "getbmp display 0 0 160 190"

    'get handle for memory bmp
    handleBmp = hbmp("display")
    #1.g "fill yellow;flush"

    h=hwnd(#1.g)  'graphicbox handle
    'get device context for window:
    calldll #user32, "GetDC",_
    h as ulong,_ 'graphicbox handle
    hdc as ulong 'returns handle to device context

    calldll #gdi32, "CreateCompatibleDC",_
    hdc as ulong,_  'graphicbox DC
    hMemDC as ulong 'memory DC

    CallDLL #gdi32,"SelectObject",_
    hMemDC as uLong,_     'memory DC
    handleBmp as uLong,_  'handle of bmp
    oldBmp as uLong       'returns previously selected bitmap

    lColor = (200*256*256) + (100*256) + 10

    open "Msimg32.dll" for DLL as #msimg
    CallDll #msimg, "TransparentBlt",_
    hdc as ulong,_     'The destination DC = graphicbox
    30 as long,_       'x location on destination
    0 as long,_        'y location on destination
    250 as long,_      'width to stretch to
    300 as long,_      'height to stretch to
    hMemDC as ulong,_  'The source DC = memory
    0 as long,_        'x location in source
    0 as long,_        'y location in source
    100 as long,_      'width to take from source
    90 as long,_       'height to take from source
    lColor as ulong,_  'RGB/long color value to make transparent
    result as long  'nonzero if successful
    close #msimg
wait

[quit]
    calldll #gdi32, "DeleteDC",_
    hMemDC as ulong,_   'DC to delete
    re as long          'nonzero=success

    calldll #user32, "ReleaseDC",_
    h as ulong,_    'window handle
    hdc as ulong,_  'device context
    ret as long

    close #1:end
```

GDI Tutorials Home