

This documentation covers extensively the BitBlt Class and the StretchBlt Class created by Liu Yucheng (can be contacted at prgmrlc@pacific.net.sg). Source is available.

And as a rule for all forms of blitting. Set the the ScaleMode for all the controls to be "3-Pixel".

ALSO invoke form1.show and doevents before starting Blting or the Blted image will not be visible. Due to the lack of system redraw.

Part 1: Transparency :Page 1

Part 2: BitBlt :Page 1

Part 3: StretchBlt :Page 4

Part 4: Internal Clipboard and other functions: Page 6

Part 5: Example : Page 7

Part 6: converting VB6 classes and VBP's for VB5 users. Page 6

Part 1:

Transparency:

Lets make it simple. You AND the mask onto the background, then you PAINT the image onto the mask.

Generally, the Black areas on the Mask become transparent and the rest are visible. No semi-transparency supported as yet.

Sorry missed out something in the previous version of the documentation. For the it to be transparent, the transparent area on the mask is black and the that on the picture is white.

As current, I am trying to make a good mask creator.

Part 2:

BitBlt Class (BltSysCls)

Property List

Private Properties

Device Contexts and Bitmaps to hold image

MvarhDC as long

MvarBMP as long

Device Contexts and Bitmaps to hold transparency mask

MvarMaskBMP as Long

MvarMaskhDC as long

OldBMP as Long

Whether image is empty

MvarIsEmpty as Boolean

Whether you want transparency

MvarTransparency as Boolean

Screen coordinates to draw upon

ScreenWidth as Integer

ScreenHeight as Integer

ScreenX as integer

ScreenY as integer

MvarTransparency is accessible through the **Transparency** property

ScreenWidth is accessible through the **ScrWidth** property

ScreenHeight is accessible through the **ScrHeight** property

ScreenX is accessible through the **ScrX** property

ScreenY is accessible through the **ScrY** property

Public Properties

Target Device Context to draw upon

TargetDC As Long

Whether there is auto redraw every time the coordinates are changed

(Note, it does not clear the previous image it blitted wtil can't figure out how)

Default is {false}

NoAutoRedraw As Boolean

Methods List

Create(AnyHwnd(for reference), TargetDC, Width of image to be created, Height of image to be created)

Create the internal Device Contexts

DestroyPicture

Kill all created images. (you have to Destroy every thing you **Create** before you exit)

LoadPicture(Filename as string)

Loads a picture

LoadMask(Filename as string)

Loads a mask for transparency. You will have to set Transparency = True for Transparency to be activated.

ReadPixel(X as integer, Y as integer)

Returns a color value of the pixel to read with reference to the image created not the screen. Does not include the Mask

Redraw

As the name suggests, redraw

SetPixel(x as integer,Y as integer, RGBvalue as integer)

Sets a pixel with reference to the image created not the screen. Does not include the Mask

CopyPicture(x as integer, y as integer, W as integer H as integer)

Copies the picture into the internal clipboard(to be explained later). Not the windows clipboard.

CopyMask(x as integer, y as integer, W as integer H as integer)

Copies the mask into the internal clipboard(to be explained later). Not the windows clipboard.

PasteMaskPicture(Optional x as integer, Optional y as integer)

Copies the Mask into the internal clipboard(to be explained later). Not the windows clipboard.

PastePicture(Optional x as integer, Optional y as integer)

Copies the internal clipboard(to be explained later) into the picture. Not the windows clipboard.

Part 3:

StretchBlt Class (StretchSysCls)

Property List

Private Properties

Device Contexts and Bitmaps to hold image

MvarhDC as long

MvarBMP as long

Device Contexts and Bitmaps to hold transparency mask

MvarMaskBMP as Long

MvarMaskhDC as long

OldBMP as Long

Whether image is empty

MvarIsEmpty as Boolean

Whether you want transparency

MvarTransparency as Boolean

Screen coordinates to draw upon

ScreenWidth as Integer

ScreenHeight as Integer

ScreenX as integer

ScreenY as integer

PicWidth as integer

PicHeight as integer

MvarTransparency is accessible through the **Transparency** property

ScreenWidth is accessible through the **ScrWidth** property used to define stretching

ScreenHeight is accessible through the **ScrHeight** property used to define stretching

ScreenX is accessible through the **ScrX** property

ScreenY is accessible through the **ScrY** property

Public Properties

Target Device Context to draw upon

TargetDC As Long

Whether there is auto redraw every time the coordinates are changed
(Note, it does not clear the previous image it blitted w till can't figure out how)
Default is {false}
NoAutoRedraw As Boolean

Methods List

Create(AnyHwnd (for reference)TargetDC, Width of image to be created, Height of image to be created)
Create the internal Device Contexts

DestroyPicture
Kill all created images. (you have to Destroy every thing you **Create** before you exit)

LoadPicture(Filename as string)
Loads a picture

LoadMask(Filename as string)
Loads a mask for transparency. You will have to set Transparency = True for Transparency to be activated.

ReadPixel(X as integer, Y as integer)
Returns a color value of the pixel to read with reference to the image created not the screen. Does not include the Mask

Redraw
As the name suggests, redraw

SetPixel(x as integer, Y as integer, RGBvalue as integer)
Sets a pixel with reference to the image created not the screen. Does not include the Mask

CopyPicture(x as integer, y as integer, W as integer H as integer)
Copies the picture into the internal clipboard(to be explained later). Not the windows clipboard.

CopyMask(x as integer, y as integer, W as integer H as integer)
Copies the mask into the internal clipboard(to be explained later). Not the windows clipboard.

PasteMaskPicture(Optional x as integer, Optional y as integer)
Copies the Mask into the internal clipboard(to be explained later). Not the windows clipboard.

PastePicture(Optional x as integer, Optional y as integer)

Copies the internal clipboard(to be explained later) into the picture. Not the windows clipboard.

Part 4:

Internal Clipboard and other functions (BltSysMod, SystemSupport)

The internal Clipboard can be created and destroyed at will and is just another Device Context which can be utilized to transfer pictures to and fro inside the program.

Methods List

CreateClpBrd(AnyhWnd(for reference)

Created a Internal ClipBoard.

DestroyClpBrd

Destroys the ClipBoard. It must be destroyed before program exit.

ClearClpBrd

Clear the ClipBoard.

CopyPicture(DC As Long, X As Integer, y As Integer, H As Integer, W_ As Integer)

Used internally to copy pictures into the ClipBoard

PastePicture(DC As Long, X As Integer, y As Integer) As Boolean

Used internally to paste pictures from the ClipBoard

DirectBltCopy(SrcDC As Long, srcBMP As Long, tarDC As Long, tarBMP_ As Long, W As Integer, H As Integer)

Copy directly from one DC to another

DirectStretchCopy(SrcDC As Long, srcBMP As Long, tarDC As_ Long, tarBMP As Long, OW As Integer, OH As Integer, W As Integer, H As Integer)

Copy directly from one DC to another with support for stretching (untested)

DirectLoad(filename As String, DC As Long, BMP As Long, W As Integer, H_ As Integer)

This function is used internally to load images and has nothing, absolutely nothing to do with DirectX..

(W as H are variables returned from the Sub.)

SystemSupport contains all the declarations required

Part 5:

Examples

These are pretty short examples that work.

BitBlt Example

```
Dim S As New StretchSysCls

Form1_Load()
    S.NoAutoRedraw = True
    S.Create form1.hwnd, form1.hDC, 500, 400
    S.LoadPicture "Boo.bmp"
    S.LoadMask "BooMask.bmp"
    S.transparency = true
    For x%=1 to 600 step 2
        s.scrX=x%
        S.scrY=x%
        S.redraw
    Next x%
    S.destroyPicture
End Sub
```

StretchBlt Examples

```
Dim S As New BltSysCls

Form1_Load()
    S.NoAutoRedraw = True
    S.Create form1.hwnd, form1.hDC, 500, 400
    S.LoadPicture "Boo.bmp"
    S.LoadMask "BooMask.bmp"
    S.transparency = true
    For x%=1 to 600 step 2
        s.scrX=x%
        S.scrY=x%
        S.scrWidth=x%
        S.scrHeight=x%
        S.redraw
    Next x%
    S.destroyPicture
End Sub
```

Clipboard Examples

```
Dim S As New BltSysCls

Form1_Load()
    CreateClpBrd
    S.NoAutoRedraw = True
    S.Create form1.hwnd, form1.hDC, 500, 400
    S.LoadPicture "Boo.bmp"
    CopyPicture picture1.hdc, 0, 0, picture1.height, _
picture1.width
    s.pastepicture
For x%=1 to 600 step 2
    s.scrX=x%
    S.scrY=x%
    S.redraw
Next x%
S.destroyPicture
destroyClpBrd
End Sub
```


For VB5 users

VB5 users, you might not be able to use blt.dll unless you download all the runtime files. So you recompile it. The VBP project file won't be able to be opened., you have to edit it in notepad and remove the line "Retained=0" then it would be able to load.

After loading, check out the classes. There might be some unwanted lines of code in the {General}-{Declarations} section. They might be any of these:

```
MultiUse = -1 'True  
Persistable = 0 'NotPersistable  
DataBindingBehavior = 0 'vbNone  
DataSourceBehavior = 0 'vbNone  
MTSTransactionMode = 0 'NotAnMTSObject
```

Remove them if they exist but I am not sure.
Now it should be able to compile