ImageMan Image OLE Control

Properties Event Methods

Description

The image control provides the ability to load, display, print, process and save images in JPEG, TIFF, BMP, DIB, RLE, PCX, PNG, DXF, Photo CD, WMF, Targa, DCX, GIF, IMG and EPS formats from applications.

Distributing your Applications

If you are using the Visual Basic Setup Wizard it should be able to automatically find and install the required ImageMan files. Our installation program adds the appropriate entries into the SWDEPEND.INI file used by setup wizard to determine required files.

If you are using another installer or another development language you will need to make sure the appropriate files are copied over to the Windows/System directory on the users system. The ocx controls must also be registered in order that your application can run correctly. Most installers now support registering OCX controls or you can use the regsvr.exe or regsvr32.exe utilities to register the ocx files.

To use the regsvr utilities you must invoke the correct utility with the name of the ocx to be registered, for instance:

> **regsvr** imocx.ocx (To register the 16 Bit ImageMan OCX)
> **regsvr32** imtwain3.ocx (To register the 32 bit Twain OCX)

The following files are required when distributing applications which use the ImageMan controls:

<u>32 Bit Image Control</u>

imocx32.ocx	Image Control
imgman31.dll	ImageMan Support DLL
im31*.dil	ImageMan Import Filters
im31x*.del	ImageMan Export Filters
mfc40.dll	Microsoft Foundation Class DLL
msvcrt40.dll	Microsoft C++ Runtime DLL

<u>32 Bit TWAIN Control</u>

imtwain3.ocx	TWAIN Scanner Control
imgman31.dll	ImageMan Support DLL
mfc40.dll	Microsoft Foundation Class DLL
msvcrt40.dll	Microsoft C++ Runtime DLL

<u>16 Bit Image Control</u>

imocx.ocx	Image Control
imgman11.dll	ImageMan Support DLL
im11*.dil	ImageMan Import Filters
im11x*.del	ImageMan Export Filters
oc25.dll	OLE Control Support DLL

<u>16 Bit TWAIN Control</u>

imtwain.ocx	TWAIN Scanner Control
imgman11.dll	ImageMan Support DLL
oc25.dll	OLE Control Support DLL

You may not distribute the imocx.lic file. This file is only required to use the control at design time.

Image Control Concepts

The ImageMan/VB image control is a very powerful tool for adding image processing support to your application. This section of the help explains how to use the image control to do common tasks.

Loading Images

The image control supports loading images in a variety of file formats from disk. If you know the name of the image you wish to load you can set the Picture property to a string containing that name. The following code fragment loads an image called sample.tif:

```
        ImageMan1.Picture = "C:\SAMPLE.TIF"
```

If you want to display a File Open dialog containing the names of images that can be loaded you can invoke the GetFileName method. The control will display a dialog and show those images that can be loaded. If the user selects a filename and selects the OK button, the image will be loaded. For instance:

```
        ImageMan1.GetFilename
```

Getting Image Information

Several Image control properties return information about the current image. The image width and height are returned in the ImageWidth and ImageHeight properties. These values are expressed in pixels.

The ImageXRes and ImageYRes properties return the image's resolution in dots per inch. Be careful when using these properties though, since some images don't contain resolution information, in which case these properties will return zero.

The ImageColors property returns the number of colors that the image contains.

Scaling Images

The image control provides several ways to scale the display of the image in the control. The Magnification property specifies the percentage of the original size that the image should be displayed at. The code below displays the image at 50% of its original size:

```
ImageMan1.Magnification = 50
ImageMan1.Refresh
```

As shown in the example you must invoke the Refresh method after changing the Magnification property. The Magnification property will always maintain the aspect ratio of the image.

The AutoScale property can be used to scale the image to fit into the image control. Setting AutoScale to a value of 1 will cause the image to be scaled to fit into the image control while maintaining its proper aspect ratio. A value of 2 will cause the image to be stretched to fit into the control. Setting the AutoScale property to either of these values will override the Magnification and ScaleWidth/Height properties. A value of zero will disable the AutoScale function.

To scale the image without maintaining its aspect ratio the ScaleWidth and ScaleHeight properties should be used. The ScaleWidth property defines how many image pixels are mapped into the width of the control. While the ScaleHeight property does the same except for the height of the control. For instance, to scale the image so it fits into the control the following code would be needed:

```
ImageMan1.ScaleWidth = ImageMan1.ImageWidth
ImageMan1.ScaleHeight = ImageMan1.ImageHeight
ImageMan1.Refresh
```

Using the Clipboard

The image control supports both copying images to the clipboard and pasting   images from the clipboard into the control.

Copying an image from the control to the clipboard requires invoking the ImageCopy or ImageCut methods. These methods will copy the image to the clipboard in either CF_DIB or CF_METAFILE formats depending on whether its a raster or vector image. The following code loads an image into the image control then copies it to the clipboard:

```
ImageMan1.Picture = "c:\sample.pcx"
ImageMan1. ImageCopy
```

To paste an image into the control requires that the Clipboard contains an image in either CF_DIB or CF_METAFILE formats. Invoking the ImagePaste method will paste an image into the Image control. The following code checks for an image of the appropriate type and if it exists, pastes it into the image control.

```
If Clipboard.GetFormat(CF_DIB) or
Clipboard.GetFormat(CF_METAFILE) Then
        ImageMan1.ImagePaste
        ImageMan1.Refresh
End If
```

Printing Images

The image control provides complete support for high quality printing on any Windows supported printer. The image size, placement and cropping can all be specified using properties in the image control.

Because of the way Visual Basic handles printing , the image control sends its data   to the Visual Basic Printer object. By doing this the control can print the image at the resolution of the printer instead of the screen resolution as is the case with the PrintForm method. This also allows your application to use the Printer object methods, Print, Line, etc. to draw on the page.

The Dst properties, DstTop, DstLeft, DstRight and DstBottom, specify the placement of the image on the page. These are specified in TWIPS units which are 1/1440". The origin is located at the upper left corner of the page, with increasing values of Y down the page.

In addition to setting the Dst properties your application must also set the PrnHdc property of the Image control to the value of the Printer.hDC property.

Once these properties have been set, the image can be printed by invoking the PrintImage method.

The following code fragment sets prints an image sized to 1"x1" on the page:

```
Dim ThehDC%

' Print a space to initialize the VB printer mechanism
' This is not required if you send other output to the '                ' printer
' object before outputting the image.

Printer.Print " "

ThehDC = Printer.hDC

ImageMan1.PrnhDC = ThehDC

' Now set the size & placement of the image
ImageMan1.DstLeft = 1440
Imageman1.DstTop = 1440

ImageMan1.DstRight = 2880
ImageMan1.DstBottom = 2880

' Send the image to the printer
ImageMan1.PrintImage

' Finish the print job and eject the page
Printer.EndDoc
```

The above code prints the entire image. To print a portion of the image requires that the Src properties, SrcTop, SrcLeft, SrcRight, and SrcBottom be set to the portion of the image to be printed. These are specified in image coordinates. For instance, to print just the upper left quarter of the image, the properties would be set like this:

```
ImageMan1.SrcLeft = 0
ImageMan1.SrcTop = 0
ImageMan1.SrcRight = ImageMan1.ImageWidth / 2
ImageMan1.SrcBottom = ImageMan1.ImageHeight / 2
```

Multiple images can be printed on a page by loading the control with each image and outputting it before calling the Printer.EndDoc or Printer.NewPage methods.

When printing multiple images it may be desirable to make the image control invisible to prevent the display of each image. This is done by setting the Visible property to False.

Saving Images

The Image control supports saving images in a variety of image formats. Using the SaveAs method and Compression, AppendImage, Quality, Overwrite properties you can specify the file type and compression information used when saving an image. You can also specify that a portion of the image be saved by specifying the portion using the Src properties.
To display the file save dialog, invoke the SaveAs method with a parameter of an empty string. This will display a file save dialog preconfigured with the supported image types. The user can then select an image type and enter a filename.

To save a file with a specific name and image type, invoke the SaveAs method with the path and filename of the output file. The image will be saved in the format specified by the image extension. The sample code below show using both methods to save an image:

```
ImageMan1.Picture = "sample.pcx"
' Let the user select an export filename
ImageMan1.SaveAs   ""

' Save it as a TIFF image called gates.tif
ImageMan1.SaveAs   "gates.tif"
```

The Compression property specifies the compression used when saving images in TIFF format.

The AppendImage property specifies whether an image written to an existing TIFF or DCX file should be appended to the file or should overwrite the file.

The Quality property specifies the quality factor to be used when saving JPEG images.

Selecting a Portion of an Image

The image control provides the ability to draw a rubber band box when the user clicks and drags the left mouse button. This behavior is enabled by setting the Select property to True.

When Select is set to True and the user clicks and drags the left mouse button over the image area the control will start drawing a rubber band box. When the user releases the mouse button, the box will be erased from the screen and a Select event will be fired. The Select event contains four parameters which are the coordinates of the area the user selected. The coordinates are expressed in image units.

This selection behavior can be used to select an area of the image for most any purpose. The example code below saves the selected portion of the image into a file selected by the user:

```
...
ImageMan1.Select = True
...

Sub ImageMan1_Select (X1, Y1, X2, Y2)
        ' Set the Src properties to specify the portion
        ' of the image to save.
        ImageMan1.SrcLeft = X1
        ImageMan1.SrcTop = Y1
        ImageMan1.SrcRight = X2
        ImageMan1.SrcBottom = Y2

        ' Display the file save dialog
        ImageMan1.SaveAs ""
End Sub
```

Handling Multi Page Images

The image control currently supports two image formats, DCX and TIFF, which can contain multiple images per file. Support for multiple image files is provided by the Pages and PageNumber properties.

The Pages property returns the number of pages in the current image file. For all single image files this will be a value of one.

The PageNumber property specifies which image in the file should be displayed. It can be set to a number from zero to the value of the Pages property minus one. When using this property to move through the images in a file it is important to remember to call the Refresh method to display the new page. Also many properties can change based on the attributes of the new image; for instance, the image width or height may have changed. The following example shows the code for a button that displays the next image in a file:

```
' If there's a next page then go to it.
If ImageMan1.PageNumber < ImageMan1.Pages —1 Then
        ImageMan1.PageNumber = ImageMan1.PageNumber + 1
        ImageMan1.Refresh

        ' update the image stats
        lblWidth.Text = Str$(ImageMan.ImageWidth)
        lblHeight.Text = Str$(ImageMan1.ImageHeight)
End If
```

Using the hDIB and hImage Properties

The hDIB property allows the image control to export and import memory based images.

By querying the hDIB property you can obtain a handle to a Windows DIB (Device Independent Bitmap) in memory. This can be passed to other applications or controls for processing. Each time the hDIB property is queried, a new copy of the image is generated and its handle returned. When the image is no longer required, the memory should be freed by calling the Windows GlobalFree function with the value returned from the property.

Assigning a DIB handle to the hDIB property will load the image into the control. The image can then be processed like any other image. The hDIB property can be used with other imaging toolkits which can provide an image in DIB format.   It is important that any value passed to the hDIB property is a valid handle to a global memory block containing a DIB. If not, the control may generate the dreaded General Protection Fault (GPF).

The hImage property is used when copying images between ImageMan controls. It cannot be passed to any other control type since it is only understood by the ImageMan image control. The hImage property is very similar to the hDIB property except that the image may be scaled when copying between controls using the hImage property. By specifying the size of the new image using the DstRight and DstBottom properties, the size of the internal image can be changed. The following code saves a 100x100 thumbnail of an image in the control named image1:

```
' Specify the size of the new image

Image1.DstRight = 100              ' New Width
Image1.DstBottom = 100             ' New height
Image2.hImage = Image1.hImage      ' Create the thumbnail
Image2.SaveAs "c:\thumb.tif"                ' Save it
Image2.Picture = ""                ' Destroy the thumbnail
```

This code uses a second image control named, Image2, to contain the thumbnail. This control can be invisible so the process of creating the thumbnail is invisible. If the DstRight and DstBottom properties had not been set then the image in the Image2 control would be the same size as the image in the Image1 control.

This same process can be used to scale the internal image maintained for an image control by specifying the new image size in the DstRight and DstBottom properties then assigning the hImage property to itself like this:

```
'Scale the image in the control to half size

Image1.Dstright = Image1.ImageWidth / 2
Image1.DstBottom = Image1.ImageHeight / 2

' Now assign the hImage to scale the image
Image1.hImage = Image1.hImage
```

As you can see the hImage property is very powerful. It is important that if you assign the hImage property to a variable that you assign that variable to an image control. If you don't, then the memory occupied by that image may not be freed.

Using the Image Processing Features

The image control supports many image processing features. Support for these features is implemented via methods, making it easy to develop sophisticated applications in very little time.

Rotation of images is accomplished by invoking the Rotate method with the number of degrees the image should be rotated. Each time the method is invoked, the image will be rotated by the specified number of degrees.

Flipping or mirroring the image is done by invoking the MirrorVertical or MirrorHorizontal methods.

An image's brightness is adjusted using the Brightness method. Setting the brightness parameter to a value between 1 and 255 will brighten the image, while setting it to a value between -1 and -255 will darken the image.

Gamma adjustments to an image are made using the Gamma method. Allowable values for this property are 1.0 to 5.0.

An image can be converted to an image with fewer colors by using the DitherMethod property and ReduceTo method. The DitherMethod property specifies which dithering algorithm will be used when converting the image. The allowable values are:

0	No Dither
1	Bayer Dither
2	Burkes Dither
3	Floyd Steinberg Dither

The no-dither option results in the fastest image conversion but the poorest image quality. The bayer dither is very fast but generally yields average results. For best image quality use either the Burkes or Floyd Steinberg dithers.

The process of reducing the colors is accomplished by invoking the ReduceTo method. When calling the method you can specify the number of colors the image is to be reduced to, whether the image should be converted to greyscale and whether an optimized palette should be used.

The following code loads a 24 bit image and reduces it to a 256 image with an optimized palette using the Burkes Dither.

```
ImageMan1.Picture = "24bit.tga"
ImageMan1.DitherMethod = 2
ImageMan1.ReduceTo 256, FALSE,   TRUE

' Now draw the new image
ImageMan1.Refresh
```

The IncreaseTo method is used to increase the color depth of an image. It is invoked with a parameter that indicates the bit depth that the image should be increased to.

Drawing on the Control

The ability of the image control to return a handle to a Windows Device Context (hDC) allows you to alter the actual image in memory. Uses of this would include redlining applications, drawing text on a form in the control, a paint application, and many others.

The hDC property can be used with the Windows GDI functions to manipulate the image. The property changes with each image so it must be obtained after loading an image. Also certain actions which change the internal image such as rotation and color reduction will cause the hDC property to change.

The following code draws a line on the image.

```
	Declare Function MyMoveTo Alias "MoveTo" Lib "GDI" ( Byval hDC%, ByVal X%, ByVal Y%)
	Declare Function MyLineTo Alias "LineTo" Lib "GDI" ( Byval hDC%, ByVal X%, ByVal Y%)

	Dim myhDC%

	ImageMan1.Picture = "sample.gif"
	myhDC = ImageMan1.hDC
	MyMoveTo myhDC, 0, 0
	MyLineTo myhDC, 100, 100
	ImageMan1.Refresh
```

After altering the image using the hDC property you must call the Refresh method to show the changes on the screen. It is also important not to use the hDC after the image has been removed from the control, as this may cause Windows to crash.

The current release of the control doesn't support the hDC property for 24 bit images. This may change in future releases so check the release notes.

Vector image considerations

The ImageMan controls support several vector image formats including Window Metafile, Wordperfect graphics files (WPG), Autocad DXF and Encapsulated Postscript files. Because of the differences between raster and vector images, some of the image processing features of the image control can't be used on vector images. The following functions cannot be performed on vector images:

- Resizing the image when copying between controls using the hImage property. The DstRight and DstBottom properties are ignored.
- Getting a handle to a Window DIB from the hDIB property.
- Getting a Display Context from the hDC property.
- Rotating, mirroring, Color Reducing, Brightness Adjustment, Gamma   Adjustment.
- Copying a portion of the image to the clipboard
- Saving the image in a raster image format.

To determine whether an image is vector, you must perform a logical 'And' between the ImageFlags property and the IMG_VECTOR (1) constant defined in the imageman.txt constants file. If the result is True then the current image is in vector format.

ImageMan OCX Help
Version 5.01
March 26, 1996

Obtaining Technical Support

You may obtain technical support for ImageMan via Phone, Fax, Internet Email, CompuServe, our website or our StarMan BBS.

<u>How to contact Data Techniques, Inc.</u>

Data Techniques, Inc.
300 Pensacola Road
Burnsville, NC 28714

Support:	704-682-4111	(9-5 EST)
Fax:	704-682-0025	
BBS:	704-682-4356	
Email:	support@data-tech.com	
Web:	www.data-tech.com	
CompuServe:	GO DATATECH	

If you have Internet Email access you may also wish to join our email mailing lists. As new versions are released we will automatically send email notifying you of the upgrade/update and other important ImageMan news.

To join the mailing list, visit our WebSite or send email to: *imageman-request@data-tech.com* with the word subscribe in the message body. You will receive a message confirming you have been added to the list.

Changes from the ImageMan VBX Controls

Changes to Properties

- The ClipboardCommand property has been replaced by the ImageCopy, ImageCut and ImagePaste methods.
- The Invert property has been changed to a method.
- The DoPrint property has been replaced with the Print method.
- The Mirror property has been replaced with the MirrorVertical and MirrorHorizontal methods.
- The Rotate property has been changed to a method.
- The SaveAs property has been changed to a method. It also now returns the name of the saved file.
- The GetFileName property has been changed to a method.
- The ReduceTo property has been changed to a method.
- The IncreaseTo property has been changed to a method.
- The Gamma property has been changed to a method.
- The ErrCode and ErrString properties are not implemented. The controls now generate standard runtime errors.

New Functionality

- Support for Kodak Photo CD Images (Read Only)
- 16 & 32 Bit Image and Scanner OLE Controls
- Support for CompuServe PNG images
- Support for AutoCad DXF files (Read Only)
- Single Degree rotation
- Scale to Grey (Anti-aliasing) via ScaleMethod property
- 2-3x Faster JPEG decompression
- DisplayColors property returns the number of colors supported by Video Driver.
- ScaleImage method can be used   to easily scale images and for creating thumbnails.
- MouseIcon and MousePointer properties have been added to allow custom cursors to be display when the mouse pointer is over the control.
- The StatusEnabled property has been added to allow the Status event to be disabled therefore speeding up the loading and saving of images.
- The Repaint method has been added to allow the developer to force a repaint of the image control.

Supported Image Formats

BMP
DIB
RLE
PCX
DCX*
JPEG
Photo CD
PNG
DXF
IMG
GIF
TIF*
WPG
TGA
WMF
EPS

* These formats are support multiple pages.

Custom Image Control Properties

AppendImage	AutoDraw
AutoScale	Blue
Compression	DisplayColors
DitherMethod	DstBottom
DstLeft	DstRight
DstTop	EmbedLength
EmbedOffset	Ext
ExtensionCount	Extensions
Green	hDC
hDIB	hImage
hPalette	hWnd
ImageColors	ImageFlags
ImageHeight	ImageWidth
ImageXRes	ImageYRes
Magnification	MouseIcon
MousePointer	Overwrite
PageNumber	Pages
PaletteEntries	Picture
PrnHDC	Quality
Red	SaveOptions
ScaleHeight	ScaleLeft
ScaleMethod	ScaleTop
ScaleWidth	Scrollbars
Select	SrcBottom
SrcLeft	SrcRight
SrcTop	StatusEnabled
UseDDB	VBPicture

Custom Image Control Events

Click	DblClick
MouseDown	MouseMove
MouseUp	Scroll
Select	Status

Custom Image Control Methods

Brightness	Gamma
GetFilename	ImageCopy
ImageCut	ImagePaste
IncreaseTo	Invert
MirrorHorizontal	MirrorVertical
PrintImage	ReduceTo
Repaint	Rotate
SaveAs	ScaleImage

AppendImage Property

Description
When saving an image to a TIFF or DCX file, this controls whether the image is appended to an existing file or the file is overwritten.

Usage
*ImageControl***.AppendImage**[= setting%]

Remarks
The AppendImage property settings are:

Setting	**Description**
True	Append image when saving to TIFF or DCX file formats
False	Do not append. Will overwrite file based on value of Overwrite property.

Data Type
Boolean

AutoScale Property

Description
Determines whether the control will be redrawn automatically when a new image is loaded.

Usage
ImageControl.**AutoScale**[= setting%]

Remarks
The AutoScale property settings are:

Setting	**Description**
0	Do not scale the image to fit into the control.
1	Scale the image to fit into the control and maintain the aspect ratio.
2	Stretch the image to fit into the control.

Note
Setting this property to a value of 1 or 2 will override the Magnification and ScaleWidth/ScaleHeight properties.

Data Type
Integer(Enumerated)

AutoDraw Property
Example

Description
Determines whether the control will be redrawn automatically when a new image is loaded.

Usage
ImageControl.**AutoDraw**[= Bool%]

Remarks
The AutoDraw property settings are:

Setting	Description
True	The control will be redrawn when a new image is loaded.
False	The control must be redrawn by calling the Refresh method after loading an image.

Note
Setting this property to **False** allows you to adjust the Magnification or Scale properties before the image is displayed.   When set to **True** the image will be drawn and scaled to fit the control.

Data Type
Integer(Boolean)

AutoDraw Property Example

Close Copy Print

```
' Load an image and use default scaling

ImageMan1.AutoDraw = True
ImageMan1.Picture = "c:\sample.tif"


' Load an image and set the Magnification to 50%

ImageMan1.AutoDraw = False
ImageMan1.Picture = "c:\sample.tif"          ' Image Loaded but not displayed
ImageMan1.Magnification = 50
ImageMan1.Refresh                            ' Draw the Image
```

Blue Property
Example

Description
Sets the blue value for a color entry in the image's palette.

Usage
ImageControl.**Blue**(Index%)[= Setting%]

Remarks
Use this property along with the Red and Green properties to change the colors in a 1, 16 or 256 color image. After changing these properties you must call the Refresh method to redraw the image with the new colors. The Index% value must be between 0 and the value of the PaletteEntries property. The allowable values for the property are 0 to 255.

Data Type
Integer Array

Red, Green, Blue Properties Example

```
' Darken an image by subtracting a value from each palette entry

Dim Idx%

If ImageMan1.ImageColors <= 256 Then              ' There's No Palette on 24 bit Images
        For Idx = 0 to ImageMan1.PaletteEntries - 1
                ImageMan1.Red(Idx) = ImageMan1.Red(Idx) - 10
                ImageMan1.Green(Idx) = ImageMan1.Green(Idx) - 10
                ImageMan1.Blue(Idx) = ImageMan1.Blue(Idx) - 10
        Next Idx
End If
```

Brightness Method

Description
Adjusts the brightness for an image.

Usage
ImageControl.**Brightness** Setting%

Remarks
Setting Brightness to a value of 0 will display the image with the default brightness. Set this property to a number between 0 and -255 to lighten the image or set it to a number between 1 and 255 to darken it.

Data Type
Integer

Click Event

Description
Occurs when the user presses and then releases a mouse button over a control.

Syntax
Sub ctlname_Click(Index as Integer)

Remarks
The argument Index uniquely identifies a control in a control array.

Note
The Click procedure is only generated for left mouse button activity. Use the MouseDown and MouseUp events to handle other mouse buttons.

Compression Property

Description
Determines what compression method will be used when saving an image using the SaveAs method.

Usage
ImageControl.**Compression**[= setting%]

Remarks
The Compression property settings are:

Setting	**Description**
0	No Compression.
1	LZW - Used in TIFF and GIF formats (Requires license from Unisys Corp.)
2	Huffman - Used in TIFF
3	Packbits - Used in TIFF
4	Fax Group 3 - Used in TIFF
5	Fax Group 4 - Used in TIFF

Note
Some formats like GIFand PCX always store the images in compressed format so this property will be ignored.

If a compression method is selected using this property and the requested image format does not support that method then the format's default compression method will be used.

Data Type
Integer(Enumerated)

DblClick Event

Description
Occurs when the user presses and releases a mouse button, then presses it again over a control.

Syntax
Sub ctlname_DblClick(Index as Integer)

Remarks
The argument Index uniquely identifies a control in a control array.

Note
The DblClick procedure is only generated for left mouse button activity. Use the MouseDown and MouseUp events to handle other mouse buttons.

DisplayColors Property

Description
Returns the number of colors supported by the current video driver mode.

Usage
ImageControl.**DisplayColors**

Remarks
This property can be used in conjunction with the ReduceColors property to properly display images which contain more colors that the video driver is capable of displaying.

Data Type
Single

DitherMethod Property

Description
Sets or Returns the dither method which is used when reducing the number of colors in the image,

Usage
ImageControl.**DitherMethod** = 0 | 1 | 2 | 3

Remarks
The DitherMethod property settings are:

Setting	**Description**
NONE(0)	No Dithering - map to nearest color.
BAYER (1)	Selects the Bayer dither.
BURKES(2)	Selects the Burkes dither.
STEINBERG(3)	Selects the Floyd/Steinberg dither.

Note
This property selects which dither method is used when using the ReduceTo property to change the color format of an image.

Data Type
Integer(Enumerated)

DstLeft, DstTop Properties

Example

Description
Sets the coordinates of the upper left hand corner of the bounding rectangle used when printing an image.

Usage
ImageControl.**DstLeft** = left!
ImageControl.**DstTop** = top!

Remarks
The bounding rectangle defines the area on the page where the image will appear. These units are expressed in TWIPS.

Data Type
Single

Note
These values must be set before invoking the Print method to print the image.

EmbedLength Property
Example

Description
Specifies the length of an embedded image.

Usage
ImageControl.**EmbedLength** [= length]

Remarks
This property is used in conjunction with the EmbeddOffset property to load images which are embedded in other files.

Data Type
Single

Note
This property should be set prior to setting the Picture property to load the embedded image.

EmbedOffset Property
Example

Description
Specifies the offset in bytes of an embedded image in a file.

Usage
ImageControl.**EmbedOffset** [= offset]

Remarks
This property is used in conjunction with the EmbedLength property to load images which are embedded in other files.

Data Type
Single

Note
This property should be set prior to setting the Picture property to load the embedded image.

Loading embedded Files Sample code

Close Copy Print

```
' Load an image that is embedded in the file sample.dat
' The image starts at byte offset 1024 and is 25000 bytes in length

ImageMan1.EmbedOffset = 1024
ImageMan1.EmbedLength = 25000
ImageMan1.Picture = "sample.dat"            ' Load the image
```

PrintImage Method

Example

Description
Causes the image to be printed.

Usage
ImageControl.**PrintImage**

Remarks
Invoking this method causes the image to be printed in the bounding box specified by DstLeft, DstTop, DstRight, and DstBottom. . Before invoking this method you must set the PrnHdc, DstLeft, DstTop, DstRight and DstBottom properties. To print only a portion of the image you must also set the SrcLeft, SrcTop, SrcRight, and SrcBottom properties.

Note
The image will be printed at the resolution of the printer not that of the screen.

Printing an Image

```
' Print an Image at actual size centered on the page
' Declare the SetMapMode Windows API function

' The actual print code
Dim nPageXMid%, nPageYMid%
Dim ImgWidth%, ImgHeight%


' Calculate the center of the page
nPageXMid = Printer.ScaleWidth / 2
nPageYMid = (Printer.ScaleHeight / 2)

' Calculate the Image size in TWIPS
ImgWidth = ImageMan1.ImageWidth * 1440 / Res
ImgHeight = ImageMan1.ImageHeight * 1440 / Res

' Tell the control where to print the size
ImageMan1.DstLeft = nPageXMid - ImgWidth / 2
ImageMan1.DstTop = nPageYMid - ImgHeight / 2
ImageMan1.DstRight = nPageXMid + ImgWidth / 2
ImageMan1.DstBottom = nPageYMid + ImgHeight / 2

' Initialize the VB printing mechanism by printing this empty string
Printer.Print ""

ImageMan1.PrnHdc = Printer.hDC

' Set the Hourglass Cursor, this may take a few seconds
MousePointer = 11

ImageMan1.PrintImage

Printer.EndDoc

' Restore the Cursor
MousePointer = 0
```

DstRight, DstBottom Property

Example

Description
Sets the coordinates of the bottom right hand corner of the bounding rectangle used when printing an image.

Usage
ImageControl.**DstRight** = right!
ImageControl.**DstBottom** = bottom!

Remarks
The bounding rectangle defines the area on the page where the image will appear. These units are expressed in TWIPS.

Data Type
Single

Note
These values must be set before invoking the Print method to print the image.

Ext Property

Description
Specifies the three letter extension of the image format which ImageMan should use when opening images with non-standard extensions.

Usage
ImageControl.**Ext**[= ext$]

Remarks
ImageMan will auto-detect the image format even if the extension is non-standard. Setting this property to the appropriate extension will make image loading faster in those cases where the extension is non-standard.

Data Type
String

ExtensionCount Property
Example

Description
Returns the number of image extensions supported by the control. This property is read-only.

Usage
ImageControl.**ExtensionCount**

Remarks
Use this property with the Extensions property to see a list of the supported image extensions.

Data Type
Integer

Extensions Property

Example

Description
Returns all support image extensions for a control.

Usage
ImageControl.**Extensions**(index%)

Remarks
This property works in conjunction with the ExtensionCount property which returns the number of supported extensions for the object.

This list is built when the ImageMan is loaded. Because of ImageMan's object-oriented architecture the number of supported formats will vary based on which DIL files are installed.

Data Type
String Array   (Read Only)

ExtensionCount, Extension Properties Example

Close Copy Print

```
' Add the list of the supported extensions to a listbox
Dim I%

For I = 0 to ImageMan1.ExtensionCount - 1
        List1.AdddItem   ImageMan1.Extension(I)
Next I
```

Gamma Method

Description
When set to a value causes the image to be corrected by the specified gamma value.

Usage
ImageControl.**Gamma**   GammaVal

Remarks
The allowable range for Gamma values is 1.0 to 5.0. After invoking this method be sure to do a Refresh on the control to display the alterted image.

GetFileName Method
Example

Description
When invoked cause the control to display it's File Open dialog.

Usage
ImageControl.**GetFileName**

Remarks
Invoke this method to display the control's file open dialog. The dialog is the Windows 3.1 common file open dialog and is configured to list all image formats which the control can display.

The return value is **TRUE** if the user selected an image otherwise if the user cancelled the dialog the return value is **FALSE**.

Using the GetFileName method

```
If ImageMan1.GetFileName Then
	' User loaded an image so lets process it
	ImageMan1.Invert
	Imageman1.Refresh
Else
	' User cancelled the File Open dialog
Endif
```

Green Property

Example

Description
Sets the green value for a color entry in the image's palette.

Usage
ImageControl.**Green**(Index%)[= Setting%]

Remarks
Use this property along with the Red and Blue properties to change the colors in a 256 color image. After changing these properties you must call the Refresh method to redraw the image with the new colors. The Index% value must be between 0 and the value of the PaletteEntries property. The allowable values for the property are 0 to 255.

Data Type
Integer Array

hDC Property
Example

Description
Returns a Windows Device Context for drawing into the image in the control.

Usage
ImageControl.**hDC**

Remarks

This property must be obtained for each image and allows you to use the Windows API functions to draw into the image. The changes made by using the hDC property change the internal bitmap maintained by the control. The altered image can be printed, saved or copied to the clipboard.

This property is read-only and available only at runtime.

Data Type
Integer

Using the hDC Property

```
Dim ImghDC

'Declare some functions from the Windows GDI API
Declare Sub MyMoveTo Alias "LineTo" Lib "GDI" ( ByVal hDC%, ByVal X%, ByVal Y% )
Declare Sub MyLineTo Alias "MoveTo" Lib "GDI" ( ByVal hDC%, ByVal X%, ByVal Y% )

' Draw an X through the image

ImghDC = ImageMan1.hDC

' These calls will draw a X over the image

MyMoveTo ImghDC, 0, 0
MyLineTo ImghDC, ImageMan1.ImageWidth, ImageMan1.ImageHeight

MyMoveTo ImghDC, ImageMan1.ImageWidth, 0
MyLineTo ImghDC, 0, ImageMan1.ImageHeight

' Update the image on screen

ImageMan1.Refresh

' Save the updated image
ImageMan1.SaveAs "c:\altered.bmp"
```

hWnd Property

Description
Returns the handle of the control's   Windows.

Usage
ImageControl.**hWnd**[= hWnd%]

Data Type
Integer

hDIB Property

Description
Returns or Sets the handle of a Windows global memory block containing the image in DIB format.

Usage
ImageControl.**hDIB**[= hDib%]

Remarks

The application must free the handle when it is done by calling the Windows API function, GlobalFree, with the value of the memory block returned from the hDIB property. Each time this property is queried the control will allocate a new memory block containing the image in DIB format.

Data Type
Integer

hImage Property
Example

Description
Returns or Sets the internal ImageMan/VB handle for an image.

Usage
ImageControl.**hImage** [= hImage%]

Remarks
This property is used to copy images between ImageMan/VB controls. By using the DstRight and DstBottom properties the image can be scaled as it is copied into the new control. By default the image is copied at the same size.

The portion of the image that is copied can also be specified by using the SrcLeft, SrcTop, SrcRight and SrcBottom properties. By default, these properties are set to the entire image.

Data Type
Integer

Using the hImage Property

```
' Make a 100x100 Thumbnail of the Image in ImageMan1 in ImageMan2
ImageMan1.DstRight = 100
ImageMan1.DstBottom = 100

ImageMan2.hImage = ImageMan1.hImage


' Resize the image in the control to half its original size
ImageMan1.DstRight = ImageMan1.ImageWidth / 2
ImageMan1.DstBottom = ImageMan1.ImageHeight / 2

ImageMan1.hImage = ImageMan1.hImage
```

ImageColors Property

Description
Returns the number of colors in the image.

Usage
ImageControl.**ImageColors**

Data Type
Single (Read Only)

Note
This is the number of colors in the image not the number of colors the display driver supports.

ImageFlags Property

Description
Returns a set of flags which describe the current image.

Usage
ImageControl.ImageFlags

Remarks
Currently only the IMG_VECTOR (1) flag is supported. If this bit is set then the image is a vector image and certain operations cannot be performed on it. Currently the following operations cannot be performed on vector images:

- Color Reduction
- Rotation
- Palette Access
- Gamma & Brightness adjustment
- Getting an hDIB for the image

Data Type
Integer (Read Only)

ImageHeight Property

Description
Returns the height of the image in image units. .

Usage
ImageControl.**ImageHeight**

Data Type
Single (Read Only)

ImageWidth Property

Description
Returns the width of the image in image units. This   is a read-only property.

Usage
ImageControl.**ImageWidth**

Data Type
Single

ImageXRes Property

Description
Returns the horizontal resolution of the image in dots per inch.

Usage
ImageControl.**ImageXRes**

Remarks
Some images may not contain resolution information therefore this property may be set to zero. Make sure to check for this condition before using this value. This value should be the same as the ImageYRes property in almost all images.

Data Type
Single (Read Only)

ImageCopy Method

Description
Copies the current image to the Windows clipboard.

Usage
ImageControl.**ImageCopy**

ImageCut Method

Description
Copies the current image to the Windows clipboard and clears the image from the control

Usage
ImageControl.**ImageCut**

ImagePaste Method

Description
Copies an image from the Windows Clipboard into the Image control.

Usage
ImageControl.**ImagePaste**

ImageYRes Property

Description
Returns the vertical resolution of the image in dots per inch.

Usage
ImageControl.**ImageYRes**

Remarks
Some images may not contain resolution information therefore this property may be set to zero. Make sure to check for this condition before using this value. This value should be the same as the ImageXRes property in almost all images.

Data Type
Single (Read Only)

IncreaseTo Method

Description
Causes the color depth of an image to be increased to the specified bit depth.

Usage
ImageControl.**IncreaseTo** *BitDepth*

Note
To decrease the number of colors in an image use the ReduceTo method.

Invert Method

Description
Invoking this method causes the colors in the image to be inverted.

Usage
ImageControl.**Invert**

Magnification Property
Example

Description
Sets or Returns the percentage the image should be scaled by when being displayed.

Usage
ImageControl.**Magnification**[= Percent%]

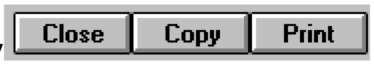
Remarks
The property should be set to the percentage scaling desired, i.e. to scale the image by 50% set this property to 50. This property should be used when the image is to be scaled by the same percentage on both axes. If each axis needs a different scale percentage then the ScaleWidth and ScaleHeight properties should be used to scale the image.

When the ScaleWidth or ScaleHeight properties have been set manually then this property's value will be invalid.

This property does not scale the image only its screen representation. Use the hImage and Dst properties to scale the actual Image.

Data Type
Integer

Using the Magnifcation Property



```
' Display the image at 50% of actual size
ImageMan1.Magnification = 50

' Do the refresh to redraw the Image
ImageMan1.Refresh
```

MirrorHorizontal Method

Description
Invoking this method causes the image to be mirrored horizontally.

Usage
ImageControl.**MirrorHorizontal**

MirrorVertical Method

Description
Invoking this method causes the image to be mirrored vertically.

Usage
ImageControl.**MirrorVertical**

MouseDown Event

Description
Occurs when the user presses a mouse button.

Syntax
Sub ctlname_**MouseDown**([Index as Integer,] Button as Integer, Shift as Integer, X as Single, Y as Single)

Remarks
MouseDown uses these arguments:

Argument	Description
Index	Uniquely identifies a control in a control array.
Button	Identifies which button was pressed. The Button argument is a bit field with bits corresponding to the left button (bit 0), right button (bit 1), and the middle button (bit 2) - values 1,2,4, respectively. Only one bit will be set, indicating which button caused the event.
Shift	The state of the Shift and Ctrl keys when the button was pressed. The Shift argument is a bit field with bits corresponding to the SHIFT key(bit 0), and the CTRL Key(bit 1) - values 1,2 respectively. One or both of the bits can be set indicating that one or both of the keys was pressed.X,Y. The current location of the mouse pointer. X and Y are always expressed in terms of the coordinate system set by the ScaleHeight, ScaleWidth, ScaleLeft and ScaleTop properties of the object.

Note
If the Select property is set to TRUE(-1) then no events will be generated for the left mouse button.

MouseIcon Property

Description
Sets a custom mouse icon to be used.

Usage
ImageControl.**MouseIcon** = Picture
ImageControl.**MouseIcon** = LoadPicture(Pathname$)

Remarks
This properties specifies the cursor to be used when the mouse pointer is over the image control. It may be set to the Picture property if the Image or Picture controls or used with the LoadPicture method to load a cursor or icon.

Data Type
Picture

MouseMove Event

Description
Occurs when the user moves the mouse.

Syntax
Sub ctlname_**MouseMove**([Index as Integer,] Button as Integer, Shift as Integer, X as Single, Y as Single)

Remarks
MouseMove uses these arguments:

Argument	**Description**
Index	Uniquely identifies a control in a control array.
Button	The state of the mouse buttons, in which a bit is set if the button is down. The Button argument is a bit field with bits corresponding to the left button (bit 0), right button (bit 1), and the middle button (bit 2) - values 1,2,4, respectively. All or some or none of these bits can be set.
Shift	The state of the Shift and Ctrl keys. The Shift argument is a bit field with bits corresponding to the SHIFT key(bit 0), and the CTRL Key(bit 1) - values 1,2 respectively. One or both of the bits can be set indicating that one or both of the keys was pressed
.X,Y	The current location of the mouse pointer. X and Y are always expressed in terms of the coordinate system set by the ScaleHeight, ScaleWidth, ScaleLeft and ScaleTop properties of the object.

Note
If the Zoom property is set to TRUE(-1) then no events will be generated for the left mouse button.

MousePointer Property

Description
Returns or sets a value indicating the type of mouse pointer displayed when the mouse is over the control.

Usage
*ImageMan1***.MousePointer** [= value]

Remarks

The settings for value are:

Setting	**Description**
0	(Default) Shape determined by the object.
1	Arrow.
2	Cross (cross-hair pointer).
3	I-Beam.
4	Icon (small square within a square).
5	Size (four-pointed arrow pointing north, south, east, and west).
6	Size NE SW (double arrow pointing northeast and southwest).
7	Size N S (double arrow pointing north and south).
8	Size NW SE (double arrow pointing northwest and southeast).
9	Size W E (double arrow pointing west and east).
10	Up Arrow.
11	Hourglass (wait).
12	No Drop.
99	Custom icon specified by the MouseIcon property.

You can use this property when you want to indicate changes in functionality as the mouse pointer passes over the control. The Hourglass setting (11) is useful for indicating that the user should wait for a process or operation to finish.

MouseUp Event

Description
Occurs when the user releases a mouse button

Syntax
Sub ctlname_**MouseUp**([Index as Integer,] Button as Integer, Shift as Integer, X as Single, Y as Single)

Remarks
MouseDown uses these arguments:

Argument	**Description**
Index	Uniquely identifies a control in a control array.
Button	Identifies which button was released. The Button argument is a bit field with bits corresponding to the left button(bit 0), right button (bit 1), and the middle button (bit 2) - values 1,2,4, respectively. Only one bit will be set, indicating which button caused the event.
Shift	The state of the Shift and Ctrl keys when the button was released. The Shift argument is a bit field with bits corresponding to the SHIFT key(bit 0), and the CTRL Key(bit 1) - values 1,2 respectively. One or both of the bits can be set indicating that one or both of the keys was pressed.
X,Y	The current location of the mouse pointer. X and Y are always expressed in terms of the coordinate system set by the ScaleHeight, ScaleWidth, ScaleLeft and ScaleTop properties of the object.

Note
If the Zoom property is set to TRUE(-1) then no events will be generated for the left mouse button.

Overwrite Property

Description
Specifies what occurs when writing a file and a file with the same name already exists.

Usage
ImageControl.**Overwrite**[= setting%]

Remarks
The Overwrite property settings are:

Setting	**Description**
0	Do not overwrite - image will not be saved.
1	Overwrite the existing file.
2	Display a messagebox to allow the user to cancel the save.

Note
If saving to a TIFF or DCX file with the **AppendImage** property set to **TRUE** this property will be ignored.

Data Type
Integer(Enumerated)

Pages Property

Description
Returns the number of images in the current file.

Usage
ImageControl.Pages

Remarks
Currently only TIFF and DCX image formats support multiple images.

Data Type
Integer (Read Only)

PageNumber Property

Description
Sets or Returns the number of the image in the current   file.

Usage
ImageControl.**PageNumber** [= num%]

Remarks
The allowable values for this property are zero to Control.Pages - 1.

Data Type
Integer

PaletteEntries Property

Description
Returns the number of   entries in an image's palette.

Usage
ImageControl.**PaletteEntries**

Remarks
Use this property with the Red, Green and Blue properties to get or set the colors for an image. This property will return zero for a 24 bit image.

Data Type
Integer (Read Only)

Picture Property

Description
Specifies the name of the image file to be displayed in the control.

Usage
ImageControl.**Picture**[= filename$]

Remarks
This property specifies the filename of the image to be displayed. If this property is set to a new value which is not an existing file, an error will occur. To clear a picture from the screen set this property to an empty string.

Data Type
String

PrnHdc Property

Example

Description
Specifies the printer device context used for printing images.

Usage
ImageControl.PrnHdc = Printer.hdc

Remarks
This property must be set to the value of the Printer   display context before printing an image. To do this set PrnHdc equal to   Printer.hdc.

Data Type
Integer

Note
Failure to set this property before printing an image can result in Unrecoverable Application errors.

Quality Property

Description
Specifies the quality factor used when saving JPEG images.

Usage
ImageControl.**Quality**[= setting%]

Note
The range of legal values for this property are 0 (best compression - worst quality) to 99 (worst compression - best quality).

Data Type
Integer

Red Property

Example

Description
Sets the red value for a color entry in the image's palette.

Usage
ImageControl.**Red**(Index%)[= Setting%]

Remarks
Use this property along with the Green and Blue properties to change the colors in a 256 color image. After changing these properties you must call the Refresh method to redraw the image with the new colors. The Index% value must be between 0 and the value of the PaletteEntries property. The allowable values for the property are 0 to 255.

Data Type
Integer Array

ReduceTo Method

Example

Description
Invoking this method causes the image to be converted to an image of the specified color format.

Usage
ImageControl.**ReduceTo** *nColors, bGreyScale, bOptimizedPalette*

Note
If the *bGreyScale* parameter is true the the image will be converted to a greyscale image. If *bGreyScale* is FALSE and the nColors parameter is set to a value of 16 or 256 then the *bOptimizedPalette* parameter will be used to determine the type of palette to be created.

Using the ReduceTo Method

```
' Convert an image to a 256 color image with an optimized palette

ImageMan1.ReduceTo 256, FALSE, TRUE

'Convert an Image to 16 color Greyscale
' In this case the bOptimizedPalette parameter is ignored since its is not applicable

ImageMan1.ReduceTo 16, TRUE, FALSE
```

Repaint Method

Description
Invoking this method causes the image to be redrawn on the screen.

Usage
ImageControl.**Repaint**

Note
This method is the same as the Refresh method but is supported in enviromnents other than Visual Basic.

Rotate Method

Example

Description
Invoking this method causes the image to be rotated to the specified angle.

Usage
nDegreesRotated = ImageControl.**Rotate** *nAngl, rgbColor*

Remarks
The rgbColor parameter specifies the color of the background when an image is rotated to a non ninety degree multiple.

This method returns the total number of degrees the image has been rotated.

Note
Each time the method is invoked, the specified transformation will take place.

Using the Rotate Method

Close Copy Print

```
' Rotate an image by 45 degrees and specify a Red background

ImageMan1.Rotate   45, RGB(255,0,0)

' Query the Image Width & Height again since the rotation changed them

ImgWidth = ImageMan1.ImageWidth
ImgHeight = ImageMan1.ImageHeight
```

SaveAs Method

Example

Description

Saves the current image or the specified portion into a file of the specified name and type.

Usage

*FileName$ = ImageControl.***SaveAs** *filename$*

Remarks

Invoking this method will cause the file to be written. If an empty string is specified,   a File Save dialog will be displayed so the user can select a file type and name.

If   a filename is specified, the name must contain a supported image extension since the extension is used to determine which format the image is saved in.

To save only a portion of an image use the SrcTop, SrcLeft, SrcRight and SrcBottom properties to specify the portion.

The return value is the name of the saved file or an empty string if the user selected cancel in the dialog.

SaveOptions Property
Example

Description
Specifies the options to be used when saving an image using the SaveAs method.

Usage
ImageControl.**SaveOptions** = *options*$

Remarks

The following options can be specified using the SaveOptions property:

TIFF_ROWSPERSTRIP

Specified the number of image rows per TIFF strip in the resulting image. Most applications will not need to manipulate this parameter.

TIFF_FILLORDER **1 | 2**

Specifies the Fill Order to be used when writing Group 3 or Group 4 compressed data. The default is 1.

Data Type
String

Saving Images using the SaveAs and SaveOptions Properties

```
' Save the current image and let the user select the name
' Use compression if supported by the saving image format
ImageMan1.SaveOptions = "COMPRESS   = ON"

' Save the Image, displaying a SaveAs dialog
ImageMan1.SaveAs = ""



' Save the image to a file called page1.tif using Group3 FAX compression

' Use chr$(13) to delimit the two option settings
ImageMan1.SaveOptions = "COMPRESS = ON"+CHR$(13)+"TIFF_COMPRESS = GROUP3"

ImageMan1.SaveAs = "c:\page1.tif"
```

ScaleHeight, ScaleWidth Properties

Description
Sets or returns the range of the vertical (ScaleHeight) and horizontal (ScaleWidth) axis for a control's internal coordinate system. The coordinate system is used for scaling and displaying the image in the control.

Usage
ImageControl.**ScaleHeight**[= scale!]
ImageControl.**ScaleWidth**[=scale!]

Remarks
By default, these properties are set to the width and height of an image in ScaleMode 0. If the ScaleWidth and/or ScaleHeight properties are set to less than the ImageWidth and/or ImageHeight properties then only a portion of the image will be displayed in the control. The ScaleLeft and ScaleTop properties can be used to scroll the image so another portion is displayed.

Note
All mouse, scroll, and zoom events generated by the control will have coordinates that are relative to the scaling mode in effect.

Data Type
Single

ScaleLeft, ScaleTop Property

Description
Sets or returns the horizontal (ScaleLeft) and vertical (ScaleTop) coordinates that describe the left and top corners of the control's internal area.

Usage
ImageControl.**ScaleLeft**[= scale!]
ImageControl.**ScaleTop**[=scale!]

Remarks
By default, these properties are set to 0. Modifying these properties will affect which portion of the image is displayed in the control. Changing these coordinates will cause the image to be scrolled in the control. These coordinates are automatically updated when the control has the scrollbar property enabled and scrollbars are displayed.

Data Type
Single

ScaleImage Method

Example

Description

Scales the current raster image.

Usage

*ImageControl.***ScaleImage** *nWidth, nHeight*

Remarks

Invoking this method cause the current image to be scaled to the specified width and height. This method is only valid for raster images.

Using the ScaleImage Method

```
' Load a big image and save a 100x100 thumbnail of it to another file
ImageMan1.Picture = "big.tif"

' Create the Thumbnail
ImageMan1.ScaleImage 100, 100

' Now save it
ImageMan1.SaveAs "thumb.tif"
```

ScaleMethod Property

Description
Specifies the method ImageMan/VB will use when scaling images.

Usage
ImageControl.**ScaleMethod**[= setting%]

Remarks
The ScaleMethod property settings are:

Setting	Description
0	Delete bits when scaling.
1	AND bits
2	OR bits
3	Use AntiAliasing when scaling

Data Type
Integer(Enumerated)

Scroll Event

Description
Occurs when the user scrolls the image using the built-in scrollbars.

Syntax
Sub ctlname_**Scroll**([Index as Integer,]   ScaleLeft as Single, ScaleTop as Single)

Remarks
Scroll uses these arguments:

Argument	**Description**
Index	Uniquely identifies a control in a control array.
ScaleLeft	The new value of the ScaleLeft property.
ScaleTop	The new value ScaleTop property.

Scrollbars Property

Description
Determines whether scrollbars will be displayed when only a portion of the image is displayed in the control.

Usage
ImageControl.**Scrollbars**[= Bool%]

Remarks
The Scrollbars property settings are:

<u>Setting</u>	**Description**
TRUE	Enables the scrollbars when the image is scaled so that it is larger than the control. The user can scroll the image by manipulating the scrollbars. When the user scrolls the image your application will receive a Scroll event.
FALSE	Disables scrollbars.

Data Type
Integer(Boolean)

SrcLeft, SrcTop Properties

Description
Specifies the coordinates of the upper left hand corner of the image which should appear in the bounding rectangle when printing an image.

Usage
ImageControl.**SrcLeft**[= left!]
ImageControl.**SrcTop**[= top!]

Remarks
These properties define what portion of the image is scaled to fit in the bounding rectangle defined by the Dst family of properties. By default, the SrcLeft and SrcTop properties are set to 0. SrcLeft must be set to a value between 0 and ImageWidth-1, while SrcTop must be set to a value between 0 and ImageHeight-1.

Data Type
Single

Note
If you want to change these values, you must change them   before invoking the Print method to print the image.

SrcRight, SrcBottom Property

Description
Specifies the coordinates of the lower right hand corner of the image which should appear in the bounding rectangle when printing an image.

Usage
ImageControl.**SrcRight**[= right!]
ImageControl.**SrcBottom**[= bottom!Remarks]

These properties define what portion of the image is scaled to fit in the bounding rectangle defined by the Dst family of properties. By default, the SrcRight and SrcBottom properties are set to ImageWidth-1 and ImageHeight-1, respectively. SrcRight must be set to a value between 0 and ImageWidth-1, while SrcBottom must be set to a value between 0 and ImageHeight-1.

Data Type
Single

Note
If you want to change these values, you must change them   before setting invoking the Print method to print the image.

Select Event
Example

Description
Occurs when the user selects a portion of the image by clicking and dragging and the Select property is set to TRUE.

Syntax
Sub ctlname_**Select**([Index as Integer,] X1 as Single, Y1 as Single , X2 as Single, Y2 as Single)

Remarks
Select uses these arguments:

Argument	**Description**
Index	Uniquely identifies a control in a control array.
X1,Y1	The coordinates of the upper left corner of the users selection. X1 and Y1 are always expressed in terms of the coordinate system set by the ScaleHeight, ScaleWidth, ScaleLeft and ScaleTop properties of the object.
X2,Y2	The coordinates of the bottom right corner of the users selection. X1 and Y1 are always expressed in terms of the coordinate system set by the ScaleHeight, ScaleWidth, ScaleLeft and ScaleTop properties of the object.

Status Event

Description
Occurs when loading or saving an image and the StatusEnabled property is TRUE.

Syntax
Sub ctlname_**Status**([Index as Integer,]   Percent As Integer)

Remarks
Status uses these arguments:

Argument	**Description**
Index	Uniquely identifies a control in a control array.
Percentage	Specifies the percentage of completion of the operation.

StatusEnabled Property

Description
Specifies whether Status events will be fired when loading and saving images.

Usage
ImageControl.**StatusEnabled**[= **TRUE** | **FALSE**]

Disabling Status events will speed up the loading and saving process.

Data Type
Boolean

UseDDB Property

Description
Determines whether the control will display an image using a Device Dependent Bitmap (DDB) or as a Device Independent Bitmap (DIB).

Usage
ImageControl.**UseDDB**[= Bool%]

Remarks
The default setting for this property is TRUE which enables the use of DDBs.

Using DDBs will speed screen redraw speed while comsuming more memory. On machines with small memory configurations setting this property to FALSE   will conserve memory.

Data Type
Integer(Boolean)

Select Property
Example

Description
Determines whether the control will display a rubber box when the user clicks and drags using the left mouse button. When the user releases the button a Select event will be generated.

Usage
ImageControl.**Select**[= Bool%]

Remarks
The Select property settings are:

Setting	**Description**
TRUE	The control will display a rubber box when the user clicks and drags the cursor with the left mouse button.
FALSE	All left mouse button activity will generate MouseDown, MouseMove and MouseUp events.

Data Type
Integer(Boolean)

Using the Select Event & Property

```
' Enabled the user to select an area of the image and copy the selected
' portion onto the clipboard. The Select property must be set to True.

Sub ImageMan1_Select (X1 As Single, Y1 As Single, X2 As Single, Y2 As Single)

        ' Set the Src properties to the portion the user selected
        ImageMan1.SrcLeft = X1
        ImageMan1.SrcTop = Y1
        ImageMan1.SrcRight = X2
        ImageMan1.SrcBottom = Y2

        ' Copy   the selected portion to the clipboard
        ImageMan1.ImageCopy

End Sub
```

hPalette Property

Description
Returns a handle to a Windows   Palette Object containing the palette for the image.

Usage
ImageControl.hPalette

Data Type
Integer (Read Only)

VBPicture Property
Example

Description
Sets or Returns the handle to an image in a format compatible with the Visual Basic Picturebox 'Picture' property.

Usage
ImageControl.**VBPicture**[= Picture]

Remarks
This property allows images to be moved between the ImageMan image control and the Visual Basic Picturebox and image controls.

Data Type
Integer

Using VBPicture Property

Close Copy Print

```
' Load an image into an ImageMan Control and then copy it to a picturebox

ImageMan1.Picture = "sample.tif"
Picture1.Picture = ImageMan1.VBPicture

' Copy an image from a Picture Control to the ImageMan image Control
Picture1.Picture = LoadPicture("sample.bmp")
ImageMan1.VBPicture = Picture1.Picture
ImageMan1.Refresh
```

TIFF   (Tag Image File Format)

ImageMan/VB supports single and multi page TIFF 5.0 files including the following compression schemes:

- No Compression
- Packbits
- LZW*
- Modified CCITT
- CCITT Group 3 1d & 2d
- CCITT Group 4

Supported color formats include:

- Monochrome
- 16 Color
- 256 Color
- 24 Bit Color

* Requires license from Unisys Corp.

PCX

Supported color formats include:

- Monochrome
- 16 Color
- 256 Color
- 24 Bit Color

Windows Bitmap

ImageMan/VB supports the following color formats:

- Monochrome
- 16 Color
- 256 Color
- 24 Bit Color

OS/2 formatted Bitmap files are also supported.

4 & 8 Bit RLE compressed bitmaps are also supported.

GIF

ImageMan supports all non-interlaced GIF files with up to 256 colors.

This format requires a license from Unisys Corp.

WMF (Windows Metafile)

ImageMan/VB supports all placeable Metafiles.

Targa

ImageMan/VB supports all version 1 and version 2 targa files with or without compression in the following color formats:

- 8 bit Greyscale
- 8 Bit Color
- 15/16/24/32 Bit color

EPS

ImageMan/VB supports all encapsulated Postscript files.

IMG (Gem Image Format)

ImageMan/VB supports compressed, monochrome IMG files.

JPG (JPEG Image Format)

ImageMan/VB supports JFIF standard JPEG files.

WPG (WordPerfect Graphics Format)

ImageMan/VB supports both raster and vector version 1.0 WPG files.

Version 2.0 files are not currently supported.

For Information on licensing the LZW compression code Contact:

Mark T Starr
Unisys Corporation
PO Box 500
Blue Bell, PA 19424-0001

Voice: 215-986-4411
Fax: 215-986-5721

You must obtain a license from Unisys before we can unlock the TIFF (w/LZW) and GIF readers and writers.

Can't Allocate Bitmap (Error 32003)

This error is fired by the control when attempting to use the VBPicture property and the picture is a bitmap. This should only occur under low memory situations.

Can't Get DDB Error (Error 32002)

This error is fired by the control when attempting to use the VBPicture property and the picture is a bitmap. This should only occur under low memory situations.

Can't Get WMF Handle(Error 32001)

This error is fired by the control when attempting to use the VBPicture property and the picture is a metafile. This should only occur under low memory situations.

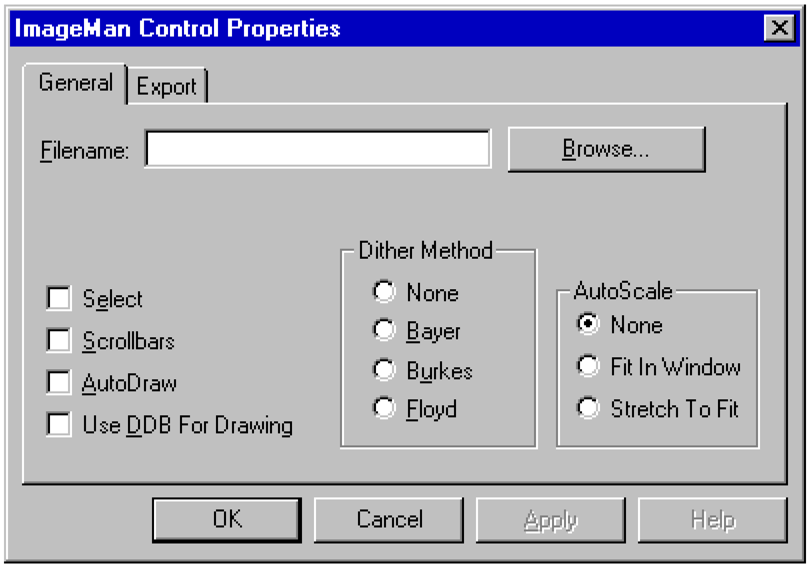
Output File Already Exists (Error 33210)

This error is fired when the control attempts to write an image to a file that already exists and the Overwrite property is set to a value of zero (Dont Overwrite).

No Image Loaded (Error 32000)

Certain properties require an image to be loaded into the control before the property can be used.

General Property Page


ImageMan Control Properties

General | Export

Filename:

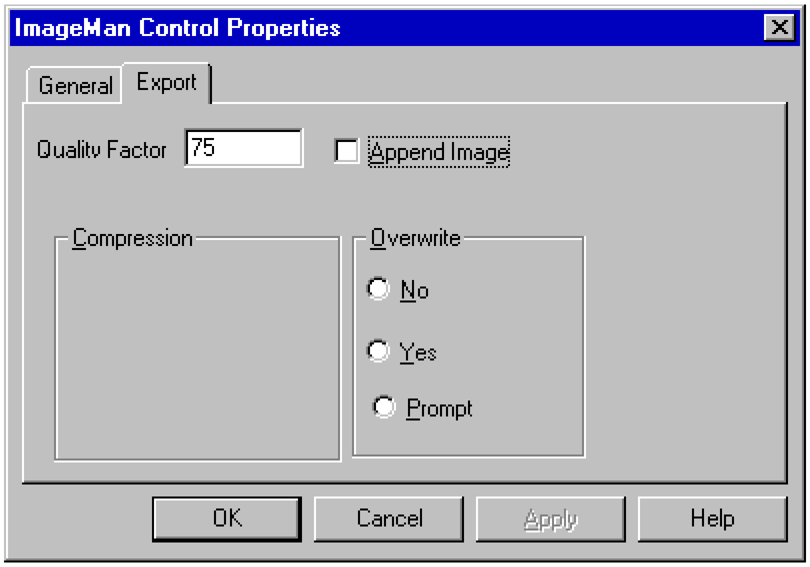
Browse...

- [ ] Select
- [ ] Scrollbars
- [ ] AutoDraw
- [ ] Use DDB For Drawing

Dither Method
- ○ None
- ○ Bayer
- ○ Burkes
- ○ Floyd

AutoScale
- ● None
- ○ Fit In Window
- ○ Stretch To Fit

Export Property Page



ImageMan TWAIN OLE Control

Properties Event Methods

Description

The Twain control provides the ability to acquire images from any TWAIN compatible scanners, frame grabbers and digital cameras.

Scanner Control Concepts
Obtaining Technical Support
Changes from the ImageMan VBX controls
ImageMan Image Control
About ImageMan Help

Scanner Control Concepts

Scanning

Scanning in initiated by invoking the Scan method. The *bShowUI* parameter of the method specifies whether the TWAIN source should display it's user interface dialog. Most scanners will honor this request although some inexpensive models do not. In this case there is no way to disable the dialog.

When a page has been successfully scanned, the Scan event will be fired. The hDIB parameter in the event contains a handle to the scanned image. When using the scanner control in conjunction with an ImageMan image control, this handle should be assigned to the ImageMan hDIB property. After doing this the image can be saved using the control if desired. The hDIB handle must be assigned to either an ImageMan control or to some other control or code. If not the image data and the memory it occupies will be lost. If you are passing the hDIB to a non ImageMan control then that code will; have to free the memory allocated for that image when it is done using it. This can be done using the GlobalFree Windows API function.

The Resolution, Brightness, Contrast, MaxPages, PixelType and UseADF properties can be set prior to invoking the Scan method to modify the scan paramaters.

Selecting a Scanner

Since some users may have more than one TWAIN device, the SelectScanner method can be invoked to display the TWAIN dialog to allow the user to select a scanner.

If TWAIN is not installed on the users system, the Select Method method will fire an error # 32000.

Custom Twain Control Properties

AppName	Brightness
Contrast	Device
MaxPages	PixelType
Resolution	ScanLeft
ScanTop	ScanRight
ScanBottom	Sources
SourceCount	UseADF

Custom Twain Control Methods

Scan   SelectScanner

Custom Twain Control Events

Scan

AppName Property
Example

Description
Specifies a string containing the name of the application.

Usage
TwainControl.**AppName**[= ApplicationName]

Remarks
Use this property to specify the name of the application which is scanning. Certain scanners will display this name in a dialog box when scanning.

Data Type
String

Brightness Property

Description
Specifies or returns the brightness to be used when scanning.

Usage
TwainControl.**Brightness**[= brightness%]

Remarks
The Brightness property can be set to a value of -1000 to 1000. A setting of zero will use the device's default brightness setting.

Data Type
Integer

Contrast Property

Description
Specifies or returns the contrast to be used when scanning.

Usage
TwainControl.**Contrast**[= contrast%]

Remarks
The Contrast property can be set to a value of -1000 to 1000. A setting of zero will use the device's default contrast setting.

Data Type
Integer

Device Property

Description
Specifies the name of the TWAIN device to be used when scanning.

Usage
TwainControl.**Device**[= DeviceName$]

Remarks
Setting this property is optional. By default the control will scan using the default twain device as set by the SelectScanner method.
To specify a scanner other than the default, set this property to the name of the TWAIN device to be used. A list of available device names can be found using the Sources and SourceCount properties.

Setting this property to an invalid name will generate an 'Invalid Property Value' runtime error.

Data Type
String

MaxPages Property

Description
Specifies the maximum number of pages to be scanned after invoking the Scan method.

Usage
TwainControl.**MaxPages**[= Pages%]

Remarks
This property is primarily of use with scanners that are equipped with Automatic Document Feeders (ADF) since it allows the application to specify how many pages should be scanned after invoking the Scan method. If specifying a value greater than one then the UseADF property should also be set to **TRUE**.

Data Type
Integer

PixelType Property

Description
Specifies the color format of the image data to be scanned.

Usage
TwainControl.PixelType[= ColorType%]

Remarks
The PixelType property settings are:

<u>**Setting**</u>	**Description**
-1	Specifies the device's default color format.
0	Black & White (1 Bit)
1	GreyScale (8 Bit)
2	RGB Color (24 Bit)
3	Palette Color (8 Bit)

Data Type
Integer (Enumerated)

Resolution Property

Description
Specifies the resolution in DPI to be used when scanning

Usage
TwainControl.**Resolution**[= dpi%]

Remarks
If the scanner doesn't support the specified resolution then it will select the closest resolution it can support.

Data Type
Integer

ScanLeft, ScanTop Properties

Description
Specifies the left and top coordinates of the scan area to acquire.

Usage
TwainControl.**ScanLeft**[= leftEdge]
TwainControl.**ScanTop**[= topEdge]

Remarks
These properties along with the ScanRight and ScanBottom properties define a bounding rectangle which defines the area of the image to be acquired when scanning. These values are expressed in inches.

Data Type
float

ScanRight, ScanBottom Properties

Description
Specifies the right and bottom coordinates of the scan area to acquire.

Usage
TwainControl.**ScanRight**[= rightEdge]
TwainControl.**ScanBottom**[= bottomEdge]

Remarks
These properties along with the ScanLeft and ScanTop properties define a bounding rectangle which defines the area of the image to be acquired when scanning. These values are expressed in inches.

Data Type
float

SourceCount Property

Example

Description
Returns the number of TWAIN devices found on the system.

Usage
TwainControl.**SourceCount**

Remarks
If this property has a value of zero then no TWAIN devices are installed and therefore no scanning can take place.

Data Type
Integer

Sources Property

Example

Description
Returns the names of the TWAIN devices found on the system

Usage
TwainControl.**Sources**(Index%)

Remarks

This array contains the names of all the installed TWAIN devices. The array contains SourceCount number of entries.

Data Type
String

Using the Sources and SourceCount properties

Close Copy Print

```
Dim I%

For I = 0 to Twain1.SourceCount - 1
        lstSources.AddItem Twain1.Source(I)        ' Add the source name to the listbox
named lstSources
Next I
```

UseADF Property

Description
Specifies whether paper should be fed from the scanner's Automatic Document Feeder (ADF).

Usage
TwainControl.**UseADF**[= TRUE | FALSE]

Remarks

Data Type
Boolean

Scan Method

Description
Initiates a scan.

Usage
TwainControl.**Scan** *bShowUI*

Remarks

The *bShowUI* parameter specifies if the scanner should show it's user interface when scanning. Some scanners ignore this flag and will always show their user interface.

The Scan Event will be fired for each page acquired from the device.

This method doesn't return a value.

SelectScanner Method

Description
Displays the TWAIN device selection dialog.

Usage
TwainControl.**SelectSources**

Remarks

This dialog allows the user to select the default TWAIN device. This dialog is provided by the TWAIN DLL so if twain is not installed then no action will occur when invoking this method.

This method doesn't return a value.

Scan Event
Example

Description
Occurs when the TWAIN device has acquired an image and when the device is closed.

Syntax
Sub ctlname_**Scan**([Index as Integer,] EventType as Integer, Status As Integer, hDIB as Integer)

Remarks
Scan uses these arguments:

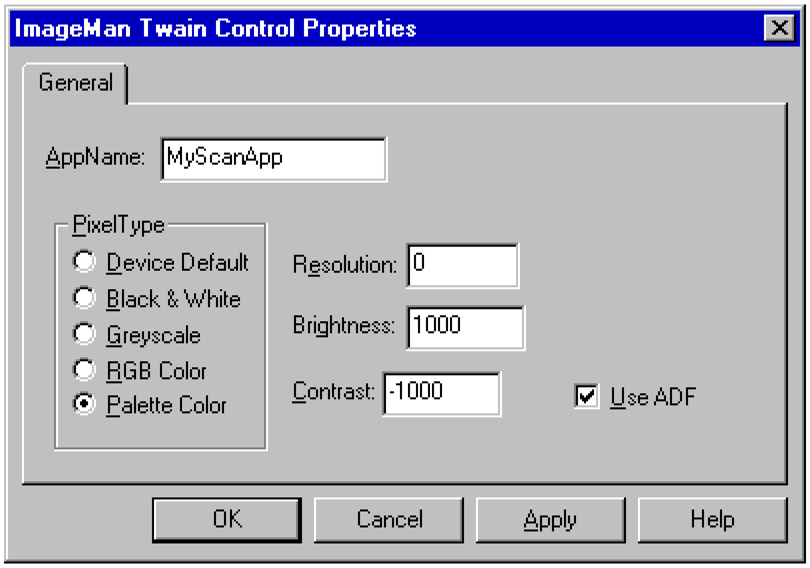
Argument	**Description**
EventType	Set to a value of 1 upon successful scanning of a page, set to a value of 2 when the TWAIN device has closed.
Status	Specifies if scanning completed successfully. A non zero value indicates a TWAIN error occurred.
hDIB	If the EventType is equal to 1 and Status is equal to 0 then this parameter contains the handle to a DIB for the acquired image. This handle would generally be assigned to the ImageMan Image Control's hDIB property. It is the responsibility of the application to free the memory associated with this handle (this is done automatically by the image control).

Using the Scan Event

```
Twain1_Scan( EventType As Integer, Status As Integer, hDIB as Integer )

        ' If the scan was successful then transfer the image to an imageman image control
        if EventType = 1 and Status = 0 Then
                ImageMan1.hDIB = hDIB
        Elseif EventType = 2 Then
                ' The scanning device has closed up and gone away
        Endif
End Sub
```

TWAIN Control Property Page



AppName: MyScanApp

PixelType
- Device Default
- Black & White
- Greyscale
- RGB Color
- [✓ Palette Color]

Resolution: 0

Brightness: 1000

Contrast: -1000

[✓ Use ADF]