

Pixel Translations File ActiveX Control

User's Guide
ImageBASIC 3.1

IMAGE  *BASIC*

Diamond Head Software, Inc.
1217 Digital Drive Ste. 125
Richardson, Texas 75081
(972) 479-9205

COPYRIGHT NOTICES

Under the copyright laws, neither the documentation nor the software may be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine-readable form, in whole or in part, without the prior written consent of Diamond Head Software, Inc., except in the manner described in the documentation.

This software product contains proprietary software components developed by a number of different software companies, referred herein as "Third Party Licensors". This documentation and the software that you purchased are protected by one or more of the following copyright notices:

Portions of this product, © 1993 - 1996 Diamond Head Software, Inc. All rights reserved.

Portions of this product, © 1996 Pixel Translations All rights reserved.

Company and product names mentioned in this documentation are trademarks or registered trademarks of their respective companies. Lotus and Lotus Notes are registered trademarks of Lotus Development Corporation. Windows is a trademark and Microsoft is a registered trademark of Microsoft Corporation.

DIAMOND HEAD SOFTWARE INC. AND ITS THIRD PARTY LICENSORS MAKE NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING THE SOFTWARE. DIAMOND HEAD SOFTWARE, INC. AND ITS THIRD PARTY LICENSORS DO NOT WARRANT, GUARANTEE OR MAKE ANY REPRESENTATIONS REGARDING THE USE OR THE RESULTS OF THE USE OF THE SOFTWARE IN TERMS OF ITS CORRECTNESS, ACCURACY, RELIABILITY, CURRENTNESS OR OTHERWISE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE IS ASSUMED BY YOU. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME JURISDICTIONS. THE ABOVE EXCLUSION MAY NOT APPLY TO YOU.

IN NO EVENT WILL DIAMOND HEAD SOFTWARE INC. OR ITS THIRD PARTY LICENSORS AND/OR THEIR DIRECTORS, OFFICERS, EMPLOYEES OR AGENTS BE LIABLE TO YOU FOR ANY CONSEQUENTIAL, INCIDENTAL OR INDIRECT DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, AND THE LIKE) ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE EVEN IF DIAMOND HEAD SOFTWARE INC. OR ITS THIRD PARTY LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU. Diamond Head Software Inc.'s and its Third Party Licensors' liability to you for actual damages from any cause whatsoever, and regardless of the form of the action (whether in contract, tort (including negligence), product liability or otherwise), will be limited to \$50.

Contents

Chapter 1 : Using the Document Control	1
Linking ImageBASIC Controls.....	1
Licensing Configuration and Verification.....	2
Where the Licenses are Kept.....	2
How the Licenses are Used.....	3
License Distribution Options.....	3
Introduction to the Document Control.....	4
Purpose of the Document Control.....	4
Definitions of Commonly Used Terms.....	5
Supported File Formats.....	6
Creating and Modifying a Document.....	7
Loading an Image from File.....	8
Loading an Image from Memory.....	9
Modifying Pages in an Existing Document.....	11
Document and Page Information.....	12
Image Data Output.....	13
Chapter 2 : Reference	17
Properties, Methods and Events.....	17
Appendix A : File Formats and Color Files	41
File and Compression Formats.....	41
File Formats.....	42
Compression Types.....	43
Color Limitations of File Formats.....	44
Comparative Charts of Colors Supported by File Format.....	44
Index	47

Chapter 1 : Using the Document Control

Linking ImageBASIC Controls

Most ImageBASIC controls can accept image data from other ImageBASIC controls. The process of designating where each ImageBASIC component gets its image data is referred to as linking the controls. With the exception of those controls that can directly access files or scanners, all ImageBASIC controls must be linked to another ImageBASIC control to get any image data.

Linking of controls is the primary method of moving an image through a series of processing steps. For example, an image may be originally captured through the PixScan control, passed to a TMSSequoia Display control for operator verification, optionally routed through a ScanFix control for enhancement, then to a Pixel Translations Document control to be written to disk, and finally to a TextBridge control for OCR processing to generate indexing information for that file.

Creating the Link

Those ImageBASIC controls that can accept image data from other ImageBASIC controls have a property named **ImageDataSource**. To create the data link between controls, this property must specify the ImageBASIC control that will be supplying image data. Any image that is received by the source control will also be sent to the linked control. For example, if the **ImageDataSource** property of a TextBridge control is set to a TMSSequoia Display control, each time a new image is loaded into the display window, the TextBridge control will receive that image.

Linking at Design Time

As each ImageBASIC control is added to a Form at design time, its **ImageDataSource** property is automatically set to an ImageBASIC control already on that Form. The assignment may be changed at design time by selecting from the drop-down list of available ImageBASIC components. This list is shown with the **ImageDataSource** property in the Properties Window or Object Inspector.

Linking at Runtime

Linking controls at runtime requires only one line of code that can be executed at any time. The source of image data can be changed during program execution by naming another ImageBASIC control in the **ImageDataSource** property, as shown here:

```
PixFile1.ImageDataSource = TMSDispl.Link
```

The **Link** property reports a unique identification string that is calculated for each instance of every ImageBASIC control as it is created. Each time the image data flowing from one ImageBASIC control changes, the receiving control's **ImageDataChanged** event is triggered. From this event, any procedure that is to be performed on each image can be started. For example, each time an OCR control's **ImageDataChanged** event occurs, an OCR attempt could be started on the new image data.

Licensing Configuration and Verification

Licensing in ImageBASIC is based on enabling a set number of concurrent seats. The number of seats that can use ImageBASIC is controlled by access to licenses. A license is an entry in a database that allows one computer to run a specific ImageBASIC component.

For example, there is a special type of license for the TMSSequoia Display control, another for TextBridge, another for ScanFix, etc. Each one of these licenses also comes in two varieties, runtime and development.

As is suggested by the names, *runtime licenses* are necessary for an *executable* to function properly, and *development licenses* are necessary to *develop* an application using ImageBASIC. A design time license will also function as a runtime license, which obviates the need to add runtime licenses for testing applications during development.

Where the Licenses are Kept

ImageBASIC will find licenses stored in either one of two locations -- in a licensing database or on a hardware key. The hardware key is plugged into the parallel port of the computer using ImageBASIC and will be automatically found each time an ImageBASIC component is used. The licensing database must be created on the site where it will be used and may not be moved from the location in which it is installed.

The inability to move a licensing database is one of its protection features. This attachment to a single location is formed when the licensing database is first created. Although it may not be moved once created, the licensing database can be written to a network drive to which all the machines running ImageBASIC have access. A file called IMGBASIC.INI must be on each of these networked machines. This file contains an entry pointing to the location of the licensing database. ImageBASIC can now search the licensing database for an available copy of each license it needs to run.

This licensing database may also be created on a local drive, activating ImageBASIC on only that one machine. The same INI file is still necessary to pinpoint the location of the database. If you opt to create a separate licensing database on each individual machine using ImageBASIC, it will of course contain only one copy of each type of license. This one copy is sufficient to activate any number of ImageBASIC-based applications on this one computer.

How the Licenses are Used

As each application that uses ImageBASIC is initiated, or the control is loaded into the development environment, the ImageBASIC licensing server attempts to find the proper license for each component. For example, if an executable has been developed that uses ImageBASIC to display existing images, and then calls on TextBridge to perform OCR on that image, that application must be able to find one available *TMSSequoia Display runtime license* and one available *TextBridge runtime license*.

If the application is successful in finding both these licenses, it will load normally and function exactly as it was programmed. When the application finds these licenses and is thereby informed that the correct licenses are available, it simultaneously locks the licenses so that no other application can use the same licenses to run at the same time. However, if two copies of these same licenses are available, another computer will be able to run the same application and will then lock the second license.

When an application that has locked one or more licenses terminates, the licenses are released and can immediately be taken by another user. This is the process by which concurrent licensing for any number of seats may be enabled. If the application ends abnormally -- the user might reboot or a concurrently running Windows application might lock up -- then the release of the licenses is conditional on the network/disk operating system.

For Novell networks, the default time for releasing locks made from a lost connection is five minutes. For other networks, your system administrator should be able to tell you how long the NOS takes to release the lock. In any case, the system administrator should be able to change this default release time to satisfy your particular needs.

If the licensing database has been installed on a stand-alone machine, the locks are immediately released and will again be available when the application is started again.

License Distribution Options

Because the ImageBASIC software is used in two distinct environments, development and runtime, different distribution methods are available to simplify

the installation of licenses in each of these sites. As far as the license verification algorithms in the ImageBASIC software are concerned, the application can run as long as it is licensed as described in the following sections.

Development License Distribution

For developers, the options are hardware licensing and software licensing. Hardware licensing is accomplished through the use of a device called a hardware key that plugs into any parallel port on the development machine. The hardware key separately enables each ImageBASIC component. *The hardware key is the default method for distribution of development licenses*

Under certain circumstances, developers may be unable to use a hardware key. In these cases, the developer can install a licensing database that contains development licenses. Installing this database and changing the database or the hardware key require the developer to run the Licensing Configuration Manager and to call Diamond Head to receive an authorization code that enables the installation or change.

Runtime License Distribution

For your clients who are running your compiled application, the same two sources of licensing authorization (the hardware key and the licensing database) are also available. Both have two primary control elements: a configuration program and an authorization program. The configuration program is run by the client and returns a unique site identification for that installation. The authorization program provides a corresponding authorization key to complete the installation.

Diamond Head Software (DHS) can provide you an application that generates the authorization code. Because you will be able to generate authorization codes for the installation of a runtime licensing database, your clients will call you for an authorization code, and DHS need not be directly involved.

Component Licensing is available for high-volume distribution or for circumstances that make the installation of a licensing database or the distribution of a hardware key unusually difficult. With this option, the application is licensed as it is compiled. It will then run without a hardware key or licensing database.

Introduction to the Document Control

Purpose of the Document Control

Each ImageBASIC component performs some specialized action such as image display, scanner control, or OCR. The Pixel Translations Document control

performs the file input and output. Because most other ImageBASIC controls cannot directly access images saved to file, the ultimate source of images will usually be either a Document control or a Scan control.

The Pixel Translations Document control is designed to provide the application developer the ability to create, manipulate and save documents formed from one or more separate image files or image sources. After its formation by the Document control, the document can then be referenced as a single source by other ImageBASIC controls. The formation of the virtual document includes the options to insert, delete and reposition pages within the document.

Image Input Options

The Document control can form a virtual document by several methods. For all methods, the source of the incoming images may be either a file or another ImageBASIC control:

- The incoming image may be used to form a new document after clearing any existing document.
- The image may be appended to the existing document
- The image may be inserted at an arbitrary point within the existing document.

Image Output Options

Once a document is formed, its constituent pages can be output either to another ImageBASIC control or written directly to file.

- The **SaveDocument**, **SavePage**, **SavePageError!**, **Bookmark not defined** and **SaveRegion** methods write the current document, page and region, respectively, to file.
- Other ImageBASIC controls can access document pages by setting the destination control's **ImageDataSource** property to specify the Pixel Translations Document control, as illustrated here for the Nestor ICR control:

```
Nestor1.ImageDataSource = PixFile1.Link
```

After an image has been read into memory by the Pixel Translations Document control, any other ImageBASIC component can then use that image, even if the underlying engine could not otherwise understand the file.

Definitions of Commonly Used Terms

The following terms are used in the following senses through this manual:

<i>document</i>	One or more image pages currently referenced by the Document control; a document may be compounded of pages from one or more sources including image files and other ImageBASIC controls
<i>file</i>	A valid image file on a Windows-accessible storage medium
<i>page</i>	A single image contained within a file or document; once added to a document, each page is no longer directly associated with its original source; as pages from each additional image source are added to an existing document, the entire collection of pages, regardless of their original source, can be referenced as a single source through the Document control.

Supported File Formats

The following file formats are supported by the Pixel Translations Document control. Some of these file types are supported only for reading, while some are supported for both reading and writing. The read-only file formats are indicated in the list below:

BMP Compressed
 CALS Type1 Group 4
 DCX
 JFIF
 PCX
 PCX
 TIFF Group 3 1-D
 TIFF Group 3 Modified Huffman
 TIFF Group 4
 TIFF Packbits
 TIFF Uncompressed

BMP is a common Windows format for storage of relatively small, low resolution color images. Because BMP files do not generally perform any compression, the files can be very large and have long access times if the image is complex or large.

CALS is a U.S. government file format. It is similar to TIFF in that the file is composed of header information followed by the compressed image data. CALS Type I supports only single page bitonal files and uses the CCITT Group 4 algorithm to compress the image data.

JFIF is an extended JPEG specification used for compression of high-color (24-bit) images.

PCX and *DCX* are commonly used Windows formats. *PCX* supports only single page files of up to 256 colors (paletted). *DCX* supports multiple page bitonal files.

The *TIFF* formats are the most common in document imaging. Group 4 typically offers the best compression ratios for bitonal images and is the default for output from the Pixel Translations Document control.

A more thorough description of these file formats and the level of color that each supports may be found in "Appendix A : File Formats and Color Files" on page 41.

Creating and Modifying a Document

In order for the Document control to output any image information either to file or to another ImageBASIC control, a document must be created. The next few pages detail the creation and modification of documents. For details on the output options once a document is available, refer to 'Image Data Output' on page 13.

- Creating a document can be as simple as loading a single file or a single page from another ImageBASIC control.
- Creating a document can also be as complex as loading several files, appending some to the end and inserting others in the middle of the document, and then replacing some of those pages with new pages, perhaps from a scanner, followed by rearranging the pages in this complex document.

Refer to the following sections for details on creating a document by loading images from file or from memory.

After a document is created, you can select a single page as the current, or active, page:

- At all times, the **PageCount** property reports the total number of pages in the current document.
- Select a single active page of the current document by setting the **PageIndex** property to an integer value from 1 to **PageCount**, inclusive.
- The active page, as selected through the **PageIndex** property, will be the page that is available to any other ImageBASIC control that is linked to the document control. The active page is also the point at which new images will be inserted and may be individually saved.

To link another control to the Document control, set the recipient control's **ImageDataSource** property to specify the link ID of the Document control. The link ID is a unique string calculated for each control as it is created. The link ID is reported in the **Link** property. For example, to display the active page in a Pixel Translations Display control that is on a different Form, link the controls together as illustrated here:

```
TMSDispl.ImageDataSource = Form1!PixFile1.Link
```

Several properties report information about the current page. This information ranges from the resolution and dimensions of the image to its color definition. Refer to 'Document and Page Information' on page 12 for a complete list of these informational properties.

Loading an Image from File

An existing image file may be opened and its contents made available through several methods. Depending upon whether you wish to start a new document or add to an existing document, use one of these methods to load the file:

Clear	Removes all current document information from memory and resets all document and page description properties to defaults. This method accepts no parameters: <code>PixFile1.Clear</code> Before executing this method, it is generally advisable to query the DocumentChanged and PageChanged properties to determine if the image has been modified since loaded.
LoadFile	Clears the current document, if any, and starts a new document containing all pages from the specified file. This method accepts a single parameter of the name of the file to load. The filename may include a relative or absolute path: <code>PixFile1.LoadFile "c:\images\img001.tif"</code>
AppendFile	Adds all pages in the specified file to the end of the current document. This method accepts a single parameter of the name of the file to load. The filename may include a relative or absolute path: <code>PixFile1.AppendFile "c:\images\img001.tif"</code>
InsertFile	Inserts all pages of the specified file beginning at the current PageIndex in the current document. This method accepts a single parameter of the name of the file to load. The filename may include a relative or absolute path: <code>PixFile1.InsertFile "c:\images\img001.tif"</code>

Note: When opening multiple-page files, images are loaded into memory only when explicitly instructed. For example, suppose you are reading a 4 (four) page file and set **PageIndex** to 3. The only pages actually read into memory are page 1 (one), because it is automatically loaded when the file is selected, and page 3 (three). Pages 2 (two) and 4 (four) are not read into memory until necessary.

Loading an Image from Memory

The Document control can accept images from both an existing image file and from another ImageBASIC control through the **ImageDataSource** property. The Document control's **ImageDataSource** property can specify any other ImageBASIC control that can serve as a source. The controls that typically supply image data to the Document control are Display controls and Scan controls.

The **ImageDataSource** property is set as illustrated below, linking the Document control to a Display control that is on another Form:

```
PixFile1.ImageDataSource = Form2!TMSDisp1.Link
```

Any image shown in the specified Display control will be available to the Document control. The image from one of these controls may be added to the existing document by executing one of the following methods:

Clear	Removes all current document information from memory and resets all document and page description properties to defaults. No parameters are required for this method: <code>PixFile1.Clear</code>
LoadPage	Clears the current document, if any, and starts a new document with the single page available from the control specified in the ImageDataSource property. No parameters are required for this method: <code>PixFile1.LoadPage</code>
LoadRegion	Clears the current document, if any, and starts a new document with the Working Region of the page available from the control specified in the ImageDataSource property. No parameters are required for this method: <code>PixFile1.LoadRegion</code>
AppendPage	Adds the page available from the control specified in the ImageDataSource property to the end of the current document. No parameters are required for this method: <code>PixFile1.AppendPage</code>
AppendRegion	Adds only the Working Region of the page available from the control specified in the ImageDataSource property to the end

of the current document. The region is added as a new page in the document. No parameters are required for this method:

`PixFile1.AppendRegion`

InsertPage

Inserts the page available from the control specified in the **ImageDataSource** property at the current **PageIndex**. No parameters are required for this method:

`PixFile1.InsertPage`

InsertRegion

Inserts the Working Region from the page available through the **ImageDataSource** property at the current **PageIndex**. No parameters are required for this method:

`PixFile1.InsertRegion`

Modifying Pages in an Existing Document

The Pixel Translations Document control can arrange, delete, copy, and move pages within the current document. For information on creating a new document rather than arranging the pages of the current document, refer to the following sections:

"Loading an Image from File" on page 8

"Loading an Image from Memory" on page 9

Page management operations are performed through the following methods:

CopyPage	Makes a copy of the page at the current PageIndex and inserts the copy at the specified index position. The index of all subsequent pages is modified and the PageCount property is updated. This method accepts a single integer parameter specifying the position of the new page in the document: <code>PixFile1.CopyPage <i>index</i></code>
DeletePage	Deletes the document page at the current PageIndex . No parameters are required for this method: <code>PixFile1.DeletePage</code>
MovePage	Moves the document page at the current PageIndex to the specified position. This method accepts a single integer parameter specifying the position of the new page in the document: <code>PixFile1.MovePage <i>index</i></code>
ReplacePage	Accepts the image from the control specified in the ImageDataSource property and replaces the current document page with the new page. No parameters are required for this method: <code>PixFile1.ReplacePage</code>
ReplaceRegion	Accepts only the current Working Region from the control specified in the ImageDataSource property and replaces the current document page with the region as a new page. No parameters are required for this method: <code>PixFile1.ReplaceRegion</code>

Document and Page Information

As each image page is loaded into memory, several read-only properties are populated with information about that image or the document. In a multiple-page document, any given page is selected by setting the **PageIndex** property. The informational properties are as follows:

DocumentChanged	Reports True if the document has been modified since the initial image or file was loaded.
PageChanged	Reports True if the current page has been modified (e.g., by annotation, cut & paste, etc.) since it was loaded.
PageCount	Reports the total number of pages in the document. The PageIndex property can be set to load any of these pages.
InputFileFormat	Reports the type of the input file. Will report a string similar to one of the following values: BMP CALS JPEG PCX TIFF
InputFileSize	Reports the number of bytes of image data in the current image, if loaded from file
PageBitsPerSample	Reports the bits per sample value for the currently loaded image
PageByteWidth	Reports the number of bytes of data describing the width of the image
PageHeight	Reports the pixel height of the image page specified in PageIndex
PagePhotoInterp	Reports the photometric interpretation value of the image page specified in PageIndex
PageSamplesPerPixel	Reports the samples per pixel value of the image page specified in PageIndex
PageWidth	Reports the pixel width of the image page specified in PageIndex
PageXRes	Reports the horizontal resolution in DPI of the image page specified in PageIndex

Image Data Output

Before the Document control can supply images to any other ImageBASIC control to write the images to file, a document must be created. Refer to 'Creating and Modifying a Document' on page 7 for a description of document creation.

Sending Images to Other Controls

When any document is defined within the Document control, the individual pages of that document are available for output either to file or to another ImageBASIC control.

- 1) To select a single page of a document as the active page, set the **PageIndex** property to any integer value between 1 and the total number of image pages in the file as reported in the **PageCount** property.
- 2) The specified page will then be sent to any other ImageBASIC control that is linked to the Document control. The ImageBASIC controls that can accept this image page must have their **ImageDataSource** property set equal to the **Link** property of the Document control.

For example, the following code segment will link a Pixel Display control, load a file and then select the second page of the newly created document for display:

```
PixDisp1.ImageDataSource = PixFile1.Link  
PixFile1.LoadFile "c:\images\form1.tif"  
PixFile1.PageIndex = 2
```

Saving Images to File

After creating a document of one or more pages, the Document control can save either individual pages or the entire document as a single file. For details on creating a document, refer to 'Creating and Modifying a Document' on page 7.

To save a page or an entire document to file, follow these steps:

- 1) Specify the Page to Save
- 2) Set Output Parameters
- 3) Write the File to Disk

1) Specify the Page to Save

If saving a single page or a region of a specific page, select the page from the existing document by setting the **PageIndex** property. This property may be set to any integer value between 1 and the number of pages in the document as reported in the **PageCount** property. For example, to select the second page of the document, set **PageIndex** as shown here:

```
PixFile1.PageIndex = 2
```

When saving the entire document, it is not necessary to select any particular page. The entire document will be saved without regard to the value of the **PageIndex** property.

2) Set Output Parameters

When saving a file, the following properties must be set before calling one of the save methods:

OutputFileFormat	Specifies the type of file to create. The enumerated options for this property are as follows: 2 CALS Type1 Group 4 12 PCX 18 TIFF Packbits 20 TIFF Group 3 1-D 21 TIFF Group 3 2-D 23 TIFF Group 3 Modified Huffman 24 TIFF Group 4 28 TIFF Uncompressed
OutputFileAppend	If True and the file already exists, the output will be appended after the last page in the file. Of the supported file types, TIFF allows the storage of multiple pages in a single file. If False and the output file already exists, the existing file will be overwritten.
OutputScalingDen	Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1). For example, setting OutputScalingDen to 2 and OutputScalingNum to 1 will create an output file of one-half the resolution and pixel dimensions.

OutputScalingNum	Specifies the destination half of a ratio defining the input pixel dimensions relative to the output pixel dimensions of an image as it is saved. Used in conjunction with OutputScalingDen to scale output data. Defaults to one (1).
OutputBitsPerSample	Specifies the bits per sample of the output file; if 0, the current value of the image data is used.
OutputSamplesPerPixel	Specifies the samples per pixel of the output file; if 0, the current value of the image data is used.
OutputPhotoInterp	Specifies the photometric interpretation of the output file. Refer to "Color Limitations of File Formats" on page 44 for information on the color levels supported by each file format and the definition of color using bits per sample, samples per pixel, and photometric interpretation.

3) Write the File to Disk

Depending upon whether you wish to save the entire document, a single page of the document, or a region of single document page, one of the following methods must be executed to write the image file.

SaveDocument	Saves all pages of the current document, applying the same output parameters listed above to all pages. If the specified output file type does not support multiple page files, an error will be generated. Accepts a single parameter specifying the file name, with or without path, of the file to write: <code>PixFile1.SaveDocument "c:\images\document.tif"</code>
SavePage	Saves only the page specified in the PageIndex property. All output parameter properties, as listed above, are applied to this single page. Accepts a single parameter specifying the file name, with or without path, of the file to write: <code>PixFile1.SavePage "c:\images\page.tif"</code>
SaveRegion	Saves only the Working Region of the page specified in the PageIndex property. All output parameter properties, as listed above, are applied to this single page. Accepts a single parameter specifying the file name, with or without path, of the file to write: <code>PixFile1.SaveRegion "c:\images\region.tif"</code>

Chapter 2 : Reference

Properties, Methods and Events

AboutBox Method

Definition:	Displays a message box showing version and copyright information when queried.
Syntax:	<code>PixFile1.AboutBox</code>
Data Type:	Long
Return Values:	None
Comments:	The message box is application modal and has a single OK button on it. The message box is unloaded when the button is clicked.

Active Property

Definition:	<p>If set to True at design time, the control will fully initialize and verify licensing immediately upon initialization of the runtime application.</p> <p>If set to False at design time, full initialization of the control will be delayed at initialization of the runtime application. In this case, this property must be explicitly set to True at runtime before the control is used.</p>
Data Type:	Boolean
Design Access:	Read/Write
Runtime Access:	Read/Write (see limits below)
See Also:	"Error! Reference source not found!" on page Error! Bookmark not defined.
Comments:	<p>If this property is set to True (the default) at design time, the control is fully initialized and licensing is verified immediately upon initialization of the application at runtime. The related technology libraries are loaded and the control is ready to be used.</p> <p>If this property is set to False at design time, the control will only partially initialize when the application loads at runtime. By delaying these two actions, the application should be able to load more quickly:</p> <ol style="list-style-type: none">1) The related technology libraries for the control will not be loaded.

- 2) The licensing server will not verify an available token for the control.

If the control initializes with **Active** set to False, this property must be explicitly set to True by the application. Until **Active** is set to True, the control will ignore all instructions to it.

If the control fails to find a license token, the **Active** property will be automatically set to False. The application can check this value on Form Load to determine if each control is licensed and can be used.

AppendFile Method

Definition: Reads the specified image file and appends its pages to the end of the current document.

Parameters: filename Name of the file to read

Syntax: `PixFile1.AppendFile filename`

Data Type: Long

Return Values: None

See Also: AppendPage Method, LoadFile Method, InsertFile Method

Comments: When successfully executed, the **PageCount** property is updated, and the current **PageIndex** remains unchanged. If a multiple-page file is specified, all of the pages in the file will be appended to the document.

The file name specified should generally include a fully qualified path. The drive specification in this property may include either a mapped drive letter or a UNC path.

The following file formats are supported for reading by the Pixel Translations Document control:

BMP Compressed

CALS Type1 Group 4

DCX

JFIF

PCX

PCX

TIFF Group 3 1-D

TIFF Group 3 Modified Huffman

TIFF Group 4

TIFF Packbits

TIFF Uncompressed

AppendPage Method

Definition:	Accepts the current image from the control specified in the ImageDataSource property and appends that page to the end of the current document.
Parameters:	None
Syntax:	<code>PixFile1.AppendPage <i>filename</i></code>
Data Type:	Long
Return Values:	None
See Also:	AppendRegion Method, AppendFileMethod, LoadPage Method, InsertRegion Method
Comments:	When successfully executed, the PageCount property is updated to reflect the additional page, but the PageIndex property remains unchanged.

AppendRegion Method

Definition:	Accepts the Working Region from the current image supplied by the control in the ImageDataSource property and appends that region to the end of the current document as another page.
Parameters:	None
Syntax:	<code>PixFile1.AppendRegion <i>filename</i></code>
Data Type:	Long
Return Values:	None
See Also:	AppendFile Method, LoadPage Method, InsertPage Method
Comments:	<p>When successfully executed, the PageCount property is updated to reflect the additional page, but the PageIndex property remains unchanged.</p> <p>If the new image page included additional information such as the definition of a Working Region or annotations, this information will be maintained by the Document control. The defined region may be saved as a separate file through the SaveRegion method. Any annotations associated with the image will be saved into the file header if the output file format is TIFF.</p>

Clear Method

Definition:	Removes all references to the current document from memory and resets all informational properties to defaults.
Parameters:	None
Syntax:	<code>PixFile1.Clear</code>
Data Type:	Long

Return Values: None

See Also: DeletePage Method, LoadFile Method, LoadPage Method

Comments: If any of the pages in the current document have changed since they were originally loaded, the **DocumentChanged** property will be True. This property may be queried at any time to determine if the current document has been changed and should be saved.

CopyPage Method

Definition: Copies the page at the current **PageIndex** to the specified index position.

Parameters: index Page index of the destination

Syntax: `PixFile1.CopyPage index`

Data Type: Long

Return Values: None

See Also: MovePage Method, DeletePage Method, PageIndex Property

Comments: The copied page is inserted in the current document at the index specified in the parameter to the method. The index for all subsequent pages will be incremented by one.

DeletePage Method

Definition: Deletes the page at the current **PageIndex**

Parameters: None

Syntax: `PixFile1.DeletePage`

Data Type: Long

Return Values: None

See Also: Clear Method, MovePage Method, PageIndex Property

Comments: Only the page at the current **PageIndex** will be deleted. The **PageCount** property will be updated to reflect the new number of pages in the document. The **PageIndex** will not change, so the next page in the file will become the active page.

DocumentChanged Property

Definition: Reports True if the current document has changed in any way since the initial page or file was originally loaded.

Data Type: Boolean

Design Access: Read-only

Runtime Access: Read-only

See Also: PageChanged Property

Comments: Any change to the document will cause this property to report True, indicating that a change has occurred. The change to the document include addition or deletion of pages, annotating a page, cut and paste to a page, and any other action that changes the image data.

Successful execution of any of the following methods will cause the **DocumentChange** property to be True:

- AppendFile
- AppendPage
- InsertFile
- InsertPage
- InsertRegion
- ReplacePage
- ReplaceRegion

Error Event

Definition: Occurs for each error internal to the control.

Parameters:

Number	A long error code that identifies the error
Description	Descriptive string of the error
SCode	A composite long number indicating the severity of the error, the facility code, the origin of the error, and the native error code
Source	Descriptive string of the source of the error
HelpFile	Suggested help file name that should have a detailed explanation of the error
HelpContext	Context ID in the help file
CancelDisplay	If set to True during this event, the standard error dialog will not be displayed

Comments: Any time an error occurs inside the Pixel Translations Document control, the **Error** event is triggered. As of this writing, if a runtime error that generated this event is trapped though the *OnError* statement, the built-in error dialog will not be displayed.

ImageDataChanged Event

Definition: Occurs each time the image data supplied to this control is changed.

Parameters: None

See Also: ImageDataSourceProperty

Comments: When the image data supplied by the control specified in the **ImageDataSource** property changes, this event is fired.

Typically, saving is triggered in this event when two or more ImageBASIC components are linked in a pipe to perform a series of processes.

ImageDataSource Property

Definition: Specifies the ImageBASIC control that will supply image data to this control.

Data Type: String

Design Access: Read/Write

Runtime Access: Read/Write

See Also: LoadPage Method, ImageDataChangedEvent

Comments: This property may be set to the Link ID of any ImageBASIC control that is capable of outputting image data. Each time the data supplied by that control changes, the Pixel Translations Document control is notified in the **ImageDataChanged** event. For example, to receive image data from a TMSSequoia Display control and then save that image, the Pixel Translations Document control must be linked as shown here:

```
PixFile1.ImageDataSource = TMSDispl.Link
```

The image data available through this property is used when the following methods are executed:

- AppendPage
- InsertPage
- InsertRegion
- LoadPage
- ReplacePage
- ReplaceRegion

InputFileFormat Property

Definition: Reports the format of the most recently loaded image file.

Data Type: String

Design Access: Not Available

Runtime Access: Read-only

See Also: InputFileName Property, InputFileSize Property

Comments: This property will always report the file type of the most recently open image file, even if additional image pages have been loaded through the **ImageDataSource** property.

When an existing image file is opened by the control, this property reports the format of the file. This property will report a value similar to one of the following:

BMP
CALS
PCX
TIFF

The following file formats are supported for reading by the Pixel Translations Document control:

BMP Compressed
CALS Type1 Group 4
DCX
JFIF
PCX
PCX
TIFF Group 3 1-D
TIFF Group 3 Modified Huffman
TIFF Group 4
TIFF Packbits
TIFF Uncompressed

InputFileName Property

Definition: When set to a valid image file name, the current document is cleared and the specified file is loaded as a new document.

Note: This property has been maintained for backward compatibility and has been superseded by the **LoadFile** method.

Data Type: String

Design Access: Read/Write

Runtime Access: Read/Write

See Also: LoadFile Method, PageIndex Property

Comments: The fully qualified path and file name should be supplied when setting this property. A relative path can be used, but care should be taken when doing so because of possible confusion because of changes in the current directory.

When set to a valid image file name, the following properties are updated:

- InputFileFormat
- InputFileSize
- PageCount

The **PageIndex** property is set to 1 and the first page of the specified file is loaded into memory. The following properties are populated with information about the currently loaded page:

PageBitsPerSample
PageByteWidth
PageHeight
PagePhotoInterp
PageSamplesPerPixel
PageWidth
PageXRes
PageYRes

The following file formats are supported for reading by the Pixel Translations Document control:

BMP Compressed
CALS Type1 Group 4
DCX
JFIF
PCX
PCX
TIFF Group 3 1-D
TIFF Group 3 Modified Huffman
TIFF Group 4
TIFF Packbits
TIFF Uncompressed

InputFileSize Property

Definition: Reports the size of the currently loaded image file in bytes.
Data Type: Long
Design Access: Not Available
Runtime Access: Read-only
See Also: LoadFile Method
Comments: This property reports the total byte size of the most recently loaded image file. For information on individual pages, refer to the **PageIndex** property.

InsertFile Method

Definition: Inserts the specified file starting at the page at the current **PageIndex**.
Parameters: filename Name of the file to insert; path is optional
Syntax: `PixFile1.InsertFile filename`
Data Type: Long
Return Values: None

See Also:	InsertPage Method, AppendFile Method, MovePage Method, PageIndex Property
Comments:	<p>If a path is not specified with the file name, or if the path is relative, the file will be opened based on the current directory. The drive specification in this property may include either a mapped drive letter or a UNC path.</p> <p>All pages of the specified file will be inserted starting at the current PageIndex. The first page of the newly loaded file will be the active page; i.e., it will be the document page specified by the PageIndex property. The PageCount property will be updated after the new images are added.</p> <p>The following file formats are supported for reading by the Pixel Translations Document control:</p> <ul style="list-style-type: none"> BMP Compressed CALS Type1 Group 4 DCX JFIF PCX PCX TIFF Group 3 1-D TIFF Group 3 Modified Huffman TIFF Group 4 TIFF Packbits TIFF Uncompressed

InsertPage Method

Definition:	Accepts the page available from the control specified in the ImageDataSource property and inserts the new page at the current PageIndex
Parameters:	None
Syntax:	<code>PixFile1.InsertPage</code>
Data Type:	Long
Return Values:	None
See Also:	ImageDataSource Property, PageIndex Property, InsertFile Method, InsertRegion Method, AppendPage Method, LoadPage Method
Comments:	Upon successful completion of the method, the PageCount property will be incremented by one, and the newly inserted page will be the active page; i.e., it will be the page specified by the PageIndex property.

InsertRegion Method

Definition:	Inserts the Working Region from the ImageDataSource at the current PageIndex .
Parameters:	None
Syntax:	<code>PixFile1.InsertRegion</code>
Data Type:	Boolean
Return Values:	True on success False on error
See Also:	ImageDataSourceProperty, InsertPage Method, InsertFile Method
Comments:	Upon successful completion of the method, the PageCount property will be incremented by one, and the newly inserted page will be the active page; i.e., it will be the page specified by the PageIndex property. Note: As of this writing, only the ImageBASIC Display controls may be used to specify a Working Region, therefore this method will be applicable only when the ImageDataSource property specifies an ImageBASIC Display control. An attempt to insert a region from a control that does not support region specification will result in an error.

Link Property

Definition:	Reports the Link ID calculated for this control at its creation.
Data Type:	String
Syntax:	<code>TMSDispl.ImageDataSource = PixFile1.Link</code>
Design Access:	Not Available
Runtime Access:	Read-only
Comments:	Each ImageBASIC control is assigned a unique Link ID at its creation. This Link ID can be specified in the ImageDataSource , DisplaySource , RegionSource , and AnnoteSource properties of various ImageBASIC controls. These source properties specify the ImageBASIC control that is supplying information or services to a control. Note: The Pixel Translations Document control may be specified only for the ImageDataSource and AnnoteSource property of other ImageBASIC controls.

LoadFile Method

Definition:	Clears the current document and starts a new document composed of all pages of the specified file.
Parameters:	Filename Name of the file to load; path is optional

Syntax: `PixFile1.LoadFile filename`

Data Type: Long

Return Values: None

See Also: AppendFile Method, InsertFile Method, LoadPage Method, Clear Method

Comments: The file name specified should generally include a fully qualified path. The drive specification in this property may include either a mapped drive letter or a UNC path.

Before this method is executed, it is advisable to query the **DocumentChanged** property. This property will report if the current document has been modified from its original form and should be saved.

The following file formats are supported for reading by the Pixel Translations Document control:

- BMP Compressed
- CALS Type1 Group 4
- DCX
- JFIF
- PCX
- PCX
- TIFF Group 3 1-D
- TIFF Group 3 Modified Huffman
- TIFF Group 4
- TIFF Packbits
- TIFF Uncompressed

LoadPage Method

Definition: Clears the current document and starts a new document composed of a single page that contains the image available from the **ImageDataSource** property.

Parameters: None

Syntax: `PixFile1.LoadPage`

Data Type: Boolean

Return Values: True on success
False on error

See Also: ImageDataSourceProperty, AppendPage Method, InsertPage Method, LoadFile Method

Comments: When a new page from another ImageBASIC control is loaded through this method, the application should generally save the existing document. For this purpose, the **DocumentChanged**

property will report if the document has been modified since the original page or file was loaded.

LoadRegion Method

- Definition:** Clears the current document and starts a new document composed of a single page that contains the Working Region of the image available from the **ImageDataSource** property.
- Parameters:** None
- Syntax:** `PixFile1.LoadRegion`
- Data Type:** Boolean
- Return Values:** True on success
False on error
- See Also:** ImageDataSource Property, AppendRegion Method, LoadPage Method, LoadRegion Method
- Comments:** When a new page from another ImageBASIC control is loaded through this method, the application should generally save the existing document. For this purpose, the **DocumentChanged** property will report if the document has been modified since the original page or file was loaded.

MovePage Method

- Definition:** Moves the page at the current **PageIndex** to the specified index.
- Parameters:** Index Page index of the destination position
- Syntax:** `PixFile1.MovePage index`
- Data Type:** Boolean
- Return Values:** True on success
False on error
- See Also:** PageIndex Property, CopyPage Method, DeletePage Method
- Comments:** The current page will be moved to the specified index, changing the index value for subsequent pages. The **PageIndex** property will not change, so the new page that is at that index will be the active page.

OutputBitsPerSample Property

- Definition:** Specifies the bits per pixel for the file that is to be saved through the **SaveDocument**, **SavePage** or **SaveRegion** method.
- Data Type:** Integer
- Design Access:** Read/Write
- Runtime Access:** Read/Write

See Also: OutputSamplesPerPixelProperty, OutputPhotoInterpProperty, SaveDocumentMethod

Comments: The color type of an image is defined by the **OutputBitsPerSample**, **OutputSamplesPerPixel** and **OutputPhotoInterp** properties. Refer to "Color Limitations of File Formats" on page 44 for a discussion of color and file types.

OutputFileAppend Property

Definition: If True, the image will be saved as the last page in the file specified in the call to the **SaveDocument**, **SavePage** or **SaveRegion** method. If the file does not exist, it will be created.

Data Type: Boolean

Design Access: Read/Write

Runtime Access: Read/Write

See Also: SaveDocumentMethod, OutputFileFormatProperty

Comments: Of the supported file types, only TIFF files support the creation of multiple pages in a single file. If another file format is selected, this property will be ignored and the specified output file will be overwritten if it exists.

OutputFileFormat Property

Definition: Specifies the file format of the output file.

Data Type: Enumerated

Design Access: Read/Write

Runtime Access: Read/Write

See Also: OutputFileAppendProperty, SaveDocumentMethod

Possible Values: See below

Comments: The following list shows the file formats supported by the Pixel Translations Document control and the valid values for this property:

- 0 BMP Compressed
- 2 CALS Type1 Group 4
- 4 DCX
- 6 JFIF
- 12 PCX
- 18 TIFF Packbits
- 20 TIFF Group 3 1-D
- 23 TIFF Group 3 Modified Huffman
- 24 TIFF Group 4
- 28 TIFF Uncompressed

Note: Different file format are limited in the level and type of color images that they support. Files saved by the Pixel Translations Document control will be of the color definition as specified in the following properties:

OutputBitsPerSample

OutputPhotoInterp

OutputSamplesPerPixel

Ensure that the file format selected supports the color format of the file. To change the color format of the file. Refer to 'Color Limitations of File Formats' on page 44 for a detailed chart.

OutputPhotoInterp Property

Definition: Specifies the photometric interpretation for the saved file.

Data Type: Enumerated

Design Access: Read/Write

Runtime Access: Read/Write

See Also: OutputBitsPerSampleProperty, OutputSamplesPerPixel Property, SaveDocumentMethod

Comments: The possible values for this property are as follows:

- 0 White0
- 1 White1
- 2 RGB
- 3 Paletted

Refer to "Color Limitations of File Formats" on page 44 for a more detailed discussion of color and file formats.

OutputSamplesPerPixel Property

Definition: Specifies the samples per pixel for the file that is written by the **SaveDocument**, **SavePage** or **SaveRegion** method.

Data Type: Integer

Design Access: Read/Write

Runtime Access: Read/Write

See Also: OutputBitsPerSampleProperty, OutputPhotoInterp Property, SaveDocumentMethod

Comments: The color type of an image is defined by the **OutputBitsPerSample**, **OutputSamplesPerPixel** and **OutputPhotoInterp** properties. Refer to "Appendix A : File Formats and Color Files" on page 41 for a discussion of color and file types.

OutputScalingDen Property

Definition:	Specifies the scaling factor applied to the incoming image data as it is saved.
Data Type:	Integer
Design Access:	Read/Write
Runtime Access:	Read/Write
See Also:	OutputScalingNumProperty, SaveDocumentMethod
Comments:	<p>Specifies one-half of the scaling ratio that is applied to image data as it is output to file. This property specifies the source half of a ratio defining the number of original pixels relative to output pixels. Used in conjunction with OutputScalingNum to scale output data for saving.</p> $\text{OutputScalingNum} / \text{OutputScalingDen} = \text{OutputSize} / \text{InputSize}$ <p>Scaling the image data will not alter the resolution or color definition of the data. Instead, only the width and height are modified.</p>

OutputScalingNum Property

Definition:	Specifies the scaling factor applied to the incoming image data as it is saved.
Data Type:	Integer
Design Access:	Read/Write
Runtime Access:	Read/Write
See Also:	OutputScalingDenProperty, SaveDocumentMethod
Comments:	<p>Specifies one-half of the scaling ratio that is applied to image data as it is output to file. This property specifies the destination half of a ratio defining the number of original pixels relative to output pixels. Used in conjunction with OutputScalingDen to scale output data for saving.</p> $\text{OutputScalingNum} / \text{OutputScalingDen} = \text{OutputSize} / \text{InputSize}$ <p>Scaling the image data will not alter the resolution or color definition of the data. Instead, only the width and height are modified.</p>

PageBitsPerSample Property

Definition:	Reports the bits per sample of the document page that is specified in the PageIndex property.
Data Type:	Enumerated

Design Access: Not Available

Runtime Access: Read-only

See Also: PageIndex Property, OutputBitsPerSampleProperty, PagePhotoInterp Property, PageSamplesPerPixelProperty

Comments: Used in conjunction with the samples per pixel and photometric interpretation, this value defines the level and type of color in the image page. Refer to "Appendix A : File Formats and Color Files" on page 41 for details on supported color types.

PageByteWidth Property

Definition: Reports the number of bytes required in the image file to describe the width of the current document page.

Data Type: Integer

Design Access: Not Available

Runtime Access: Read-only

See Also: PageWidth Property, PageIndex Property

Comments: This property is meaningful only when the file type and compression method of the image file requires the storage of image data in byte segments, such as files which use Group 3 and Group 4 compression.

PageChanged Property

Definition: Reports True if the current page has been modified since it was loaded. Reports False if the current page is unchanged.

Data Type: Boolean

Design Access: Not Available

Runtime Access: Read-only

See Also: DocumentChanged Property, PageIndexProperty, "Creating and Modifying a Document" on page 7

Comments: Any of the pages in the current document may be made the current page by setting the **PageIndex** property to an integer value between 1 (one) and **PageCount**.

PageCount Property

Definition: Reports the total number of pages in the current document.

Data Type: Integer

Design Access: Not Available

Runtime Access: Read-only

See Also: PageIndex Property, "Creating and Modifying a Document" on page 7

Comments: Any of the pages in the current document may be made the active page by setting the **PageIndex** property to an integer value between 1 (one) and **PageCount**.

PageHeight Property

Definition: Reports the pixel height of the image page specified in the **InputFileName** and **PageIndex** properties.

Data Type: Long

Design Access: Not Available

Runtime Access: Read-only

See Also: PageWidth Property, PageYRes Property, PageIndex Property

Comments: The height of an image is always reported in original image pixels without regard for any scaling performed on the image for display or saving (until the new image is loaded).

PageIndex Property

Definition: Specifies the active page in the current document.

Data Type: Long

Design Access: Not Available

Runtime Access: Read/Write

See Also: PageCount Property, See additional properties below

Comments: The **PageCount** property reports the total number of pages that are in the current document. Any of the available pages may be made the active page by setting the **PageIndex** property to a value between 1 and **PageCount**.

When the **PageIndex** is set to a valid value, the specified image page is loaded into memory, and the following descriptive properties are updated:

- PageBitsPerSample
- PageByteWidth
- PageChanged
- PageHeight
- PagePhotoInterp
- PageSamplesPerPixel
- PageWidth
- PageXRes
- PageYRes

The following methods are affected by the **PageIndex** value:

- CopyPage
- DeletePage

InsertFile
InsertPage
InsertRegion
MovePage
SavePage
SaveRegion

PagePhotoInterp Property

Definition: Reports the photometric interpretation of the document page that is specified in the **PageIndex** property.

Data Type: Enumerated

Design Access: Not Available

Runtime Access: Read-only

See Also: PageIndex Property, OutputPhotoInterpProperty, PageBitsPerSampleProperty, PageSamplesPerPixelProperty

Comments: The possible values for this property are as follows:

- 0 White0
- 1 White1
- 2 RGB
- 3 Paletted

0--White0 indicates that white image pixels are represented in the binary image data as zeros (0). Darker colors are represented by higher values. Applies to bitonal and grayscale images.

1--White1 indicates that black image pixels are represented in the binary image data as zeros (0). Lighter colors are represented by higher values. Applies to bitonal and grayscale images.

2--RGB indicates that each color component (red, green, blue) is defined separately for each pixel. Requires a samples per pixel of three (3).

3--Paletted indicates that a color palette is available to define the pixel colors in this image. Applies to 16-color and 256-color images.

PageSamplesPerPixel Property

Definition: Reports the samples per pixel of the document page that is specified in the **PageIndex** property.

Data Type: Enumerated

Design Access: Not Available

Runtime Access: Read-only

See Also: PageIndex Property, OutputSamplesPerPixelProperty, PageBitsPerSampleProperty, PagePhotoInterp Property

Comments: Used in conjunction with the bits per sample and photometric interpretation, this value defines the level and type of color in the image page. Refer to "Appendix A : File Formats and Color Files" on page 41 for details on supported color types.

PageWidth Property

Definition: Reports the pixel width of the current document page.

Data Type: Integer

Design Access: Not Available

Runtime Access: Read-only

See Also: PageHeight Property, PageXResProperty, PageIndex Property

Comments: The width of an image is always reported in original image pixels without regard for any scaling performed on the image for display or saving (until the new image is loaded).

PageXRes Property

Definition: Reports the horizontal resolution in DPI of the current document page.

Data Type: Integer

Design Access: Not Available

Runtime Access: Read-only

See Also: PageYRes Property, PageWidthProperty, PageIndex Property

Comments: The resolution of the image file is typically the same horizontally and vertically. The most common exceptions to this rule are facsimile documents received electronically rather than printed. The resolution reported in this property is in the units specified in the file, typically dots per inch.

PageYRes Property

Definition: Reports the vertical resolution in DPI of the current document page.

Data Type: Integer

Design Access: Not Available

Runtime Access: Read-only

See Also: PageXRes Property, PageHeightProperty, PageIndex Property

Comments: The resolution of the image file is typically the same horizontally and vertically. The most common exceptions to this rule are facsimile documents received electronically rather than printed.

The resolution reported in this property is in the units of dots per inch.

ReplacePage Method

Definition: Replaces the document page at the current **PageIndex** with the entire page from the **ImageDataSource**

Parameters: None

Syntax: `PixFile1.ReplacePage`

Data Type: Boolean

Return Values: True on success
False on error

See Also: ReplaceRegion Method, InsertPage Method, DeletePage Method

Comments: The newly inserted page will be the active page (i.e., it will be specified by the current **PageIndex**).

ReplaceRegion Method

Definition: Replaces the document page at the current **PageIndex** with only the Working Region from the **ImageDataSource**

Parameters: None

Syntax: `PixFile1.ReplaceRegion`

Data Type: Boolean

Return Values: True on success
False on error

See Also: ReplacePage Method, InsertRegion Method, DeletePage Method

Comments: The newly inserted page will be the active page (i.e., it will be specified by the current **PageIndex**).

Save Method

Definition: Saves the document page at the current **PageIndex** to the specified file.

Note: This method is maintained for backward compatibility but has been superseded by the following methods:
SaveDocument
SavePage

Parameters: filename Fully qualified path and file name to write

Syntax: `PixFile1.Save filename`

Data Type: Long

Return Values: 0 on failure
1 on success

See Also:	SaveDocument Method	
Comments:	Several properties set parameters to the saving of any image file. These properties are as follows:	
	OutputFileFormat	Specifies the type of file to create
	OutputFileAppend	If True and the file already exists, the image will be saved as the last page in the file
	OutputScalingDen	Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1).
	OutputScalingNum	Specifies the destination half of a ratio defining the input pixel dimensions relative to the output pixel dimensions of an image as it is saved. Used in conjunction with OutputScalingDen to scale output data. Defaults to one (1).
	OutputBitsPerSample	Specifies the bits per sample of the output file
	OutputSamplesPerPixel	Specifies the samples per pixel of the output file
	OutputPhotoInterp	Specifies the photometric interpretation of the output file

SaveDocument Method

Definition:	Saves all pages of the current document.	
Parameters:	filename	Fully qualified path and file name to write
Syntax:	<code>PixFile1.SaveDocument filename</code>	
Data Type:	Boolean	
Return Values:	True on success False on error	
See Also:	SavePage Method, SaveRegionMethod	
Comments:	Several properties set parameters to the saving of any image file. These properties are as follows:	
	OutputFileFormat	Specifies the type of file to create
	OutputFileAppend	If True and the file already exists, the image will be saved as the last page in the file

OutputScalingDen	Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1).
OutputScalingNum	Specifies the destination half of a ratio defining the input pixel dimensions relative to the output pixel dimensions of an image as it is saved. Used in conjunction with OutputScalingDen to scale output data. Defaults to one (1).
OutputBitsPerSample	Specifies the bits per sample of the output file
OutputSamplesPerPixel	Specifies the samples per pixel of the output file
OutputPhotoInterp	Specifies the photometric interpretation of the output file

SavePage Method

Definition:	Saves the current page to the specified file.						
Parameters:	filename Fully qualified path and file name to write						
Syntax:	<code>PixFile1.SavePage filename</code>						
Data Type:	Boolean						
Return Values:	True on success False on error						
See Also:	SaveDocumentMethod, SaveRegionMethod, PageIndex Property						
Comments:	The current page is selected by setting the PageIndex property. Several properties set parameters to the saving of any image file. These properties are as follows: <table> <tr> <td>OutputFileFormat</td><td>Specifies the type of file to create</td></tr> <tr> <td>OutputFileAppend</td><td>If True and the file already exists, the image will be saved as the last page in the file</td></tr> <tr> <td>OutputScalingDen</td><td>Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1).</td></tr> </table>	OutputFileFormat	Specifies the type of file to create	OutputFileAppend	If True and the file already exists, the image will be saved as the last page in the file	OutputScalingDen	Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1).
OutputFileFormat	Specifies the type of file to create						
OutputFileAppend	If True and the file already exists, the image will be saved as the last page in the file						
OutputScalingDen	Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1).						

OutputScalingNum	Specifies the destination half of a ratio defining the input pixel dimensions relative to the output pixel dimensions of an image as it is saved. Used in conjunction with OutputScalingDento to scale output data. Defaults to one (1).
OutputBitsPerSample	Specifies the bits per sample of the output file
OutputSamplesPerPixel	Specifies the samples per pixel of the output file
OutputPhotoInterp	Specifies the photometric interpretation of the output file

SaveRegion Method

Definition:	Saves the Working Region of the current page to the specified file.
Parameters:	filename Fully qualified path and file name to write
Syntax:	<code>PixFile1.SaveRegion filename</code>
Data Type:	Long
Return Values:	0 on failure 1 on success
See Also:	SaveDocumentMethod, SavePage Method, PageIndex Property
Comments:	Several properties set parameters to the saving of any image file. These properties are as follows:
OutputFileFormat	Specifies the type of file to create
OutputFileAppend	If True and the file already exists, the image will be saved as the last page in the file
OutputScalingDen	Specifies the source half of a ratio defining the input pixel dimensions relative to output pixel dimensions. Used in conjunction with OutputScalingNum to scale output data. Defaults to one (1).
OutputScalingNum	Specifies the destination half of a ratio defining the input pixel dimensions relative to the output pixel dimensions of an image as it is saved. Used in conjunction with OutputScalingDento to scale output data. Defaults to one (1).

OutputBitsPerSample	Specifies the bits per sample of the output file
OutputSamplesPerPixel	Specifies the samples per pixel of the output file
OutputPhotoInterp	Specifies the photometric interpretation of the output file

Appendix A : File Formats and Color Files

File and Compression Formats

Most of us do not have a lot of experience with image data. We may know that our favorite fax program produces PCX files, and that other programs produce files called TIFF files, but we may not know much beyond that. Many of the features in ImageBASIC will be much more easily understood if we present a short primer on how image data is captured, stored and maintained.

When a black and white image is captured at the scanner, it is held as raster data, or a stream of pixels that map together to form a picture. This data stream is relatively large; e.g., the pixels making up a bitonal 8.5" X 11" image scanned at 300 DPI (dots per inch) resolution will take up about 1 megabyte. The scanner usually sends this data either to the CPU or to a special board for compression.

As the data reaches the CPU there are numerous compression options. Each compression option presents different trade-offs with respect to compactness and decompression speed. For data that is being sent to the screen for display, we use a form of compression called Run Length Encoding (RLE). Run Length Encoding is a form of compressing the data that provides for very rapid decompression for quick display. RLE reduces the file size from 1 megabyte to about 250K, so it is somewhat more costly in terms of memory size than some other options.

By default, Diamond Head Software uses TIFF Group 4 compression in files and RLE in memory. Other options are available, however, and an overview of the various compression methods is given in the next section.

File Formats

BMP File Format

A common Windows graphics format, BMP files can be read by almost any Windows-based image viewer. BMP files are paletted, and can represent up to 24-bit color. However, these files are not compressed and can therefore be very large when storing high color or high resolution images.

CALS File Format

CALS is a raster format for compressing bitonal image data. It was developed by the U.S. Department of Defense and is now in wide use throughout federal applications. The image data is either uncompressed or compressed using CCITT Group 4 algorithm. The image data is appended to a raster header block, very much like TIFF files. Although common in U.S. government document imaging applications, CALS is uncommon elsewhere.

GIF File Format

GIF has been popularized by years of use on CompuServe and other Internet resources. Typically grayscale or paletted, GIF images are widely supported and provide reasonable compression -- a typical GIF file will be 20% of the uncompressed data size. This file format can encode an image up to 64K X 64K pixels using from 1 to 8 bits of color.

GIF files use LZW compression and are, therefore, not recommended for use unless you have entered into a licensing agreement with UNISYS Corp., holders of rights to LZW.

JPEG File Format

JPEG is a compression method originally developed to compress photographs. As might be expected, most JPEG files are color or grayscale, not bitonal. JPEG is a "lossy" compression, meaning that some resolution and details are lost when an image undergoes compression. However, the loss of resolution is usually not visible and does not discourage the use of JPEG.

JPEG can encode an image of up to 24-bit color to a maximum size of 64K X 64K pixels. Many image files can be compressed to just 10% of their original size with very little net loss.

PCX / DCX File Format

PCX was developed and distributed by Microsoft and ZSoft and is common throughout all Windows installations. Image data is compressed using an RLE

scheme, and is therefore quick but not markedly efficient, particularly when storing high color images.

PCX can encode bitonal, 4-bit, 8-bit, or 24-bit color images up to 64K X 64K pixels. Typical compression for images or 16 colors or fewer is approximately 50%, while high color and complex images can actually be increased in size.

DCX is simply a version of PCX that allows for the storage of multiple images in a single file.

TIFF File Format

TIFF was initially developed by Aldus Corporation to standardize the storage of bitonal images in document imaging and desktop publishing applications. TIFF is a versatile format composed of a header record listing details of the image followed by the compressed image data. Image data in a TIFF can be compressed using a number of different algorithms, and the exact compression scheme is recorded in the header to allow for easier decompression.

TIFF Group 4, the default format for ImageBASIC, compresses typical documents to approximately 5% of the original size. Group 4 is a bitonal standard and will not reliably store any grayscale or color images.

Compression Types

CCITT Group 3 Compression

Group 3 compression is a standard developed by CCITT, a standards organization committee responsible for the sanctioning of many file compression methods. This technique uses a combination of RLE, differential, and Huffman encoding.

CCITT Group 4 Compression

CCITT Group 4 compression is similar to Group 3 except that Group 3 limits its compression to one dimension, while Group 4 compresses both horizontally and vertically. Group 4 compression results in the smallest file size of all the options available in ImageBASIC, which is why it is the default compression method for file saving.

LZW Compression

LZW is a compression algorithm first developed in the late 1970's that is dictionary-based. LZW compression substitutes portions of image data that have been seen before with shorter strings stored in its dictionary.

Run Length Encoding (RLE)

RLE (Run Length Encoding) is a simple compression algorithm that encodes a run of identical pixels as a count and the pixel value. This compression will generally result in a file size approximately one-quarter the original size. RLE compression is very fast but relatively inefficient; however, because of its speed advantages, it is used as the compression method for images held in memory by ImageBASIC and also in BMP files.

Uncompressed Image Files

Uncompressed images are simply that -- no compression has been applied to the raw image data. Therefore, the image files tend to be very large. For example, the raw data to describe a bitonal letter size image at 300 DPI is about 1 MB. Adding to the area of the image or to the level of color can drastically increase the file size and the time required to transfer and display it.

Color Limitations of File Formats

The level of color that can be saved in each file format is a function of the specification of that format. For example, TIFF Group 4 is a bitonal standard, but JPEG can store up to 24-bit color. The next section introduces the definition of color in image files and contains tables that shown the level of color that can be saved in each file type.

Comparative Charts of Colors Supported by File Format

In the following tables, the entries in the top row of each table indicate the color format of an image file. The three numbers indicate, respectively, BitsPerSample, SamplesPerPixel, and PhotometricInterpretation.

The left-most column indicates the file format. By finding the intersection of the file type and color format, you can find the most appropriate, fully functional combination for your needs.

BitsPerSample(BPS) indicates how many bits define the different values for each sample (see SamplesPerPixel) that defines a pixel's color. The only valid values are 1, 4, and 8.

SamplesPerPixel(SPP) indicates how many individual values are used to define a pixel's color. The only valid values are 1 and 3. A value of 1 indicates that this image is not RGB, leaving bitonal, gray scale, and paletted as possibilities. BitsPerSample and PhotometricInterpretation will indicate which of these possible formats is actually used.

PhotometricInterpretation(PI) indicates the interpretation of color values for display. The valid values of PhotometricInterpretation are as follows:

- 0 White 0 Typical bitonal/gray interpretation in which 0 indicates white and higher numbers are darker shades.
- 1 White 1 The inverse of White 0, in which 0 is interpreted as black and higher numbers are lighter shades.
- 2 RGB Red, Green, and Blue values of each pixel are specified; this setting requires a value of 3 for SamplesPerPixel
- 3 Paletted An array is constructed with a color assigned to each value in than array. This value is generally only applied to color images and limits the image to fewer colors than RGB, typically 16-color or 256-color.

The key to the results codes in the tables is as follows:

- Y The combination file type and color definition is fully functional
- ~ Recognizable image but palette is not converted properly
- N This combination does not work at all

Bitonal and Grayscale Image File Formats

	Binary BPS 1 SPP 1 PI 0/1	16-level gray BPS 4 SPP 1 PI 0/1	256-level gray BPS 8 SPP 1 PI 0/1
TIFF Group 4	Y	N	N
TIFF Group 3	Y	N	N
TIFF Packed	Y	Y	Y
TIFF Uncompressed	Y	Y	Y
CALS	Y	N	N
BMP	Y	N	Y
JPEG	N	N	N

Paletted and RGB Image File Formats

	16-color palette BPS 4 SPP 1 PI 3	256-color palette BPS 8 SPP 1 PI 3	24-bit RGB BPS 8 SPP 3 PI 2
TIFF Group 4	N	N	N
TIFF Group 3	N	N	N
TIFF Packed	Y	Y	Y
TIFF Uncompressed	Y	Y	Y
CALS	N	N	N

BMP	Y	Y	N
JPEG	N	N	Y

Index

A

- AboutBox Method 17
- Active Property 17
- Adding to a Document
 - AppendFile Method 8, 18
 - AppendPage Method 9, 19
 - AppendRegion Method 19
 - InsertFile Method 8, 24
 - InsertPage Method 10, 25
 - InsertRegion Method 26
 - ReplacePage Method 11, 36
 - ReplaceRegion Method 11, 36
- AppendFile Method 8, 18
- AppendPage Method 9, 19
- AppendRegion Method 19
- Arranging Pages of a Document
 - AppendFile Method 8, 18
 - AppendPage Method 9, 19
 - AppendRegion Method 19
 - CopyPage Method 11, 20
 - DeletePage Method 11, 20
 - InsertFile Method 8, 24
 - InsertPage Method 10, 25
 - InsertRegion Method 26
 - MovePage Method 11, 28
 - ReplacePage Method 11, 36
 - ReplaceRegion Method 11, 36
- B
- BMP File Format 6, 42
- C
- CALS File Format 6, 42
- Changes to Document
 - DocumentChanged Property 20
 - PageChanged Property 32
- Clear Method 8, 9, 19
- Color Definition 44
 - Bits Per Sample 44
 - OutputBitsPerSample Property 15, 28
 - OutputPhotoInterp Property 15, 30
 - OutputSamplesPerPixel Property 15, 30
 - Photometric Interpretation 45

- Samples Per Pixel 44
- Compression Options 41
- CopyPage Method 11, 20

D

- Data Input
 - ImageDataSource Property 1
- Definitions of Commonly Used Terms 5
- DeletePage Method 11, 20
- Development Licensing 4
- DocumentChanged Property 20

E

- Error Event 21
- Events
 - Error 21
 - ImageDataChanged 2, 21

F

- File Format
 - OutputFileFormat Property 14, 29
- File Formats Supported 6
- File Information
 - InputFileFormat 12, 22
 - InputFileSize 12, 24
 - PageBitsPerSample Property 31
 - PageChanged Property 32
 - PageCount 12
 - PageCount Property 32
 - PageHeight Property 33
 - PagePhotoInterp Property 34
 - PageSamplesPerPixel Property 34
 - PageWidth Property 35
 - PageXRes Property 35
 - PageYRes Property 35
 - Property 32
- Formats Supported 6

G

- GIF File Format 42
- Group 3 Compression 43
- Group 4 Compression 43

I

- Image Data Output 13

- ImageDataChanged Event 2, 21
- ImageDataSource Property 1, 8, 22
- Input Options 5
- InputFileFormat Property 12, 22
- InputFileName Property 23
- InputFileSize Property 12, 24
- InsertFile Method 8, 24
- InsertPage Method 10, 25
- InsertRegion Method 26
- Introduction to the Document Control 4

J

- JPEG File Format 42

L

- Licensing
 - Active Property 17
- Link Property 26
- Linking Controls
 - ImageDataChanged Event 2, 21
 - ImageDataSource Property 1, 22
- Load Time Improvement
 - Active Property 17
- LoadFile Method 8, 26
- Loading a Multiple Page File
 - PageIndex Property 33
- Loading an Image
 - InputFileName 23
- Loading an Image from File 8
- LoadPage Method 9, 27
- LoadRegion Method 28
- LZW Compression 43

M

- Methods
 - AboutBox 17
 - AppendFile 8, 18
 - AppendPage 9, 19
 - AppendRegion 19
 - Clear 8, 9, 19
 - CopyPage 11, 20
 - DeletePage 11, 20
 - InsertFile 8, 24
 - InsertPage 10, 25
 - InsertRegion 26
 - LoadFile 8, 26
 - LoadPage 9, 27
 - LoadRegion 28
 - MovePage 11, 28

- ReplacePage 11, 36
- ReplaceRegion 11, 36
- Save 15, 36
- SaveDocument 5, 37
- SavePage 5, 38
- SaveRegion 5, 39

Modifying a Document

- AppendFile Method 8, 18
- AppendPage Method 9, 19
- AppendRegion Method 19
- CopyPage Method 11, 20
- DeletePage Method 11, 20
- InsertFile Method 8, 24
- InsertPage Method 10, 25
- MovePage Method 11, 28
- ReplacePage Method 11, 36
- ReplaceRegion Method 11, 36
- MovePage Method 11, 28
- Multi-Page Files
 - OutputFileAppend Property 14, 29

N

- New Document LoadFile 8
- New Document LoadPage 9, 27
- New Document LoadRegion 28

O

- Output to File 13
- Output of Image Data
 - Link Property 26
- Output Options 5
- Output to Other ImageBASIC Controls
 - 13
- OutputBitsPerSample Property 15, 28
- OutputFileAppend Property 14, 29
- OutputFileFormat Property 14, 29
- OutputPhotoInterp Property 15, 30
- OutputSamplesPerPixel Property 15, 30
- OutputScalingDen Property 14, 31, 37, 38, 39
- OutputScalingNum Property 15, 31, 37, 38, 39

P

- PageBitsPerSample Property 31
- PageByteWidth Property 32
- PageChanged Property 32
- PageCount Property 12, 32
- PageHeight Property 33

- PageIndex Property 33
- PagePhotoInterp 34
- PageSamplesPerPixel Property 34
- PageWidth Property 35
- PageXRes Property 35
- PageYRes Property 35
- PCX / DCX File Format 42
- PCX File Format 7
- Properties
 - Active 17
 - DocumentChanged 20
 - ImageDataSource 1, 8, 22
 - InputFileFormat 12, 22
 - InputFileName 23
 - InputFileSize 12, 24
 - Link 26
 - OutputBitsPerSample 15, 28
 - OutputFileAppend 14, 29
 - OutputFileFormat 14, 29
 - OutputPhotoInterp 15, 30
 - OutputSamplesPerPixel 15, 30
 - OutputScalingDen 14, 31, 37, 38, 39
 - OutputScalingNum 15, 31, 37, 38, 39
 - PageBitsPerSample 31
 - PageByteWidth 32
 - PageChanged 32
 - PageCount 12, 32
 - PageHeight 33
 - PageIndex 33
 - PagePhotoInterp 34
 - PageSamplesPerPixel 34
 - PageWidth 35
 - PageXRes 35
 - PageYRes 35
- Purpose of the Document Control 4

R

- Reading a File 8
- Removing a Document
 - Clear Method 8, 9, 19
- ReplacePage Method 11, 36
- ReplaceRegion Method 11, 36
- Run Length Encoding (RLE) 44
- Runtime Licensing 4

S

- Save Method 15, 36
- SaveDocument Method 5, 37
- SavePage Method 38

- SaveRegion Method 5, 39
- Saving Color Files 44
- Saving Files
 - ImageDataSource Property 22
 - OutputBitsPerSample 15, 28
 - OutputFileAppend Property 14, 29
 - OutputFileFormat Property 14, 29
 - OutputPhotoInterp 15, 30
 - OutputSamplesPerPixel 15, 30
 - OutputScalingDen 14, 31, 37, 38, 39
 - OutputScalingNum 15, 31, 37, 38, 39
 - Save Method 15, 36
 - SaveDocument Method 37
 - SaveDocumentMethod 5
 - SavePage Method 5, 38
 - SaveRegion Method 5, 39
- Saving Images to File 13
- Scaling Output Files
 - OutputScalingDen Property 14, 31, 37, 38, 39
 - OutputScalingNum Property 15, 31, 37, 38, 39
- Sending Images to Other Controls 13
- Source of Image Data
 - ImageDataSource Property 8
- Starting a Document
 - LoadFile Method 8, 26
 - LoadPage Method 9, 27
 - LoadRegion Method 28
- Supported Color Formats 44
- Supported File Formats 6

T

- TIFF File Format 7, 43
- Timing of Licensing Verification
 - Active Property 17

U

- Uncompressed Image Files 44

V

- Version Information
 - AboutBox Method 17