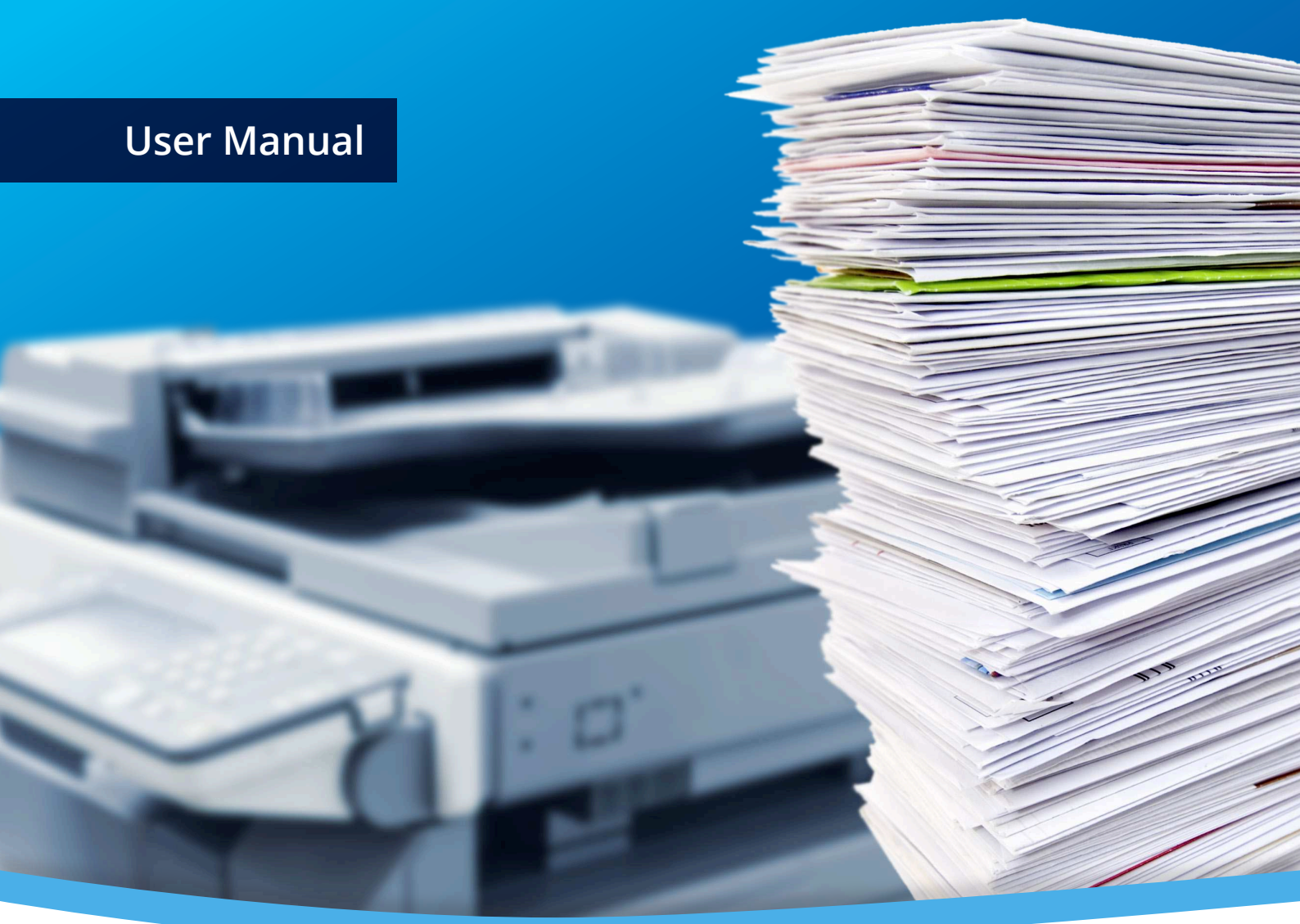
User Manual



3-Heights™ Image to PDF Converter API

Version 4.3

PDF-TOOLS.COM
Premium PDF Technology

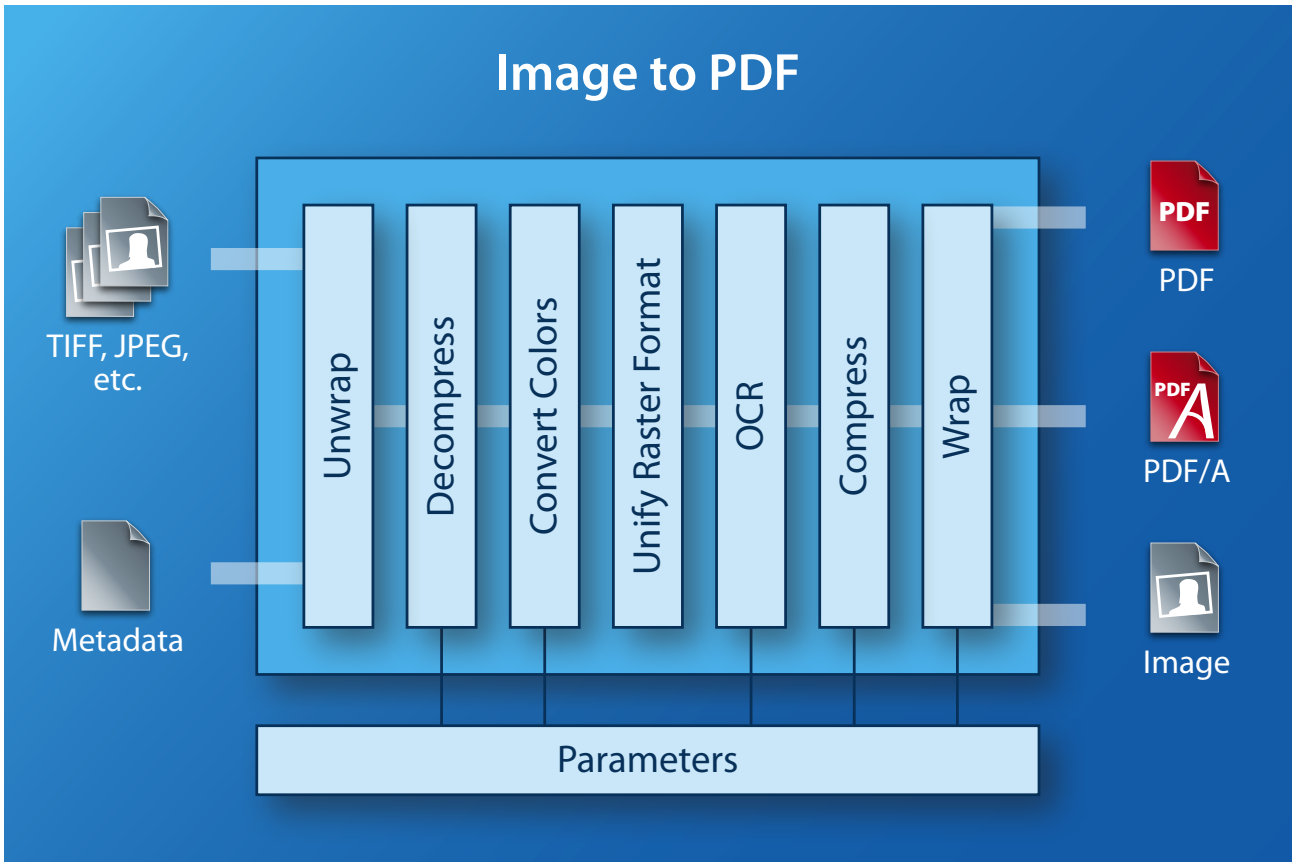
Contents

1 Introduction

1.1 Description

The 3-Heights™ Image to PDF Converter API converts raster image formats to PDF and PDF/A. PDF/A has been acknowledged world-wide as the ISO standard for long-term archiving since 2005. The Image to PDF Converter is used to convert images into a standardized format, for instance for electronic archiving or electronic data exchange.

It is also possible to include metadata from external sources. The Converter is characterized by a robust design, high throughput and accurate image reproduction. The optional OCR add-in makes output files searchable in full text mode.



1.2 Functions

The 3-Heights™ Image to PDF Converter API converts raster image formats such as JPEG, TIFF or PNG to PDF or PDF/A. It can merge pages from various image files to form a single PDF and can also split multi-page image files into single page PDF files. Further options include defining page size and resolution, image scaling and the inclusion of (external) metadata. Optical character recognition (OCR) is also available as an option.

Features

Image to PDF:

- Convert single page or multi-page raster images to PDF
- Definable output format (PDF/A-1, PDF/A-2, PDF/A-3, PDF 1.x)
- Automatic or selectable image compression, depending on the image type
- Automatic or selectable PDF page size
- Selectable page area
- Selectable image quality for lossy compression
- Set image position
- Set scaling
- Set standard resolution (DPI / X and Y coordinates)
- Set encryption and access rights
- Selectable and embeddable ICC color profile
- Define alternative texts (tagging) and image language
- Set document attributes
- Input and output document from file or memory
- Set cropbox for the generated PDF file
- Optional JPEG image recompression
- Set image orientation
- List available OCR engines
- Set OCR engine
- Set OCR engine language(s)
- Set options specific to OCR engine (performance optimization)
- Embedding metadata
- Support for image masks

Image to Image:

- Split single page or multi-page raster images into individual, single page images
- Merge multiple images to form one multi-page image
- Convert to an image format of the same color depth
- Modify TIFF image compression
- Set quality index for lossy image compression
- Create lossy and lossless JPEG2000 and JBIG2 images
- Read input and output document from file or memory

Formats

Input Formats:

- BMP (1, 2, 4, 8, 24 bit)
- GIF (2 to 8 bit)
- JBIG2 (lossy compression, lossless compression)
- JPEG, JPEG2000 and JPEG-LS (Grayscale, RGB)
- CMYK PBM and PNG (1 to 8, 24 bit)
- TIFF
  - Bitonal : uncompressed, CCITT G3, CCITT G3-2D, CCITT G4, LZW, ZIP, Packbits
  - Grayscale, RGB and CMYK: uncompressed, LZW, JPEG, JPEG (old), ZIP, Packbits

Output formats - Image to PDF Converter:

- PDF 1.x (e.g. PDF 1.4, PDF 1.5, etc.)
- PDF/A-1a, PDF/A-1b
- PDF/A-2a, PDF/A-2b, PDF/A-2u
- PDF/A-3a, PDF/A-3b, PDF/A-3u

Output formats - Image to Image Converter:

- All input formats plus EPS

Compliance

- Standards: ISO 19005-1 (PDF/A-1), ISO 19005-2 (PDF/A-2), ISO 19005-3 (PDF/A-3), ISO 32000 (PDF 1.7), TIFF V6
- Quality assurance: Isartor test suite

1.3 Interfaces

The following interfaces are available:

- C
- Java
- .NET
- COM

1.4 Operating Systems

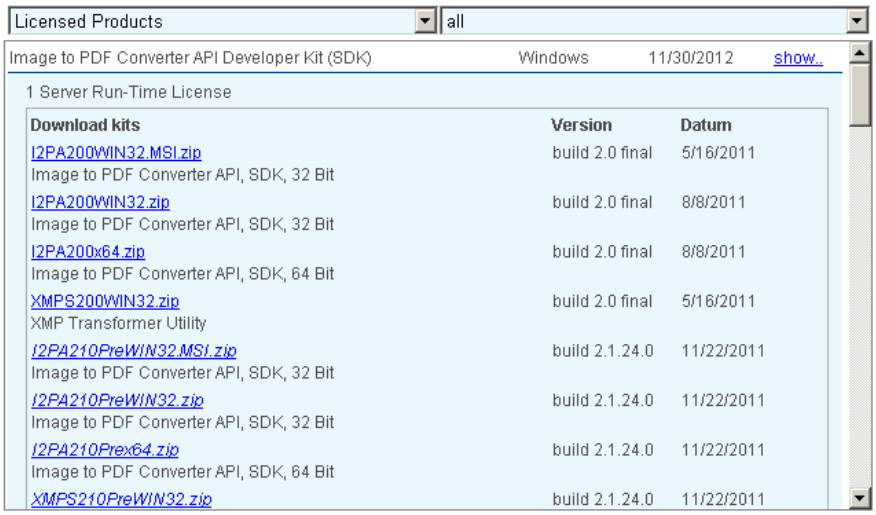
- Windows 2000, XP, Vista, 7 - 32 and 64 bit
- Windows Server 2003, 2008, 2008-R2 - 32 and 64 bit
- FreeBSD 4.7 for Intel
- HP-UX 11.0 - 32 bit and 32/64 bit Itanium
- IBM AIX (4.3: 32 Bit, 5.1: 64 bit)
- Linux (SuSE and Red Hat on Intel)
- Mac OS X
- Sun Solaris (2.7 and higher)

2 Installation

2.1 Download and Installation

The installation of the software requires the following steps.

1. You need administrator rights to install this software.
2. Log in to your download account at http://www.pdf-tools.com. Select the product “Image to PDF Converter API ”. If you have multiple versions available, select an SDK version. The download account will show you one or multiple download links. If you have no active downloads available or cannot log in, please contact pdfsales@pdf-tools.com for assistance.
   You will find product version of different builds available. We always suggest using a so called “Final Release” version, which is a well tested and stable version and labeled with “final”. Other versions are called “Pre-Release” and they normally contain new features and bug-fixes. We suggest using “Pre-Release” versions for evaluation and if you explicitly need a new feature or specific bug fix.



There are 32 and 64 bit versions available. The 32 bit version runs on both, 32 and 64 bit platforms. There is a zipped MSI (*MSI.zip) and a ZIP (*.zip) version available. The MSI (Microsoft Installer) provides an installation routine that installs and uninstalls the product for you. The ZIP version allows you to select and install everything individually. Download the version you wish to install.

3. If you select an MSI version, extract the MSI, start it and follow the steps in the installation routine. No further steps are needed.
   If you are using the ZIP version, follow the steps below.
4. Open the ZIP archive. Check the appropriate option to preserve file paths (folder names) and unzip the archive to a local folder (e.g. *C:\program files\pdf-tools*).
5. The unzip process now creates the following subdirectories:

   bin: Contains the runtime executable binary code and the .NET assemblies

   bin\fonts: Contains the fonts “Symbol” and “ZapfDingbats” and the font mapping file

   bin\icc: Contains color profile and link to download more color profiles

   doc: Contains documentation files

   include: Contains header files to include in your C / C++ project

   jar: Contains the Java wrapper when using JNI

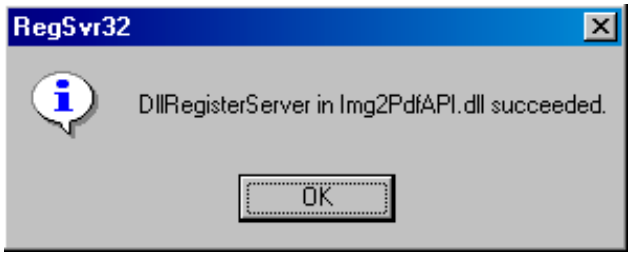
   lib: Contains the object file library to include in your C / C++ project

   samples: Contains samples in various programming languages

6. COM interface only: Before you can use the 3-Heights™ Image to PDF Converter API component in your COM application program you have to register the component using the `regsvr32.exe` program that is provided with the Windows operating system; it resides in the directory System32. If you are using a newer operating system, such as Vista or Windows 7, start the command prompt as Administrator. The following screenshot shows the registration of the Image to PDF Converter API DLL.

If you are using a 64-bit operating system and would like to register the 32-bit version of the 3-Heights™ Image to PDF Converter API, you need to use the regsvr32 from the directory `%SystemRoot%\SysWOW64` instead of `%SystemRoot%\System32`.[1]
If the registration process succeeds, the following dialog window is displayed:

[1] Otherwise you get the following message: *“LoadLibrary(“Img2PdfAPI.dll”) failed - The specified module could not be found.”*



The registration can also be done silently (e.g. for deployment) using the switch `/s`.
The other DLLs do not need to be registered, but for simplicity it is suggested that they are in the same directory as the *Img2PdfAPI.dll*.

2.2 Windows

General

Here is an overview of the relevant files that come with the 3-Heights™ Image to PDF Converter API Tool:

`bin\Img2PdfAPI.dll`	This is the DLL that contains the main functionality (required).
`bin\pdcjk.dll`	This DLL contains support for Asian languages (optional). It is loaded from the module path.
`bin\*NET.dll`	The .NET assemblies (required if using the .NET interface).
`bin\*.ocr`	These are an OCR interface DLLs that are used in combination with the 3-Heights™ OCR Enterprise Add-On which can be purchased as a separate product. Windows XP and later: This file must reside in the same directory as Img2Pdf.dll. Windows 2000: This file must reside in the same directory where the executable resides. E.g. if using .NET this is the directory where the compiled .exe is.
`lib\Img2PdfAPI.lib`	Import library for C programs.
`jar\I2PA.jar`	Java API archive.
`doc\Img2PdfAPI.idl`	COM interface definition.
`include\Img2PdfAPI.h`	C API include file.
`Include\Img2PdfAPI_c.h`	COM API include file.
`Include\Img2PdfAPI_i.c`	COM API identifier definitions.
`Include\pdferror.h`	Supplementary C header-file containing error codes.

2.3 Uninstall, Install a New Version

If you used the MSI for the installation, go to Start ->3-Heights™ Image to PDF Converter API ... ->Uninstall...
If you used the ZIP file: In order to uninstall the product undo all the steps done during installation, e.g. unregister using regsvr32 -u, delete all files, etc.

Installing a new version does not require to previously uninstall the old version. The files of the old version can directly be overwritten with the new version. If using the COM interface, the new DLL must be registered, un-registering the old version is not required.

2.4 Unix

Here is an overview of the shared libraries and other files that come with the Image to PDF Converter API:

Table: File Description	
Name	**Description**
`bin/libImg2PdfAPI.so`	This is the shared library that contains the main functionality.
`doc/*.*`	Documentation
`bin/I2PA.jar`	Java API archive.
`include/*.h`	C API include file.

Example code written in different programming languages are available at product page of the PDF Tools AG website (www.pdf-tools.com).

All Unix Platforms

1. Unpack the archive in an installation directory, e.g. */usr/pdftools.com/*
2. Copy or link the shared object into one of the standard library directories, e.g:
   `ln -s /usr/pdftools.com/bin/libImg2PdfAPI.so /usr/lib`
3. In case you have not yet installed the GNU shared libraries, get a copy of these from http://www.pdf-tools.com; extract the shared images and copy or link them into */usr/lib or /usr/local/lib*.

MAC OS/X

The shared library must have the extension .jnilib for use with Java. We suggest that you create a file link for this purpose by using the following command:

```
ln libImg2PdfAPI.dylib libImg2PdfAPI.jnilib
```

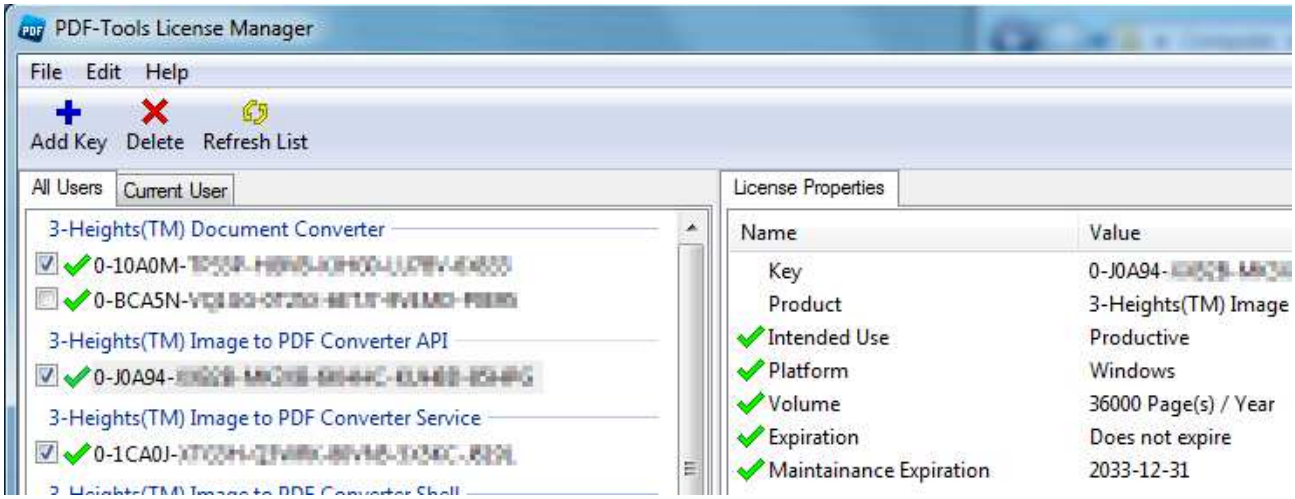
3 License Management

There are three possibilities to pass the license key to the application:

1. The license key is installed using the GUI tool (Graphical user interface). This is the easiest way if the licenses are managed manually. It is only available on Windows.
2. The license key is installed using the shell tool. This is the preferred solution for all non-Windows systems and for automated license management.
3. The license key is passed to the application at runtime via the “LicenseKey” property. This is the preferred solution for OEM scenarios.

3.1 Graphical License Manager Tool

The GUI tool *LicenseManager.exe* is located in the *bin* directory of the product kit.



List all installed license keys

The license manager always shows a list of all installed license keys in the left pane of the window. This includes licenses of other PDF Tools products. The user can choose between:

- Licenses available for all users. Administrator rights are needed for modifications.
- Licenses available for the current user only.

Add and delete license keys

License keys can be added or deleted with the “Add Key” and “Delete” buttons in the toolbar.

- The “Add key” button installs the license key into the currently selected list.
- The “Delete” button deletes the currently selected license keys.

Display the properties of a license

If a license is selected in the license list, its properties are displayed in the right pane of the window.

Select between different license keys for a single product

More than one license key can be installed for a specific product. The checkbox on the left side in the license list marks the currently active license key.

3.2 Command Line License Manager Tool

The command line license manager tool `licmgr` is available in the bin directory for all platforms except Windows.

A complete description of all commands and options can be obtained by running the program without parameters:

```
licmgr
```

List all installed license keys

```
licmgr list
```

Add and delete license keys

Install new license key:

```
licmgr store X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Delete old license key:

```
licmgr delete X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Both commands have the optional argument `-s` that defines the scope of the action:

- `g`: For all users
- `u`: Current user

Select between different license keys for a single product

```
licmgr select X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

3.3 License Key Storage

Depending on the platform the license management system uses different stores for the license keys.

Windows

The license keys are stored in the registry:

- `HKLM\Software\PDF Tools AG` (for all users)
- `HKCU\Software\PDF Tools AG` (for the current user)

Mac OS X

The license keys are stored in the file system:

- `/Library/Application Support/PDF Tools AG` (for all users)
- `~/Library/Application Support/PDF Tools AG` (for the current user)

Unix/Linux

The license keys are stored in the file system:

- `/etc/opt/pdf-tools` (for all users)
- `~/.pdf-tools` (for the current user)

Note: The user, group and permissions of those directories are set explicitly by the license manager tool. It may be necessary to change permissions to make the licenses readable for all users. Example:

```
chmod -R go+rx /etc/opt/pdf-tools
```

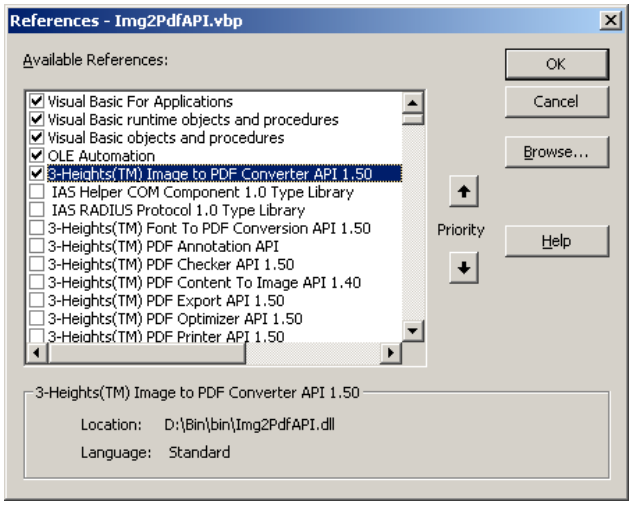
4 Programming Interfaces

4.1 Visual Basic 6

After installing the 3-Heights™ Image to PDF Converter API and registering the COM interface (see chapter Download and Installation), you find a Visual Basic 6 example *Img2PdfAPI.vbp* in the directory *samples/VB/*. You can either use this sample as a base for an application, or you can start from scratch.

If you start from scratch, here is a quick start guide for you:

1. First create a new Standard-Exe Visual Basic 6 project. Then include the 3-Heights™ Image to PDF Converter API component to your project.



2. Draw a new Command Button and optionally rename it if you like.
3. Double-click the command button and insert the few lines of code below. All that you need to change is the path of the file name.

Example:

```
Private Sub Command1_Click()
  Dim conv As New IMG2PDFAPILib.Img2Pdf
  conv.Create App.Path & "\output.pdf"
  conv.CreatePageFromImage App.Path & "\input.jpg"
  conv.Close
End Sub
```

The four steps of the above code are very simple: (1) Create a Img2Pdf object, (2) create an PDF file for output, (3) open an image file for input and copy its page(s), (4) close PDF- and image file. And that's all - a few lines of code. To modify your program and set options, consult the Reference Manual section.

4.2 C/C++

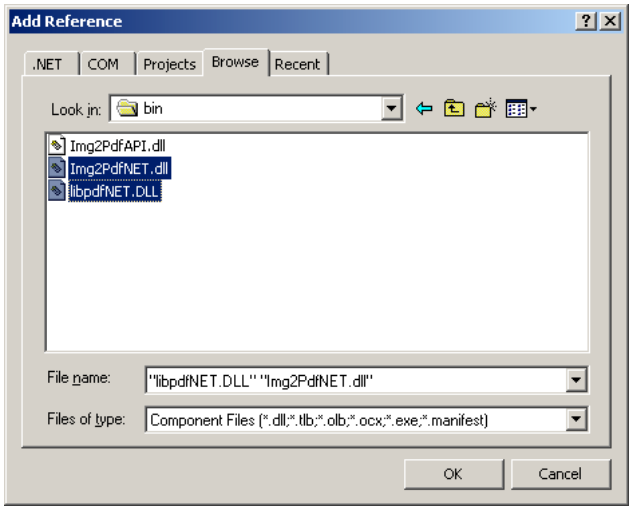
The header file *img2pdf_c.h* needs to be included in the C program. The library *lib\img2pdf.lib* needs to be linked to the project.

4.3 .NET

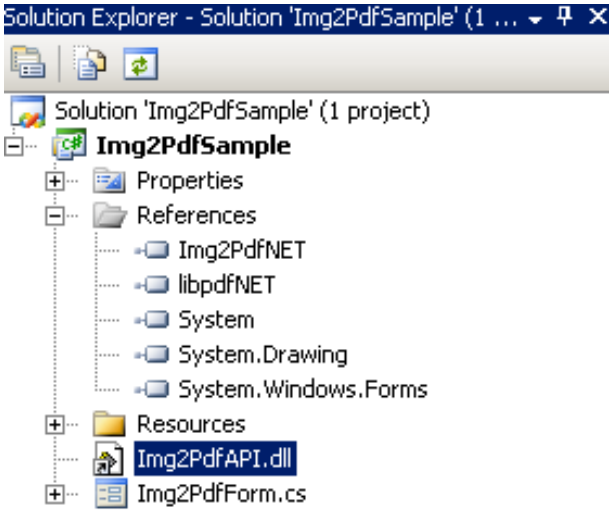
References

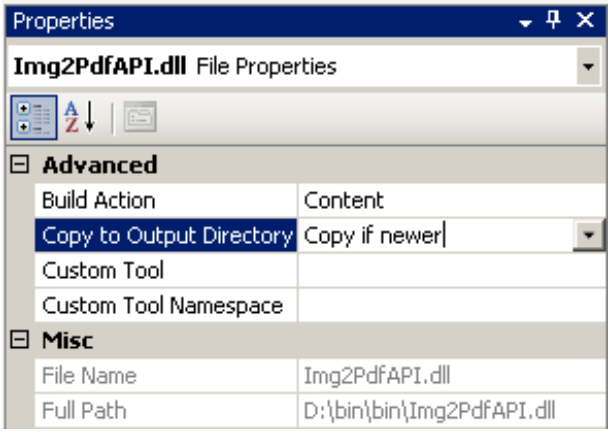
The 3-Heights™ Image to PDF Converter API does not provide a pure .NET interface. Instead, it consists of .NET assemblies and a native DLL. This has to be accounted for when installing and deploying the software.

1. The .NET assemblies (**NET.dll*) are to be added as references to the project (see screenshot below).
   To do so, in the "Solution Explorer" right-click your project and select "Add Reference...". The "Add Reference" dialog will appear. In the tab "Browse", browse for the .NET assemblies *libpdfNET.dll* and *Img2PdfNET.dll*, add them to the project as shown below:



2. *Img2PdfAPI.dll* is not a .NET assembly, but a native DLL. It is not to be added as a reference to the project. (Doing so would use its COM interface and create an Interop DLL). *Img2PdfAPI.dll* is called by the .NET assembly *Img2PdfNET.dll*. *Img2PdfAPI.dll* must be found at execution time. The common way to do this is adding it as an existing item to the project and set its property "Copy to Output Directory" to "Copy if newer".
   Alternatively the directory where *Img2PdfAPI.dll* resides can be added to the environment variable "PATH" or it can simply be copied manually to the output directory.





It is required to use the 32 bit version of the software on a 32 bit platform. On a 64 bit platform either version of the software can be used.
Should you use the 32 bit version of the software on a 64 bit platform, the platform configuration must not be set to "Any CPU", but explicitly to "x86".

The version of the .NET assemblies and the native DLL must be the same, e.g. 4.2.26.0. The version can be found in the version tag of the file's properties.

Create a New Project

There should be at least one .NET sample for MS Visual Studio 2005 available in the ZIP archive of the Windows Version of the 3-Heights™ Image to PDF Converter API. The easiest for a quick start is to refer to this sample.

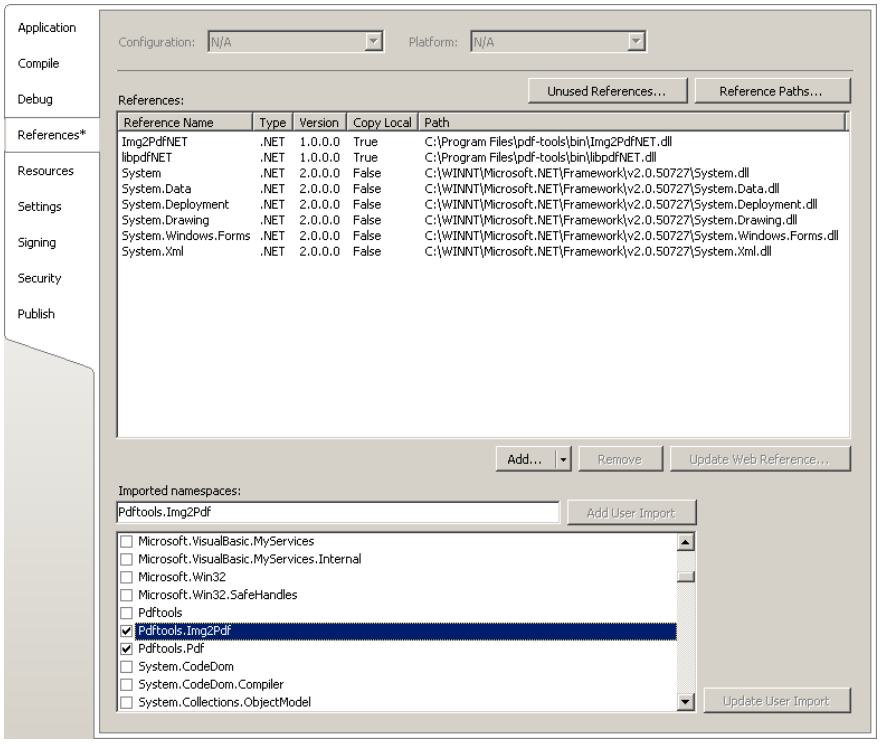
In order to create a new project from scratch, do the following steps:

1. Start Visual Studio and create a new C# or VB project.
2. Add a reference to the .NET assemblies.
3. import namespaces (Note: This step is optional, but useful.)
4. Write Code

Steps 3 and 4 are shown separately for C# and Visual Basic.

Visual Basic

3. Double-click "My Project" to view its properties. On the left hand side, select the menu "References". The .NET assemblies you added before should show up in the upper window.
   In the lower window import the namespaces *Pdftools.Pdf* and *Pdftools.Img2Pdf.*
   You should now have settings similar as in the screenshot below:



4. The .NET interface can now be used as shown below:

```
Dim converter As New Pdftools.Img2Pdf.Img2Pdf()
converter.Create(...)
converter.CreatePagesFromFile(...)
...
converter.Close()
```

C#

3. Add the following namespaces:

```
using Pdftools.Pdf;
using Pdftools.Img2Pdf;
```

4. The .NET interface can now be used as shown below:

```
Img2Pdf converter = new Img2Pdf();
converter.Create(...);
converter.CreatePagesFromFile(...);
...
converter.Close();
```

Deploying in .NET

When deploying a .NET solution, please refer to the following FAQ "Deploying in .NET":
http://www.pdf-tools.com/pdf/Support/FAQ/Article.aspx?name=Deployment-In-NET

Troubleshooting: TypeInitializationException

The most common issue when using the .NET interface is if the native DLL is not found at execution time. This normally manifests when the constructor is called for the first time and exception is thrown - normally of type System.TypeInitializationException.

To resolve that ensure the native DLL is found at execution time. For this, see section .NET Interface in the chapter Installation or the following FAQ:
https://www.pdf-tools.com/pdf/Support/FAQ/Article.aspx?name=Exception-type-initializer

4.4 Java

This chapter briefly describes how to compile and execute the sample application img2pdfconv.java on Windows or Unix.

Compilation

When using the Java interface, the Java-wrapper *jar\I2PA.jar* needs to be on the CLASSPATH. This can be achieved in two ways:

- It is added to the variable CLASSPATH.
- It is provided at compilation time using the switch -classpath.

Windows:

```
javac -classpath .;C:\pdf-tools\jar\I2PA.jar img2pdfconv.java
```

Unix:

```
javac -classpath .:home/path/I2PA.jar img2pdfconv.java
```

Execution

The Java archive I2PA.jar must reside on the classpath.

The path to the native library must be provided. This can be achieved in two ways:

- It is added on the environment variable PATH (Windows).
- It is added at execution time using -Djava.library.path=...

Windows:

```
java -classpath .;C:\pdf-tools\jar\I2PA.jar -Djava.library.path=C:\pdf-tools\bin
    img2pdfconf {parameters}
```

Unix:

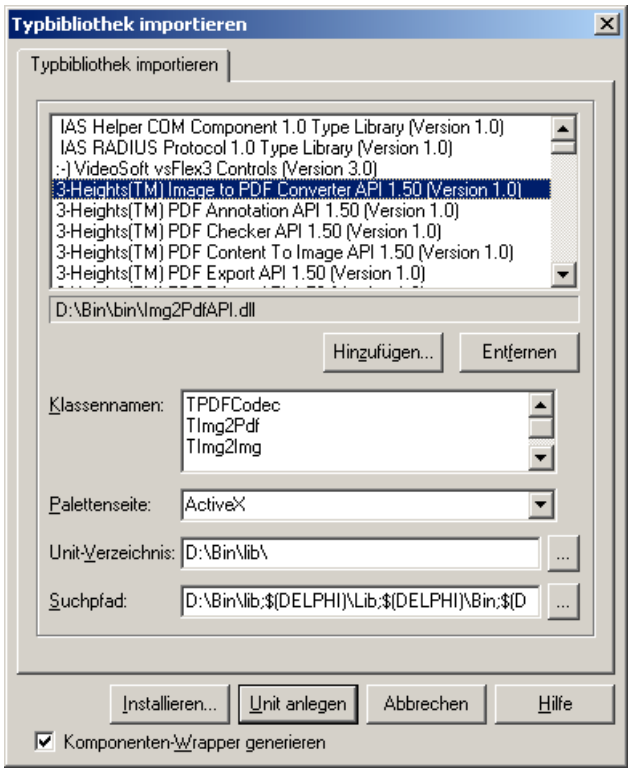
```
java -classpath .:path/to/jar/I2PA.jar  -Djava.library.path=path/to/bin
    img2pdfconf {parameters}
```

(Note: I2PA.jar is a Java wrapper and does not provide any functionality natively, therefore the library is always required for execution.)

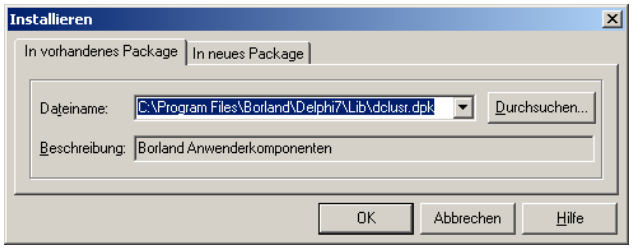
4.5 Delphi

This chapter guides through installing the 3-Heights™ Image to PDF Converter API on Borland Delphi 7. The screenshots are taken on a German Windows.

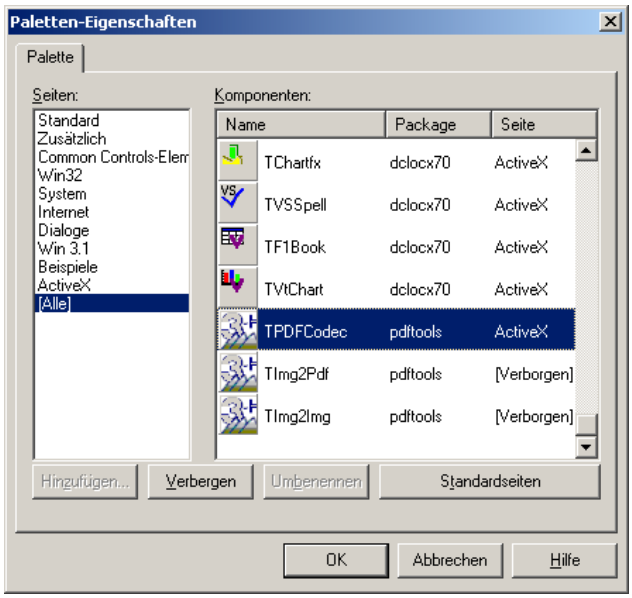
1. Once the COM interface of the DLL is registered as described in the section COM Interface, start your Delphi application.
2. Go to the menu *Project* and select *Import Type Library*.



   If the DLL was successfully registered, it shows up in the list. Select the 3-Heights™ Image to PDF Converter API. If there are any conflicts with class names, adjust the class names accordingly. Select a Unit path, e.g. D:\bin\lib\ that defines in what directory the Type Library should be created, add this path to the search path.
3. Select the 3-Heights™ Image to PDF Converter API component and click Install.
   Add the class to the existing package $(DELPHI)\Lib\dclusr.dpk.



4. You should then receive a confirmation message box saying the component has been successfully registered. Close the package dclusr.dpk and save changes.
5. An icon of the class 3-Heights™ Image to PDF Converter API should show up in the ActiveX tab. If it does not show up, i.e. it is marked hidden, select “Configure Palette” from the menu “Components”. Drag and drop it from [All] to [ActiveX].



6. Now you can open the Delphi sample which is included in *samples/Delphi/*, or create a new sample from scratch.

5 User's Guide

5.1 Overview of the API

What is the 3-Heights™ Image to PDF Converter API about?

The API can be used in any application that requires a process to convert images to PDF documents or split or merge images. Here is a typical use case:

An application takes raster images as input. These can come from any source, such as a scanner or are uploaded from the internet. The application processes these images, e.g. resizes them, applies down-sampling, compresses them, merges them with other images, etc. Finally it creates an output document. The output document can be an image again or a PDF or a PDF/A document. The output is used for any purpose, such as sending it back to the submitter of the original image, archive it or forward it for post-processing.

How does the API work?

The way to use the 3-Heights™ Image to PDF Converter API is output-oriented. An Img2Pdf object is bound to a PDF output-document, which can be a PDF file or a PDF in memory. One or multiple images can be opened and their pages, or a selection of pages are converted to PDF pages and added to the PDF output-document.

This allows for single document conversion as well as merging multiple image documents into one PDF document or split one multi-page image (e.g. a TIFF) to single page PDF documents.

The basic call sequence is:

- Create object
- Set PDF output-document compliance (such as PDF 1.5 or PDF/A 2b)
- Create PDF output-document
- Apply settings (page size, quality, color profiles, etc.)
- Create page(s) from image input-document(s)
- Close PDF

In Visual Basic 6, these calls could look as below:

```
Dim conv As New IMG2PDFAPILib.Img2Pdf
conv.Compliance = IMG2PDFAPILib.ePDFA1b
If Not conv.Create(outputPDF.txt) Then ...
conv.AdjustPage = 1
If Not conv.CreatePagesFromFile( inputImage.txt, 1, -1) Then ...
conv.Close
```

Use in Conjuction with the PDF Prep Tool Suite

The 3-Heights™ Image to PDF Converter API is also bundled to the PDF Prep Tool Suite (PTS) in order to convert raster images to PDF images, which then can be added to PDF documents.

The PTS does not support .NET, therefore any comments in this manual about .NET can be neglected if working in combination with the PTS. The .NET assemblies are not bundled with the PTS.

OCR Recognition of Images

The 3-Heights™ Image to PDF Converter API can also be used to perform OCR on an image and extract the detected text. During this process, no PDF output document is created. This feature can for example be used to read a barcode from an image.

The basic call sequence in Visual Basic 6 is as follows:

- Create a PDFCodec object.
- Open the image file and set the page number.
- Create an *ImgOcr* object and configure it (OCR engine, parameters, language).
- Set the image using the *SetImage()* method of the *ImgOcr* object and call the *Recognize* method to perform OCR recognition.
- Read the OCR text using the GetFirstOcrText() and GetNextOcrText() methods.

6 Reference Manual

The reference manual is based on the COM interface. However there is an equivalent function to each COM function in the C, .NET and Java interface. (See img2pdfapi_c.h and i2pa.jar)

The main DLL contains five classes:

- Img2Img: This class can be used to convert images, or a page range of them, from one type to another.
- Img2Pdf: This class can be used to convert images to PDF documents.
- PDFCodec: This class can be used to retrieve various information from images, such as image compression, color depth, resolution, size, image mask, etc. This class can also be used to interface with other libraries, such as the PDF Prep Tool to import images into a PDF document.
- ImgOcr: This class can be used to perform OCR recognition on an image and extract the detected text.
- OcrText: This class represents a text fragment detected by the ImgOcr class.

6.1 The Img2Pdf Interface

The interface Img2Pdf provides the functionality to create a PDF documents from various image formats.

Image-related properties, such as compression or quality are related to the target output file. For example, if BitonalCompression is set to eComprGroup4, any bi-tonal image that is converted to a PDF document is saved with compression CCITT G4. In order to read the property (e.g. the compression) of an exiting image file, use the interface PDFCodec.

AdjustPage

```
Property Boolean AdjustPage
Accessors: Get, Set
Default: True
```

When set to true, the page dimensions of the PDF will be chosen, so that the image fits exactly on the page. If set to true, the property FitImage is automatically set to false.

Alt

```
Property String Alt
Accessors: Get, Set
Default: "Imported image"
```

In order to create a PDF/A level A compliant document (PDF/A-1a, PDF/A-2a, PDF/A-3a), each image must have an alternate text with a description of the image in support of accessibility to users with disabilities. This property sets this alternate text used for images added subsequently. The property should be set before adding images. It is only relevant in combination with PDF/A level A. See also properties Lang and Compliance.

BitonalCompression

```
Property TPDFCompression BitonalCompression
Accessors: Get, Set
Default: eComprGroup4
```

Get or set the compression type for bi-tonal images. Normally either CCITT G4 or JBIG2 is used for bi-tonal compression. Due to the simpler algorithm CCITT G4 has the advantage of being faster. JBIG2 can achieve compression ratios that are up to twice as high as CCITT G4 at the cost of longer computation time. See also enumeration TPDFCompression.

BorderSize

```
Property Single BorderSize
Accessors: Get, Set
Default: 0
```

This property sets or gets the border between the image and the page border. The units are points (1 point = 1/72 inch). The border does not change the dimension of the page set by the property SetPageSize.

CenterImage

```
Property Boolean CenterImage
Accessors: Get, Set
Default: False
```

Center the image on the page horizontally and vertically.

Close

```
Method Boolean Close()
```

This method closes the PDF file. It is called after a PDF document has been created and the desired pages from images are added. Avoiding the call to this function may still result in a valid output, but it can also cause memory leaks.

Return value:

- True: The PDF file was closed successfully.
- False: Otherwise

Compliance

```
Compliance Property TPDFCompliance Compliance
Accessors: Get, Set
Default: ePDF17
```

This property allows setting a PDF compliance level. It must be set before calling Create. Supported compliance modes are:

ePDF1x	Regular PDF Versions such as 1.4, 1.5, 1.6, 1.7
ePDFA1b	PDF/A-1b format
ePDFA1a	PDF/A 1a format (accessibility)
ePDFA2b	PDF/A 2b format
ePDFA2u	PDF/A 2u format (Unicode)
ePDFA2a	PDF/A 2a format (accessibility)
ePDFA3b	PDF/A 3b format
ePDFA3u	PDF/A 3u format (Unicode)
ePDFA3a	PDF/A 3a format (accessibility)

In order to create PDF/A compatible documents, there are additional requirements besides setting the compliance level:

- Metadata: Selecting a PDF/A compliance level will automatically generate the XML metadata and other requirements to meet the PDF/A specification.
- Tagging: For PDF/A level A (accessibility) it is also requested to have an alternate descriptive text for images. This text can be set using the properties Alt and Lang.
- Color Profiles: For non-calibrated colors, a color profile must be embedded. See methods SetOutputIntent and SetColorSpaceProfile. If no color profile is set, then for RGB and Grayscale colors, calibrated color spaces are generated while for CMYK colors, a default CMYK output intent is set.

If JPEG2000 images are to be converted to PDF/A and the JPEG2000 compression shall be retained, a compliance level of PDF/A-2 or later must be selected.

ContinuousCompression

```
Property TPDFCompression ContinuousCompression
Accessors: Get, Set
Default: eComprJPEG
```

Get or set the compression type of color and grey scaled images in the PDF document. See also enumeration TPDFCompression.

Create

```
Method Boolean Create(String PDFFileName, String UserPwd, String OwnerPwd,
                                          TPDFPermission PermissionFlags)
```

This method creates a PDF file.

Note that in order to meet PDF/A compliance, the document mustn't be encrypted.

Parameters:
- PDFFileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules.
- UserPwd (optional): Set the user password of the PDF document. If this parameter is omitted, the default password is used. Use 0 to set no password.
- OwnerPwd (optional): Set the owner password of the PDF document. If this parameter is omitted, the default password is used. Use 0 to set no password.
- PermissionFlags (optional): Set the permission flags of the PDF document. This option requires an owner password to be set. By default no permissions are granted. To not encrypt the output document, set PermissionFlags to -1, user and owner password to 0. In order to allow high quality printing, both flags ePermPrint and ePermDigitalPrint need to be set. See also enumeration TPDFPermission. To combine multiple flags, use a bitwise or operator (ex. VB: PermissionFlags = ePermPrint OR ePermDigitalPrint).

Return value:
- True: The file was created successfully.
- False: The file could not be created, because e.g. the file already exists and is locked/read-only.

CreateInMemory

```
Method Boolean CreateInMemory()
```

This method creates a PDF in memory. Once the document is completed and after the Close() call, it can be accessed using the method GetPDF().

CreatePageFromCodec

```
Method Boolean CreatePageFromCodec(PDFCodec pCodec)
```

This method creates a page from an image object. It must be called after Create or CreateInMemory.

Parameters:
- pCodec: A PDFCodec object holding an image.

Return value:
- True: The page in the PDF document was created successfully.
- False: otherwise.

CreatePageFromImageFile

```
Method Boolean CreatePageFromImageFile(String FileName)
```

This method adds the page (or pages for multi-page TIFF images) of an image file to the current PDF output. It must be called after Create or CreateInMemory.

Parameters:
- FileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules.

Return value:
- True: The page(s) in the PDF document were created successfully.
- False: otherwise.

CreatePagesFromFile

```
Method Boolean CreatePagesFromFile(String FileName, Long FromPageNo, Long ToPageNo)
```

This method adds the page (or a page range for multi-page TIFF images) of an image file to the current PDF output. It must be called after Create or CreateInMemory.

Parameters:
- FileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules.
- FromPageNo (optional): The starting page number. Default = 1.
- ToPageNo (optional): The last page number. Default = -1 (last page).

Return value:
- True: The page(s) in the PDF document were created successfully
- False: otherwise.

DefaultDPI

```
Property Single DefaultDPI
Accessors: Get, Set
Default: 96
```

Set the default resolution in DPI (dots per inch), if it's not provided by the image. Default is 96 dpi. If the resolution is given by the image then this option doesn't have any effect. Basically it changes the amount of dots per inch by changing the size of the image in the PDF document. The size of the raster image in pixel is not changed.

ErrorCode

```
Property TPDFErrorCode ErrorCode
Accessors: Get
```

This property can be accessed to receive the latest error code. See also enumeration TPDFErrorCode. PDF-Tools error codes are listed in the header file *pdferror.h*. Please note, that only few of them are relevant for the 3-Heights™ Image to PDF Converter API.

ExportText

```
Method Boolean ExportText(String FileName)
```

This function is used in combination with OCR only. It allows to write the text, which is detected by the OCR engine during conversion, not only as invisible text in the PDF, but additionally to a text file. The text file is closed when output PDF document is closed using the function Close.

Parameters:
- FileName: Defines the text file and optionally its path. If the parameter is an empty string, no text file is created.

Return value:
- True: if the function call was successful
- False: otherwise.

FitImage

```
Property Boolean FitImage
Accessors: Get, Set
Default: False
```

Scale the image to fit the size of the page of the PDF. The image is scaled so that either width or height match the dimensions of the page, the other dimension is scaled proportionally. If set to true, the property AdjustPage is automatically set to false.

GetOCREngine

Deprecated, use GetOCRPluginName instead.

GetOCREngineCount

Deprecated, use GetOCRPluginCount instead.

GetOCRPluginCount

```
Method Integer GetOCRPluginCount()
```

OCR engines are accessed through the corresponding OCR interface DLLs. At present there are four OCR engine supported:

Abbyy FineReader 10.0 OCR Engine: This engine is accessed by the OCR interface DLL *pdfocrpluginAbbyy10.ocr*.

Abbyy FineReader 8.1 OCR Engine: This engine is accessed by the OCR interface DLL *pdfocrpluginAbbyy.ocr*.

3-Heights™ OCR Service: This service is accessed by the OCR interface DLL *pdfocrpluginService.ocr*. The service accesses the Abbyy FineReader 8.1 OCR Engine.

Tesseract: This engine is accessed by the OCR interface DLL *pdfocrpluginTesseract.ocr*.

The OCR interface DLL is provided by the 3-Heights™ Image to PDF Converter API.

The OCR engine is provided as a separate product: 3-Heights™ OCR Enterprise Add-On.

In order to make use of the OCR engine, the OCR interface DLL and the OCR engine must be installed. The property *GetOCRPluginCount* returns the number of available OCR interface DLLs. It does not verify the corresponding OCR engines are installed and can be initialized. The OCR engine is loaded with the method *SetOCREngine*.

Return value:
- The number of available OCR engines (i.e. their corresponding OCR interface DLLs).

GetOCRPluginName

```
Method String GetOCRPluginName(Integer iOCREngine)
```

An OCR engine is accessed through an OCR plug-in. Each plug-in corresponds to one OCR engine. The number of OCR plug-ins is retrieved using n = GetOCRPluginCount. The method GetOCRPluginName(n) returns the name of the nth OCR Engine which corresponds to that OCR plug-in. At present there are three OCR engines available: “Abbyy”, “service” and “tesseract”.

Parameters:
- iOCREngine: The nth OCR engine. The total number of engines is retrieved using GetOCRPluginCount.

Return value:
- The name of the nth OCR engine. Null if it doesn’t exist.

GetPDF

```
Method Variant GetPDF()
```

Get the output file from memory. See also method CreateInMemory.

Return value:
- A byte array containing the output PDF. In certain programming languages, such as Visual Basic 6, the type of the byte array must explicitly be Variant.

ImageQuality

```
Property Single ImageQuality
Accessors: Get, Set
Default: 75
```

Get or set the quality index of the lossy compression. This is a value from 1 to 100. This can be applied for JPEG, JPEG2000 and JBIG2 compression. For JBIG2 only the values from 10 to 100 that are multiples of 10 are supported. For both JPEG2000 and JBIG2, a quality index of 100 means lossless compression. JPEG compression is always lossy.

IndexedCompression

```
Property TPDFCompression IndexedCompression
Accessors: Get, Set
Default: eComprFlate
```

Get or set the compression type of indexed images in the PDF document. Supported compressions are Flate and LZW, see also enumeration TPDFCompression.

InfoEntry

```
Property String InfoEntry(String Key)
Accessors: Get, Set
```

Parameters:
- Key: A key as string

Return value:
- Value: The value as string

Retrieve or add a key-value pair to the documents info object. Values of predefined keys are also stored in the XMP metadata package.

Popular entries defined by the PDF Specification and used by most PDF viewers are “Title”, “Author”, “Subject”, “Creator” (sometimes referred to as Application) and “Producer” (sometimes referred to as PDF Creator).

Examples in Visual Basic 6:

- Get document title: `t = InfoEntry("Title")`
- Set document title: `InfoEntry( "Title ") = "my title"`
- Set the creation date to 13:55:33, April 5, 2010, UTC+2:
  `InfoEntry("CreationDate") = "D: 20100405135533 + 02'00'"`

Lang

```
Property String Lang
Accessors: Get, Set
Default: "US-EN"
```

Set the language for the alternate text that is set using the property Alt. The default language is US-EN. Other languages can be set using the corresponding abbreviations, e.g. “DE” (German), “FR” (French), etc.

Linearize

```
Property Boolean Linearize
Accessors: Get, Set
Default: False
```

Get or set whether to linearize the PDF output file for fast web access.

Linearization is the process of preparing a PDF file in a way that permits random page access by a web browser. While the whole non-linearized PDF file must be downloaded before the first page can be displayed, this is not the case for a linearized file.

OCREmbedOCRImage

```
Property Boolean OCREmbedOCRImage
Accessors: Get, Set
Default: True
```

This property specifies whether or not the deskewed and denoised image created by the OCR engine is used in the output file. If this property is set to False, the input image is copied to the output file.

OCRDeskewImage

```
Property Boolean OCRDeskewImage
Accessors: Get, Set
Default: True
```

This property specifies whether the image and text are deskewed according to the recognized skew angle.

`True` Rotage image, such that lines of text are made horizontal. This might change the appearance of the page. This setting is recommended for scanned documents.

`False` Do not change skew of images. This setting is recommended for born-digital documents.

OCREmbedBarcodes

```
Property Boolean OCREmbedBarcodes
Accessors: Get, Set
Default: False
```

This property specifies whether the recognized barcodes are embedded in the XMP metadata.

OCRResolutionDPI

```
Property Single OCRResolutionDPI
Accessors: Get, Set
Default: 300
```

Resample images to target resolution before they are sent to the OCR engine. The default is 300 dpi, which is the preferred resolution for most OCR engines.

OCRThresholdDPI

```
Property Single OCRThresholdDPI
Accessors: Get, Set
Default: 400
```

Only images with a higher resolution than the threshold are re-sampled before OCR. The default is 400 dpi. If set to -1: no re-sampling is applied.

Orientation

```
Property TPDFOrientation Orientation
Accessors: Get
```

Return the orientation rounded to the next 90 degrees. The orientation is an enumeration with eight different values (rotation times flipping). See enumeration TPDFOrientation.

Quality

Deprecated, use ImageQuality instead.

Recompress

```
Property Boolean Recompress
Accessors: Get, Set
Default: False
```

If set to true, JPEG, JPEG2000 and CCITT Fax Group4 streams are re-compressed.

Advantages:

- Invalid streams are repaired (as far as possible)
- Standard JPEG streams are created (which should be readable by any application)

Disadvantages:

- Recompressing a lossy stream usually increases the file size and lowers the Quality

ResolutionDPI

```
Property Single ResolutionDPI
Accessors: Get, Set
Default: 150
```

Get or set the resolution in dpi after re-sampling images. This property affects all three image compression types (bi-tonal, monochrome, color). The typical value for the resolution when optimizing for the web is 150 dpi. For printing typically no re-sampling is applied (see property ThresholdDPI). Pre-blended images, images with a color key mask, mask, and soft mask images are not re-sampled.

SetColorSpaceProfile

```
Method Boolean SetColorSpaceProfile(String Profile)
```

Set a color space profile for embedding in the output PDF. See also SetOutputIntent for color profiles. The color profile provided here is used directly for the image's color space.

Parameters:

- Profile: The file name of the color profile

Return value:

- True: The color profile was set successfully.
- False: The file name points to an invalid color profile. (Only PDF/A compliant profiles are accepted.)

At maximum three profiles (one RGB profile, one CMYK profile, and one Gray profile) can be set by using at most one call to SetOutputIntent and/or at most three calls to SetColorSpaceProfile.

SetMetadata

```
Method Boolean SetMetadata(Single FileName)
```

Set the document's XMP metadata. The XMP metadata is inserted as is, which means it is not parsed and validated. If no XMP metadata is provided, the 3-Heights™ Image to PDF Converter API generates it automatically.

Parameters:

- FileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules of the file containing the XMP metadata.

Return value:

- True: The XMP metadata file was set successfully.
- False: otherwise.

SetOCREngine

```
Method Boolean SetOCREngine(String Engine)
```

This method require the 3-Heights™ OCR Add-On, which is a separate product, to be installed. See also documentation for the 3-Heights™ OCR Add-On.

Set the OCR engine that is used when OCR information shall be added during the conversion. If the engine's name is set to an empty string, OCR is not applied.

See also documentation for the 3-Heights™ OCR Add-On.

Parameters:

- Engine: The name of the OCR engine (e.g. "abbyy"). For every available OCR engine, there is a corresponding OCR interface DLL. The OCR interface DLLs (e.g. *pdfocrAbbyy.ocr*) are distributed with the 3-Heights™ Image to PDF Converter API and are required to communicate with the OCR engine. The names of all available OCR engines can be retrieved using the properties *GetOCREngineCount* and *GetOCREngine*.

Return value:

- True: The OCR interface DLL was found, the OCR engine was found and the OCR engine was successfully initialized.
- False: otherwise.

SetOCRLanguages

```
Method Boolean SetOCRLanguages(String Languages)
```

This method requires the 3-Heights™ OCR Add-On, which is a separate product, to be installed. See also documentation for the 3-Heights™ OCR Add-On.

Setting a languages helps the OCR engine to minimize errors by means of using dictionaries of the defined languages.

This method must be called after SetOCREngine.

If SetOCRParams is used, *SetOCRLanguages* must be called after SetOCRParams.

Parameters:

- Languages: A string of one or multiple, comma-separated languages. The supported names are OCR engine dependant. The OCR engine will only use dictionaries of the set languages.

Return value:

- True: The Language(s) were successfully set
- False: Otherwise

Example:

```
SetOCREngine("abbyy")
```

```
SetOCRLanguages("English, German")
```

SetOCRParams

```
Method Boolean SetOCRParams(String Params)
```

This method requires the 3-Heights™ OCR Add-On, which is a separate product, to be installed. See also documentation for the 3-Heights™ OCR Add-On.

By means of this method, OCR engine specific settings can be applied in the form of key-value pairs. These pairs are OCR engine dependant and are described in the corresponding manual.

Parameters:
- Params: A list of comma-separated key value pairs. See example.

Return value:
- True: The OCR parameters were successfully set.
- False: otherwise.

Example:

```
SetOCREngine("abbyy")

SetOCRParams("BalancedMode=TRUE, DetectBold=FALSE")
```

SetOutputIntent

```
Method Boolean SetOutputIntent(String Profile)
```

The output intent represents the output color profile. Color profiles are usually provided with the OS. On Windows for example, they can be found at *C:\WINNT\system32\spool\drivers\color*.

Alternatively profiles can be found here:
- www.pdf-tools.com/public/downloads/resources/colorprofiles.zip
- www.color.org/srgbprofiles.html
- www.adobe.com/support/downloads/iccprofiles/icc_eula_win_dist.html

Please note that most color profiles are copyrighted, therefore you should read the license agreements on the above links before using the color profiles.

Parameters:
- Profile: The name of the color profile. An example could be: "*C:\WINNT\system32\spool\drivers\color\sRGB Color Space Profile.icm*"

If compliance level PDF/A is selected and no output intent is defined, a default output intent (sRGB Color Space Profile.icm) is embedded.

This method must be called after Create has been called.

SetPageSize

```
Method Boolean SetPageSize(Single Width, Single Height)
```

Set the page size of the current and following pages in the PDF document in points. (1 point = 1/72 inch).

Parameters:
- Width: The width of the page in points.
- Height: The height of the page in points.

Return value:
- True: The page size was be set successfully.
- False: otherwise.

The default values, if the property AdjustPage is set false, is Width=595 and Height=842 (A4).

ThresholdDPI

```
Property Single ThresholdDPI
Accessors: Get, Set
Default: 225
```

Get the threshold in dpi to selectively activate re-sampling. Only images with a resolution above the threshold dpi will be re-sampled. The typical threshold value when optimizing for the web is 225 dpi (default). This property affects all three image compression types (bi-tonal, monochrome, color). Set to -1 to deactivate re-sampling.

6.2 The PDFCodec Interface

The codec interface provides information about the image. Such as bits per component, components per pixel, color space, the image data itself, etc. This data can be used by other applications such as the PDF Prep Tool Suite.

Keep in mind that most properties are not read before a page number is defined using the method PageNo. This is also true for images with just one page.

BitsPerComponent

```
Property Integer BitsPerComponent
Accessors: Get
```

Return the number of bits that are used to represent a single color component of an image sample. The number of color components per image data sample can be retrieved through the image's color space interface.

Close

```
Method Boolean Close()
```

Close an opened input file. If the document is already closed the method does nothing.

Return value:
- True: The file was closed successfully.
- False: Otherwise

ColorSpace

```
Property IPDFColorSpace* ColorSpace
Accessors: Get
```

Return an interface to the color space of the image (see ColorSpace Interface).

ComponentsPerPixel

```
Property Integer ComponentsPerPixel
Accessors: Get
```

Return the number of components per pixel.

Compression

```
Property TPDFCompression Compression
Accessors: Get
```

This property returns the compression type. See also enumeration TPDFCompression. This property is initially set to 0.

Create

```
Method Boolean Create(String FileName)
```

Create an empty image file.

Parameters:
- FileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules of the image file. Supported extensions are listed in the chapter Supported Image Extensions.

Return value:
- True: The file was created successfully.
- False: The file was not created, e.g. the file already exists and is read-only.

CreateInMemory

```
Method Boolean CreateInMemory(String Extension)
```

Create an image in memory.

Parameters:
- Extension: The type of the image to be created. Supported extensions are listed in the chapter Supported Image Extensions.

Return value:
- True: The image was created successfully in memory.
- False: otherwise.

fXDPI, fYDPI

Deprecated, use XDPI and YDPI instead.

GetImage

```
Method Variant GetImage()
```

This method returns an image which was previously created in memory using the methods CreateInMemory and Close.

Height

```
Property Long Height
Accessors: Get
```

Return the height of the image in pixels (also called samples). The unit of pixels can be converted to a distance unit such as inch, millimeter etc. using a resolution value, i.e. 72 dpi (dots per inch).

ImageQuality

```
Property Single ImageQuality
Accessors: Get, Set
Default: 75
```

Get or set the quality index of the lossy compression. This is a value from 1 to 100. This can be applied for JPEG, JPEG2000 and JBIG2 compression. For JBIG2 only the values from 10 to 100 that are multiples of 10 are supported. For both JPEG2000 and JBIG2, a quality index of 100 means lossless compression. JPEG compression is always lossy.

IsPremultipliedAlpha

```
Property Boolean IsPremultipliedAlpha
Accessors: Get
```

This property returns true if the image pixels are stored as the original pixel times the alpha value. (i.e. pixel = backdrop * (alpha - 1) + image * alpha)

Mask

```
Property Variant Mask
Accessors: Get
```

Return the image's explicit mask if available. The mask's sample data is organized the same way as the image data except that the data contains one bit per pixel. A one bit indicates an opaque pixel and a zero bit indicates a transparent pixel.

Open

```
Method Boolean Open(String FileName)
```

This method opens an image file.

Parameters:
- FileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules of the image file.

Return value:
- True: The file was opened successfully.
- False: Otherwise

OpenMem

```
Method Boolean OpenMem(Variant varMem)
```

This method opens an image from memory.

Parameters:
- varMem: A byte array containing the image.

Return value:
- True: The file was opened successfully.
- False: Otherwise

Page

Deprecated, use PageNo instead.

PageCount

```
Property Long PageCount
Accessors: Get
```

Return the number of pages of an open document. If the document is closed then 0 is returned.

PageNo

```
Property Long PageNo
Accessors: Get, Set
Default: -1
```

Set or get the current page number in the image.The page number must always be set, also for single page images.

Palette

```
Property Variant Palette
Accessors: Get
```

This property returns the palette of the image (if existing).

Quality

Deprecated, use ImageQuality instead.

Recompress

```
Property Boolean Recompress
Accessors: Get, Set
Default: True
```

If set to false, JPEG, JPEG2000 and CCITT Fax Group4 streams are not de-compressed. As a result, the Samples property will return the compressed stream as indicated by the Compression property. If possible, the Recompress property should be set before calling the Open() method, because for some image formats changing the Recompress property might result in reloading some image data.

Samples

```
Property Variant Samples
Accessors: Get
```

Return the image's data samples in a byte array. The sample data is ordered by line from top to bottom and within a line from left to right. The lines are byte aligned. If the number of bits per component is less than one byte then the samples are ordered beginning with the most significant bit first.

SMask

```
Property Variant SMask
```

With this property the soft mask of an image can be extracted.

Width

```
Property Long Width
Accessors: Get
```

Return the width of the image in pixels (also called samples). The unit of pixels can be converted to a distance unit such as inch, millimeter etc. using a resolution value, i.e. 72 dpi (dots per inch).

XDPI, YDPI

```
Property Single XDPI
Property Single YDPI
Accessors: Get
```

These properties return the resolution in dots per inch in X and Y direction.

6.3 The Img2Img Interface

The image to image interface is a separate interface that provides functionality to convert images from one format to another. It allows changing the compression type and up-sampling.

BitonalCompression

```
Property TPDFCompression BitonalCompression
Accessors: Get, Set
Default: eComprGroup4
```

Get or set the compression type for bi-tonal images. Normally either CCITT G4 or JBIG2 is used for bi-tonal compression. Due to the simpler algorithm CCITT G4 has the advantage of being faster. JBIG2 can achieve compression ratios that are up to twice as high as CCITT G4 at the cost of longer computation time. See also enumeration TPDFCompression.

ContinuousCompression

```
Property TPDFCompression ContinuousCompressionCompression
Accessors: Set
```

Set the compression type for color and grey-scale images in the output image. See also enumeration TPDFCompression.

ContinousCompression

Deprecated, use ContinuousCompression instead.

ConvertFile

```
Method Boolean ConvertFile(String InputFileName, String OutputFileName,
Long FromPageNo, Long ToPageNo)
```

Convert an image from one type to another and save it to a file. The image type is defined by the extension of the parameter OutputFileName. See also chapter "Supported Image Extensions".

Parameters:

- InputFileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules.
- OutputFileName: The file name and optionally the file path, drive or server string according to the operating systems file name specification rules.
- FromPageNo (optional): The first page of the page range to be copied from a multi-page input file. Default = 1.
- ToPageNo (optional): The last page of the page range to be copied from a multi-page input file. Default = -1 = last page.

Return value:

- True: The file of the image was created successfully.
- False: otherwise.

CopyPage

```
Method Boolean CopyPage(PDFCodec InputCodec, PDFCodec OutputCodec)
```

This method copies the current page (PDFCodec.Page) from a PDFCodec object to another PDFCodec object. Target codec has to be an opened file using Open or Create.

Parameters:

- InputCodec: A PDFCodec object containing a valid image at the currently set page number of the input codec.
- OutputCodec: A PDFCodec object, to which the page is appended. The currently set page number in the output codec is not relevant.

Return value:

- True: The page was copied successfully.
- False: otherwise.

DPI

```
Property Long DPI
Accessors: Set, Get
Default: 0 (not applied)
```

Set the resolution in dpi (dots per inch). This property supports up-sampling, but not down-sampling.

ImageQuality

```
Property Single ImageQuality
Accessors: Get, Set
Default: 75
```

Get or set the quality index of the lossy compression. This is a value from 1 to 100. This can be applied for JPEG, JPEG2000 and JBIG2 compression. For JBIG2 only the values from 10 to 100 that are multiples of 10 are supported. For both JPEG2000 and JBIG2, a quality index of 100 means lossless compression. JPEG compression is always lossy.

IndexedCompression

```
Property TPDFCompression IndexedCompression
Accessors: Set
```

Set the compression type for indexed images in the output image. See also enumeration TPDFCompression.

Quality

Deprecated, use ImageQuality instead.

6.4 The ImgOcr Interface

The image OCR interface allows you to extract OCR text from an image opened using the PDFCodec interface. During that process, no output file is created. The ImgOcr interface is not needed to create a searchable PDF, use the Img2Pdf interface for that task.

GetFirstOcrText

```
Method OcrText GetFirstOcrText()
```

Get the first text fragment recognized, or NULL if none available.

GetNextOcrText

```
Method OcrText GetNextOcrText()
```

Get the next text fragment recognized, or NULL if none available.

GetOCREngineName

```
Method String GetOCREngineName()
```

Get the name of the currently set OCR engine.

GetOCRPluginCount

```
Method Integer GetOCRPluginCount()
```

Get the number of available OCR plugins (see GetOCRPluginCount of the Img2Pdf interface).

GetOCRPluginName

```
Method String GetOCRPluginName(Integer iOCREngine)
```

Get the name of the i-th OCR plugin engine.

Recognize

```
Method Boolean Recognize()
```

Perform OCR recognition. The return value indicates whether or not the recognition has been successful.

SetImage

```
Method Boolean SetImage(PDFCodec Image)
```

Set the image to OCR. Before calling this method the image must be opened and the correct page set. The return value indicates whether or not the image could be set.

SetOCRLanguages

```
Method Boolean SetOCRLanguages(String Languages)
```

Set the OCR languages (see SetOCRLanguages of the Img2Pdf interface).

SetOCRParams

```
Method Boolean SetOCRParams(String Parameters)
```

Set the OCR parameters (see SetOCRParams of the Img2Pdf interface).

6.5 The OcrText Interface

The OCR text interface represents a text fragment detected by the image OCR interface.

BaseLine

```
Property Single BaseLine
Accessors: Get
```

Get the Y coordinate of the text's base line.

FontName

```
Property String FontName
Accessors: Get
```

Get the name of the font. For barcodes the font name is "Barcode".

FontSize

```
Property Single FontSize
Accessors: Get
```

Get the size of the font in points.

Rect

```
Property Variant Rect
Accessors: Get
```

Get the bounding box rectangle of the text.

StringLength

```
Property Integer StringLength
Accessors: Get
```

Get the number of characters of the recognized string.

Text

```
Property String Text
Accessors: Get
```

Get the recognized text.

6.6 Enumerations

Note: Depending on the interface, enumerations may have “TPDF” as prefix (COM, C) or “PDF” as prefix (.NET) or no prefix at all (Java).

TPDFColorSpace

eColorGray	Gray
eColorGrayA	Gray with alpha channel
eColorRGB	Red Green Blue
eColorRGBA	RGB with alpha channel
eColorCMYK	Cyan Magenta Yellow Key
eColorYCbCr	YCbCr
eColorYCbCrK	YCbCrK
eColorPalette	Color space using a palette
eColorLAB	CIE L*a*b*
eColorOther	Other

TPDFCompliance

ePDF10	PDF Version 1.0
ePDF11	PDF Version 1.1
ePDF12	PDF Version 1.2
ePDF13	PDF Version 1.3
ePDF14	PDF Version 1.4 (corresponds to Acrobat 5)
ePDF15	PDF Version 1.5
ePDF16	PDF Version 1.6 (corresponds to Acrobat 7)
ePDF17	PDF Version 1.7
ePDFA1a	PDF/A 1a, ISO 19005-1, Level A compliance
ePDFA1b	PDF/A 1b, ISO 19005-1, Level B compliance

ePDFA2a	PDF/A 2a, ISO 19005-2, Level A compliance
ePDFA2b	PDF/A 2b, ISO 19005-2, Level B compliance
ePDFA2u	PDF/A 2u, ISO 19005-2, Level U compliance
ePDFA3a	PDF/A 3a, ISO 19005-3, Level A compliance
ePDFA3b	PDF/A 3b, ISO 19005-3, Level B compliance
ePDFA3u	PDF/A 3u, ISO 19005-3, Level U compliance
ePDFUnk	Unknown format (default)

TPDFCompression

eComprRaw	No compression
eComprJPEG	Joint Photographic Expert Group
eComprFlate	Flate compression
eComprLZW	Lempel-Ziv-Welch
eComprGroup3	CCITT Fax Group 3
eComprGroup3_2D	CCITT Fax Group 3 2D
eComprGroup4	CCITT Fax Group 4
eComprJBIG2	Joint Bi-level Image Experts Group
eComprJPEG2000	JPEG2000
eComprUnknown	Unknown compression

Note that not all image formats/color depths support all compression types.

TPDFErrorCode

All TPDFErrorCode enumerations start with "PDF_" followed by a single letter which is one of "S", "E", "W" or "I", an underscore and a descriptive text. The single letter gives in an indication of the type of error. These are: Success, Error, Warning, Information. With respect to corrupt PDF files: An error indicates a corruption in the PDF, the file may or may not be readable. A warning indicates the file is readable but not valid.

A full list of all PDF Tools error codes is available in the header file *pdferror.h*. Note that only a few are relevant for the Image to PDF Converter API. The most common are listed here.

PDF_S_SUCCESS	The operation was completed successfully.
LIC_E_NOTSET, LIC_E_NOTFOUND, …	Various license management related errors.
PDF_E_FILEOPEN	Failed to open the file.
PDF_E_FILECREATE	Failed to create the file.

The following warnings can occur when creating PDF/A:

PDF_I2P_W_OUTPUTINTENT	An output intent was required. An sRGB profile was created.
PDF_I2P_W_SMASK	The soft mask of the image was removed during the conversion (PDF/A-1 only).
PDF_I2P_W_JPXDECODE	JPEG2000 compression was replaced by JPEG compression (PDF/A-1 only).

TPDFOrientation

eOrientationUndef	Undefined
eOrientationTopLeft	Image is untransformed.
eOrientationTopRight	Before viewing, image is flipped horizontally.
eOrientationBottomRight	Before viewing, image is rotated by 180°.
eOrientationBottomLeft	Before viewing, image is flipped vertically.
eOrientationLeftTop	Before viewing, image is rotated by 90° clockwise and then flipped horizontally.
eOrientationRightTop	Before viewing, image is rotated by 90° clockwise.
eOrientationRightBottom	Before viewing, image is rotated by 90° clockwise and flipped vertically
eOrientationLeftBottom	Before viewing, image is rotated by 90° counter-clockwise.

TPDFPermission

An enumeration for permission flags. If a flag is set, the permission is granted.

ePermAll	Grant all Permissions
ePermPrint	Low resolution printing
ePermModify	Changing the document
ePermCopy	Content copying or extraction
ePermAnnotate	Annotations
ePermFillForms	Filling of form fields
ePermSupport Disabilities	Support for disabilities
ePermAssemble	Document Assembly
ePermDigitalPrint	High resolution printing

Changing permissions or granting multiple permissions is done using a bitwise or operator. Changing the current permissions in Visual Basic should be done like this:

Allow Printing:	`Permission = Permission Or ePermPrint`
Prohibit Printing:	`Permission = Permission And Not ePermPrint`

6.7 Supported Image Extensions

The following extensions are supported:

`.tif, .tiff`	Tagged Image File Format
`.jpg, .jpe, .jpeg`	Joint Photographic Expert Group
`.png`	Portable Network Graphics
`.gif`	Graphics Interchange Format
`.bmp`	Window Bitmap
`.jb2`	Joint Bi-level Image Experts Group
`.jp2`	JPEG2000
`.jpx`	Extended JPEG2000
`.pbm,.pgm,.pnm,.ppm`	Portable Bitmap File Format
`.eps`	Encapsulated PostScript (Output only)

6.8 Supported Image Compression Types

For additional information about compressions in PDF, see also ISO 32000, chapter 7.4.

No Compression (Raw) (0)

Raw means no compression is applied.

DCT (JPEG)

Developer	Joint Photographic Experts Group committee
Version	PDF 1.2, PDF/A-1
Color depth	8, 24 bits per pixel
Compression type	Lossy
Compression algorithm	The image is broken up into blocks that are 8 by 8 samples. On each of these blocks and color channel a discrete cosine transformation (DCT) is applied and its coefficients are quantized. The visual quality of the resulting image depends on the loss of information defined by the step size of the quantization and on the image that is being compressed. The compression can be controlled via an image quality parameter - a value from 1 to 100 (default 75). Typical compression ratios are 15:1 (no perceptible loss of information) to 30:1.
Application area	Sampled continuous-tone pictures (photographs)

Flate (ZIP)

Developer	Flate compression is based on the public-domain zlib / deflate compression method
Version	PDF 1.2, PDF/A-1
Color depth	1-8, 24 bits per pixel
Compression type	Lossless
Compression algorithm	A lossless data compression algorithm that uses a combination of the LZ77 algorithm and Huffman coding.
Application area	Images

LZW

Developer	Abraham Lempel, Jacob Ziv and Terry Welch Copyright based issues, which expired in most countries in 2003/2004, reduced the popularity of this compression. As one of its consequences it is not included in PDF/A standard.
Version	PDF 1.2
Color depth	2-8 bits per pixel
Compression type	Lossless
Compression algorithm	An indexed based compression that is also used in the GIF and TIFF image formats.
Application area	Gray-scale images, artificial images

CCITT Fax Group 3 and 4

Developer	International Telecommunications Union (ITU), formerly known as the Comité Consultatif International Téléphonique et Télégraphique
Version	PDF 1.0, PDF/A-1
Color depth	1 bit per pixel
Compression type	Lossless
Compression algorithm	Group 3: 1-dimensional version of the CCITT Group 3 Huffman encoding algorithm. (4) Group 3 2D: 2-dimensional version of the CCITT Group 3 Huffman encoding algorithm. (5) Group 4: An advanced version of a bi-tonal algorithm based on the CCITT Fax Group 3 2D compression. (6)
Application area	Line-art image, bi-tonal, faxes

JBIG2

Developer	Joint Bi-Level Image Experts Group
Version	PDF 1.4, PDF/A-1
Color depth	1 bit per pixel
Compression type	Lossless if the image quality index is set to 100 Lossy otherwise
Compression algorithm	The image is broken down into individual symbols, which are stored in a table. A symbol is added to the table if it doesn't exist yet. If a matching symbol already exists, it is used as a reference. This algorithm works especially well for images with a lot of similar symbols such as scanned text or images that use patterns. Generally JBIG2 provides a better compression ratio than CCITT G3 or G4 compression. Typical compression ratios for text pages are 20:1 to 50:1.
Application area	Line-art image, bi-tonal

JPEG2000

Developer	Joint Photographic Experts Group committee
Version	PDF 1.5, PDF/A-2
Color depth	8, 24 bits per pixel
Compression type	Lossless if the image quality index is set to 100 Lossy otherwise
Compression algorithm	JPEG 2000 is a wavelet-based image compression standard. It was developed with the intention of superseding the original discrete cosine transform-based JPEG standard.
Application area	Sampled continuous-tone pictures (photographs)

7 Samples

The latest versions of samples for various programming languages are available at www.pdf-tools.com/asp/products.asp?name=I2PA.

8 Licensing and Copyright

9 Contact

PDF Tools AG
Kasernenstrasse 1
8184 Bachenbülach
Switzerland
www.pdf-tools.com