



PDF-XChange Drivers API SDK 2012

User Manual

Head Quarters:

Tracker Software Products (Canada) Ltd.,

P.O. Box 79, 9622 Chemainus Road
Chemainus, V0R 1K0
British Columbia, Canada

Sales

Tel: Canada (+00) 1-250-324-1621
Fax: Canada (+00) 1-250-324-1623

In Europe:

Unit 17, Raleigh Court, Priestly Way,
Crawley, Sussex, RH10 9PD,
England.

Sales

Tel: +44 (0) 20 8555 1122
Fax: (+00) 1-250-324-1623

<http://www.tracker-software.com>

sales@tracker-software.com

Created: Thursday, April 17, 2014

Copyright © 2001-2014 Tracker Software Products. All Rights Reserved.

PDF-XChange Drivers API SDK 2012

copyright 2001-2014© Tracker Software Products. All rights reserved.
<http://www.tracker-software.com>

The information contained in this document is subject to change without notice.
This document contains proprietary information which is protected by copyright.
All rights are reserved. No part of this document may be photocopied, reproduced,
or translated to another language without the prior written consent of Tracker Software Products.

Table of Contents

Chapter 1: Welcome	1
Chapter 2: Using Drivers API SDK	2
Chapter 3: System Requirements	4
Chapter 4: Installation	5
Chapter 5: IPXControlEx Object	9
Chapter 6: Interface IPXCPrinter	11
Properties	11
Name	13
Option	14
Property Table Definition	15
Paper	16
General	21
Compression	23
Graphics	25
Fonts	29
Security	30
DigSig (Digital Signatures)	31
Overlay	35
Links	37
Watermarks	38
Bookmarks	38
HeadersFooters	40
DocInfo (Document Information)	42
Save	42
EMail	46
Optimization	48
Saver	50
Naming Macros	50
Methods	53
SetRegInfo	54
ResetDefaults	54
ApplyOptions	55
AddBookmarkItem	55
AddTextWatermark	58
AddImageWatermark	61
StorePrinterOptions	64
ReStorePrinterOptions	64
SetAsDefaultPrinter	65

RestoreDefaultPrinter	66
AddSMTPServer	66
Events	68
Asynchronous	69
Synchronous	73
Chapter 7: Property Declarations	77
Chapter 8: Index	78

Welcome



PDF-XChange Drivers API 2012 SDK

Help Manual

Welcome to the Tracker Software Products online help system. Browse through the help pages by clicking on the icons below or selecting pages in the table of contents to the left. To quickly find specific product information, enter search criteria in the search box above and click the search button.

Chapter Guide

[Using Drivers API SDK](#)
[System Requirements](#)
[Installation](#)
[IPXControlEx Object](#)
[Interface IPXCPrinter](#)
[Property Declarations](#)

Most popular pages

[Welcome](#)
[IPXControlEx Object](#)
[Using Drivers API SDK](#)
[Interface IPXCPrinter](#)
[Properties](#)
[Installation](#)
[Paper](#)
[AddBookmarkItem](#)
[Name](#)
[Asynchronous](#)

Ask Us



If you're unable to find what you're looking for in this help system, try these alternative resources:

- [Our Website](#)
- [Knowledgebase](#)
- [FAQ](#)

or contact our support team:

support@tracker-software.com

Using Drivers API SDK

Who should use the PDF-XChange Drivers API?

Developers requiring the ability to create PDF documents from their own applications (*Clarion Developers - See below) or indeed managing the output of other file formats to PDF from 3rd party applications such as MS Office, AutoCAD, Internet Explorer etc, providing a set of 32/64 bit Virtual print drivers - the PDF-XChange Drivers API is an excellent choice. Creating arguably the smallest and most reliable facsimile in PDF format of the original document available today - at a very competitive price.

Other developer kits available for PDF Creation, Manipulation and Viewing from Tracker Software Products include, **PDF-Tools SDK**, **PDF-XChange PRO SDK** and the **PDF-XChange Viewer SDK**. For more information on all of our developer products please visit our website:

<http://www.tracker-software.com/developers>

Windows Support

From Version 4 we have dropped Drivers for all Windows versions prior to Windows 2000 - **should you still need to support Windows 9x/ME/NT**, we have continued to offer our highly successful PDF-XChange Version 3 product range, though it should be noted that NO further development will take place with our Version 3 product line.

Version 5 supports all Windows versions Windows 2000 and later.

The PDF-XChange Drivers API ActiveX

Allows developers to control the properties of the PDF-XChange driver for the creation of PDF documents. Developers are able to harness the full power of our award-winning Driver by specifying parameters for PDF creation as used by many millions of contented End Users - directly from within the developer's own Software applications.

The PDF-XChange Drivers API is supplied as an ActiveX component. Using its methods and properties all operations can be performed.

The sequence for creating a PDF file from a developer's application is:

1. Create a new unique (and temporary) virtual printer using the [IPXCControlEx interface](#).
2. Using the [IPXCPrinter](#) interface returned by [IPXCControlEx](#), set up all properties as required for PDF creation.
3. Print to the virtual printer created by step 1, either directly from developer's application or from any 3rd party application run from this application.
4. Close the created printer and destroy the temporary printer instance created.

***Clarion for Windows Developers** generally do not need to use Virtual Print Drivers to output their application reports using our products. Clarion uses a method to output reports to a Windows Metafile format for viewing that our PDF-Tools SDK is able to convert directly to PDF, whilst still retaining the reports formatting and text data - ensuring it is still a fully text searchable PDF file once created - we therefore recommend that unless Clarion developer's are converting 3rd party formats such as MS Office, HTML, AutoCAD etc to PDF - they use our PDF-Tools SDK and not the PDF-XChange Drivers API SDK.

Troubleshooting

In most environments the provided COM facilities are available transparently after the installation and registration are complete. However, for some 32-bit based IDE's (for example, Microsoft Visual Basic 6) running in 64-bit Windows, this may be problematic. In this event, you should manually reference the type library (pxccom.tlb) located in the PDF-XChange installation directory.

System Requirements

PDF-XChange Drivers API 2012 SDK supports all Windows (32/64 bit) operating systems from Windows XP** and later.



Version 5 (2012): Microsoft/Citrix Terminal Server compatible*.
Version 4: Microsoft/Citrix Terminal Server compatible*.
Version 3: Still available for Windows 95/98

* **Note:** Though many users have virtualized some of our component products such as the PDF-XChange Viewer and PDF-Tools application using XenApp, we do not support this at this time. Particularly the printer drivers are not designed to work in a virtualized environment.

** [Limitations to product support for Windows XP](#)

To experience the best performance of our products on all Windows operating systems, the onus is on users to ensure that they have all the latest available Microsoft Windows Service Packs & Updates installed.

Installation

We provide a comprehensive installation executable for developer's to distribute with their application and this is the ONLY method authorized for distribution, re-engineering or disassembly of the installation package provided is specifically and absolutely prohibited. Our installer contains all the required driver options required for all versions of Windows from Windows 2000 through to Windows 7 - and all points in between, both 32 and 64 bit!

Developers may either pack the provided installer within their own installation package - or provide separately and simply call from their own installer script using the extensive command line switches provided below.

The latest PDF-XChange Driver for redistribution purposes is always [available from our web site](#).

Or from this link : http://www.tracker-software.com/PDFX5SA_sm.zip

Please note - it is a required that you use the above installer and do not attempt to reverse engineer or de-compile and create your own
- if you are not willing to adhere to this specific condition - you should not use the PDF-XChange Driver within your application.

PDF-XChange Drivers - Silent Installation

This functionality is provided for users with multiple licenses and Developer SDK's - not for single (end) user licenses to be copied to multiple machines - which is illegal. A license is required for each machine to which PDF-XChange is installed - except for the Developer's SDK and evaluation use in demonstration mode.

Assuming you have a multiple user license - installing PDF-XChange to many workstations
- it may be useful to be able to silently install without the need for any user interaction. This allows you to execute the install either from the command line with certain parameters and switches applied - or create a batch file (.bat file) and distribute this with the main installation executable for your users to run directly alleviating the need for the System Administrator to visit each user and install.

Please note as with any system device - such as a printer, full Administrator privileges are required to install on later 'Win32' versions of Microsoft Windows.

Here is an example of the full command line string available:

Most items are optional unless otherwise stated

```
C:\PDFX5SA_sm.exe /VERYSILENT /NORESTART /COMPONENTS="pdfSaver,PDF-XChange driver, Help,Languagess" /DIR="C:\Program Files\PDF-XChange 5\" "/UserName:I am" "/Organization:Tracker software" "/UserEmail:my@email.com"
```

Broken down into sections - these are the switches and what each does.

Note: Please note that the indicated string content must begin and end with double quote marks (")!

SWITCHES	DESCRIPTION
MAIN install executable and path (required)	This is the full path to the installation executable, note it is the standard install executable - both silent and non silent installations are available from the single executable. C:\PDFX5SA_sm.exe
/VERYSILENT	Designates that the installation once started should be silent and require no user interaction.
/SILENT	Designates that the install should only request essential information from the user once started.
/NORESTART	Once the install is complete - no reboot will occur - but beware - PDF-XChange does require a reboot before updates will take effect - not recommended.
/DIR=(required)	The full path to install all required files to, created if not already available. /DIR="C:\Program Files\PDF-XChange\"
/Group=(Optional)	The Windows 'Start Menu' folder in which to locate the Menu options provided when installing PDF-XChange - this will always be a 'Child' menu option of the default 'Programs Files' menu and this cannot be changed. /GROUP="Tracker Software\PDF-XChange 2012"
/COMPONENTS=	If this switch is specified only those components actually listed will be installed, otherwise all components are installed. The available components are: - pdfSaver (Always Required) - PDF-XChange driver (Always Required) - Help - Languages (please note 'ss' is required) - Autounload /COMPONENTS="pdfSaver, PDF-XChange driver, Help,Languages"

<p>/AutoUnload:nn</p>	<p>If this option is specified, after the specified timeout (nn, in minutes) the PDFSaver4 module of the PDF-XChange driver will unload.</p> <p>Useful when installed on a Windows Terminal Server or Citrix Metaframe server with multiple Users to minimize memory in use when PDF creation is not being used by a user - will automatically reload when required.</p> <p>" /Autounload:03 "</p> <p>(Strictly speaking Autounload is not a component - but a default setting parameter mainly used in Citrix Metaframe or Terminal Services installations to minimize memory use on a server where PDFSaver4.exe could be loaded multiple times.)</p>
<p>/Key (Registration) - only to be used if you are also providing an END USER enabled license - not included as part of your Developers SDK License - EVER</p>	<p>Allows the license key to be registered during silent installation - for live use. Without this information PDF-XChange will function in demo mode only.</p> <p>" /Key:XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX "</p>
<p>/UserName Registration</p>	<p>Allows the registered username to be registered during silent installation.</p> <p>Username " /UserName:MY Name "</p>
<p>/Organization Registration</p>	<p>Allows the Company details to be registered during silent installation.</p> <p>" /Organization:Tracker software "</p>
<p>/PDEFAULT= (Optional)</p>	<p>Sets PDF-XChange to be the system default Printer - the default option without this entry is that PDF-XChange will not be the default printer.</p>
<p>/PName= (Optional) printer renaming</p>	<p>Sets the PDF-XChange printer name as required - the default option without this entry is currently "PDF-XChange 5.0".</p> <p>/PName="PDF-XChange for my application"</p> <p>Please note that the string content must begin and end with double quote marks (").</p>
<p>/UserEmail Registration</p>	<p>Allows the users email address details to be registered during silent installation.</p> <p>" /UserEmail:my@email.com "</p>

<p>/Lang</p>	<p>Sets a language as the default language for the User Interface for all the installed programs and Print drivers (PDF-XChange, PDF-Tools, OFFice2PDF and the MS Office Addin's etc).</p> <p>/LANG=language en (English), nl (Dutch), uk (Ukrainian), fr (French), gr (German), it (Italian), sp (Spanish), ko (Korean), ch (Chinese), jp (Japanese), fi (Finnish)</p> <p>This may not be the complete list - please see the installation folder.</p> <p>'C:\Program Files\Tracker Software\PDF-XChange 5\Languages' for the up to date list of supported language files.</p> <p>Example (for German)</p> <pre>PDFX5SA_sm.exe [other parameters] /LANG=gr [Additional parameters]</pre>
<p>/DRVwoPRN</p>	<p>Installs the printing drivers without a visible "printer".</p>

Warning! Failure to ensure that special switch characters such as quote marks (") and "/" are included in your command line or batch file, in the correct position - will result in failure to behave as required.

IPXCControlEx Object

The **IPXCControlEx** ActiveX object is used to create a new **temporary** PDF-XChange printer instance, and returns the **IPXCPrinter** object, this in turn allows the developer's application to set and control the properties of the newly created printer instance. Once the job is completed the printer instance is destroyed.

IPXCControlEx.Printer

Property **IPXCControlEx.Printer** allows the creation of a new temporary printer based on the PDF-XChange 2012 driver and returns the IPXCPrinter object associated with this printer.

Declaration

C#

```
object get_Printer(string pServerName,  
string pPrinterName,  
string pRegKey,  
string pDevCode);
```

C++

```
HRESULT get_Printer([in] BSTR pServerName,  
[in] BSTR pPrinterName,  
[in] BSTR pRegKey,  
[in] BSTR pDevCode,  
[out, retval] VARIANT* ppPrinter);
```

VB

```
Property Printer(pServerName As String,  
pPrinterName As String,  
pRegKey As String,  
pDevCode As String)
```

Parameters

pServerName

Reserved for future use. Should be a NULL or empty string.

pPrinterName

Specifies the desired name for the printer to be created. If a printer of the same name already exists, then to ensure it is uniquely named, the new printer name will have a suffix automatically generated, based on the originally-passed value.

Note: Differences between Licensed and Evaluation use.

The following 2 items are provided as part of your license purchase. If you are evaluating the **PDF-XChange Drivers API SDK**, all generated output will have evaluation watermarks printed on each page and these cannot be subsequently be removed. Once licensed you must re-compile your project, embedding your license strings provided within your code as outlined above and all new PDF files generated will be free of the evaluation watermarks placed on all evaluation output. All output created in evaluation mode will need to be recreated from the original source documents and materials.

pRegKey

String containing the Developer registration key. If this registration key is not present or is invalid, the PDF-XChange demo label will be added to all PDF pages generated by the printer.

pDevCode

String containing the developer code. If this developer code is not present or is invalid, the PDF-XChange demo label will be added to all PDF pages generated by the printer.

Return value

If successful, the function returns an **IPXCPrinter** object associated with newly created printer. On failure the function returns a NULL.

Example (VB)

```
01 Dim PDFFactory As New PXCComLib5.CPXControlEx
02 Dim WithEvents PDFPrinter As PXCComLib5.CPXPrinter
03
04 Public Sub Form_Load()
05 Set PDFPrinter = PDFFactory.Printer("", "PDF-XChange 2012 Sample", "<reg>",
06 "<dev>");
07
08 Public Sub Form_Unload(Cancel As Integer)
09 Set PDFPrinter = Nothing
10 End Sub
11 </dev></reg>
```

Interface IPXCPrinter

The **IPXCPrinter** ActiveX interface object is used to control all options of the PDF creation process. The object with this interface is returned by the **IPXCControlEx** ActiveX object. When printing is completed, the **IPXCPrinter** object should be released. Note that releasing this object will also remove the virtual printer associated with this object.

Events provided by **IPXCPrinter** object can be used to control printing process. **IPXCPrinter** ActiveX interface provides the following **Properties**, **Methods**, and **Events**.

Properties

1. **Property Name** returns the name of the printer associated with the object.
2. **Property Option** is used to specify almost all the properties of the printer.

The following sections define the specific Property Options:

1. **Section Paper** properties in this section allow you to define page layouts for the created PDF document.
 1. **Normal** - Used to specify position and size of the page on the sheet.
 2. **n-Up** - Defines how many and in which order pages should be placed on the sheet.
 3. **Booklet** - Defines booklet (or book) creation parameters.
2. **Section General** defines the PDF Specification to which the created PDF file should conform, and the options for the PDF viewing application.
3. **Section Compression** specifies which compression options should be used for the PDF document's text and images.
 1. Subsection **Color** specifies compression for True color images.
 2. Subsection **Indexed** specifies image compression for images with a color palette.
 3. Subsection **Mono** specifies the compression method to be used to compress monochrome/black-&-white images.
4. **Section Graphics** specifies how different types of images should be processed before inserting into a PDF document.
 1. Subsection **Color** determines image downsampling and conversion options for True color images.
 1. **Color.Downsampling** defines whether and how images should be "downsampled," i.e. have their resolution reduced.
 2. **Color.Convert** defines whether color images should be converted to grayscale, monochrome, or black & white.
 2. Subsection **Indexed** determines image downsampling and conversion options for indexed images with palettes.
 1. **Indexed.Downsampling** defines whether and how images should be "downsampled," i.e. have their resolution reduced.
 2. **Indexed.Convert** defines whether color images should be converted to grayscale, monochrome, or black & white.
 3. Subsection **MonoAndDrawings** determines settings for downsampling and converting monochrome images and black & white drawings.
 1. **MonoAndDrawings.Downsampling** defines whether and how images should be "downsampled," i.e. have their resolution reduced.
 2. **MonoAndDrawings.Convert** defines whether color images should be converted to

-
- grayscale, monochrome, or black & white.
5. **Section Fonts** contains properties which define embedding and other options for fonts to be used in the PDF document.
 6. **Section Security** defines security options for the PDF document.
 7. **Section DigSig (Digital Signatures)** defines Digital Signature properties of the document and the properties for the signature used for signing.
 8. **Section Overlay** contains properties which define how a newly created document can be overlaid using an existing PDF file.
 9. **Section Links** determines whether to insert live links within the PDF document, what kinds of links, and how they are to be displayed.
 10. **Section Watermarks** allows the developer to enable or disable adding watermark(s) into the document, and to define a set of these watermarks.
 11. **Section HeadersFooters** defines headers and footers to be placed on all pages of the PDF document.
 12. **Section DocInfo** allows PDF document information fields to be defined and allow adding of XMP Metadata within the PDF document.
 13. **Section Save** allows you to define the folder where PDF document should be stored, the file name of the document and other document saving options.
 14. **Section EMail** is used when the created PDF file should be emailed after creation and define the parameters for emailing. The created PDF file will be sent as an attachment of this email.
 15. **Section Optimization** controls Optimizations of the generated PDF file.
 16. **Section Bookmarks** activates automatic bookmark tree creation for the PDF document and defines the parameters for this process.
 17. **Section Saver** controls the **ShowProgress** property of the **pdfSaver** application.

Name

Property Name

This property is a read only property and returns the name of the printer associated with the object.

Declaration

C#

```
string Name;
```

C++

```
HRESULT get_Name([out, retval] BSTR* pName);
```

(VB)

```
Property Name As String
```

Please note that this name can be different from that passed in the Printer properties of the IPXCControlEx object, if a printer with the requested name already existed.

Option

Property Option

This is a Write only parameterized property option and is used to specify almost all the properties of the PDF-XChange printer.

Declaration

C#

```
void set_Option(string pOptionName, object _p2);
```

C++

```
HRESULT put_Option([in] BSTR pOptionName, [in] VARIANT _arg2);
```

VB

```
Property Option(pOptionName As String)
```

A detailed list of all available properties with detailed descriptions is listed in Properties. All properties are identified by a string for that pOptionName. The name of the property will also usually contain some parts separated by the dot (.) character. Values of these properties can be strings, integers, boolean or binary values (an array of bytes).

Property Table Definition

Each property section has a table detailing the possible property values. The three columns of the tables provided are always:

- **Name.** The name of the property. This is what you assign the value to. For example: "**Paper.LayoutType**".
- **Possible Values.** These are the allowed values for the property specified in **Name**.
- **Description.** Describes the use or function of the property.

The possible values are described as numbers and corresponding strings, and either may optionally be used to set the value of the property.

For example: **0 (Normal)** means that this property can be set as the integer value **0** or as the string "**Normal**".

Therefore, both of these assignments are equivalent:

- `pPrinter.Option["Paper.LayoutType"] = 0`
- `pPrinter.Option["Paper.LayoutType"] = "Normal"`

Within the table column **Possible Values** if the entry is Boolean Value then an acceptable entry may be any of the following:

For the value **0** any one of the following strings would also be acceptable: **No**, **False**, or **Disabled**. For the value of **1** any one of the following strings would also be acceptable: **Yes**, **True**, or **Enable**. For example, both of these two assignments are equivalent:

- `pPrinter.Option["Compress.Graphics"] = 1`
- `pPrinter.Option["Compress.Graphics"] = "Enable"`

Please note, names of properties and named values are **case sensitive**.

Paper

Properties in this section allow you to define page layouts for the created PDF document.

Name	Possible values	Description
LayoutType	0 (Normal) 1 (n-UP) 2 (Booklet)	Defines the layout type used for page creation of the PDF document. This property determines which other property sections are to be used: Normal, n-Up, or Booklet. Default value: 0 (Normal)
MirrorX	Boolean Value	Enables or disables horizontal mirroring of all drawings. Default value: False
MirrorY	Boolean Value	Enables or disables vertical mirroring of all drawings. Default value: False
SheetSizeIndex	From -2 to 41	Specifies the sheet used for PDF creation. Value -2 means custom sheet size which is defined by properties SheetWidth and SheetHeight. Value -1 means Auto sheet size. In this case sheet size is dependant on the layout type and page size. All other values are indices of the standard paper sizes defined in PDF-XChange. See List of Standard Paper Sizes. Default value: -1
SheetWidth	13-5080	Specifies the width and height of the sheet. Values are in tenths of millimeters and must be within the specified range. Default values: 2100 (width) and 2970 (height)
SheetHeight	13-5080	The maximum height/width of a PDF page are 200" x 200" or approx : 5080x5080 mm as defined in the PDF Specification.

Normal

The maximum height/width of a PDF page are 200" x 200" or approx : 5080x5080 mm as defined in the PDF Specification.

Name	Possible values	Description
LeftPos	Integer	These properties define the position of the top left corner of the page relative to the top left corner of the sheet. Values are in tenths of millimetres.
TopPos	Integer	These properties are used when both the Centre and Fit properties have <code>False</code> values. Default values: 0
PageScale	Integer	Specifies the scale level to use when positioning the page on the sheet. Value is in tenths of 1 percent. This property is only used if the Fit property is <code>False</code> . Default value: 1000
Center	Boolean Value	If this property is <code>True</code> , the page is centred on the sheet. Otherwise its position is determined by LeftPos and TopPos properties. This property is not used when the Fit property is <code>True</code> . Default value: <code>True</code>
Fit	Boolean Value	Specifies that the page should be 'fitted' to the sheet. Default value: <code>True</code>

n-Up

When the LayoutType specified is nUP, the following properties define how many and in which order, pages should be placed on the sheet.

Name	Possible values	Description
Nup	0 - one page per sheet 1 - two pages per sheet 2 - four pages per sheet 3 - six pages per sheet 4 - eight pages per sheet 5 - nine pages per sheet 6 - sixteen pages per sheet	Defines the number of pages to be placed on one sheet. Default value: 0
NupOrder	0 (<code>AcrossFromLeft</code>) 1 (<code>DownFromLeft</code>) 2 (<code>AcrossFromRight</code>) 3 (<code>DownFromRight</code>)	Defines the order in which pages should be placed on the sheet. Default value: 0 (<code>AcrossFromLeft</code>)

Borders	Boolean Value	Specifies if a border should be drawn around the pages on the sheet. Default value: <code>False</code>
---------	---------------	---

Booklet

When the **LayoutType** value is **Booklet**, the following properties define the booklet creation parameters.

Name	Possible values	Description
BookletType	0 (Brochure) 1 (Book)	If the value of this property is Book, pages will be grouped in typographic gatherings by the number specified in the SheetsInBook property. Otherwise, the booklet will contain only one book with all the pages included. Default value: 0 (Brochure)
SheetsInBook	1-64	Defines the number of sheets in one book when the property BookletType has the value Book. Default value: 4
GutterSize	Positive integer	Defines the gutter size for the booklet. Value is specified in tenths of millimeters. Default value:

Standard Paper Sizes

List of predefined papers in the PDF-XChange 2012. Number, specified into column **Index**, can be used to specify the sheet size into property **Paper.SheetSizeIndex**.

Column **Standard Index** contains index for the paper as it is defined into Windows API. Please note that some paper definitions are specific for PDF-XChange and have no standard index.

Dimension column defines the size of the paper for its standard orientation (portrait). Dimensions are specified in inches or in millimeters.

Index	Standard Index	Paper name	Dimension
0	1	Letter/ANSI A	8,5 x 11 "
1	3	Tabloid/ANSI B	11 x 17 "
2	-	ANSI Super B	13 x 19 "
3	24	ANSI C	17 x 22 "
4	25	ANSI D	22 x 34 "
5	26	ANSI E	34 x 44 "
6	-	ANSI F	28 x 40 "
7	5	US Legal	8,5 x 14 "
8	6	Statement	5,5 x 8,5 "
9	7	Executive	7,25 x 10,5 "
10	-	A0	841 x 1189 mm
11	-	A1	594 x 841 mm
12	62	A2	420 x 594 mm
13	8	A3	297 x 420 mm
14	9	A4	210 x 297 mm
15	11	A5	148 x 210 mm
16	-	B0	1000 x 1414 mm
17	-	B1	7070 x 1000 mm
18	-	B2	5000 x 7070 mm
19	42	B4 (ISO)	250 x 353,6 mm
20	12	B4 (JIS)	257 x 364 mm
21	34	B5	176 x 250 mm
22	13	B5 (JIS)	182 x 257 mm
23	14	Folio	8,5 x 13 "
24	15	Quarto	215 x 275 mm
25	18	Note	8,5 x 11 "
26	19	Envelope #9	3,875 x 8,875 "
27	20	Envelope #10	4,125 x 9,5 "
28	21	Envelope #11	4,5 x 10,375 "
29	23	Envelope #14	5 x 11,5 "
30	27	Envelope DL	110 x 220 mm

31	28	Envelope C5	162 x 229 mm
32	29	Envelope C3	324 x 458 mm
33	30	Envelope C4	229 x 324 mm
34	31	Envelope C6	114 x 162 mm
35	32	Envelope C65	114 x 229 mm
36	35	Envelope B6	125 x 176 mm
37	37	Envelope Monarch	3,875 x 7,5 "
38	38	Envelope Personal	3,625 x 6,5 "
39	-	18x24	18 x 24 "
40	-	24x36	24 x 36 "
41	-	30x42	30 x 42 "

General

Properties in this section define the PDF Specification to be applied to the created PDF file and the parameters for the PDF viewing application, which may be used to define the properties used when opening the document for Viewing.

Name	Possible values	Description
Specification	0 (Auto) 3 (1.3) 4 (1.4) 5 (1.5) 6 (1.6) 7 (1.7) -1 (PDF/A-1b) -2 (PDF/A-1a) -3 (PDF/A-2a) -4 (PDF/A-2b) -5 (PDF/A-2u)	Defines the PDF Specification level to which the PDF file should conform. Value <code>Auto (0)</code> , allows PDF-XChange to choose the most efficient specification based on other specified options. When the value from -5 to -1 specified, PDF-XChange will create a PDF file which conforms to ISO standard PDF/A. Please note that some PDF functions and content are not available/allowed when specifying PDF/A formatting. Default value: 0 (Auto)
PageLayout	0 (Default) 1 (ShowNone) 2 (Bookmarks) 3 (Thumbnails) 4 (FullScreen)	Default value: 0 (Default)
PageMode	0 (Default) 1 (ShowNone) 2 (Bookmarks) 3 (OptionalContent)	Default value: 0 (Default)

FullScreenMode	0 (Default) 1 (ShowNone) 2 (Bookmarks) 3 (OptionalContent)	Default value: 0 (Default)
MagnificationType	0 (Default) 1 (ActualSize) 2 (FitPage) 3 (FitWidth) 4 (FitHeight) 5 (Level)	Default value: 0 (Default)
MagnificationLevel	1-6400	Default value: 100
OpenToPage		Default value: 0
HideToolbars	Boolean Value	Default value: False
HideMenubar	Boolean Value	Default value: False
HideUI	Boolean Value	Default value: False
FitWindow	Boolean Value	Default value: False
CenterWindow	Boolean Value	Default value: False
ShowDocTitle	Boolean Value	Default value: False

Compression

The properties of this section specify which compression should be used for the PDF document's text and images.

Name	Possible values	Description
Graphics	Boolean Value	Specifies PDF document image compression. If this value is <code>True</code> , compression methods for different types of images are defined in the subsections for Color , Indexed and Mono images. Default value: <code>True</code>
Text	Boolean Value	If this property is <code>True</code> , textual content of the PDF pages will be compressed which reduces the size of the resulting PDF document. Default value: <code>True</code>

Color

Specifies compression for True color images.

Name	Possible values	Description
Enabled	Boolean Value	Default value: <code>True</code>
Method	0 (Auto) 1 (JPEG) 2 (ZIP) 3 (JPEGZIP) 4 (J2K) 5 (J2KZIP)	Specifies compression method to be used for compressing True color images. Please note, when the PDF/A specification is used for creating a PDF file, compression methods J2K and J2KZIP cannot be used. Default value: 0 (Auto)
JPEGQuality	1-100	Specifies the image quality when using JPEG compression (JPEG and JPEGZIP). Higher quality means better image definition, and a larger compressed stream, so a larger file. Default value: 75

Indexed

Name	Possible values	Description
Enabled	Boolean Value	If this property is <code>True</code> , Indexed images will be compressed by the method specified by property Method . Otherwise they will be stored into PDF without compression. Default value: <code>True</code>
Method	0 (<code>Auto</code>) 1 (<code>PackBits</code>) 2 (<code>ZIP</code>) 3 (<code>LZW</code>)	Defines the compression method to be used to compress images with palette. Please note, the compression method LZW cannot be used with the PDF/A output specification. Default value: 0 (<code>Auto</code>)

Mono

Defines the compression method to be used to compress Monochrome/black & white images.

Name	Possible values	Description
Enabled	Boolean Value	Default value: <code>True</code>
Method	0 (<code>Auto</code>) 1 (<code>ZIP</code>) 2 (<code>CCITT3</code>) 3 (<code>CCITT4</code>) 4 (<code>RLE</code>) 5 (<code>JBIG2</code>)	Defines the compression method to be used to compress monochrome/black & white images. Please note , the compression method JBIG2 cannot be used with the PDF/A output specification. Default value: 0 (<code>Auto</code>)
JBIG2Method	0 (<code>Standard</code>) 1 (<code>CropBorders</code>) 2 (<code>Symbols</code>)	Specifies preprocessing method of images which will be compressed using JBIG2 method. Default value: 0 (<code>Standard</code>)

Graphics

This section contains three subsections which define how different types of images should be processed before inserting into a PDF document.

Color

Determines image downsampling and conversion options for True color images.

- Subsection **Downsample** defines when and how images should be downsampled.
- Subsection **Convert** specifies whether to use color reducing for True color images and into which format they should be converted.

Color.Downsampling

This subsection defines when and how images should be downsampled.

Name	Possible values	Description
Method	0 (None) 1 (Linear) 2 (Bilinear) 3 (Bicubic)	Specifies which method should be used for downsampling. If the value of this property is 0 (None), images of this type will not be downsampled. Default value: 0 (None)
To	50-2399	These two properties define when an image should be downsampled. Default value: 96
IfAbove	51-2400	If the original DPI of the image is equal to or above that of the value specified by the property IfAbove, the image will be downsampled to the DPI specified by the property To. Default value: 128

Color.Convert

Specifies whether color images should be converted to grayscale, monochrome or black & white.

Name	Possible values	Description
Method	0 (None)	Specifies how images of this type should be converted.
	1 (ToGrayscale)	Value 0 (None) means that no images of this type will be converted. Default value: 0 (None)
	2 (ToMonochrome)	
	3 (ToBlackAndWhite)	
Dither	Boolean Value	Specifies whether images should be dithered during conversion. This has meaning only for methods ToMonochrome and ToBlackAndWhite . Default value: True

Indexed

Determines image downsampling and conversion options for indexed images with palettes.

- Subsection **Downsample** defines when and how images should be downsampled.
- Subsection **Convert** specifies whether to use color reducing for indexed images and into which format they should be converted.

Indexed.Downsampling

This subsection defines when and how images should be downsampled.

Name	Possible values	Description
Method	0 (None)	Specifies which method should be used for downsampling.
	1 (Linear)	If the value of this property is 0 (None), images of this type will not be downsampled.
	2 (Bilinear)	
	3 (Bicubic)	Default value: 0 (None)
To	50-2399	These two properties define when an image should be downsampled. Default value: 96
IfAbove	51-2400	If the original DPI of the image is equal to or above that of the value specified by the property <code>IfAbove</code> , the image will be downsampled to the DPI specified by the property <code>To</code> . Default value: 128

Indexed.Convert

Specifies whether indexed images should be converted to grayscale, monochrome, or black & white.

Name	Possible values	Description
Method	0 (None)	Specifies how images of this type should be converted.
	1 (ToGrayscale)	Value 0 (None) means that no images of this type will be converted.
	2 (ToMonochrome)	
	3 (ToBlackAndWhite)	Default value: 0 (None)
Dither	Boolean Value	Specifies whether images should be dithered during conversion. This has meaning only for methods ToMonochrome and ToBlackAndWhite . Default value: True

MonoAndDrawings

This section determines settings for downsampling and converting monochrome images and black & white drawings.

- Subsection **Downsample** defines when and how images should be downsampled.
- Subsection **Convert** specifies whether mono or black & white images should be converted.

MonoAndDrawings.Downsampling

This subsection defines when and how images should be downsampled.

Name	Possible values	Description
Method	0 (None)	Specifies which method should be used for downsampling.
	1 (Linear)	If the value of this property is 0 (None), images of this type will not be downsampled.
	2 (Bilinear)	
	3 (Bicubic)	Default value: 0 (None)
To	50-2399	These two properties define when an image should be downsampled. Default value: 96
IfAbove	51-2400	If the original DPI of the image is equal to or above that of the value specified by the property IfAbove , the image will be downsampled to the DPI specified by the property To . Default value: 128

MonoAndDrawings.Convert

Specifies images should be converted as either mono or black & white images.

Name	Possible values	Description
Method	0 (None)	Defines how mono images and line art should be converted.
	1 (ToGrayscale)	Value 0 (None) means that no conversions will be performed.
	2 (ToMonochrome)	
	3 (ToBlackAndWhite)	Default value: 0 (None)
Threshold	1-255	Defines the level behind which color of line arts will be treated as black. For comparison the grayscale level of line art color is used. Default value: 128

Fonts

This section contains properties which define embedding and other options for fonts to be used in the PDF document.

Name	Possible values	Description
EmbedAll	Boolean Value	Default value: False
EmbedProtected	Boolean Value	Default value: True
WriteToUnicode	Boolean Value	Default value: False
Subset	Boolean Value	<p>If this option is enabled and if the percentage of characters used from the font specified is less than that specified in the <code>SubsetLevel</code> property, only the actually used characters from the font will be embedded within the PDF document.</p> <p>Otherwise the complete font file will be embedded within the PDF - resulting in a larger PDF file - but this will ensure consistent formatting/viewing is always possible.</p> <p>Default value: True</p>
SubsetLevel	1-100	Default value: 100

Additionally there are two identical subsections **AlwaysEmbed** and **NeverEmbed**, which define lists of fonts which should always be embedded or which never should be embedded.

Each of these subsections has the following properties:

Name	Possible values	Description
Enable	Boolean Value	<p>Enable using the list defined by the <code>Fonts</code> property.</p> <p>Default value: False</p>
Fonts	String	Contains a comma separated list of fonts. Default value: empty list.

Security

This section defines security options for the PDF document.

Note: When the PDF/A format is specified the security options for the document will not be used and will be ignored, it is advisable therefore to actually disable security to enable faster file processing.

Name	Possible values	Description
Enabled	Boolean Value	Specifies if the PDF document should be secured or not. If this property has the value <code>False</code> , no other properties from this section will be used. Default value: <code>False</code>
HasUserPassword	Boolean Value	These properties define if the PDF document will contain a valid User and Owner password. If the PDF document has no user password, this document can be opened without the need to enter a password in any PDF viewing or manipulation tool - even if an Owner password is set. To ensure robust security - set both an Owner and User password! Default value: <code>False</code>
HasOwnerPassword	Boolean Value	Default value: <code>False</code>
UserPassword	String	Defines the User password. This password is used for opening the document with user privileges. Default value: empty string.
OwnerPassword	String	Defines the Owner password. This password is required to change permissions/security settings in the PDF document. If this password is empty or the same as the User password, changing the permissions /security will be impossible. Default value: empty string.

Type	<p>0 (40RSA)</p> <p>1 (128RSA)</p> <p>2 (128AES)</p> <p>3 (256AES)</p> <p>4 (256AES_R6)</p>	<p>Defines the security method used for encrypting the document. The value of this property specifies which subsection (Perms40 or Perms128) will be used to define users permission for the document.</p> <p>The last option - 256AES_R6 - represents encryption algorithm that should be introduced in PDF 2.0 Specification, but already used by Adobe Acrobat products.</p> <p>Default value: 0 (40RSA)</p>
Perms40.AllowPrinting	Boolean Value	Default value: False
Perms40.AllowChanging	Boolean Value	Default value: False
Perms40.AllowCopying	Boolean Value	Default value: False
Perms40.AllowComments	Boolean Value	Default value: False
Perms128.ContentAccess	Boolean Value	Default value: False
Perms128.AllowCopying	Boolean Value	Default value: False
Perms128.Changes	0 - 4	<p>Specifies the level of security to be applied regarding changes to the document.</p> <p>0 = No changes will be allowed.</p> <p>1 = Only document assembly is allowed for the user.</p> <p>2 = Only form field fill-in and signing is allowed.</p> <p>3 = Allows comment authoring, form field fill-in and signing of the document.</p> <p>4 = Allows general editing, comments and form field authoring for the user.</p> <p>Default value: 0</p>
Perms128.Printing	<p>0 - not allowed</p> <p>1 - low resolution</p> <p>2 - fully allowed</p>	<p>Specifies user's permissions for printing the document.</p> <p>Default value: 0</p>
Perms128.EncryptMeta	Boolean Value	<p>Specifies if metadata In the PDF document should be encrypted or not.</p> <p>N.B. This property is applicable for PDF specifications 1.6 and higher.</p> <p>Default value: True</p>

DigSig (Digital Signatures)

This section defines Digital Signature properties of the document and the properties for the signature used for signing.

Name	Possible values	Description
Enabled	Boolean value	If this property is <code>False</code> , digital signing is not used and no other properties need to be specified. Default value: <code>False</code>
Store	0 (System) 1 (File)	Defines which stored certificate should be used. If the value of this property is 0 (System), the default user's stored system certificate will be used. If the value of this property is 1 (File), an external PFX file must be specified. Default value: 0 (System)
Hash	Binary	This property specifies the hash value of the certificate to be used for signing the document. When Store value is 0 (System), PDF-XChange will find the corresponding certificate within the system certificate storage. When Store value is 1 (File), this property is not used.
PFXFile	String	When Store value is 1 (File), this specifies the PFX file which contains the digital signature for signing the document. When Store value is 0 (System), this property is not used.
PFXPassword	String	When Store value is 1 (File), this specifies the password for the PFX file. When Store value is 0 (System), this property is not used.
Reason	String	These properties define additional text properties to be included with the digital signature within the PDF document. Default values: empty string.
Location		
Contact		

Graphic	<p>0 (None)</p> <p>1 (Image)</p> <p>2 (Name)</p>	<p>Defines how the graphic (left) portion of the signature fields should be displayed.</p> <p>If the value is <code>None</code>, this will not be used, and information about the digital signature will be placed in the entire area of the signature field.</p> <p>If the value is <code>Image</code>, the image file specified by <code>GraphicFileName</code> will be used.</p> <p>With the value <code>Name</code>, the signer's name as in the certificate will be used.</p> <p>Default value: 2 (Name)</p>
GraphicFileName	String	<p>When <code>Graphic</code> is <code>Image</code>, this is the fully qualified path name for the image to be used on the signature.</p> <p>This image can be any one of the supported image formats.</p>
ShowFlags	Integer	A combination of flags which define how the signature field should look. See details below.
Page	Integer	<p>Zero based page number - on which the digital signature should be placed. A value of -1 for this property places the signature on the last page of the document.</p> <p>Default value: -1</p>
Left	Integer	<p>This property define the position (relative to the top left corner of the page). Value are specified in tenths of a millimetre.</p> <p>Default value: 254</p>
Top	Integer	<p>This property define the position (relative to the top left corner of the page). Value are specified in tenths of a millimetre.</p> <p>Default value: 254</p>
Width	Integer	<p>This property define the dimension of the signature field. Value are specified in tenths of a millimetre.</p> <p>Default value: 508</p>
Height	Integer	<p>This property define the dimension of the signature field. Value are specified in tenths of a millimetre.</p> <p>Default value: 168</p>

ShowFlags defines the information which should be placed on Signature field appearance. It can be combination of the following values:

Value	Meaning
1	If this value is used, the name of the signer (given from the certificate) will be shown on the signature field.
2	If this value is used, signing date will be placed on signature field.
4	If this value is used, the text specified by Location property will shown.
8	If this value is used, the text specified by Reason property will shown.
16	If this value is used, Distinguished Name field from certificate will be added to the signature field.
32	If this value used, each text item placed on the signature field will be prepended by its corresponding label (for example, Name: Tracker Software Ltd.)

Overlay

This section contains properties which define how a newly created document can be overlaid using an existing PDF file.

Note: Using overlays automatically disables the PDF/A feature!

Name	Possible values	Description
Enabled	Boolean value	Enables or disables overlay feature of the PDF-XChange. If this flag is <code>False</code> , all other properties from this section will be ignored. Default value: <code>False</code>
OverlayFile	String	Specifies a file with which the newly created PDF will be overlaid. The value should be a fully-qualified path name for the PDF file. Default value: empty string.
OverlayPassword	String	If the file specified by the <code>OverlayFile</code> property is encrypted with a valid user password, <code>OverlayPassword</code> must contain a valid password for the file. Default value: empty string.
AsForeground	Boolean value	Defines where overlay pages should be located. Background is the default unless specified. Default value: <code>False</code>

Repeat	<p>0 (NoRepeat)</p> <p>1 (RepeatLastPage)</p> <p>2 (RepeatContinuous)</p>	<p>Specifies the repeat option for overlaying. Meaning of values:</p> <p>0 (NoRepeat): Do not Repeat - begin at the start of the overlay file and apply each corresponding page from this file to the corresponding page number in the newly generated PDF file. Should the newly created file have a greater number of pages than available in the overlay file - no overlay is applied to the remaining pages created in the newly generated PDF document.</p> <p>1 (RepeatLastPage): Apply the PDF overlay file pages to the corresponding pages on the newly generated PDF file - should the newly created file have a greater number of pages - apply the last page of the overlay file to all remaining pages. If the overlay file is a single page - this will be placed on all of the generated files pages by default.</p> <p>2 (RepeatContinuous): Begin at the start of the overlay file and apply each corresponding page from this file to the corresponding page number in the generated PDF file. Should the newly created file have a greater number of pages than available in the overlay file, start at the beginning of the Overlay file and begin the process again, applying the first page in the Overlay PDF file to the next page in the generated output PDF document in the sequence, then page 2 of the overlay file to the next generated PDF document page and so on - until the generated document is completed. If the overlay file is a single page - this will be placed on all of the generated files pages by default.</p> <p>Default value: 0 (NoRepeat)</p>
AlignH	<p>0 (Left)</p> <p>1 (Center)</p> <p>2 (Right)</p>	<p>Specifies the horizontal alignment for the overlay pages relative to the new document's pages.</p> <p>Default value: 1 (Center)</p>
AlignV	<p>0 (Top)</p> <p>1 (Middle)</p> <p>2 (Bottom)</p>	<p>Specifies the vertical alignment for the overlay pages relative to the new document's pages.</p> <p>Default value: 1 (Middle)</p>
Fit	Boolean value	<p>'Fits' the OverlayFile pages to match that of the host page.</p> <p>Default value: False</p>
KeepAspect	Boolean value	<p>The OverlayFile pages' proportions should be retained when using other options.</p> <p>Default value: False</p>

Links

This property determines whether to insert live links within the PDF document, what kinds of links, and how they are to be displayed (**Border** property).

Name	Possible values	Description
Enabled	Boolean value	This will detect URL and mail links - please note to be valid, links must be valid and occupy a single line of the page. Embedded links cannot be recognized (e.g. 'Click Here') Default value: <code>False</code>
DetectFiles	Boolean value	Links to files - must be valid and contain a fully qualified path. Default value: <code>False</code>
CheckExistance	Boolean value	Check a links validity as part of the recognition and PDF page inclusion process. Default value: <code>False</code>

The **Border** property of this section defines the links border parameters and the way the link will appear on the page.

Name	Possible values	Description
Type	0 (None) 1 (Underline) 2 (Rectangle)	Specifies the type of links border. Default value: 0 (None)
Style	0 (Solid) 1 (Dashed) 2 (Dotted)	Default value: 0 (Solid)
Color	Integer	Specifies RGB value of color for links border. To specify this value RGB macro defined into Windows API can be used. Default value: 255 (red color)
Width	5 - 50	Defines the width of links border. Value is specified in tenth of millimeters. Default value: 10

Watermarks

These properties allow the developer to enable or disable adding watermark(s) into the document, and to define a set of these watermarks.

Name	Possible values	Description
Enabled	Boolean value	If this property is <code>True</code> , watermarks defined by <code>Watermarks</code> property will be added to the PDF file. Otherwise watermarks will not be used. Default value: <code>False</code>
Watermarks	String	A semicolon (;) separated list of watermark names that should be added to the document. Each of these watermark should be added using AddTextWatermark or AddImageWatermark methods. Default value: empty string.

Bookmarks

This section activates automatic bookmark tree creation for the PDF document and defines the parameters for this process.

Name	Possible values	Description
Enabled	Boolean value	Enables automatic creation of a bookmark tree within the document. If the value of this property is <code>False</code> , no bookmarks tree will be created, and the rest of the properties of this section are ignored. Default value: <code>False</code>
AllowMultiLine	Boolean value	Activates detection of multiline bookmark items. Default value: <code>True</code>
DocTitleAsRoot	Boolean value	When <code>True</code> , the <code>Root</code> element of the bookmarks tree will be the document title (see DocInfo.Title property) and all other top level bookmarks will be children of this element. Default value: <code>False</code>

MatchLevel	Boolean value	<p>When <code>True</code>, a detected non-top level item will only be added to the tree if a previously detected item of higher level already exists.</p> <p>When <code>False</code>, a detected item will always be added to the tree, and if there was no parent detected at a higher level, dummy parents will be added.</p> <p>Default value: <code>False</code></p>
DisplayMode	<p>0 (<code>AllOpened</code>)</p> <p>1 (<code>AllClosed</code>)</p> <p>2 (<code>ByItem</code>)</p> <p>3 (<code>OpenToLevel</code>)</p>	<p>Defines which items of the bookmarks tree should be initially opened into the PDF viewer application.</p> <p>Value <code>AllOpened</code> means that all items will be opened.</p> <p>Value <code>AllClosed</code> means that all items will be closed.</p> <p>Value <code>ByItem</code> means the state of the item defined by its template definition will be used.</p> <p>Value <code>OpenToLevel</code>, means that all items up to the level specified by <code>UpToLevel</code> property will be opened. All other items will be closed by default.</p> <p>Default value: 2 (<code>ByItem</code>)</p>
UpToLevel	Integer	<p>Defines the level into bookmarks tree up to which items will be initially opened into the PDF viewer application.</p> <p>Only active if <code>DisplayMode</code> has the value 3 (<code>OpenToLevel</code>).</p> <p>Default value: 1</p>
Template	String	<p>Specified the name of Template used for detection and creation bookmarks tree within the PDF document.</p> <p>This property is not used with the ActiveX control and should be an empty string which means current template. The current template can be created by using method AddBookmarkItem.</p> <p>Default value: empty string</p>

HeadersFooters

These properties define headers and footers to be placed on all pages of the PDF document. Headers and footers contain three parts: left, center and right.

Each part specifies the texts color and font that should be inserted into the document.

- Text placed in the left part is left aligned and placed at the **left of the page** (position for header's left part are defined by `OffsetTop` and `OffsetLeft` properties).
- Text of center part is center aligned on the page.
- Text of the right part is right aligned.

Any of the text fields may contain macros, which will be replaced with their values during printing of the PDF.

Name	Possible values	Description
Enabled	Boolean value	Default value: <code>False</code>
FirstPageNumber	Integer	Specifies the first page number for the headers and footers. The value of this property is used in the <code>%[Page]</code> macro. Default value: <code>1</code>
OffsetLeft	Integer	Values of these properties define the positions on the page where the headers and footers should be placed. All values are in tenths of a millimetre. Default value: <code>127</code>
OffsetTop		
OffsetRight		
OffsetBottom		

Header and footer are defined with two subsections, **Header** and **Footer** which are identical and contain three identical subsections **Left**, **Center**, and **Right**. Each part has properties:

Name	Possible values	Description
Font	String	The Font used for text which will be added into the document. Default value: <code>Arial</code> .
FontSize	10 - 32768	Specifies font size for the text. Font size is specified in tenth of point (1 point is 1/72 inch). Default value: <code>100</code>
FontBold	Boolean value	Specifies if the font should be bold. Default value: <code>False</code>

FontItalic	Boolean value	Specifies if italic variant of the font should be used. Default value: False
Color	colorREF	Defines the color of the text. Default value: 0 (black color)
Value	String	Defines the text which should be placed into the part. Text can contains macros. Default value: empty string.

For example, to place a red label with the current page number at the right-top corner of each page, using Verdana Bold font with 12 pt size, the following properties should be set:

C++

```

1 printer.Option["HeadersFooters.Enabled"] = TRUE;
2 printer.Option["HeadersFooters.Header.Right.Font"] = L"Verdana";
3 printer.Option["HeadersFooters.Header.Right.FontBold"] = TRUE;
4 printer.Option["HeadersFooters.Header.Right.FontSize"] = 120;
5 printer.Option["HeadersFooters.Header.Right.color"] = RGB(255, 0, 0);
6 printer.Option["HeadersFooters.Header.Right.Value"] = L"Page %[Page]";

```

VB

```

1 With PDFPrinter
2 .Option("HeadersFooters.Enabled") = TRUE;
3 .Option("HeadersFooters.Header.Right.Font") = "Verdana";
4 .Option("HeadersFooters.Header.Right.FontBold") = "Yes";
5 .Option("HeadersFooters.Header.Right.FontSize") = 120;
6 .Option("HeadersFooters.Header.Right.color") = 255;
7 .Option("HeadersFooters.Header.Right.Value") = "Page %[Page]";
8 End With

```

DocInfo (Document Information)

The properties of this section allow PDF document information fields to be defined and allow adding of XMP Metadata within the PDF document.

Name	Possible values	Description
Enabled	Boolean value	If this property is <code>False</code> , no information fields will be specified within the PDF document except the Producer field, which is always specified and contains the PDF-XChange version information for the creation of this PDF file. Default value: <code>False</code>
AskForDocInfo	Boolean value	If this option is <code>True</code> , a popup window where Document information fields can be entered will be shown after the document is spooled. Default value: <code>False</code>
AddXMP	Boolean value	If this option is <code>True</code> , XMP metadata will be added within the PDF document. This metadata will contain values of information fields, date of document creation etc. Please note, when PDF/A mode used, XMP metadata will be added into the document irregardless of the value of this property and Enabled property. Default value: <code>False</code>
Title	String	Specifies the Title information field of the document. Default value: empty string
Subject	String	Specifies the Subject information field of the document. Default value: empty string
Author	String	Specifies the Author information field of the document. Default value: empty string
Keywords	String	Specifies the Keywords information field of the document. Keywords should be comma separated. Default value: empty string
Creator	String	Specifies the Creator information field of the document. Default value: empty string

Save

Using properties of this section it is possible to define the folder where PDF document should be stored, the file name of the document and other document saving options.

Name	Possible values	Description
SaveType	0 (pdfSaver) 1 (AppendToLast) 2 (Save) 3 (JustEmail)	<p>Specifies the action which should be performed when document is spooled.</p> <ul style="list-style-type: none"> pdfSaver specifies that the pdfSaver application (the document creation application of PDF-XChange) will display the Job Management window where the action which should be performed can be specified. AppendToLast appends the newly printed document to the last job in the jobs list. pdfSaver may then be used to create a PDF file from this job. Save (most used) generates and saves the PDF file from the printed document using other properties defined into this section. JustEmail creates the PDF file and emails it using properties defined in the EMail section. <p>Default value: 2 (Save)</p>
ShowSaveDialog	Boolean value	<p>If this property is True, pdfSaver will display the Save As dialog where the path and file name of the PDF file must be specified.</p> <p>Canceling the Save As dialog will cancel the PDF file creation.</p> <p>Default value: True</p>
Path	String	<p>Defines the path where newly created PDF file should be saved. If this property is an empty string, the user's My Documents folder will be used for document storing.</p> <p>In some cases this property is not used. See the description of the File property.</p> <p>Default value: empty string</p>
File	String	<p>Defines file name of the newly created PDF file. This name can contain macros and a path. If this property contains a path, this path will be used instead of path specified in Path property.</p> <p>Default value: %[DocName]</p>

StripPath	Boolean value	<p>Defines how value of macro %[DocName] will be calculated.</p> <p>If this property is <code>True</code> and the document name passed from the printing application contains the path, this path will then be stripped during assigning %[DocName] property.</p> <p>Default value: <code>False</code></p>
WhenExists	<p>0 (ShowWarning)</p> <p>1 (Overwrite)</p> <p>2 (AutoNumber)</p> <p>3 (Append)</p> <p>4 (InsertBefore)</p>	<p>Specifies the action which will be performed during saving if the file with the name specified by Path and File properties already exists.</p> <p>When value is <code>ShowWarning</code>, pdfSaver will show a warning dialog where the user can choose what to do.</p> <p>Value <code>Overwrite</code> allows pdfSaver to overwrite existing file without any warning dialogs.</p> <p>If the value is <code>AutoNumber</code>, pdfSaver will change the name of the destination file by appending with a numeric value. The format of this number is specified by AutoNumber.Start and AutoNumber.NumDigits properties.</p> <p>Values <code>Append</code> and <code>InsertBefore</code> allow pdfSaver to merge existing and newly created PDF documents by placing the newly created PDF at the end (for value <code>Append</code>) or at the beginning (<code>InsertBefore</code>) of the existing file.</p> <p>Please note, these options are not available when PDF/A feature is used.</p> <p>Default value: 0 (ShowWarning)</p>
AutoNumber.Start	0 - 999999	<p>Define the start number when auto-numbering used.</p> <p>Default value: 1</p>
AutoNumber.NumDigits	1 - 6	<p>Define the format of the number when auto-numbering used. The number of digits defines the minimum number of digits used for the number (if number is shorted it will be prepended by '0' characters).</p> <p>Default value: 3</p>
RunApp	Boolean value	<p>Specifies if the newly created PDF document should be opened after creation.</p> <p>Default value: <code>True</code></p>

RunCustom	Boolean value	<p>Defines which application should be used to open created PDF document. If the value of this property is True, application specified by AppName will be used; otherwise the default PDF application will be used.</p> <p>This property used only when RunApp is True.</p> <p>Default value: False</p>
AppName	String	<p>This property is used only when both RunApp and RunCustom properties have the value True.</p> <p>It names the application which should be used for opening the created PDF file. The value of this property must be the fully-qualified path name of the application.</p> <p>Default value: empty string</p>
AppParams	String	<p>This property specifies the parameters that will be passed to the AppName application when opening the PDF file. String %f represents the full path name of the created PDF file.</p> <p>Default value: %f</p>

EMail

Email properties are used when the created PDF file should be emailed after creation and define the parameters for emailing.

The created PDF file will be sent as an attachment of this email.

Name	Possible values	Description
Enabled	Boolean value	<p>Specifies if the PDF document should be emailed after creation.</p> <p>If the value of this property is <code>False</code>, none of the other properties in this section are used.</p> <p>Default value: <code>False</code></p>
Method	0 (<code>LaunchClient</code>) 1 (<code>SendViaClient</code>) 2 (<code>SendViaSMTP</code>)	<p>Specifies how the email should be sent.</p> <p>Value <code>LaunchClient</code> launches the default email client with parameters filled in from the other properties, and the user will send the email manually after making any corrections or emendations as required.</p> <p>Value <code>SendViaClient</code> launches the default email client and sending the email automatically without intervention from the user.</p> <p>Value <code>SendViaSMTP</code> will use the SMTP server specified by the <code>Server</code> property to send the email.</p> <p>Default value: 0 (<code>LaunchClient</code>)</p>
From	String	Specifies the email account to use for the From email property.
To	String	<p>Specifies various recipient(s) of the email. If more than one recipient is specified, they should be separated by the semicolon (;) character.</p> <p>Each recipient can be specified as recipient name (for example, John Smith) and the email address will be resolved from the email client's address book; email address (for example, john@server.com); or as a combination of name and address: John Smith <john@server.com>.</p> <p>Default values: empty string</p>
Cc	String	
Bcc	String	
Subject	String	<p>Defines the subject string for the email.</p> <p>Default value: empty string</p>

Content	String	<p>Defines content of the email.</p> <p>Default value: empty string</p>
ZIP	Boolean value	<p>Specifies how the PDF file should be added as an attachment to the email.</p> <p>When <code>True</code>, the PDF file should be sent as a ZIP archive.</p> <p>When <code>False</code>, the PDF file should be sent as the original PDF file.</p> <p>Default value: <code>True</code> - send as ZIP archive</p>
Server	String	<p>Specifies the name of the SMTP server for sending email.</p> <p>This property is used only when the <code>Method</code> property has the value <code>SendViaSMTP</code>.</p> <p>The SMTP server must be defined using AddSMTPServer method.</p> <p>Default value: empty string</p>

Optimization

This section controls Optimizations of the generated PDF file.

Name	Possible values	Description
Blending	One of following integer value: 0 - Normal 1 - Multiply 2 - Screen 3 - Overlay 4 - Darken 5 - Lighten 6 - colorDodge 7 - colorBurn 8 - HardLight 9 - SoftLight 10 - Difference 11 - Exclusion	Specifies blending mode used for blending line art during printing from some application (for example from AutoCAD when lines overlaying is enabled in it). Default value: 4
DeTessellation	0 (None) 1 (Simple)	Specifies if detessellation should be used for a sequence of triangles. Detessellation is useful when printing from AutoCAD and allow to produce smaller and optimal PDF files. Default value: 1 (Simple)
UseTransparency	Boolean value	Enables transparency drawing into the PDF file. PDF-XChange can detect some drawing sequences and can then replace them with transparency drawing. Please note: when PDF/A mode is used, this property is forced to <code>False</code> value. Default value: <code>True</code>

<code>seMaskedImages</code>	Boolean value	Using masked images produces smaller PDF files for some drawing sequences. Default value: <code>True</code>
<code>GrayPaletteAsRGB</code>	Boolean value	If this option is enabled, the palettes of grayscaled images are stored as an RGB palette where each element of this palette has the same value for each of its RGB components. Default value: <code>False</code>
<code>ImagesOptimization</code>	Boolean value	Enables the optimization of image sequences. Such optimization requires many manipulations with images but causes the resulting PDF to be much smaller and better for viewing. Default value: <code>True</code>
<code>EnableSwapping</code>	Boolean value	Enables swapping images to the disk during image optimization. When swapping is turned off optimization time is smaller as a rule, but requires more memory. Default value: <code>True</code>
<code>SwapToDiskLimit</code>	1-8	Defines the image size (in megabytes) in memory starting from which this image will be swapped to disk (if option <code>EnableSwapping</code> has <code>True</code> value). Default value: 1

Saver

This section controls the **ShowProgress** property of the **pdfSaver** application.

Name	Possible values	Description
ShowProgress	Boolean value	<p>If this property is <code>True</code>, the progress dialog will be shown during the printing and PDF creation process.</p> <p>Value <code>False</code> disables this progress window.</p> <p>Default value: <code>True</code></p>

Note: Other properties may be added in the future.

Naming Macros

PDF-XChange supports macros for file naming and for headers/footers text. All macros have syntax:

```
%[Macro{ :extra }]
```

Where **Macro** is a name of the macro value, and **extra** defines extra specifier for the macro.

Items between **{ }** are optional. Not all macros use an extra specifier, and will ignore all text after **:** up to closing bracket **]**.

The following macros are supported into PDF-XChange:

Name	Possible values	Description
AppName	not defined	Specifies the executable module of printing application. It contains only a file name of executable, and doesn't contain the path of the this module. For example, when printing is from Microsoft Word, this macro will have the value Winword.exe .
DocName	not defined	Specifies the document name as it was passed to the PDF-XChange driver by printing application (via DOCINFO structure at printing start). If property Save.StripPath is <code>True</code> and document name, passed by application, contains a path, this path will be stripped before assigning to the macro.
DocPath	not defined	Specifies the path part (if it is present) of the document name passed by the printing application.

Date	Format string. See below for syntax of format string.	Has a value of date of document printing. Value can be formatted using format specified in extra part of the macro. When no extra is specified, default regional settings are used for formatting.
Year	not defined	Year number. Format of this number (two digits or four) depends from regional settings.
Month	not defined	Month number. Format depends from regional settings.
Day	not defined	Day of month (as digits). Format depends from regional settings.
Time	Format string. See below for syntax of format string.	Printing time as string. Value can be formatted using format specified in extra part of the macro. When no extra is specified, default regional settings are used for formatting.
Hour	not defined	Hour of printing. Format depends from regional settings.
Minute	not defined	Minute of printing. Format depends from regional settings.
Second	not defined	Seconds of printing. Format depends from regional settings.
Computer	not defined	The name of user's computer.
User	not defined	The name (login name) of user who makes printing.
Page	not defined	Current page number. Can be used only for headers and footers. Starting page number is defined by HeadersFooters.FirstPageNumber property.
Pages	not defined	Total number of pages into the document.
Env	Environment variable name	Has a value of environment variable specified by extra string. If there are no such environment variable, or extra string isn't specified, value of this macro will be an empty string. For example, %[Env:UserName] will have the login name of the user who prints.

Format of picture string used to form the date string. The following elements can be used to construct a format picture string. If there are spaces to separate the elements in the format string, these spaces will appear in the same location in the result string. The letters must be in uppercase or lowercase as shown in the table, for example, "MM" not "mm". Characters in the format string that are enclosed in single quotation marks appear in the same location and unchanged in the output string.

Picture	Meaning
d	Day of month as digits with no leading zero for single-digit days.
dd	Day of month as digits with leading zero for single-digit days.
ddd	Day of week as a three-letter abbreviation. Depends of regional settings.
dddd	Day of week as its full name. Depends of regional settings.
M	Month as digits with no leading zero for single-digit months.
MM	Month as digits with leading zero for single-digit months.
MMM	Month as three-letter abbreviation. Depends of regional settings.
MMMM	Month as its full name. Depends of regional settings.
y	Year as last two digits, but with no leading zero for years less than 10.
yy	Year as last two digits, with leading zero for years less than 10.
yyyy	Year represented by full four or five digits, depending of the calendar used. Thai Buddhist and Korean calendars both have five digit year. The "yyyy" pattern will show five digits for these two calendars, and four digits for all other supported calendars.
yyyyy	Same as "yyyy".
gg	Period/era string. This element is ignored if the date does not have an associated era or period string. Depends of regional settings.

Format of picture string used to form the time string. The following elements can be used to construct a format picture string.

Picture	Meaning
h	Hours with no leading zero for single-digit hours; 12-hour clock.
hh	Hours with leading zero for single-digit hours; 12-hour clock.
H	Hours with no leading zero for single-digit hours; 24-hour clock.
HH	Hours with leading zero for single-digit hours; 24-hour clock.
m	Minutes with no leading zero for single-digit minutes.
mm	Minutes with leading zero for single-digit minutes.
s	Seconds with no leading zero for single-digit seconds.
ss	Seconds with leading zero for single-digit seconds.
t	One character time marker string, such as A or P. Depends of regional settings.
tt	Multi-character time marker string, such as AM or PM. Depends of regional settings.

Methods

Methods are used to perform ActiveX control actions that simple property assignments cannot accomplish.

- **Method SetRegInfo** is used to set registration information specific for developer for the printer.
- **Method ResetDefaults** resets all properties to their default values.
- **Method ApplyOptions** is used to apply all properties changed with property **Option**.
- **Method AddBookmarkItem** adds a new item into bookmarks template tree.
- **Method AddTextWatermark** is used to define a new text watermark which may be added to the document.
- **Method AddImageWatermark** is used to define a new image watermark which may be added to the document.
- **Method StorePrinterOptions** is used to store all printer options into any external storage which supports an IStream interface.
- **Method ReStorePrinterOptions** restores all printer options previously stored by method **StorePrinterOptions**.
- **Method SetAsDefaultPrinter** sets the printer associated with the **IPXCPrinter** object as the system default printer.
- **Method RestoreDefaultPrinter** restores the system default printer which was stored during a call to the **SetAsDefaultPrinter** method.
- **Method AddSMTPServer** adds a definition of an SMTP server, which can be used for sending PDF files via email.

SetRegInfo

Method SetRegInfo

This method is used to set registration information specific for developer for the printer. This method may be used when registration information was not passed into the **Printer** property of **IPXCControlEx** object during printer creation.

Declaration

C#

```
void setreginfo(string pRegKey, string pDevCode);
```

C++

```
HRESULT setreginfo([in] BSTR pRegKey, [in] pDevCode);
```

VB

```
Sub setreginfo(pRegKey As String, pDevCode As String)
```

ResetDefaults

Method ResetDefaults

This method resets all properties to their default values, as defined in the **Properties** sections above.

Declaration

C#

```
void ResetDefaults();
```

C++

```
HRESULT ResetDefaults();
```

VB

```
Sub ResetDefaults()
```

ApplyOptions

Method ApplyOptions

This method is used to apply all properties changed with property **Option**.

This method should be called before printing onto the printer associated with the **IPXCPrinter** object.

Declaration

C#

```
void ApplyOptions(int dwFlags);
```

C++

```
HRESULT ApplyOptions([in] DWORD dwFlags);
```

VB

```
Sub ApplyOptions(dwFlags As Long)
```

AddBookmarkItem

Method AddBookmarkItem

Method **AddBookmarkItem** adds a new item into the bookmarks template tree. Returned value from this method can be used for subsequent calls of this method.

Declaration

C#

```
int AddBookmarkItem(int nParent,  
    int dwDetectFlags,  
    string sDet_FontName,  
    int nDet_FontStyle,  
    int nDet_Size,  
    int nDet_SizeDelta,  
    int nDet_TextColor,  
    int dwDisplayFlags,  
    int dwDisplayColor);
```

C++

```

HRESULT AddBookmarkItem([in] long nParent,
    [in] long dwDetectFlags,
    [in] BSTR sDet_FontName,
    [in] long nDet_FontStyle,
    [in] long nDet_Size,
    [in] long nDet_SizeDelta,
    [in] long nDet_Textcolor,
    [in] long dwDisplayFlags,
    [in] long dwDisplaycolor,
    [out, retval] long* pID);

```

VB

```

Function AddBookmarkItem(nParent As Long,
    dwDetectFlags As Long,
    sDet_FontName As String,
    nDet_FontStyle As Long,
    nDet_Size As Long,
    nDet_SizeDelta As Long,
    nDet_Textcolor As Long,
    dwDisplayFlags As Long,
    dwDisplaycolor As Long) As Long

```

Parameters**nParent**

Specifies the parent ID of the item to be created. This could be a value returned from a previous call to this function, or -1, if the item should be a top level item.

dwDetectFlags

A bit combination integer value which specifies which properties of the body text are to be used to match it as a bookmark item. A combination of the following values can be used:

Bit	Value	Meaning
0	1	If this bit is set, the font name specified is used by the sDet_FontName parameter to determine all matching text/strings. sDet_FontName may not be NULL or an empty string when this flag is set.
1	2	If this bit is set, the font style specified is used by the nDet_FontStyle parameter to determine all matching text/strings.
2	4	If this bit is set, the font size specified is used by the nDet_Size and nDet_SizeDelta parameters to determine all matching text/strings Text with the font size $nDet_Size \pm nDet_SizeDelta$ matches and is therefore added as a bookmark item.
3	8	If this bit is set, the text color specified by nDet_Textcolor is also used as part of the process in detecting potential matches.

The value of this parameter cannot be **zero (0)**; at least one matching parameter **must** be used.

sDet_FontName

Specifies the font name for matching.

nDet_FontStyle

Specifies the font style for matching as an integer value which may have any one of the following values:

Value	Meaning
0	Normal Font
1	Bold Font
2	Italic Font
3	Bold Italic Font

nDet_Size

Specifies the font size to be used for matching. The font size is specified in tenths of a point (one point = 1/72 inches). This unit is 1/720 of an inch.

nDet_SizeDelta

Specifies the font size delta to be used for matching. The delta value is specified in tenths of a point (one point = 1/72 inches). This unit is 1/720 of an inch.

nDet_Textcolor

Specifies text color to be used for matching.

dwDisplayFlags

A bit combination integer value which defines how the bookmark item will be shown in the bookmark tree of the PDF viewing application. May be a combination of the following flags:

Bit	Value	Meaning
0	1	When this bit is set, the bookmark item will be drawn using a bold font.
1	2	When this bit is set, the bookmark item will be drawn using an italic font.
2	4	When this bit is set, the bookmark item will be initially opened in the bookmark tree window of the viewing application. This flag is used only when the property Bookmarks.DisplayMode has a value ByItem.

dwDisplaycolor

Specifies the color bookmark items will be displayed in, if matching this template, in the viewer's bookmark tree window. The RGB Macro defined in the Windows API may be used to specify the color value.

Return Value

This function returns an internal ID for the newly created bookmark item. This value may be used as a **nParent** parameter for later calls to the **AddBookmarkItem** method.

Example

The following example creates simple bookmarks detection tree, which makes text drawn using Times New Roman, 12 pt, bold font as top-level items. And items drawn by the same font, but not bold - as second level items. Also items drawn using Arial, bold, 14 pt, will be added as top level item and will be displayed using red color.

VB

```

1 Dim parent as Long
2 parent = PXCPrinter.AddBookmarkItem(-1, 7, "Times New Roman", 1, 120, 5, 0,
0, 0)
3 PXCPrinter.AddBookmarkItem(parent, 7, "Times New Roman", 0, 120, 5, 0, 0, 0)
4 PXCPrinter.AddBookmarkItem(-1, 7, "Arial", 1, 140, 5, 0, 0, 255)

```

AddTextWatermark

Method AddTextWatermark

This method is used to define a new text watermark to be added to the document. This methods parameters correspond to the parameters available in the **PDF-XChange Driver 2012** GUI during the adding/editing of a Text Watermark (topic Watermarks; Add... button).

Declaration

C#

```

void AddTextWatermark(
    string sName,
    string sText,
    string sFontName,
    int dwFontWeight,
    int blTalic,
    int bOutline,
    int nFontSize,
    int nLineWidth,
    int nTextColor,
    int dwAlign,
    int xOffset,
    int yOffset,
    int nAngle,
    int dwOpacity,
    int dwFlags,
    int dwPlaceType,
    string sRange);

```

C++

```
HRESULT AddTextWatermark(  
    [in] BSTR sName,  
    [in] BSTR sText,  
    [in] BSTR sFontName,  
    [in] long dwFontWeight,  
    [in] long bltalic,  
    [in] long bOutline,  
    [in] long nFontSize,  
    [in] long nLineWidth,  
    [in] long nTextColor,  
    [in] long dwAlign,  
    [in] long xOffset,  
    [in] long yOffset,  
    [in] long nAngle,  
    [in] long dwOpacity,  
    [in] long dwFlags,  
    [in] long dwPlaceType,  
    [in] BSTR sRange);
```

VB

```
Sub AddTextWatermark(  
    sName As String,  
    sText As String,  
    sFontName As String,  
    dwFontWeight As Long,  
    bltalic As Long,  
    bOutline As Long,  
    nFontSize As Long,  
    nLineWidth As Long,  
    nTextColor As Long,  
    dwAlign As Long,  
    xOffset As Long,  
    yOffset As Long,  
    nAngle As Long,  
    dwOpacity As Long,  
    dwFlags As Long,  
    dwPlaceType As Long,  
    sRange As String)
```

Parameters

sName

Specifies the name of the newly created watermark item. This name is used to identify this watermark in the **Watermarks** property string. If there is already a watermark with the same name - it will be overwritten!

sText

Specifies the watermark text.

sFontName

Specifies the font (by name) to use when drawing the watermark. **N.B. Required!**

dwFontWeight

Specifies the weight of the font. Possible values are: 400 for Normal and 700 for Bold.

bItalic

When True, specifies the italic variant of the font identified by the sFontName parameter.

bOutline

When True, the text will be drawn as an outline; when False, as solid filled text.

nFontSize

Specifies the font size of the text. Unlike V4 of the Drivers API, in V5 the font size is specified in tenth points (1 pt = 1/72 inch). I.e., to specify font size of **12pt** the value of **120** should be passed in nFontSize.

When this parameter is 0 (zero), the font size will be adjusted to fit the entire page.

nLineWidth

Specifies the line width for outlining the contour of the text. Line width is specified in tenths of points. Used only when bOutline is TRUE.

nTextColor

Specifies the text color of the watermark.

dwAlign

Specifies the text alignment on the page. When nFontSize is 0 (zero) this parameter is ignored and the watermark will be centered vertically and horizontally (value 9).

dwAlign may be a combination any of the following values:

Bit	Value	Meaning
0	1	Horizontal Center Align.
1	2	Right Align: Cannot be set when bit 0 (Horizontal center align is set) When both bits 0 and 1 are not set, Left align used.
4	16	Vertical Center align.
5	32	Bottom Align. Cannot bit set when bit 4 is set. When both bits 4 and 5 are not set, Top align used.

xOffset**yOffset**

Specify Horizontal and Vertical offsets for the watermark relative to the top-left corner of the page. Values are specified in tenths of millimeters.

nAngle

Specifies the Rotation angle for the watermark. Possible values are from -90 to 90 degrees. Positive values mean counterclockwise rotation.

dwOpacity

Specifies the Opacity level of the watermark. Value must be in the range of 0 (fully transparent) to 100 (fully opaque). If the value is outside of the range it will be clipped.

dwFlags

Bit combination integer value which defines how the watermark should be placed and when it should be visible, may be a combination of the following flags:

Bit	Value	Meaning
0	1	Not used.
1	2	If this bit is set the watermark will be placed as a background on the page. Otherwise it will be placed in the foreground - over the content of the existing page.
2	4	Watermark should be visible during viewing the PDF document in the PDF viewing application.
3	8	Watermark should be visible when the PDF document is exported (for example as an image format) from the viewing application.
4	16	Watermark should be printed when the PDF document is printed from the PDF viewing application.

dwPlaceType

Defines on which pages the watermark should be placed. May be any of the following values:

Value	Meaning
0	Watermark will be placed on all pages of the document
1	Watermark will be placed only on the first page of the document.
2	Watermark will be placed only on the last page of the document.
3	Watermark will be placed only on even pages of the document.
4	Watermark will be placed only on odd pages of the document.
5	Watermark will be placed on pages specified by the sRange parameter.

sRange

A String defining the page range(s) onto which the watermark should be placed. Each range can be: page number, or range of pages in the form [p1]-[p2], which means from p1 to p2 inclusive. One of p1 or p2 may be omitted. In this case all pages prior to p2 or all pages after p1 will be included into the range. Ranges should be separated by a comma(,) or semicolon (;).

Example of range string: "-3, 5, 7-11, 15-". This decodes as "pages 1, 2, 3, 5, 7, 8, 9, 10, 11, 15 to the end of the document."

AddImageWatermark

Method AddImageWatermark

This method is used to define a new image watermark to be added to the document. The parameters of this method correspond to the parameters available in the **PDF-XChange Driver 2012** UI during the adding/editing of a text watermark (page Watermarks; Add... button).

Declaration

C#

```
void AddImageWatermark(  
    string sName,  
    string sImageFileName,  
    int dwTranscolor,  
    int dwWidth,  
    int dwHeight,  
    int dwAlign,  
    int xOffset,  
    int yOffset,  
    int nAngle,  
    int dwOpacity,  
    int dwFlags,  
    int dwPlaceType,  
    string sRange);
```

C++

```
HRESULT AddImageWatermark(  
    [in] BSTR sName,  
    [in] BSTR sImageFileName,  
    [in] long dwTranscolor,  
    [in] long dwWidth,  
    [in] long dwHeight,  
    [in] long dwAlign,  
    [in] long xOffset,  
    [in] long yOffset,  
    [in] long nAngle,  
    [in] long dwOpacity,  
    [in] long dwFlags,  
    [in] long dwPlaceType,  
    [in] BSTR sRange);
```

VB

```
Sub AddImageWatermark(
    sName As String,
    sImageFileName As String,
    dwTranscolor As Long,
    dwWidth As Long,
    dwHeight As Long,
    dwAlign As Long,
    xOffset As Long,
    yOffset As Long,
    nAngle As Long,
    dwOpacity As Long,
    dwFlags As Long,
    dwPlaceType As Long,
    sRange As String)
```

Parameters**sName**

Specifies the name of the newly created watermark item. This name may be used in the Watermarks property. If there is already an existing watermark with the same name, it will be overwritten.

sImageFileName

Specifies the fully qualified path name to the image file to be used as a watermark. The image may be in any of the following supported image formats: BMP, PNG, JNG, JPEG, JPEG 2000, TIFF, JBIG, JBIG2, GIF, PCX, DCX, EMF, WMF. If the image is a multipage image, only the first page will be used as a watermark.

dwTranscolor

Specifies the transparent color for the image. If this parameter is anything other than **-1**, each pixel in the sImageFileName image which has a color specified by the dwTranscolor parameter will be transparent in the PDF file generated.

dwWidth**dwHeight**

Specifies the dimensions of the watermark. The image specified by the sImageFileName parameter will be resized to 'fit' within the specified area and will retain its aspect ratio.

All other parameters of this method have the same meaning as the corresponding parameters of [AddTextWatermark](#) method, except of dwFlags parameter which may use two additional flags:

dwFlags

Bit combination integer value which defines how the watermark should be placed and when it should be visible, may be a combination of the following flags:

Bit	Value	Meaning
8	256	If specified, image will be fit the page. parameters dwWidth and dwHeight will be ignored.
9	512	If specified, image will not keep aspect ratio on scaling.

Return Value

The method returns S_OK if successful, or an error code if it fails.

StorePrinterOptions

Method StorePrinterOptions

This method is used to store all printer options into any external storage method supporting the **IStream** interface.

Declaration

C#

```
void StorePrinterOptions(IStream pStream);
```

C++

```
HRESULT StorePrinterOptions(IStream* pStream);
```

VB

```
Sub StorePrinterOptions(pStream As IStream)
```

Parameters

pStream

Pointer to the **IStream** object into which all printer properties will be serialized.

Return Value

The method returns S_OK if successful; otherwise an error code is returned.

ReStorePrinterOptions

Method ReStorePrinterOptions

This method restores all original printer options previously saved using the method [StorePrinterOptions](#) to save to an external storage method supporting the **IStream** interface.

Declaration

C#

```
void ReStorePrinterOptions(IStream pStream);
```

C++

```
HRESULT ReStorePrinterOptions(IStream* pStream);
```

VB

```
Sub ReStorePrinterOptions(pStream As IStream)
```

Return Value

The method returns S_OK if successful; otherwise an error code is returned.

SetAsDefaultPrinter

Method SetAsDefaultPrinter

This method sets the printer associated with the **IPXCPrinter** object as the system default printer. The original system default printer can be restored later using the method [RestoreDefaultPrinter](#).

The original system default printer will be restored automatically when releasing the **IPXCPrinter** object.

Declaration**C#**

```
void SetAsDefaultPrinter();
```

C++

```
HRESULT SetAsDefaultPrinter();
```

VB

```
Sub SetAsDefaultPrinter()
```

Return Value

The method returns S_OK if successful; otherwise an error code is returned.

RestoreDefaultPrinter

Method RestoreDefaultPrinter

This method restores the original system default printer modified when calling [SetAsDefaultPrinter](#) method. If the method **SetAsDefaultPrinter** was not previously called this method call will be ignored.

Declaration

C#

```
void RestoreDefaultPrinter();
```

C++

```
HRESULT RestoreDefaultPrinter();
```

VB

```
Sub RestoreDefaultPrinter()
```

Return Value

The method returns S_OK if successful; otherwise an error code is returned.

AddSMTPServer

Method AddSMTPServer

This method adds the parameters of a defined SMTP server, which may then be used for sending PDF files via email.

For more details see description of [EMail](#) section.

Declaration

C#

```
void AddSMTPServer(  
string sName,  
int nPort,  
int bNeedAuth,  
string sUser,  
string sPassword,  
long bSecureConnection);
```

C++

```
HRESULT AddSMTPServer(  
    [in] BSTR sName,  
    [in] long nPort,  
    [in] long bNeedAuth,  
    [in] BSTR sUser,  
    [in] BSTR sPassword,  
    [in] long bSecureConnection);
```

VB

```
Sub AddSMTPServer(  
    sName As String,  
    nPort As Long,  
    bNeedAuth As Long,  
    sUser As String,  
    sPassword As String,  
    bSecureConnection As Long)
```

Parameters**sName**

Specifies the name or IP address of the SMTP server.

nPort

Specifies the port number to be used to communicate with the SMTP server. The usual SMTP default is port **25**.

bNeedAuth

Set to True if the specified server requires authorization for sending email.

sUser

User name used for authorization on the server. Not used when bNeedAuth is False.

sPassword

Password used for access authorization to the server. Not used when bNeedAuth is False.

bSecureConnection

Set to True if the server requires a secure connection or False otherwise.

Return Value

Method returns S_OK if success, and other values on fail.

Events

The **IPXCPrinter** interface provides events which can be used by a Developer's applications to determine the stage of the PDF creation process. Some events are fired during document spooling and these events are asynchronous to prevent dead-locking. Events which are fired during the generation of PDF documents are synchronous.

An **asynchronous event** means that PDF-XChange fires the event and will not wait while application receives this event and returns from it. It also means that the application will receive these events immediately when they are fired.

When a **synchronous event** is fired, PDF-XChange will wait until the application has finished this event.

All events have a **JobID** parameter which specifies an internal Job identifier of the printed document. This ID can be used when multiple documents are printing to determine from which document the event was fired.

The following events are supported in **PDF-XChange Driver 2012**.

Asynchronous Events:

- **Event OnStartDoc** is fired when a new document starts to print.
- **Event OnStartPage** is fired for each page when it starts to print.
- **Event OnEndPage** is fired for each page after it prints.
- **Event OnEndDoc** is fired when the complete document is spooled by the GDI.

Synchronous Events:

- **Event OnDocSpooled** is fired immediately when PDF-XChange starts processing the document that was printed by the GDI.
- **Event OnFileSaved** is fired when the PDF file is saved.
- **Event OnFileSent** is fired when emailing of the PDF file was enabled and after the email was sent. If emailing is not enabled this event will not fire.
- **Event OnError** is fired when an error occurs during the processing of the PDF file.

Asynchronous

Asynchronous Events

The following Asynchronous events are supported in **PDF-XChange Driver 2012**.

- **Event OnStartDoc** is fired when a new document starts to print.
- **Event OnStartPage** is fired for each page when it starts to print.
- **Event OnEndPage** is fired for each page after it prints.
- **Event OnEndDoc** is fired when the complete document is spooled by the GDI.

Event OnStartDoc

This event is fired when a new document starts to print.

Declaration

C#

```
event OnStartDoc(int JobID, string lpszDocName, string lpszAppName);
```

C++

```
HRESULT OnStartDoc(LONG JobID, BSTR lpszDocName, BSTR lpszAppName);
```

VB

```
Event OnStartDoc(JobID As Long, lpszDocName As String, lpszAppName As String)
```

Parameters

JobID

Internal Job identifier.

lpszDocName

Specifies the document name as it was passed from the printing application.

lpszAppName

Specifies the module name of the printing application.

Return Value

PDF-XChange ignores returned values from this event.

Event OnStartPage

This event is fired for each page when it starts to print.

Declaration

C#

```
event OnStartPage(int JobID, int nPageNumber);
```

C++

```
HRESULT OnStartPage(LONG JobID, LONG nPageNumber);
```

VB

```
Event OnStartPage(JobID As Long, nPageNumber As Long)
```

Parameters

JobID

Internal Job identifier.

nPageNumber

Zero based page number when spooling.

Return Value

PDF-XChange ignores returned values from this event.

Event OnEndPage

This event is fired for each page after printing completes.

Declaration

C#

```
event OnEndPage(int JobID, int nPageNumber);
```

C++

```
HRESULT OnEndPage(LONG JobID, LONG nPageNumber);
```

VB

```
Event OnEndPage(JobID As Long, nPageNumber As Long)
```

Parameters

JobID

Internal Job identifier.

nPageNumber

Zero based page number spooled.

Return Value

PDF-XChange ignores returned values from this event.

Event OnEndDoc

This event is fired when the complete document has been spooled by the GDI.

Declaration

C#

```
event OnEndDoc(int JobID, int bOK);
```

C++

```
HRESULT OnEndDoc(LONG JobID, LONG bOK);
```

VB

```
Event OnEndDoc(JobID As Long, bOK As Long)
```

Parameters

JobID

Internal Job identifier.

bOK

When this parameter is True, the document has fully spooled and PDF-XChange will continue to process this document to produce the PDF file. If there was an error during printing (e.g. printing cancelled) there will be no further document processing.

Return Value

PDF-XChange ignores returned values from this event.

Synchronous

Synchronous Events

The following Synchronous events are supported by the PDF-XChange 2012 Driver.

- **Event OnDocSpooled** is fired immediately PDF-XChange starts processing the document printed by the GDI.
- **Event OnFileSaved** is fired when the PDF file is saved.
- **Event OnFileSent** is fired when PDF emailing is enabled and after the email has been sent. If emailing is not enabled this event will not fire.
- **Event OnError** is fired if an error occurs during the processing of the PDF file.

Event OnDocSpooled

This event is fired immediately PDF-XChange begins processing the document printed by the GDI.

Declaration

C#

```
event OnDocSpooled(int JobID, string lpszDocName, string lpszAppName);
```

C++

```
HRESULT OnDocSpooled(LONG JobID, BSTR lpszDocName, BSTR lpszAppName);
```

VB

```
Event OnDocSpooled(JobID As Long, lpszDocName As String, lpszAppName As String)
```

Parameters

These event parameters are identical to the parameters for the OnStartDoc event.

During handling for this event the application can set document specific properties (for example, the path where the PDF document should be stored as defined by the printing application).

The application should return S_OK from this event handler.

Event OnFileSaved

This event is fired when the PDF file is saved. This allows the application to perform any application-based processing to this file.

Declaration

C#

```
event OnFileSaved(int JobID, string IpszFileName);
```

C++

```
HRESULT OnFileSaved(LONG JobID, BSTR IpszFileName);
```

VB

```
Event OnFileSaved(JobID As Long, IpszFileName As String)
```

Parameters

JobID

Internal Job identifier.

IpszFileName

Specifies the fully qualified path name for where the PDF document has been saved.

Return Value

PDF-XChange ignores returned values from this event.

Event OnFileSent

This event is fired when emailing of a PDF file is enabled and after the email has been sent. If emailing is not enabled, this event will not be fired.

Declaration

C#

```
event OnFileSent(int JobID, string IpszFileName);
```

C++

```
HRESULT OnFileSent(LONG JobID, BSTR IpszFileName);
```

VB

```
Event OnFileSent(JobID As Long, IpszFileName As String)
```

Parameters

Parameters for this event are the same as the parameters for the OnFileSaved event. PDF-XChange ignores returned values from this event.

Event OnError

This event is fired when an error occurs during the processing of the PDF file. It further indicates that the events [OnFileSaved](#) and/or [OnFileSent](#) will not be fired for this document and processing of the document is terminated.

Declaration

C#

```
event OnError(int JobID, int dwErrorCode);
```

C++

```
HRESULT OnError(LONG JobID, LONG dwErrorCode);
```

VB

```
Event OnError(JobID As Long, dwErrorCode As Long)
```

Parameters

JobID

Internal Job identifier.

dwErrorCode

Specifies an error (code) occurred during document processing.

Return Value

PDF-XChange ignores the returned values from this event.

Property Declarations

Unsupported Property Declarations

All these methods will return an E_NOTIMPL error code:

- Event **LastPrinterEvent**
- Method **PrintDocument**
- Method **WaitForPrintEvent**

Index

B

Bookmarks 38

C

Compression 23

D

Digital Signatures 31

Document Information 42

E

E-Mail 46

Event OnDocSpooled 73

Event OnEndDoc 69

Event OnEndPage 69

Event OnError 73

Event OnFileSaved 73

Event OnFileSent 73

Event OnStartDoc 69

Event OnStartPage 69

Events 68

Events: Asynchronous 69

Events: Synchronous 73

F

Fonts 29

G

General 21

Graphics 25

H

HeadersFooters 40

I

Installation 5

Interface IPXCPrinter 11

IPXCControlEx Object 9

L

Links 37

M

Method AddBookmarkItem 55

Method AddImageWatermark 61

Method AddSMTPServer 66

Method AddTextWatermark 58

Method ApplyOptions 55

Method ResetDefaults 54

Method RestoreDefaultPrinter 66

Method ReStorePrinterOptions 64

Method SetAsDefaultPrinter 65

Method SetRegInfo 54

Method StorePrinterOptions 64

Methods 53

N

Naming Macros 50

O

Optimization 48

Options Bookmarks 38

Options Compression 23

Options Digital Signatures 31

Options DocInfo 42

Options E-Mail 46

Options Fonts 29

Options General 21

Options Graphics 25

Options HeadersFooters 40

Options Links 37

Options Optimization 48

Options Overlay 35

Options Paper 16

Options Save 42

Options Saver 50

Options Security 30

Options Watermarks 38

Overlay 35

P

Paper 16

Properties 11

Property Declarations 77

Property Name 13

Property Option 14

Property Table Definition 15

S

Save 42

Saver 50

Security 30

Silent Install Switches 5

U

Using Drivers API SDK 2

W

Watermarks 38

Welcome 1