API Reference for Application Communication

Fast, Affordable & Secure PDF Solutions

Foxit Software Inc.

Contents

Preface

The Foxit® PhantomPDF® Software Development Kit (SDK) provides a set of API calls for creating PDF from other document types and controlling PhantomPDF to do something you are interested in automatically without manual intervention.

What's in this guide?

This document provides a detailed reference of all the APIs that are used to communicate with PhantomPDF.

Who should read this guide?

This guide is for developers that want to communicate with PhantomPDF from another application, or who are writing tools that need to do something about PDF automatically.

You can use all the APIs easily if you are familiar with VBA or C++.

Related documentation

For information about	See
FoxitPhantomPDF_Manual.pdf	*FoxitPhantomPDF83_Manual.pdf*

OLE Automation

This chapter describes the objects, data types, and methods in the OLE automation interface.

The name PhantomPDF.Application is the external strings OLE clients use to create object of application type. For convenience, the PhantomPDF developer type libraries also call it PhantomPDF.Application.

For all the other automation objects (for example, Document and Creator), you can get them through PhantomPDF.Application.

The following table summarizes the available objects and data types.

Object	**Description**
PhantomPDF.Application	The application itself.
PhantomPDF.Creator	Represents a creator object that is used to convert non-pdf to PDF
PhantomPDF.Document	Represents a document that is currently opened in PhantomPDF.
PhantomPDF.Page	Represents a page in a document.
PhantomPDF.WatermarkElementInfo	Represents a watermark structure.
PhantomPDF.CombineFlags	An enum use for Creator.CombineFiles's parameter uCombineFlags.

PhantomPDF.Application

Represents the Foxit application. The Application object includes properties and methods that return top-level objects. For example, the CurrentDocument property returns a Document object.

Methods

The Application object has the following methods.

Method	Description
CloseDocument	Close a Document.
CreateBlankDoc	Create a blank Document.
Exit	Exit PhantomPDF.
GetDocument	Get a Document object by index.
IsValidDocument	Check whether a document object is valid.
OpenDiskFileAsMenPDF	Open a PDF document as a memory document in PhantomPDF.
OpenDocument	Open a PDF document in PhantomPDF.
CreateWatermarkElementInfo	Create a WatermarkElementInfo object.

Properties

The Application object has the following Properties.

Property	Description
CountDocuments	Count of documents opened in PhantomPDF, read only.
Creator	Creator object that is used to convert non-pdf to PDF, read only.
CurrentDocument	The current document object, read/write.
LastError	The Last error code, read only.

CloseDocument

Close a Document.

Syntax

HRESULT CloseDocument(IDocument* doc, VARIANT_BOOL bPromptToSave);

Parameters

doc	The document to be closed.
bPromptToSave	Prompt user to save if the document is modified.

Return value

S_OK if successful, S_FALSE if not.

See Also

CreateBlankDoc

OpenDiskFileAsMenPDF

GetDocument

OpenDocument

CreateBlankDoc

Create a blank document.

Syntax

IDocument* CreateBlankDoc(FLOAT fWidth, FLOAT fHeight);

Parameters

fWidth	Width, the unit is inches. (0.14~200).
fHeight	Height, the unit is inches. (0.14~200).

Return value

Pointer of Document if successful, NULL if failed.

See Also

CloseDocument

OpenDiskFileAsMenPDF

GetDocument

OpenDocument

Exit

Exit PhantomPDF. You should close all the documents before exiting PhantomPDF. The function will return a value immediately after sending the exit command to PhantomPDF.

Syntax

HRESULT Exit();

Return value

S_OKif exit command is sent to PhantomPDF, S_FAILif not.

GetDocument

Get a Document object by page index.

Syntax

IDocument* GetDocument(SHORT index);

Parameters

index	Page index, beginning with 0.

Return value

Pointer of Document if successful, NULL if failed.

See Also

CloseDocument

OpenDiskFileAsMenPDF

CreateBlankDoc

OpenDocument

IsValidDocument

Check whether a Document object is valid.

Syntax

VARIANT_BOOL IsValidDocument(IDocument* doc);

Parameters

doc	A Document object.

Return value

-1 if the Document object is valid, 0 if not.

See Also

CloseDocument

OpenDiskFileAsMenPDF

GetDocument

OpenDocument

OpenDiskFileAsMemPDF

Open a PDF document as a memory document in PhantomPDF.

Syntax

IDocument* OpenDiskFileAsMemPDF(BSTR PDFFilePath, BSTR Title);

Parameters

PDFFilePath	The PDF document Path.
Title	The memory document's title.

Return value

Pointer of Document if successful, NULL if failed.

See Also

CloseDocument

CreateBlankDoc

GetDocument

OpenDocument

OpenDocument

Open a PDF document in PhantomPDF.

Syntax

IDocument* OpenDocument(BSTR PDFFilePath, BSTR Password, VARIANT_BOOL bMakeVisible, VARIANT_BOOL bAddToMRU);

Parameters

PDFFilePath	The PDF document Path.
Password	If the document does not have a password, you should set it to an empty string.
bMakeVisible	Whether to show the document in PhantomPDF.
bAddToMRU	Whether to add the document to most recent documents list.

Return value

Pointer of Document if successful, NULL if failed.

See Also

CloseDocument

OpenDiskFileAsMenPDF

GetDocument

CreateBlankDoc

CreateWatermarkElementInfo

Create a WatermarkElementInfo object.

Syntax

IWatermarkElementInfo* CreateWatermarkElementInfo();

Return value

A IWatermarkElementInfo object.

See Also

Document.AddWatermark

PhantomPDF.Creator

The Creator object is used to convert non-pdf to PDF.

Methods

The Creator object has the following methods.

Method	Description
ConvertToPDF	Convert non-pdf to PDF.
CombineFiles	Combine multiple documents to a PDF.

ConvertToPDF

Convert non-pdf to PDF.

Syntax

HRESULT ConvertToPDF(BSTR bstrSrcPathName, BSTR bstrDestPathName);

Parameters

bstrSrcPathName	The source document path.
bstrDestPathName	The dest pdf document path.

Return value

S_OKif successful, S_FAILif not.

See Also

CombineFiles

CombineFiles

Combine multiple documents to a PDF.

Syntax

SHORT CombineFiles(BSTR bstrFiles, BSTR DestPDFFile, SHORT uCombineFlags);

Parameters

bstrFiles	The source files.
DestPDFFile	The dest pdf document path.
uCombineFlags	The flags, Enum CombineFlags

Return value

Number of combined files, less than zero if there are any errors.

Remarks

The parameter bstrFiles can be either a folder path or a string that is made from different file paths joining together and between each file path separated by '|'.

The flag COMBINE_TRAVEL_SUBDIR of uCombineFlags can be ignored if bstrFiles is not a folder path.

See Also

ConvertToPDF

PhantomPDF.Document

Represent a document that is currently opened in PhantomPDF.

Methods

The Document object has the following methods.

Method	Description
Export	Export the document to disk.
Save	Save the document.
RotatePage	Rotate specified pages.
GetFieldValue	Get the value of a specified form.
SetFieldValue	Set the value of a specified form.
AddWatermark	Add a watermark.
OCRAndExportToExcel	OCR and export the document to an Excel document.

Export

Export the document to disk.

Syntax

HRESULT Export(BSTR SaveAsPath);

Parameters

SaveAsPath	The dest pdf path.

Return value

S_OKif successful, S_FAILif not.

See Also

Save

Save

Save the document.

Syntax

```
HRESULT Save();
```

Return value

S_OK if successful, S_FAIL if not.

See Also

Export

RotatePage

Rotate specified pages.

Syntax

```
HRESULT RotatePage(SHORT nPage, SHORT nRotate);
```

Parameters

nPage	The pages index that begins with 0.
nRotate	nRotate , 0 for 0°, 1 for 90°, 2 for 180°, -1 for -90°

Return value

S_OKif successful, S_FAILif not.

Remarks

The parameter nRotate is only 0, 1, 2 or -1, and any other value will result in failure.

GetFieldValue

Get the value of a specified form.

Syntax

```
BSTR GetFieldValue(BSTR FieldName);
```

Parameters

FieldName	The form name.

Return value

Value of specified form.

Remarks

At present for the form types supported, see the following:

- Text Field
- Push Button
- List Box
- Combo Box
- Radio Button
- Check Box

See Also

SetFieldValue

SetFieldValue

Set the value of a specified form.

Syntax

```
HRESULT SetFieldValue(BSTR FieldName, BSTR value);
```

Parameters

FieldName	The form name.
value	Value to set.

Return value

S_OKif successful, S_FAILif not.

Remarks

At present for the form types supported, see the following:

- Text Field
- Push Button
- List Box
- Combo Box
- Radio Button

- Check Box

See Also

GetFieldValue

AddWatermark

Add a watermark.

Syntax

HRESULT AddWatermark(IWatermarkElementInfo* pWatermarkInfo);

Parameters

pWatermarkInfo	A watermark information object.

Return value

S_OKif successful, S_FAILif not.

Remarks

The parameter pWatermarkInfo is always created through calling Application.CreateWatermarkElementInfo,

See Also

Application.CreateWatermarkElementInfo

OCRAndExportToExcel

OCR and export the document to an Excel document.

Syntax

HRESULT OCRAndExportToExcel(BSTR ExcelPathname, SHORT start, SHORT end, VARIANT_BOOL bAllPages);

Parameters

ExcelPathname	An Excel Pathname.
start	Page index of start page, ignored if bAllPages is true.
end	Page index of end page, ignored if bAllPages is true.
bAllPages	Whether to work on all pages.

Return value

S_OKif successful, S_FAILif not.

Remarks

The parameter ExcelPathname mustadd .xlsx as suffix, otherwise it will automatically append .xlsx as suffix. The parameters Start and End both begin with 1, and will both be ignored if bAllPages is true.

If ExcelPathname already exists, the old file will be replaced with a new generated excel file directly, and you need to pay attention to it.

See Also

Export

PhantomPDF.Page

Represent a Page in a Document. Not yet supported.

PhantomPDF.WatermarkElementInfo

Represent a data type of watermark info. WatermarkElementInfo is usually a parameter of the interface Document.AddWatermark.

Methods

The WatermarkElementInfo object has the following methods.

Method	Description
Serialize	Serialized as a string.

Properties

The WatermarkElementInfo object has the following Properties.

Property	Description
Type	0-Text, 1-File. The default value is 0.
WMFile	Watermark file path. Ignored if Type is 0.
WMText	Watermark text. Ignored if Type is 1.
FontName	Watermark font name. Ignored if Type is 1.
FontSize	Watermark font size. Ignored if Type is 1 or WMScale greater than or equal to zero.

TextColorRef	A color, for example 0x000000, 0xFFFFFF and 0x888888. Ignored if Type is 1.
Rotation	Rotation Angle (0 ~ 360) °.
Opacity	Opacity (0 ~ 1).
WMScale	Scale relative to the target page, (-1 ~ 1). -1 means no scale, and FontSize will be ignored if WMScale greater than or equal to zero.
Top	0 - Appear behind page, 1 - Appear on top of page.
VerticalDistance	Vertical offset, inch, (-100000.00 ~ 100000.00)
VerticalDistanceFrom	Top 0, Center 1 , Bottom 2
HorizontalDistance	Horizontal offset, inch, (-100000.00 ~ 100000.00)
HorizontalDistanceFrom	Left 0, Center 1 , Right 2
Start	The Start page that needs watermark, with the page index beginning with 1.
End	The End page that needs watermark, with the page index beginning with 1.
Even	The even pages between the Start page and the End page that need watermark.
Odd	The odd pages between the Start page and the End page that need watermark.
WMFilePageIndex	The page index of the Watermark file. Beginning from 1. Ignored if Type is 0.

Remarks

When the properties Start and End are -1 at the same time, it will mean all pages, and without doubt that the properties Even and Odd are still working.

For VBA code, the property TextColorRef can be assigned to &H0, &HFFFFFF.For more details of common color, please see the following table.

Constants	value	describe
vbBlack	RGB(0, 0, 0)	Black
vbRed	RGB(255, 0, 0)	Red
vbGreen	RGB(0, 255, 0)	Green
vbYellow	RGB(255, 255, 0)	Yellow
vbBlue	RGB(0, 0, 255)	Blue
vbMagenta	RGB(255, 0, 255)	Magenta
vbCyan	RGB(0, 255, 255)	Cyan

vbWhite	RGB(255, 255, 255)	White

Serialize

Serialized as a string.

Syntax

BSTR Serialize();

Return value

A string representing the WatermarkElementInfo.

Remarks

In general, you don’t need to care the interface. But if you want to save a WatermarkElementInfo permanently, serializing it as a string is a great way.

PhantomPDF.CombineFlags

An enum use for Creator.CombineFiles’s parameter uCombineFlags.

Constants

The PhantomPDF.CombineFlags enum has the following constants.

Constants	value
COMBINE_DEFAULT	0x00
COMBINE_STOP_CONVERTFAIL	0x01
COMBINE_ADD_CONTENTS	0x02
COMBINE_TRAVEL_SUBDIR	0x04

COMBINE_DEFAULT

0x00, Default.

Remarks

Skip the error file and continue. Don't build the contents index page. Don't traverse subdirectory.

COMBINE_STOP_CONVERTFAIL

0x01, Abort the combination and exit the function if any error occurs.

COMBINE_ADD_CONTENTS

0x02, Add contents index to the first page.

COMBINE_TRAVEL_SUBDIR

0x04, Traverse subdirectory.

Demo for VBA

The following are some sample VBA codes you can use for your application.

Example 1 A simple VBA code skeleton of your application

```
Dim phApp As PhantomPDF.Application

Set phApp = CreateObject("PhantomPDF.Application")

Dim phCreator As PhantomPDF.Creator
Set phCreator = phApp.Creator

Call phCreator.ConvertToPDF("D:\image.png", " D:\image.png.pdf")

Dim phRotateDoc As PhantomPDF.Document
Set phRotateDoc = phApp.OpenDocument("D:\image.png.pdf", "", True, True)

Dim bValidDoc As Boolean
bValidDoc = phApp.IsValidDocument(phRotateDoc)

If bValidDoc Then
    Call phRotateDoc.RotatePage(0, 1)
    phRotateDoc.Export ("D: \rotateDest.pdf")

    'do other things ......

    Call phApp.CloseDocument(phRotateDoc, False)


End If
phApp.Exit
```

Example 2 Adding a text watermark to the center of all pages

```
Dim phWmInfo As PhantomPDF.WatermarkElementInfo
Set phWmInfo = phApp.CreateWatermarkElementInfo()

phWmInfo.Type = 0
phWmInfo.WMText = "Foxit PhantomPDF"
phWmInfo.FontName = "Helvetiva"
phWmInfo.FontSize = 24
phWmInfo.TextColorRef = &H0
phWmInfo.Rotation = 0
phWmInfo.Opacity = 1
phWmInfo.WMScale = 1
phWmInfo.Top = True
phWmInfo.VerticalDistance = 0
phWmInfo.VerticalDistanceFrom = 1
phWmInfo.HorizontalDistance = 0
phWmInfo.HorizontalDistanceFrom = 1
phWmInfo.Start = -1
phWmInfo.End = -1
phWmInfo.Even = True
phWmInfo.Odd = True

Call phRotateDoc.AddWatermark(phWmInfo)
phRotateDoc.Export ("D:\watermark.pdf")
```

Example 3 Adding an image watermark to the center of all pages

```
Dim phWmInfo As PhantomPDF.WatermarkElementInfo
Set phWmInfo = phApp.CreateWatermarkElementInfo()

phWmInfo.Type = 1
phWmInfo.WMFile = "D:\image.png"
phWmInfo.Rotation = 0
phWmInfo.Opacity = 1
phWmInfo.WMScale = 1
phWmInfo.Top = True
phWmInfo.VerticalDistance = 0
phWmInfo.VerticalDistanceFrom = 1
phWmInfo.HorizontalDistance = 0
phWmInfo.HorizontalDistanceFrom = 1
phWmInfo.Start = -1
phWmInfo.End = -1
phWmInfo.Even = True
phWmInfo.Odd = True
phWmInfo.WMFilePageIndex = 1

Call phRotateDoc.AddWatermark(phWmInfo)
phRotateDoc.Export ("D:\watermark.pdf ")
```

Example 4 Converting non-PDF to PDF

```
        Dim phCreator As PhantomPDF.Creator
        Set phCreator = phApp.Creator

        Call phCreator.ConvertToPDF("D:\image.png", " D:\image.png.pdf")
        Call phCreator.ConvertToPDF("D:\excel.xlsx", "D:\excel.xlsx.pdf")
```

Example 5 Combining several documents to a PDF

The path "D:\CombineFiles" is a folder which contains several files that are supported.

```
        Dim phCreator As PhantomPDF.Creator
        Set phCreator = phApp.Creator

        Dim nCombinedCnt As Integer

        nCombinedCnt = phCreator.CombineFiles("D:\image1.png|image2.png", "D:
\combineFiles_files.pdf", COMBINE_ADD_CONTENTS)

        Call phCreator.CombineFiles("D:\CombineFiles", "D:\combineFiles_folder.pdf",
COMBINE_ADD_CONTENTS)
```

Example 6 Getting/Setting a PDF form’s value

```
Dim phFormDoc As PhantomPDF.Document
Set phFormDoc = phApp.OpenDocument("D:\testForms.pdf", "", True, True)

'Text Field
Dim strTextField0 As String
strTextField0 = phFormDoc.GetFieldValue("Text Field0")
Call phFormDoc.SetFieldValue("Text Field0", "sally zhong for Testing")
strTextField0 = phFormDoc.GetFieldValue("Text Field0")


'Push Button
Dim strPushButton1 As String
strPushButton1 = phFormDoc.GetFieldValue("Push Button1")
Call phFormDoc.SetFieldValue("Push Button1", "I am Button")
strPushButton1 = phFormDoc.GetFieldValue("Push Button1")


'Check Box
Dim strCheckBox0 As String
strCheckBox0 = phFormDoc.GetFieldValue("Check Box0")

If strCheckBox0 = "Yes" Then
   Call phFormDoc.SetFieldValue("Check Box0", "Off")
Else
   Call phFormDoc.SetFieldValue("Check Box0", "Yes")
End If

strCheckBox0 = phFormDoc.GetFieldValue("Check Box0")


'Radio Button
Dim strRadioButton0 As String
strRadioButton0 = phFormDoc.GetFieldValue("Radio Button0")

If strRadioButton0 = "Yes" Then
   Call phFormDoc.SetFieldValue("Radio Button0", "Off")
Else
   Call phFormDoc.SetFieldValue("Radio Button0", "Yes")
End If

strRadioButton0 = phFormDoc.GetFieldValue("Radio Button0")
```

```
'ComboBox
Dim strComboBox0 As String
strComboBox0 = phFormDoc.GetFieldValue("Combo Box0")
Call phFormDoc.SetFieldValue("Combo Box0", " Combo Data1")
strComboBox0 = phFormDoc.GetFieldValue("Combo Box0")
Call phFormDoc.SetFieldValue("Combo Box0", "out of term")
strComboBox0 = phFormDoc.GetFieldValue("Combo Box0")


'List Box
Dim strListBox0 As String
strListBox0 = phFormDoc.GetFieldValue("List Box0")
Call phFormDoc.SetFieldValue("List Box0", " List Data1")
strListBox0 = phFormDoc.GetFieldValue("List Box0")
Call phFormDoc.SetFieldValue("List Box0", "out of term")
strListBox0 = phFormDoc.GetFieldValue("List Box0")


phFormDoc.Save
Call phApp.CloseDocument(phFormDoc, False)
```

Appendix 1 Last-Error Code List

The Last-Error Code is an integer that you can use to know about reasons for errors when calling the API fails. You can get a LastError value by the following code.

```
' phApp is a PhantomPDF.Application Object
MsgBox phApp.LastError
```

Here is the detail about the LastError List.

ErrorName	value	Description
EC_UNKNOWN_ERROR	32767	An unknown error
EC_OK	0	OK
EC_CANCEL	1	Cancel
EC_INVALID_PARAMETER	2	Invalid parameter
EC_TIME_OUT	3	Time out
EC_OUT_OF_RANGE_ADJUST	4	Input out of range. Automatic adjustment has been made.
EC_PERMISSION_DENIED	5	Permission denied
EC_FILEPATH_INVALID_OR_OPENED	6	The specified file path is invalid or already has been opened
EC_FILE_NOT_EXIST	7	The specified file does not exist
EC_FILE_FORMAT_SUPPORTED	8	File format is not supported
EC_PRINTER_NOT_EXIST	9	The specified printer does not exist
EC_COM_INIT_FAILED	10	COM initialization failed
EC_BLANK_PPT	11	Can't convert a blank PPT
EC_BLANK_EXCEL	12	Can't convert a blank Excel
EC_DOCUMENT_SAVED_BEFORE_OCR	13	The document should be saved before OCR
EC_LICENCE_INVALID	14	The license is invalid
EC_OUT_OF_RANGE	15	Input out of range
EC_NO_CURRENT_DOCUMENT	16	There is no current document
EC_CAN_ONLY_OPEN_PDF	17	Can only open a PDF file
EC_FORM_NOT_EXIST	18	The specified form does not exist
EC_CANT_FIND_ITEM	19	Can't find the specified item value

EC_CONNECTING_PH_FAIL	20	Connecting to Foxit PhantomPDF failed
EC_DOCUMENT_OBJECT_IS_INVALID	21	Document Object is invalid
EC_OCR_ERROR	22	OCR error
EC_PORT_USED	23	The specified port is being used
EC_NO_MODIFY_PERM	24	There is no Modify permission
EC_NO_PRINT_PERM	25	There is no Print permission
EC_NO_Fill_FORM_PERM	26	There is no FillForm permission
EC_OPEN_DOCUMENT_FAIL	27	Failed to open the document.
EC_TRIAL_EXPIRED	28	The trial has expired
EC_MODULE_NOT_IN_KEY	29	The module about the interface is not included in your keyfile