PDF Maker Class .NET by SkySof Software last updated: 11/13/06

PDF Maker Class .NET is a powerful .NET component for software developers containing many useful PDF related functions. These functions can be used to easily create high-quality, professional Adobe Acrobat PDF files from virtually any file type: Excel worksheets, Word documents, PowerPoint files, Access reports, Publisher documents, Visio files, Crystal reports, AutoCAD drawings, image files, text files, etc. You can also create password-protected encrypted PDF files with ease. The encrypted files use high level 128-bit RC4 encryption so you can be assured they are safe from hackers. Also, with PDF Maker Class .NET you can easily add form fields, comments, and bookmarks to your documents. Using PDF Maker Class .NET in your own application can save you plenty of time and money. All functions have been thoroughly tested and optimized to perform as quickly as possible. If you are a COM developer we have an ActiveX version named "PDF Maker DLL" which can be downloaded from our site at http://www.skysof.com The direct download link is http://www.getfilez.com/pdfmaker.exe If you have an idea for a new function please email a detailed description to SkySof Software at kusluski@bellsouth.net

Note: PDF Maker Class .NET requires Adobe Acrobat Standard or Professional 5 or greater. When installing Acrobat make certain that you also install the Adobe Distiller which is not installed by default.

The cost of PDF Maker Class .NET is only $99.95 per developer and only $499.95 for a site license. If you want to customize the PDF Maker Class .NET functions to your liking the complete Visual Basic .NET source code can be purchased for only $999.95. All upgrades are free and PDF Maker Class .NET may be distributed with your application royalty free.

A trial version of PDF Maker Class .NET can be downloaded from the following link:

http://www.getfilez.com/pdfmcls.zip

Click the link below to purchase PDF Maker Class .NET through RegNow:

https://www.regnow.com/softsell/nph-softsell.cgi?item=4459-43

You can save 10% if you purchase PDF Maker Class .NET through PayPal:

Individual Developer License – only $90.95

https://www.paypal.com/xclick/business=kusluski%40mail.ic.net&item_name=PDF+Maker+Class+.NET+Single+Developer+License&amount=90.95

Site Developer License – only $449.95

https://www.paypal.com/xclick/business=kusluski%40mail.ic.net&item_name=PDF+Maker+Developer+Site+License&amount=449.95

Complete Visual Basic .NET Source Code – only $900.00

https://www.paypal.com/xclick/business=kusluski%40mail.ic.net&item_name=PDF+Maker+Complete+Visual+Basic+.NET+Source+Code&amount=900.00

Important! Before using PDF Maker Class .NET you should configure the Adobe PDF Printing Preferences:

Note: These instructions pertain specifically to Window's XP. They may vary slightly if you are using a different operating system.

1) Click Window's Start Button on lower left corner of screen
2) Select "Printers and Faxes" from the menu
3) With the mouse cursor over the printer "Adobe PDF" (or "Acrobat Distiller") click the right mouse button to invoke the popup menu
4) Select "Printing Preferences…" from the menu
5) Uncheck "View Adobe PDF results"
6) Uncheck "Do not send fonts to Adobe PDF"
7) Uncheck "Ask to Replace existing PDF file"
8) All other options should be checked
9) Click the OK Button to save changes

To add the PDF Maker Class .NET reference to your .NET project:

1) Load Visual Studio .NET. and create a new project
2) From the Project Menu select "Add Reference…"
3) Click the Browse Button

4) Open file C:\Program Files\SkySof Software Inc\PDF Maker Class .NET\pdfmcls.dll
5) The file should appear in the "Selected Components:" list
6) Click the OK Button. The reference should now appear in the Solution Explorer window.

Using PDF Maker Class .NET in your .NET project

After adding the PDF Maker Class .NET reference, place the following code to the top of the Form Class:

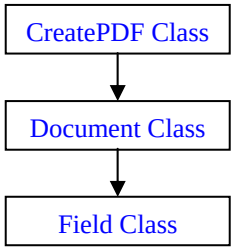
```
Imports SkySof
Imports SkySof.CreatePDF
```

To create an instance of PDF Maker Class .NET place the following statement in the Load subroutine of your form:

```
oPDFmaker = New CreatePDF
```

Note: All sample code in this document is Visual Basic .NET code

Object Hierarchy Chart



PDF Maker Class .NET consists of three classes: CreatePDF Class, Document Class, and the Field Class. The CreatePDF Class contains methods for creating PDF and PS (PostScript) files from various file types and miscellaneous methods. The Document Class represents a PDF document and contains methods and properties for changing the appearance and behavior of a document. This class includes methods for creating form fields and adding annotations. Finally, the Field Class represents a valid form field within a document. It contains methods and properties for changing the appearance and behavior of form fields of type text, combo box, list box, radio button, check box, and signature.

On the previous page we showed you how to create an instance of the CreatePDF Object via oPDFmaker = New CreatePDF. To create an instance of the Document Object use the OpenDocument method of the CreatePDF Object:

```
Dim objDoc As Document
Dim strPDF As String
strPDF = "c:\file.pdf"
objDoc = oPDFMaker.OpenDocument(strPDF)
If objDoc Is Nothing Then
   MsgBox("Unable to open file.")
Else
   MsgBox "Opened " & objDoc.Path
End If
```

Once you have an instance of the Document Object available you can create an instance of the Field Object with the GetField method of the Document Object:

```
Dim objField As Field
Dim strName As String
strName = "TextField1"
objField = objDoc.GetField(strName)
If objField Is Nothing Then
   MsgBox("Invalid field.")
Else
   MsgBox "Value: " & objField.Value
End If
```

You do not have to store the field object in a variable to access its methods and properties:

```
MsgBox objDoc.GetField(“MyField”).Value
```

Looping through the Fields Collection:

```
Dim fld As Object
Dim s As String
s = ""
For Each fld In objDoc.GetFields
   s = s & fld & vbCrLf
Next
MsgBox s, vbOKOnly, objDoc.GetFields.Count & " total fields"
```

You can have more than one document or field open at a time. This comes in handy for exchanging information between documents:

```
Dim objDoc1 As Document
Dim objDoc2 As Document
Dim objField1 As Field
Dim objField2 As Field
objDoc1 = oPDFmaker.OpenDocument("c:\temp\file1.pdf")
objDoc2 = oPDFmaker.OpenDocument("c:\temp\file2.pdf")
objField1 = objDoc1.GetField("Field1")
objField2 = objDoc2.GetField("Field1")
objDoc1.Author = objDoc2.Author
objField1.Value = objField2.Value
```

Creating Acrobat forms

With PDF Maker Class .NET you can easily create Acrobat forms programmatically. Acrobat forms can then be uploaded to a web server for use. Acrobat forms have many advantages over HTML pages. PDF Maker Class .NET contains many functions for creating and manipulating the following form field types: text box, check box, button, radio button, combo box, list box, and signature. Before creating an Acrobat form you must create a blank PDF file. The following source code demonstrates how to create a form with calculating text fields:

```
Dim strPDF As String
Dim strDOC As String

On Error Resume Next

Call oPDFmaker.CloseAcrobat() 'ensure Acrobat is closed

strPDF = AppPath & "\blank.pdf"
strDOC = AppPath & "\blank.doc"

oPDFmaker.CreatePDFfromWord(strPDF, strDOC)

If Err.Number Then
   MsgBox(Err.Number & " : " & Err.Description, vbExclamation, "Error Message")
   MsgBox("Word must be installed on this machine.", vbInformation, "Message")
End If

If Err.Number = 0 Then

   Dim objDoc As Document
   objDoc = oPDFmaker.OpenDocument(strPDF)
   If objDoc Is Nothing Then
      MsgBox("Unable to create object. You must have Adobe Acrobat Standard or Professional installed.")
   Else
      Dim a(3) As Object
      Dim varColor(3) As Object
      Dim varRadioItems(1) As String
```

```
        'set location
        a(0) = 10 'lower left X
        a(1) = 765 'lower left Y
        a(2) = 270 'upper right X
        a(3) = 715 'upper right Y

        'Create Labels - actually read-only Text Fields with transparent border
        varColor(0) = "G"
        varColor(1) = 1
        varColor(2) = 0
        varColor(3) = 0
        objDoc.CreateTextField("Label1", 1, a, AlignType.pdoAlignLeft, FieldBorderType.pdoFieldBorderSolid, , False, 0, ,
FieldDisplayType.pdoFieldDisplayVisible, True, False, , FieldLineWidthType.pdoFieldLineWidthNone, False, False, True,
False, varColor, , , FontType.pdoTimesRoman, 24, , "Enter Number")
        a(1) = a(1) - 70
        a(3) = a(3) - 70
        objDoc.CreateTextField("Label2", 1, a, AlignType.pdoAlignLeft, FieldBorderType.pdoFieldBorderSolid, , False, 0, ,
FieldDisplayType.pdoFieldDisplayVisible, True, False, , FieldLineWidthType.pdoFieldLineWidthNone, False, False, True,
False, varColor, , , FontType.pdoTimesRoman, 24, , "Enter Number")
        a(1) = a(1) - 70
        a(3) = a(3) - 70
        objDoc.CreateTextField("Label3", 1, a, AlignType.pdoAlignLeft, FieldBorderType.pdoFieldBorderSolid, , False, 0, ,
FieldDisplayType.pdoFieldDisplayVisible, True, False, , FieldLineWidthType.pdoFieldLineWidthNone, False, False, True,
False, varColor, , , FontType.pdoTimesRoman, 24, , "Result")

        'Create Text Fields
        a(0) = 200 'lower left X
        a(1) = 765 'lower left Y
        a(2) = 370 'upper right X
        a(3) = 715 'upper right Y
        objDoc.CreateTextField("Field1", 1, a, AlignType.pdoAlignLeft, FieldBorderType.pdoFieldBorderBeveled, 10, False,
0, 2, FieldDisplayType.pdoFieldDisplayNoPrint, True, False, , FieldLineWidthType.pdoFieldLineWidthMedium, False, False,
False, False, , , , FontType.pdoTimesRoman, 24, "Enter number", , FieldTriggerType.pdoOnFocus, "var txtField = event.target;
txtField.fillColor = color.red; txtField.textColor = color.white;")
        'Set backcolor to tranparent when field loses focus
        objDoc.GetField("Field1").SetFieldAction(FieldTriggerType.pdoOnBlur, "var txtField = event.target;
txtField.fillColor = color.transparent; txtField.textColor = color.black;")
        'Prevent non-numeric characters in field
        objDoc.GetField("Field1").SetFieldAction(FieldTriggerType.pdoKeystroke, "var re = /^\\d$/;if (event.willCommit ==
false) { if (re.test(event.change) == false) { app.beep(); event.rc = false; } }")
        a(1) = a(1) - 70
        a(3) = a(3) - 70
        objDoc.CreateTextField("Field2", 1, a, AlignType.pdoAlignLeft, FieldBorderType.pdoFieldBorderBeveled, 10, False,
1, 3, FieldDisplayType.pdoFieldDisplayNoPrint, True, False, , FieldLineWidthType.pdoFieldLineWidthMedium, False, False,
False, False, , , , FontType.pdoTimesRoman, 24, "Enter number", , FieldTriggerType.pdoOnFocus, "var txtField = event.target;
txtField.fillColor = color.red; txtField.textColor = color.white;")
        objDoc.GetField("Field2").SetFieldAction(FieldTriggerType.pdoOnBlur, "var txtField = event.target;
txtField.fillColor = color.transparent; txtField.textColor = color.black;")
        objDoc.GetField("Field2").SetFieldAction(FieldTriggerType.pdoKeystroke, "var re = /^\\d$/;if (event.willCommit ==
false) { if (re.test(event.change) == false) { app.beep(); event.rc = false; } }")
        a(1) = a(1) - 70
        a(3) = a(3) - 70
        objDoc.CreateTextField("Field3", 1, a, AlignType.pdoAlignLeft, FieldBorderType.pdoFieldBorderBeveled, 10, False,
2, , FieldDisplayType.pdoFieldDisplayVisible, True, False, , FieldLineWidthType.pdoFieldLineWidthMedium, False, False,
True, False, , , , FontType.pdoTimesRoman, 24, "Enter text here", , FieldTriggerType.pdoCalculate, "var f1 =
this.getField('Field1');var f2 = this.getField('Field2');var f = this.getField('Radio');if(f.isBoxChecked(0)) event.value = f1.value +
f2.value;else event.value = f1.value - f2.value;")

        a(1) = a(1) - 70
        a(3) = a(3) - 70

        'Create Button
        'set color to yellow
```

```
'varColor(0) = "RGB"
'varColor(1) = 1
'varColor(2) = 1
'varColor(3) = 0.8

objDoc.SetColor(ColorType.pdoYellow, varColor)

objDoc.CreateButtonField("Button1", 1, a, "Calculate", FieldBorderType.pdoFieldBorderBeveled, , , True,
FieldDisplayType.pdoFieldDisplayVisible, varColor, FieldLineWidthType.pdoFieldLineWidthMedium, False, , , ,
FontType.pdoTimesRoman, 20, "Button Field", FieldTriggerType.pdoMouseDown, "this.calculateNow();")

a(0) = a(0) + 180
a(2) = a(2) + 180

'Create a submit button to submit form data to a web page for processing
objDoc.CreateButtonField("Button2", 1, a, "Submit", FieldBorderType.pdoFieldBorderBeveled, , , True,
FieldDisplayType.pdoFieldDisplayVisible, varColor, FieldLineWidthType.pdoFieldLineWidthMedium, False, , , ,
FontType.pdoTimesRoman, 20, "Button Field", FieldTriggerType.pdoMouseDown,
"this.submitForm('http://www.website.com/langerhans.asp', false, true);")

'Create Radio Buttons
varColor(0) = "G"
varColor(1) = 1
a(0) = 370 'lower left X
a(1) = 755 'lower left Y
a(2) = 420 'upper right X
a(3) = 735 'upper right Y
varRadioItems(0) = "Add"
varRadioItems(1) = "Subtract"
objDoc.CreateTextField("Label4", 1, a, AlignType.pdoAlignRight, FieldBorderType.pdoFieldBorderSolid, , False, 0,
, FieldDisplayType.pdoFieldDisplayVisible, True, False, , FieldLineWidthType.pdoFieldLineWidthNone, False, False, True,
False, varColor, , , , 24, , "+")
a(0) = a(0) + 40 'lower left X
a(2) = a(2) + 40 'upper right X
objDoc.CreateRadioButtonField("Radio", 1, a, varRadioItems, FieldBorderType.pdoFieldBorderBeveled,
varRadioItems(0), FieldDisplayType.pdoFieldDisplayVisible, , FieldLineWidthType.pdoFieldLineWidthMedium, False, False, ,
, , , , varRadioItems(0), False)
a(0) = a(0) + 40 'lower left X
a(2) = a(2) + 40 'upper right X
objDoc.CreateTextField("Label5", 1, a, AlignType.pdoAlignRight, FieldBorderType.pdoFieldBorderSolid, , False, 0,
, FieldDisplayType.pdoFieldDisplayVisible, True, False, , FieldLineWidthType.pdoFieldLineWidthNone, False, False, True,
False, varColor, , , , 24, , "-")
a(0) = a(0) + 40 'lower left X
a(2) = a(2) + 40 'upper right X
objDoc.CreateRadioButtonField("Radio", 1, a, varRadioItems, FieldBorderType.pdoFieldBorderBeveled,
varRadioItems(1), FieldDisplayType.pdoFieldDisplayVisible, , FieldLineWidthType.pdoFieldLineWidthMedium, False, False,
lngGlyphStyle:=GlyphStyleType.pdoCircle)

'Get list of fields using function GetFieldsArray
Dim varFields As Object
objDoc.GetFieldsArray(varFields)
Dim i As Integer
Dim s As String
For i = 0 To UBound(varFields)
   s = s & varFields(i) & vbCrLf
Next
MsgBox(s, vbOKOnly, UBound(varFields) + 1 & " fields created")

'Get list of fields using function GetFields (returns a collection)
'Dim fld As Variant
's = ""
'For Each fld In objDoc.GetFields
'    s = s & fld & vbCrLf
```

```
        'Next
        'MsgBox s, vbOKOnly, objDoc.GetFields.Count & " fields created"

        objDoc.Save(SaveType.pdoSaveChangesOnly)

        'objDoc.SubmitFormToURL "http://www.Langerhans.gov/orders.asp"

        objDoc.CloseDoc()

        oPDFmaker.OpenPDF(strPDF)

    End If
End If
```

JavaScript

The programming language JavaScript can be used to empower Acrobat forms. With JavaScript you can create sophisticated forms with interactive interfaces, implement calculated form fields, and submit form data to a web server. These are only a few things you can do within your Acrobat documents using JavaScript. The extent to which you can use JavaScript in your documents is virtually limitless. There are four types of JavaScript in an Acrobat document:

1) Form Field – Attached to form fields. Use function SetFieldAction to set these JavaScripts.
2) Document – Associated with the opening of a Acrobat document. Use function AddScript to set this JavaScript.
3) Document Action – Executed when a predefined event occurs within an Acrobat document. Use function SetDocumentAction to set these JavaScripts.
4) Page – Associated with a specific page within an Acrobat document. Use function SetPageAction to set these JavaScripts.
For a more complete reference on using JavaScript in your documents consider purchasing the indispensable book "Extending Acrobat Forms With JavaScript" by John Deubert (Adobe Press). John Deubert's web site is http://www.acumentraining.com The PDF Maker Class .NET Visual Basic .NET Demo code contains several useful JavaScripts. With PDF Maker Class .NET and JavaScript you will be able to programmatically create awesome Acrobat forms!

Distributing PDF Maker Class .NET with your application

PDF Maker Class .NET may be distributed with your application royalty free. If you use functions such as OpenPDF(), CreateBookmarksInPDF(), CreatePDFfromWebPage(), SetOpenPassword() or any other function which opens a PDF file in your application you should include the file PDFOSEND.EXE in your setup program. This file is located in your Window's System32 directory. You should install this file on the client's machine in the same directory as the PDF Maker Class .NET DLL file or in their Window's System Directory. The file PDFOSEND.EXE is used by PDF Maker DLL to send messages to Adobe Acrobat windows using Window's API. Its purpose is to automate the manual process of typing in file paths.

How to create password-protected/encrypted PDF files

Note: this feature only available with Adobe Acrobat 6 and greater

Passwords and encryption are set from the Adobe Acrobat Printer setup screen. To access these features:

1) Click the Window's Start Button
2) Select "Printers and Faxes" from the menu
3) Right click on "Adobe PDF" to invoke the popup menu
4) Select "Printing Preferences…" from the menu
5) Find "Adobe PDF Security" on the screen and click the drop-down listbox
6) Select "Reconfirm Security for each job" from the list
7) Click the Apply Button to save your changes
8) Click the Edit.. Button to the right of the list box
9) Modify security settings to your liking and click OK Button when done
10) Click OK Button to exit the screen

Now that Adobe Acrobat Distiller has been setup to create password-protected/encrypted files you're good to go! When using functions that create PDF files, such as CreatePDFfromWord(), make sure that the parameter blnApplySecurity is set to True. For example:

```
oPDFmaker.CreatePDFfromWord "C:\MyDir\file.pdf", "C:\MyDir\file.doc", blnApplySecurity:=True
```

If you create a password-protected file and want to open it with the OpenPDF() function you must use the appropriate password:

oPDFmaker.OpenPDF “C:\MyDir\file.pdf”, strOpenPassword:=”opensezme”

Events (CreatePDF Class)

DoneCreatingPDF – Executes after the PDF file has been created.
DoneCreatingPS – Executes after the PS (PostScript) file has been created.
StartCreatingPDF – Executes before creating the PDF file.
StartCreatingPS – Executes before creating the PS (PostScript) file.

1. PDF Creation Methods (CreatePDF Class)

1.1 CreatePDF - Method to create an Adobe Acrobat (PDF) file from a variety of file types.

Note: This function requires Acrobat 6 and greater.

Arguments:

strSourceFile As String	Source file to use
strPDFFile As String	PDF file to create
Optional intSecsToWaitForPDF As Integer	Seconds to wait for PDF to complete

Example:

```
Dim strFile As String
Dim strPDF As String
Dim i As Long

On Error Resume Next

strFile = App.Path & "\debug.doc"
strPDF = App.Path & "\new.pdf"

'Use this function to quickly create a PDF from a variety of file types including
'DOC, DWG, XLS, GIF, JPG, TIF, etc.

oPDFmaker.CreatePDF strFile, strPDF
If Err.Number = 0 Then
   'Open the PDF
   oPDFmaker.OpenPDF strPDF
End If
```

1.2 CreatePDFfromAccessReport - Method to create an Adobe Acrobat (PDF) file from an Access report.

Requirements: Access and file PDFOSEND.EXE (see page 6 of this document)

Arguments:

strPDFFile As String	PDF file to create
strDatabase As String	Access database to use
strReportName As String	Report to use
Optional strPassword As String	Access database password
Optional strFilter As String	Report filter
Optional strWhereCondition As String	Report where condition
Optional strPDFWriter As String	PDF Writer name. Default: “Acrobat PDFWriter”
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strMDB As String
Dim strRpt As String
Dim i As Integer
```

```
On Error Resume Next
Screen.MousePointer = vbHourglass

strPDF = App.Path & "\invoice.pdf"
'Note: You may need to modify this file path
strMDB = "C:\Program Files\Microsoft Visual Studio\VB98\NWIND.MDB"
strRpt = "invoice"

oPDFmaker.CreatePDFfromAccessReport strPDF, strMDB, strRpt

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.3 CreatePDFfromAutoCAD - Method to create an Adobe Acrobat (PDF) file from AutoCAD DWG and DXF files.

Requirements: AutoCAD and file PDFOSEND.EXE (see page 6 of this document)
Arguments:

strPDFFile As String	PDF file to create
strDrawing As String	AutoCAD DWG/DXF file to use
Optional varLayout As Variant	Layout to use (can be a name or number greater than 0)
Optional strPDFWriter As String	PDF Writer name. Default: “Acrobat PDFWriter”
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strDWG As String

Screen.MousePointer = vbHourglass
strPDF = App.Path & "\test.pdf"
strDWG = App.Path & “\test.dwg”
oPDFmaker.CreatePDFfromAutoCAD strPDF, stDWG, “layout1”
'open PDF file
oPDFMaker.OpenPDF strPDF
Screen.MousePointer = vbDefault
```

1.4 CreatePDFfromCrystalReport - Method to create an Adobe Acrobat (PDF) file from a Crystal Report File.

Requirements: Crystal Reports and file PDFOSEND.EXE (see page 6 of this document)

Arguments:

strPDFFile As String	PDF file to create
strCrystalReport As String	Crystal Report File to use
Optional strDistiller As String	Name of Adobe Acrobat Distiller
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strRPT As String

Screen.MousePointer = vbHourglass
strPDF = App.Path & "\test.pdf"
strRPT = App.Path & “\test.rpt”
oPDFmaker.CreatePDFfromCrystalReport strPDF, strRPT
'open PDF file
```

```
oPDFmaker.OpenPDF strPDF
Screen.MousePointer = vbDefault
```

1.5 CreatePDFfromExcel - Method to create an Adobe Acrobat (PDF) file from an Excel worksheet.

Requirements: Excel

Arguments:

StrPDFFile As String	PDF file to create
StrExcelWorkbook As String	Excel workbook file to use
VarExcelWorksheet As Variant	Excel worksheet to use. Can be a name or number
strRange As String	Excel worksheet range to use
Optional strPassword As String	Excel workbook password
Optional strWriteResPassword As String	Excel workbook write reserve password
Optional blnUseGrid As Boolean	Use grid lines. Values: True,False
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional strHeader As String	Page header
Optional lngHeaderAlignment As Long	Page header alignment. Values: 0=left, 1=center, 2=right
Optional strFooter As String	Page footer
Optional lngFooterAlignment As Long	Page footer alignment. Values: 0=left, 1=center, 2=right
Optional dblTopMargin As Double	Page top margin in inches
Optional dblLeftMargin As Double	Page left margin in inches
Optional dblRightMargin As Double	Page right margin in inches
Optional dblBottomMargin As Double	Page bottom margin in inches
Optional dblHeaderMargin As Double	Page header margin in inches
Optional dblFooterMargin As Double	Page footer margin in inches
Optional lngFirstPageNumber As Long	Beginning page number
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional strJobOptionsFile As String	Print job options file.
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPDF As String
Dim strWB As String
Dim strRange As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\sample.pdf"
strWB = App.Path & "\sample.xls"
strRange = "A1:E276"

oPDFmaker.CreatePDFfromExcel strPDF, strWB, 1, strRange, , , , , "Page &P", "right", "&D &T", "center", 1.0, 0.5, 0.0, 1.0

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.6 CreatePDFfromExcel2 - Method to create an Adobe Acrobat (PDF) file from an Excel worksheet.

Requirements: Excel and file PDFOSEND.EXE (see page 6 of this document)

Note: No postscript file created. Creates PDF faster than CreatePDFfromExcel()

Arguments:

StrPDFFile As String	PDF file to create
StrExcelWorkbook As String	Excel workbook file to use
VarExcelWorksheet As Variant	Excel worksheet to use. Can be a name or number
strRange As String	Excel worksheet range to use
Optional strPassword As String	Excel workbook password
Optional strWriteResPassword As String	Excel workbook write reserve password
Optional blnUseGrid As Boolean	Use grid lines. Values: True,False
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional strHeader As String	Page header
Optional lngHeaderAlignment As Long	Page header alignment. Values: 0=left, 1=center, 2=right
Optional strFooter As String	Page footer
Optional lngFooterAlignment As Long	Page footer alignment. Values: 0=left, 1=center, 2=right
Optional dblTopMargin As Double	Page top margin in inches
Optional dblLeftMargin As Double	Page left margin in inches
Optional dblRightMargin As Double	Page right margin in inches
Optional dblBottomMargin As Double	Page bottom margin in inches
Optional dblHeaderMargin As Double	Page header margin in inches
Optional dblFooterMargin As Double	Page footer margin in inches
Optional lngFirstPageNumber As Long	Beginning page number
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat?

Example:

```
Dim strPDF As String
Dim strWB As String
Dim strRange As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\sample.pdf"
strWB = App.Path & "\sample.xls"
strRange = "A1:E276"

oPDFmaker.CreatePDFfromExcel2 strPDF, strWB, 1, strRange, , , , , "Page &P", "right", "&D &T", "center", 1.0, 0.5, 0.0, 1.0

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.7 CreatePDFfromImageFile - Method to create an Adobe Acrobat (PDF) file from an image file (JPG,GIF,BMP,TIF,PNG,PCX).

Arguments:

strPDFFile As String	PDF file to create
strImageFile As String	Image file to use
Optional strTitle As String	Title to save with PDF file
Optional strAuthor As String	Author to save with PDF file
Optional strSubject As String	Subject to save with PDF file
Optional strKeywords As String	Keywords to save with PDF file

Example:

```
Dim strPDF As String
```

```
Dim strJPG As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\image.pdf"
'Pratically any type of image file can be used including JPG,TIF,PNG,PCX,GIF,BMP
strJPG = App.Path & "\image1.jpg"

oPDFmaker.CreatePDFfromImageFile strPDF, strJPG, "My PDF file", "Frank Kusluski", "Cute girl", "keywords go here"

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
  oPDFmaker.OpenPDF strPDF
End If
```

1.8 CreatePDFfromMultiAccessReport - Method to create an Adobe Acrobat (PDF) file from multiple Access reports.

Requirements: Access and file PDFOSEND.EXE (see page 6 of this document)

Note: Reports must exist in the same database

Arguments:

strPDFFile As String	PDF file to create
strDatabase As String	Database to use
varReportNames As Variant	array of report names
Optional strPassword As String	Password to use
Optional ByVal strPDFWriter As String	PDF Writer name. Default: “Acrobat PDFWriter”
Optional blnCloseAcrobat As Boolean.	Close Acrobat after creating PDF?

Example:

```
Dim arr(1) As String
arr(0) = “Report1”
arr(1) = “Report2”
oPDFMaker.CreatePDFfromMultiAccess “c:\file.pdf”, “c:\db.mdb”, arr
```

1.9 CreatePDFfromMultiAutoCAD - Method to create an Adobe Acrobat (PDF) file from multiple AutoCAD layouts.

Requirements: AutoCAD and file PDFOSEND.EXE (see page 6 of this document)

Note: Layouts must exist in the same drawing file.

Arguments:

strPDFFile As String	PDF file to create
strDWGFile As String	Drawing file to use
varLayouts As Variant	array of layout names/numbers
Optional ByVal strPDFWriter As String	PDF Writer name. Default: “Acrobat PDFWriter”
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim arr(1) As String
arr(0) = “Layout1”
arr(1) = “Layout2”
oPDFMaker.CreatePDFfromMultiAutoCAD “c:\file.pdf”, “c:\drawing.dwg”, arr
```

1.10 CreatePDFfromMultiExcel - Method to create an Adobe Acrobat (PDF) file from multiple Excel worksheets.

Requirements: Excel

Note: Worksheets must exist in the same workbook.

Arguments:

strPDFFile As String	PDF file to create
strExcelWorkbook As String	Excel workbook file to use
varExcelWorksheets As Variant	Excel worksheets to use (array). Can be a name or number
varRanges As Variant	Excel worksheet ranges to use (array)
Optional strPassword As String	Excel workbook password
Optional strWriteResPassword As String	Excel workbook write reserve password
Optional blnUseGrid As Boolean	Use grid lines. Values: True,False
Optional varLandscape As Variant	Page orientation (array). Values: True,False
Optional strHeader As String	Page header
Optional lngHeaderAlignment As Long	Page header alignment. Values: 0=left, 1=center, 2=right
Optional strFooter As String	Page footer
Optional lngFooterAlignment As Long	Page footer alignment. Values: 0=left, 1=center, 2=right
Optional dblTopMargin As Double	Page top margin in inches
Optional dblLeftMargin As Double	Page left margin in inches
Optional dblRightMargin As Double	Page right margin in inches
Optional dblBottomMargin As Double	Page bottom margin in inches
Optional dblHeaderMargin As Double	Page header margin in inches
Optional dblFooterMargin As Double	Page footer margin in inches
Optional varFirstPageNumber As Variant	Beginning page numbers (array)
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPDF As String
Dim strWB As String
Dim varWorksheets(2) As Integer
Dim varRanges(2) As String
Dim varLandscape(2) As Boolean
Dim varFirstPageNumber(2) As Long

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\sample.pdf"
strWB = App.Path & "\sample.xls"
varWorksheets(0) = 1
varWorksheets(1) = 2
varWorksheets(2) = 3
varRanges(0) = "A1:E111"
varRanges(1) = "A1:E131"
varRanges(2) = "A1:E46"
varLandscape(0) = True
varLandscape(1) = False
varLandscape(2) = True
varFirstPageNumber(0) = 1
varFirstPageNumber(1) = 4
varFirstPageNumber(2) = 8

oPDFmaker.CreatePDFfromMultiExcel strPDF, strWB, varWorksheets, varRanges, , , , varLandscape, "Page &P", "right", "&D &T",
"center", 1.0, 0.5, 0.0, 1.0 , , , varFirstPageNumber
```

```
Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.11 CreatePDFfromMultiExcel2 - Method to create an Adobe Acrobat (PDF) file from multiple Excel worksheets.

Requirements: Excel and file PDFOSEND.EXE (see page 6 of this document)

Note: No postscript files created. Creates PDF faster than CreatePDFfromMultiExcel()

Arguments:

strPDFFile As String	PDF file to create
strExcelWorkbook As String	Excel workbook file to use
varExcelWorksheets As Variant	Excel worksheets to use (array). Can be a name or number
varRanges As Variant	Excel worksheet ranges to use (array)
Optional strPassword As String	Excel workbook password
Optional strWriteResPassword As String	Excel workbook write reserve password
Optional blnUseGrid As Boolean	Use grid lines. Values: True,False
Optional varLandscape As Variant	Page orientation (array). Values: True,False
Optional strHeader As String	Page header
Optional lngHeaderAlignment As Long	Page header alignment. Values: 0=left, 1=center, 2=right
Optional strFooter As String	Page footer
Optional lngFooterAlignment As Long	Page footer alignment. Values: 0=left, 1=center, 2=right
Optional dblTopMargin As Double	Page top margin in inches
Optional dblLeftMargin As Double	Page left margin in inches
Optional dblRightMargin As Double	Page right margin in inches
Optional dblBottomMargin As Double	Page bottom margin in inches
Optional dblHeaderMargin As Double	Page header margin in inches
Optional dblFooterMargin As Double	Page footer margin in inches
Optional varFirstPageNumber As Variant	Beginning page numbers (array)
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strWB As String
Dim varWorksheets(2) As Integer
Dim varRanges(2) As String
Dim varLandscape(2) As Boolean
Dim varFirstPageNumber(2) As Long

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\sample.pdf"
strWB = App.Path & "\sample.xls"
varWorksheets(0) = 1
varWorksheets(1) = 2
varWorksheets(2) = 3
varRanges(0) = "A1:E111"
varRanges(1) = "A1:E131"
varRanges(2) = "A1:E46"
varLandscape(0) = True
varLandscape(1) = False
varLandscape(2) = True
```

```
varFirstPageNumber(0) = 1
varFirstPageNumber(1) = 4
varFirstPageNumber(2) = 8

oPDFmaker.CreatePDFfromMultiExcel2 strPDF, strWB, varWorksheets, varRanges, , , , varLandscape, "Page &P", "right", "&D &T",
"center", 1.0, 0.5, 0.0, 1.0 , , , varFirstPageNumber

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.12 CreatePDFfromMultiImage - Method to create an Adobe Acrobat (PDF) file from multiple image files (JPG,GIF,BMP,TIF,PNG,PCX).

Arguments:

strPDFFile As String	PDF file to create
varImageFiles As Variant	Image files to use (array)

Example:

```
Dim strPDF As String
Dim img(1) As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\multiple.pdf"

img(0) = App.Path & "\image1.jpg"
img(1) = App.Path & "\image2.jpg"

'Create the PDF!
oPDFmaker.CreatePDFfromMultiImage strPDF, img

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
  oPDFmaker.OpenPDF strPDF
End If
```

1.13 CreatePDFfromMultiPDF - Method to create an Adobe Acrobat (PDF) file from multiple PDF files.

Arguments:

strPDFFile As String	PDF file to create
varPDFFiles As Variant	PDF files to use (array)

Example:

```
Dim strPDF As String
Dim strPDF1 As String
Dim strPDF2 As String
Dim strPDF3 As String
```

```
Dim pdf(2) As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\multiple.pdf"

'Create PDF1
oPDFmaker.CreatePDFfromImageFile App.Path & "\image.pdf", App.Path & "\image1.jpg", "My PDF file", "Frank Kusluski", "Cute girl", "keywords go here"
strPDF1 = App.Path & "\image.pdf"

'Create PDF2
oPDFmaker.CreatePDFfromExcel App.Path & "\sample.pdf", App.Path & "\sample.xls", 1, "October", , , , , "Page &P", "right", "&D &T", "center", 1#, 0.5, 0#, 1#
strPDF2 = App.Path & "\sample.pdf"

'Create PDF3
oPDFmaker.CreatePDFfromWord App.Path & "\debug.pdf", App.Path & "\debug.doc", , , "1, 3, 6-8, 10"
strPDF3 = App.Path & "\debug.pdf"

'store PDF file names in one dimensional array - order is important!
pdf(0) = strPDF1
pdf(1) = strPDF2
pdf(2) = strPDF3

'finally combine all the PDF files into one PDF file!
oPDFmaker.CreatePDFfromMultiPDF strPDF, pdf

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
  oPDFmaker.OpenPDF strPDF
End If
```

1.14 CreatePDFfromMultiPDFinDir - Method to create an Adobe Acrobat (PDF) file from multiple PDF files in a specific directory.

Arguments:

strPDFFile As String	PDF file to create
strDirectory As String	Directory where PDF files exist
Optional strFiles As String = "*.pdf"	PDF files to use (can use wildcard characters)
Optional lngSortType As Long	Sort field (0=None,1=File Name,2=File Size,3=File Date)
Optional lngSortOrder As Long	Sort order (0=Ascending, 1=Descending)

Example:

```
Dim strPDF As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\all.pdf"

'Create the PDF!
oPDFmaker.CreatePDFfromMultiPDFinDir strPDF, App.Path, “f*.pdf”, 1, 0

Screen.MousePointer = vbDefault
```

```
If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
  oPDFmaker.OpenPDF strPDF
End If
```

1.15 CreatePDFfromMultiPS - Method to create an Adobe Acrobat (PDF) file from multiple PostScript (PS) files.

Arguments:

strPDFFile As String	PDF file to create
varPSFiles As Variant	Postscript files to use (stored in one dimensional array)
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional strJobOptionsFile As String	Print job options file.
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPDF As String
Dim strPS1 As String
Dim strPS2 As String
Dim strPS3 As String
Dim strPS4 As String
Dim strPpt As String
Dim strDoc As String
Dim strTxt As String
Dim strPagesToPrint As String
Dim strWB As String
Dim strRange As String

Dim ps(3) As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\multiple.pdf"

strPS1 = App.Path & "\file.ps"
strPpt = App.Path & "\file.ppt"

'create PostScript file from a PowerPoint file
oPDFmaker.CreatePSfromPowerPoint strPS1, strPpt

strPS2 = App.Path & "\sample.ps"
strWB = App.Path & "\sample.xls"
strRange = "A1:E276"

'create PostScript file from an Excel workbook file worksheet
oPDFmaker.CreatePSfromExcel strPS2, strWB, 1, strRange, , , , , "Page &P", "right", "&D &T", "center", 1.0, 0.5, 0.0, 1.0

strPS3 = App.Path & "\debug.ps"
strDoc = App.Path & "\debug.doc"
strPagesToPrint = "1, 3, 6-8, 10"

'create PostScript file from a Word document file
oPDFmaker.CreatePSfromWord strPS3, strDoc, , , strPagesToPrint

strPS4 = App.Path & "\file2.ps" 'note file name of 'file2.ps' - we already used 'file.ps' above
strTxt = App.Path & "\file.txt"
```

```
'create PostScript file from a text file
oPDFmaker.CreatePSfromTextFile strPS4, strTxt, False, "arial", False, 8, 255, False, False, False, True, "right", 20.5, 20.5, , 20.5

'store PostScript file names in one dimensional array - order is important!
ps(0) = strPS1
ps(1) = strPS2
ps(2) = strPS3
ps(3) = strPS4

'finally combine all the PostScript files into one PDF file!
oPDFmaker.CreatePDFfromMultiPS strPDF, ps

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.16 CreatePDFfromPowerPoint - Method to create an Adobe Acrobat (PDF) file from a PowerPoint file.

Requirements: Power Point

Arguments:

strPDFFile As String	PDF file to create
strPPDocument As String	PowerPoint file to use
Optional lngStartSlide As Long = -1	PowerPoint start slide to use
Optional lngEndSlide As Long = -1	PowerPoint end slide to use
Optional blnHorizontalOrientation As Boolean	PowerPoint slide orientation. Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional strJobOptionsFile As String	Print job options file.
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPDF As String
Dim strDoc As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\file.pdf"
strDoc = App.Path & "\file.ppt"

oPDFmaker.CreatePDFfromPowerPoint strPDF, strDoc

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.17 CreatePDFfromPowerPoint2 - Method to create an Adobe Acrobat (PDF) file from a PowerPoint file.

Requirements: Power Point and file PDFOSEND.EXE (see page 6 of this document)

Note: No postscript files created. Creates PDF faster than CreatePDFfromPowerPoint()

Arguments:

strPDFFile As String	PDF file to create
strPPDocument As String	PowerPoint file to use
Optional lngStartSlide As Long = -1	PowerPoint start slide to use
Optional lngEndSlide As Long = -1	PowerPoint end slide to use
Optional blnHorizontalOrientation As Boolean	PowerPoint slide orientation. Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strDoc As String
On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\file.pdf"
strDoc = App.Path & "\file.ppt"

oPDFmaker.CreatePDFfromPowerPoint2 strPDF, strDoc

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.18 CreatePDFfromPS - Method to create an Adobe Acrobat (PDF) file from a PostScript (PS) file.

Arguments:

strPDFFile As String	PDF file to create
strPSFile As String	Postscript file to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional strJobOptionsFile As String	Print job options file.
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPS As String
Dim strPDF As String
Dim strWB As String
Dim strRange As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPS = App.Path & "\sample.ps"
strPDF = App.Path & "\sample.pdf"
strWB = App.Path & "\sample.xls"
strRange = "A1:E276"

'create a PostScript file from Excel workbook worksheet range
```

```
oPDFmaker.CreatePSfromExcel strPS, strWB, 1, strRange, , , , , "Page &P", "right", "&D &T", "center", 1#, 0.5, 0#, 1#

'create a PDF file from a PostScript file
oPDFmaker.CreatePDFfromPS strPDF, strPS

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.19 CreatePDFfromTextFile - Method to create an Adobe Acrobat (PDF) file from a text file.

Requirements: Word

Arguments:

strPDFFile As String	PDF file to create
strTextFile As String	Text file to use
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional strFont As String	Font name. Default is “Courier”
Optional blnFontBold As Boolean	Font bold. Values: True,False
Optional intFontSize As Integer = 8	Font size
Optional lngFontColor As Long	Font color
Optional blnFontItalic As Boolean	Font italic. Values: True,False
Optional blnFontUnderline As Boolean	Font underline. Values: True,False
Optional blnPageNumberInFooter As Boolean	Place page number in footer. Values: True,False
Optional blnPageNumberInHeader As Boolean	Place page number in header. Values: True,False
Optional lngPageNumberAlignment As Long	Page number alignment. Values: 0=left, 1=center, 2=right
Optional sngTopMargin As Single	Page top margin
Optional sngLeftMargin As Single	Page left margin
Optional sngRightMargin As Single	Page right margin
Optional sngBottomMargin As Single	Page bottom margin
Optional blnFirstPage As Boolean = True	Add page number to first page? Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional strJobOptionsFile As String	Print job options file.
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPDF As String
Dim strTxt As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\file.pdf"
strTxt = App.Path & "\file.txt"

oPDFmaker.CreatePDFfromTextFile strPDF, strTxt, False, "arial", False, 8, 255, False, False, False, True, "right", 20.5, 20.5, , 20.5

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.20 CreatePDFfromTextFile2 - Method to create an Adobe Acrobat (PDF) file from a text file.

Requirements: Word and file PDFOSEND.EXE (see page 6 of this document)

Note: No postscript file created. Creates PDF faster than CreatePDFfromTextFile()

Arguments:

strPDFFile As String	PDF file to create
strTextFile As String	Text file to use
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional strFont As String	Font name. Default is “Courier”
Optional blnFontBold As Boolean	Font bold. Values: True,False
Optional intFontSize As Integer = 8	Font size
Optional lngFontColor As Long	Font color
Optional blnFontItalic As Boolean	Font italic. Values: True,False
Optional blnFontUnderline As Boolean	Font underline. Values: True,False
Optional blnPageNumberInFooter As Boolean	Place page number in footer. Values: True,False
Optional blnPageNumberInHeader As Boolean	Place page number in header. Values: True,False
Optional lngPageNumberAlignment As Long	Page number alignment. Values: 0=left, 1=center, 2=right
Optional sngTopMargin As Single	Page top margin
Optional sngLeftMargin As Single	Page left margin
Optional sngRightMargin As Single	Page right margin
Optional sngBottomMargin As Single	Page bottom margin
Optional blnFirstPage As Boolean = True	Add page number to first page? Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strTxt As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\file.pdf"
strTxt = App.Path & "\file.txt"

oPDFmaker.CreatePDFfromTextFile2 strPDF, strTxt, False, "arial", False, 8, 255, False, False, False, True, "right", 20.5, 20.5, , 20.5

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.21 CreatePDFfromTextFile3 - Method to create an Adobe Acrobat (PDF) file from a text file.

Requirements: Notepad and file PDFOSEND.EXE (see page 6 of this document)

No postscript file created. Very fast. Creates PDF in minimal time.

Arguments:

strPDFFile As String	PDF file to create
strTextfile As String	Text file to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
Dim strPDF As String
Dim strTxt As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\file.pdf"
strTxt = App.Path & "\file.txt"

oPDFmaker.CreatePDFfromTextFile3 strPDF, strTxt

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.22 CreatePDFfromVisio - Method to create an Adobe Acrobat (PDF) file from a Visio file.

Requirements: Visio

Arguments:

strPDFFile As String	PDF file to create
strVisioFile As String	Visio file to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF?

Example:

```
oPDFMaker.CreatePDFfromVisio “c:\file.pdf”, “c:\file.vsd”
```

1.23 CreatePDFfromWebPage - Method to create an Adobe Acrobat (PDF) file from a web page.

Requirements: Internet connection and file PDFOSEND.EXE (see page 6 of this document), Acrobat 6 and greater

Arguments:

strURL As String	Web address to use
strPDFFile As String	PDF file to create
Optional blnHideAcrobat As Boolean	Hide Acrobat? Values: True,False
Optional intSecsToWaitForPDF As Integer = 2	Seconds to wait for PDF to complete

Example:

```
Dim strURL As String
Dim strPDF As String
Dim i As Long

On Error Resume Next

strURL = "http://www.microsoft.com"
strPDF = App.Path & "\webpage.pdf"

'Use this awesome function to create a PDF from a web page!

PDFMaker1.CreatePDFfromWebPage strURL, strPDF, True
```

```
If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
  oPDFMaker1.OpenPDF strPDF
End If
```

1.24 CreatePDFfromWord - Method to create an Adobe Acrobat (PDF) file from a Word document.

Requirements: Word

Arguments:

strPDFFile As String	PDF file to create
strWordDocument As String	Word document to use
Optional strPassword As String	Word document password
Optional strWritePassword As String	Word document write password
Optional strPages As String	Word document pages to use
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional blnPageNumberInFooter As Boolean	Place page number in footer. Values: True,False
Optional blnPageNumberInHeader As Boolean	Place page number in header. Values: True,False
Optional lngPageNumberAlignment As Long	Page number alignment. Values: 0=left, 1=center, 2=right
Optional blnFirstPage As Boolean	Add page number to first page? Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False
Optional strJobOptionsFile As String	Print job options file.
Optional blnApplySecurity As Boolean	Apply Distiller security settings. Values: True,False

Example:

```
Dim strPDF As String
Dim strDoc As String
Dim strPagesToPrint As Variant

On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\debug.pdf"
strDoc = App.Path & "\debug.doc"
strPagesToPrint = "1, 3, 6-8, 10"

oPDFmaker.CreatePDFfromWord strPDF, strDoc, , , strPagesToPrint

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

1.25 CreatePDFfromWord2 - Method to create an Adobe Acrobat (PDF) file from a Word document.

Requirements: Word and file PDFOSEND.EXE (see page 6 of this document)

Note: No postscript file created. Creates PDF faster than CreatePDFfromWord()

Arguments:

strPDFFile As String	PDF file to create

strWordDocument As String	Word document to use
Optional strPassword As String	Word document password
Optional strWritePassword As String	Word document write password
Optional strPages As String	Word document pages to use
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional blnPageNumberInFooter As Boolean	Place page number in footer. Values: True,False
Optional blnPageNumberInHeader As Boolean	Place page number in header. Values: True,False
Optional lngPageNumberAlignment As Long	Page number alignment. Values: 0=left, 1=center, 2=right
Optional blnFirstPage As Boolean	Add page number to first page? Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat after creating PDF file?

Example:

```
Dim strPDF As String
Dim strDoc As String
Dim strPagesToPrint As Variant
On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\debug.pdf"
strDoc = App.Path & "\debug.doc"
strPagesToPrint = "1, 3, 6-8, 10"

oPDFmaker.CreatePDFfromWord2 strPDF, strDoc, , , strPagesToPrint

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
End If
```

2. PostScript Creation Methods (CreatePDF Class)

2.1 CreatePSfromAccessReport - Method to create a PostScript (PS) file from an Access report.

Requirements: Access

Arguments:

strPSFile As String	Postscript file to create
strDatabase As String	Access database to use
strReportName As String	Report name to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional strPassword As String	Access database password
Optional strFilter As String	Report filter to use
Optional strWhereCondition As String	Report where condition to use
Optional strSaveWindowCaption As String	Save window caption Default: "save pdf file as"

2.2 CreatePSfromAutoCAD - Method to create a PostScript (PS) file from an AutoCAD DWG/DXF file.

Requirements: AutoCAD

Arguments:

strPSFile As String	Postscript file to create
strDWGFile As String	DWG/DXF file to use
varLayout As String	Layout name/number to use
Optional strPDFWriter As String	Adobe Acrobat Print Driver. Default: "Acrobat PDFWriter"

Optional blnCloseAcrobat As Boolean	Close Acrobat? Values: True,False
Optional strSaveWindowCaption As String	Save window caption Default: "save pdf file as"

2.3 CreatePSfromCrystalReport - Method to create a PostScript (PS) file from a Crystel report.

Requirements: Crystal Reports

strPSFile As String	Postscript file to create
strCrystalReport As String	Crystal Report to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnCloseAcrobat As Boolean	Close Acrobat? Values: True,False
Optional strSaveWindowCaption As String	Save window caption Default: "save pdf file as"

2.4 CreatePSfromExcel - Method to create a PostScript (PS) file from an Excel worksheet.

Requirements: Excel

Arguments:

strPSFile As String	Postscript file to create
strExcelWorkbook As String	Excel workbook file to use
VarExcelWorksheet As Variant	Excel worksheet to use. Can be a name or number
strRange As String	Excel worksheet range to use
Optional strPassword As String	Excel workbook password
Optional strWriteResPassword As String	Excel workbook write reserve password
Optional blnUseGrid As Boolean	Use grid lines. Values: True,False
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional strHeader As String	Page header
Optional lngHeaderAlignment As Long	Page header alignment. Values: 0=left, 1=center, 2=right
Optional strFooter As String	Page footer
Optional lngFooterAlignment As Long	Page footer alignment. Values: 0=left, 1=center, 2=right
Optional dblTopMargin As Double	Page top margin in inches
Optional dblLeftMargin As Double	Page left margin in inches
Optional dblRightMargin As Double	Page right margin in inches
Optional dblBottomMargin As Double	Page bottom margin in inches
Optional dblHeaderMargin As Double	Page header margin in inches
Optional dblFooterMargin As Double	Page footer margin in inches
Optional lngFirstPageNumber As Long = 1	Beginning page number
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False

Example:

```
Dim strPS As String
Dim strWB As String
Dim strRange As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPS = App.Path & "\sample.ps"
strWB = App.Path & "\sample.xls"
strRange = "A1:E276"

oPDFmaker.CreatePSfromExcel strPS, strWB, 1, strRange, , , , , "Page &P", "right", "&D &T", "center", 1#, 0.5, 0#, 1#

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  MsgBox "PostScript file created.", vbInformation, "Message"
```

```
End If
```

2.5 CreatePSfromMiscFile - Method to create a PostScript (PS) file from a miscellaneous file type.

Requirements: Word

strPSFile As String	Postscript file to create
strMiscFile As String	File to use
Optional strDistiller As String	Adobe Acrobat Distiller Name

2.6 CreatePSfromPowerPoint - Method to create a PostScript (PS) file from a PowerPoint file.

Requirements: Power Point

Arguments:

strPSFile As String	Postscript file to create
strPPDocument As String	PowerPoint file to use
Optional lngStartSlide As Long = -1	PowerPoint start slide to use
Optional lngEndSlide As Long = -1	PowerPoint end slide to use
Optional blnHorizontalOrientation As Boolean	PowerPoint slide orientation. Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False

Example:

```
Dim strPS As String
Dim strDoc As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPS = App.Path & "\file.ps"
strDoc = App.Path & "\file.ppt"

oPDFmaker.CreatePSfromPowerPoint strPS, strDoc

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  MsgBox "PostScript file created.", vbInformation, "Message"
End If
```

2.7 CreatePSfromTextFile - Method to create a PostScript (PS) file from a text file.

Requirements: Word

Arguments:

strPSFile As String	Postscript file to create
strTextFile As String	Text file to use
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional strFont As String	Font name. Default is “Courier”
Optional blnFontBold As Boolean	Font bold. Values: True,False
Optional intFontSize As Integer = 8	Font size
Optional lngFontColor As Long	Font color
Optional blnFontItalic As Boolean	Font italic. Values: True,False
Optional blnFontUnderline As Boolean	Font underline. Values: True,False
Optional blnPageNumberInFooter As Boolean	Place page number in footer. Values: True,False

Optional blnPageNumberInHeader As Boolean	Place page number in header. Values: True,False
Optional lngPageNumberAlignment As Long	Page number alignment. Values: 0=left, 1=center, 2=right
Optional sngTopMargin As Single	Page top margin
Optional sngLeftMargin As Single	Page left margin
Optional sngRightMargin As Single	Page right margin
Optional sngBottomMargin As Single	Page bottom margin
Optional blnFirstPage As Boolean = True	Add page number to first page? Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False

Example:

```
Dim strPS As String
Dim strTxt As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPS = App.Path & "\file.ps"
strTxt = App.Path & "\file.txt"

oPDFmaker.CreatePSfromTextFile strPS, strTxt, False, "arial", False, 8, 255, False, False, False, True, "right", 20.5, 20.5, , 20.5

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  MsgBox "PostScript file created.", vbInformation, "Message"
End If
```

2.8 CreatePSfromTextFile2 - Method to create a PostScript (PS) file from a text file.

Requirements: Word, Notepad

Arguments:

strPSFile As String	Postscript file to create
strTextFile As String	Text file to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional strSaveWindowCaption As String	Save window caption Default: "save pdf file as"

2.9 CreatePSfromTextFile3 - Method to create a PostScript (PS) file from a text file.

Arguments:

strPSFile As String	Postscript file to create
strTextFile As String	Text file to use
Optional intRightMargin As Integer	Right margin
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional strSaveWindowCaption As String	Save window caption Default: "save pdf file as"

2.10 CreatePSfromVisio - Method to create a PostScript (PS) file from aVisio file.

Requirements: Visio

Arguments:

strPSFile As String	Postscript file to create
strVisioFile As String	Visio file to use
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional strSaveWindowCaption As String	Save window caption Default: "save pdf file as"

2.11 CreatePSfromWord - Method to create a PostScript (PS) file from a Word document.

Requirements: Word

Arguments:

strPSFile As String	Postscript file to create
strWordDocument As String	Word document to use
Optional strPassword As String	Word document password
Optional strWritePassword As String	Word document write password
Optional strPages As String	Word document pages to use
Optional blnLandscape As Boolean	Page orientation. Values: True,False
Optional blnPageNumberInFooter As Boolean	Place page number in footer. Values: True,False
Optional blnPageNumberInHeader As Boolean	Place page number in header. Values: True,False
Optional lngPageNumberAlignment As Long	Page number alignment. Values: 0=left, 1=center, 2=right
Optional blnFirstPage As Boolean = True	Add page number to first page? Values: True,False
Optional strDistiller As String	Adobe Acrobat Distiller Name
Optional blnShowWindow As Boolean	Show job process window. Values: True,False
Optional blnSpoolJobs As Boolean	Spool print jobs. Values: True,False

Example:

```
Dim strPS As String
Dim strDoc As String
Dim strPagesToPrint As String

On Error Resume Next

Screen.MousePointer = vbHourglass

strPS = App.Path & "\debug.ps"
strDoc = App.Path & "\debug.doc"
strPagesToPrint = "1, 3, 6-8, 10"

oPDFmaker.CreatePSfromWord strPS, strDoc, , , strPagesToPrint

Screen.MousePointer = vbDefault

If Err.Number Then
   MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
   MsgBox "PostScript file created.", vbInformation, "Message"
End If
```

3. Document Methods/Properties (Document Class)

3.1 AcrobatFormsVersion - Property (string) to get the version number of the forms software running inside the viewer.

3.2 AcrobatLanguage - Property (string) to get the language of the running Acrobat Viewer.

3.3 AcrobatPlatform - Property (string) to get the platform that the script is currently executing on. Values: WIN, MAC, UNIX.

3.4 AcrobatViewerType - Property (string) to get the viewer that the application is running.

3.5 AcrobatViewerVariation - Property (string) to get the packaging of the running Acrobat Viewer.

3.6 AcrobatViewerVersion - Property (string) to get the version number of the current viewer.

3.7 AddScript - Method to add a JavaScript to a document. The script executes when the document is opened.

Arguments:

strName As String	Script name
strScript As String	JavaScript code to execute

3.8 AddThumbnails - Method to add thumbnails to an Adobe Acrobat (PDF) file. See also method RemoveThumbnails.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

3.9 AddWatermarkFromFile - Method to add a watermark to an Adobe Acrobat (PDF) file from a PDF file.

Requirements: Acrobat 7 and greater.

Arguments:

strSourcePDF As String	PDF file to use as watermark
Optional lngSourcePage As Long	Page in PDF file to use
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnOnTop As Boolean = True	Watermark on top?
Optional blnOnScreen As Boolean = True	Display on screen?
Optional blnOnPrint As Boolean = True	Print?
Optional sngScale As Single = 1.0	Size
Optional sngOpacity As Single = 1.0	Transparency level

3.10 AddWatermarkFromText - Method to add a watermark to an Adobe Acrobat (PDF) file from a text string.

Requirements: Acrobat 7 and greater.

Arguments:

strText As String	Text to use as watermark
Optional lngAlignment As Long	Text alignment
Optional lngFont As Long	Text font
Optional intTextSize As Integer = 24	Text font size
Optional varColor As Variant	Text color
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnOnTop As Boolean = True	Watermark on top?
Optional blnOnScreen As Boolean = True	Display on screen?
Optional blnOnPrint As Boolean = True	Print?
Optional sngScale As Single = 1.0	Size
Optional sngOpacity As Single = 1.0	Transparency level

3.11 AddWebLinks - Method to add web links to an Adobe Acrobat (PDF) file. See also method RemoveWeblinks.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

3.12 AttachFile - Method to embed an external file into the document.

Arguments:

strName As String	Internal file name
strFile As String	File name to attach

3.13 Author - Property (string) to get/set the author of the document.

3.14 BaseURL - Property (string) to get/set the base URL to resolve relative web links within the document.

3.15 BookmarkExists - Method to determine if a bookmark exists within the document.

3.16 Calculate - Property (Boolean) to get/set calculate property. If true, allows calculations to be performed for this document. If false, prevents all calculations from happening for this document.

3.17 CalculateNow - Method to force computation of all calculation fields in the current document.

3.18 CloseDoc - Method to close the current document without saving changes. See method Save for saving changes.

3.19 CreateBookmarksFromArray - Method to Create bookmarks in the current document from a two dimensional array.

Arguments:

varArray As Variant	Array of bookmark names
Optional blnExpand As Boolean	Expand bookmarks?

Returns: Long Integer (total number of bookmarks created)

3.20 CreateBookmarksFromArray2 - Method to create bookmarks in an Adobe Acrobat (PDF) file from an array.

Arguments:

varArray As Variant	Array of bookmark names
Optional blnExpand As Boolean	Expand bookmarks?

Returns: Long Integer (total number of bookmarks created)

3.21 CreateBookmarksFromArray3 - Method to create bookmarks in an Adobe Acrobat (PDF) file from an array.

Arguments:

varArray As Variant	Array of bookmark names
Optional blnExpand As Boolean	Expand bookmarks?

Returns: Long Integer (total number of bookmarks created)

3.22 CreateBookmarksFromOutline - Method to create bookmarks in an Adobe Acrobat (PDF) file from an outline.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

Returns: Long Integer (total number of bookmarks created)

3.23 CreateBookmarksFromPages - Method to create bookmarks in an Adobe Acrobat (PDF) file from pages.

Arguments:

Optional strText As String = "Page"	Bookmark partial name

Returns: Long Integer (total number of bookmarks created)

3.24 CreateButtonField - Method to create a button field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
Optional strText As String	Button caption
Optional lngBorder As Long	Button border type

Optional intButtonAlignX As Integer = 50	Buton X alignment
Optional intButtonAlignY As Integer = 50	Button Y alignment
Optional blnButtonFitBounds As Boolean	Button fit bounds?
Optional lngFieldDisplay As Long	Field display type
Optional varFillColor As Variant	Field background color (array)
Optional intLineWidth As Long	Field line width
Optional blnReadOnly As Boolean	Read only?
Optional varColor As Variant	Field color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional lngFont As Long	Text font
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270

3.25 CreateCheckBoxField - Method to create a check box field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
varExportValues As Variant	Field export values (array)
Optional lngBorder As Long	Field border type
Optional strDefaultValue As String	Field default value
Optional lngFieldDisplay As Long	Field display type
Optional varFillColor As Variant	Field background color (array)
Optional intLineWidth As Long	Field line width type
Optional blnReadOnly As Boolean	Read only?
Optional blnRequired As Boolean	Required?
Optional varColor As Variant	Field color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional VarValue As Variant	Field value
Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270
Optional lngGlyphStyle As Long	Field glyph style

3.26 CreateChildBookmark - Method to create child bookmarks in an Adobe Acrobat (PDF) file.

Arguments:

strChildBookmark As String	Child bookmark name to create
Optional strBookmark As String	Bookmark to search for
Optional lngPageNumber As Long	Page number
Optional intIndex As Integer	Bookmark index
Optional varColor As Variant	Bookmark color
Optional ByVal lngStyle As Long	Bookmark style
Optional blnCaseSensitive As Boolean	Case sensitive search?
Optional blnWholeWord As Boolean = True	Whole word search?
Optional intX As Integer	X scroll number
Optional intY As Integer	Y scroll number
Optional blnExpand As Boolean	Expand bookmark?

Returns: Integer (1 if child bookmark was ceated)

3.27 CreateComboBoxField - Method to create a combo box field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
varValues As Variant	Field values (array)
Optional lngBorder As Long	Field border type
Optional blnCommitOnSelChange As Boolean	Commit on selection change?
Optional intCalcOrderIndex As Integer	Calculation order index
Optional strDefaultValue As String	Field default value
Optional blnEditable As Boolean	Editable?
Optional lngFieldDisplay As Long	Field display type
Optional blnDoNotSpellCheck As Boolean	Do not spell check?
Optional varFillColor As Variant	Field background color (array)
Optional intLineWidth As Long	Field line width type
Optional blnReadOnly As Boolean	Read only?
Optional blnRequired As Boolean	Required?
Optional varColor As Variant	Field color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional lngFont As Long	Text font
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional varValue As Variant	Field value
Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270
Optional varExportValues As Variant	Field export values (array)

3.28 CreateComment - Method to create a comment in an Adobe Acrobat (PDF) file.

Arguments:

lngPage As Long	Page number
strComment As String	Comment
Optional varColor As Variant	Color (array)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional intLeftMargin As Integer	Left margin
Optional intTopMargin As Integer	Top margin
Optional lngNoteIcon As Long	Note icon
Optional strName As String	Annotation name
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional blnPopupOpen As Boolean	Popup on open?
Optional lngNoteState As Long	Note status
Optional intBorderWidth As Integer = 1	Border width
Optional sngOpacity As Single = 1.0	Opacity level (values: 0.00 to 1.00)

3.29 CreateFreeTextAnnot - Method to create a FreeText annotation in an Adobe Acrobat (PDF) file.

Arguments:

lngPage As Long	Page number
varRectangle As Variant	Annotation rectangle coordinates
strText As String	Annotation text
Optional varColor As Variant	Annotation foreground color (array)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional strName As String	Annotation name
Optional blnLock As Boolean	Lock?
Optional blnReadOnly As Boolean	Read only?
Optional lngAlignment As Long	Annotation alignment type

Optional varFillColor As Variant	Annotation background color (array)
Optional lngFont As Long	Text font name
Optional intFontSize As Integer = 8	Text size
Optional intBorderWidth As Integer = 1	Border width
Optional lngRotation As Long	Annotation rotation. Values: 0, 90, 180, 270
Optional sngOpacity As Single = 1.0	Opacity level (values: 0.00 to 1.00)

3.30 CreateJavaScriptObject - Method to create a JavaScript object from a Document object. Once created all the documents properties and methods are available.

Returns: Object (JavaScript object)

3.31 CreateListBoxField - Method to create a list box field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
varValues As Variant	Field values (array)
Optional lngBorder As Long	Field border type
Optional blnCommitOnSelChange As Boolean	Commit on selection change?
Optional strDefaultValue As String	Field default value
Optional lngFieldDisplay As Long	Field display type
Optional varFillColor As Variant	Field background color (array)
Optional blnMultipleSelection As Boolean	Multiple selection?
Optional intLineWidth As Long	Field line width type
Optional blnReadOnly As Boolean	Read only?
Optional blnRequired As Boolean	Required?
Optional varColor As Variant	Field color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional lngFont As Long	Text font
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional varValue As Variant	Field value
Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270
Optional varExportValues As Variant	Field export values (array)

3.32 CreateRadioButtonField - Method to create a radio button field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
varExportValues As Variant	Field export values (array)
Optional lngBorder As Long	Field border type
Optional strDefaultValue As String	Field default value
Optional lngFieldDisplay As Long	Field display type
Optional varFillColor As Variant	Field background color (array)
Optional intLineWidth As Long	Field line width type
Optional blnReadOnly As Boolean	Read only?
Optional blnRequired As Boolean	Required?
Optional varColor As Variant	Field color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional VarValue As Variant	Field value
Optional blnRadiosInUnison As Boolean	Radio buttons in unison?

Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270
Optional lngGlyphStyle As Long	Field glyph style

3.33 CreateSignatureField - Method to create a signature field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
Optional lngBorder As Long	Field border type
Optional lngFieldDisplay As Long	Field display type
Optional varFillColor As Variant	Field background color (array)
Optional intLineWidth As Long	Field line width type
Optional blnReadOnly As Boolean	Read only?
Optional blnRequired As Boolean	Required?
Optional varColor As Variant	Field color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional VarValue As Variant	Field value
Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270

3.34 CreateTemplate - Method to create a template from a page in an Adobe Acrobat (PDF) file.

Arguments:

Optional lngPage As Long	Page number

3.35 CreateTextAnnot - Method to create a Text annotation in an Adobe Acrobat (PDF) file.

Arguments:

lngPage As Long	Page number
strText As String	Text
Optional varColor As Variant	Color (array)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional intLeftMargin As Integer	Left margin
Optional intTopMargin As Integer	Top margin
Optional lngNoteIcon As Long	Note icon
Optional strName As String	Annotation name
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional blnPopupOpen As Boolean	Popup on open?
Optional lngNoteState As Long	Note status
Optional intBorderWidth As Integer = 1	Border width
Optional sngOpacity As Single = 1.0	Opacity level (values: 0.00 to 1.00)

3.36 CreateTextField - Method to create a text field in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Field name
lngPage As Long	Page number
varCoords As Variant	Field location coordinates
Optional lngAlignment As Long	Field alignment type
Optional lngBorder As Long	Field border type

Optional intCharLimit As Integer	Maximum number of characters allowed in field
Optional blnComb As Boolean	Display box for each character?
Optional intCalcOrderIndex As Integer	Calculation order index
Optional strDefaultValue As String	Field default value
Optional lngFieldDisplay As Long	Field display type
Optional blnDoNotSpellCheck As Boolean	Do not spell check?
Optional blnFileSelect As Boolean	Show file selection dialog?
Optional varFillColor As Variant	Field background color (array)
Optional intLineWidth As Long	Field line width type
Optional blnMultiLine As Boolean	Multiple lines?
Optional blnPassword As Boolean	Password field?
Optional blnReadOnly As Boolean	Read only?
Optional blnRequired As Boolean	Required?
Optional varBorderColor As Variant	Field border color (array)
Optional strSubmitName As String	Field submit name
Optional varTextColor As Variant	Field foreground color (array)
Optional lngFont As Long	Text font name
Optional intTextSize As Integer = 8	Text size
Optional strToolTipText As String	Tool tip text
Optional VarValue As Variant	Field value
Optional lngTrigger As Long	Field trigger type
Optional strAction As String	JavaScript code to execute
Optional lngRotation As Long	Field rotation. Values: 0, 90, 180, 270

3.37 CreationDate - Property (date) to get the document creation date.

3.38 Creator - Property (string) to get document creator.

3.39 DataObjects - Property (variant) to get an array containing all the named data objects in the document.

3.40 DeletePages() – Method to delete pages in an Adobe Acrobat (PDF) file.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

3.41 DeleteSound - Method to delete sound objects in an Adobe Acrobat (PDF) file.

Arguments:

strName As String	Sound name to delete

3.42 Dirty - Property (boolean) to get/set whether the document has been dirtied as the result of a changes to the document, and therefore needs to be saved.

3.43 Disclosed - Property (boolean) to get/set whether the document should be accessible to JavaScripts in other documents.

3.44 DocumentFilename - Property (string) to get the base filename with extension of the document referenced by the doc object. The device-independent path is not returned.

3.45 DynamicXFAForm - Property (boolean) returns true if the document is XFA, and it is dynamic . Returns false otherwise. A dynamic XFA form is one in which some of the fields can grow or shrink in size to accomodate the values they contain.

3.46 EmailDocument - Method to email the current document.

Arguments:

Optional blnShowWindow As Boolean = True	Show user interface?
Optional strTo As String	Message recipients
Optional strCC As String	Message carbon copy recipients
Optional strBCC As String	Message blind copy recipients
Optional strSubject As String	Message subject

Optional strBody As String	Message body

3.47 EmailForm - Method to email the current document's form fields as an FDF file.

Arguments:

Optional blnShowWindow As Boolean = True	Show user interface?
Optional strTo As String	Message recipients
Optional strCC As String	Message carbon copy recipients
Optional strBCC As String	Message blind copy recipients
Optional strSubject As String	Message subject
Optional strBody As String	Message body

3.48 ExecuteBookmark - Method to execute a bookmark in an Adobe Acrobat (PDF) file.

Arguments:

strBookmark As String	Bookmark name to execute
Optional blnCaseSensitive As Boolean	Case sensitive search?
Optional blnWholeWord As Boolean = True	Whole word search?

Returns: Boolean (True if successful, False if not successful)

3.49 ExportAsFDF - Method to export form fields to a FDF file.

Arguments:

Optional blnAllFields As Boolean	Export all fields?
Optional blnNoPassword As Boolean = True	Include text fields with password flag set to True?
Optional varFields As Variant	Array or string of fields to submit
Optional blnFlags As Boolean	Include field flags?
Optional strPath As String	FDF file path
Optional blnAnnotations As Boolean	Include annotations? (Acrobat version 6 and greater)

3.50 ExportAsText - Method to export form fields to a tab-delimited text file.

Arguments:

Optional blnNoPassword As Boolean = True	Include text fields with password flag set to True?
Optional varFields As Variant	Array or string of fields to submit
Optional strPath As String	Text file path

3.51 ExportAsXFDF - Method to export form fields to a XFDF file.

Arguments:

Optional blnAllFields As Boolean	Export all fields?
Optional blnNoPassword As Boolean = True	Include text fields with password flag set to True?
Optional varFields As Variant	Array or string of fields to submit
Optional blnFlags As Boolean	Include field flags?
Optional strPath As String	XFDF file path
Optional blnAnnotations As Boolean	Include annotations? (Acrobat version 6 and greater)

3.52 ExportDataObject - Method to export a specified document object to an external file.

Arguments:

strName As String	Data object name
Optional strPath As String	File path
Optional blnAllowAuthorization As Boolean	Show authorization dialog? (Acrobat version 6 and greater)

3.53 ExportXFAData - Method to export a document's XFA data file to an external file.

Requirements: Acrobat 6 and greater

Arguments:

Optional strPath As String	File path
Optional blnXDP As Boolean = True	Export in XDP format?

3.54 ExtractPages - Method to extract pages from a document and create a new Adobe Acrobat (PDF) file.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional strPath As String	File path

3.55 FileSize - Property (Long Integer) to get the file size of the document in bytes.

3.56 FindAndCommentText - Method to find specified text in an Adobe Acrobat (PDF) file and add a comment.

Arguments:

strFindText As String	Text to find
strText As String	Comment text
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnCaseSensitive As Boolean	Case sensitive?
Optional lngNoteIcon As Long	Note icon
Optional varColor As Variant	Color (array)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional intBorderWidth As Integer = 1	Border width
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional lngAlignment As Long	Alignment
Optional lngNoteState As Long	Note status
Optional blnTransparent As Boolean = True	Transparent?

Returns: Long (# of annotations created)

3.57 FindAndHighlightText - Method to find specified text in an Adobe Acrobat (PDF) file and highlight.

Arguments:

strFindText As String	Text to find
strText As String	Comment text
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnCaseSensitive As Boolean	Case sensitive?
Optional varHighlightColor As Variant	Highlight color (yellow is default)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional blnTransparent As Boolean	Transparent?
Optional strContents As String	
Optional blnHidden As Boolean	
Optional strName As String	
Optional blnPopupOpen As Boolean	
Optional blnNoView As Boolean	
Optional blnToggleNoView As Boolean	

Returns: Long (# of annotations created)

3.58 FindAndOverwriteText - Method to find specified text in an Adobe Acrobat (PDF) file and replace with free text annotation.

Arguments:

strFindText As String	Text to find
strReplaceText As String	Replacement text
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnCaseSensitive As Boolean	Case sensitive?
Optional lngFont As Long	Text font
Optional intWidth As Integer	Replacement text width
Optional varColor As Variant	Foreground color (array)
Optional varFillColor As Variant	Background color (array)
Optional blnBorder As Boolean	Border?
Optional intY As Integer	Y position

Returns: Long (# of annotations created)

3.59 FindAndSquigglyText - Method to find specified text in an Adobe Acrobat (PDF) file and add a squiggly line below it.

Arguments:

strFindText As String	Text to find
strText As String	Comment text
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnCaseSensitive As Boolean	Case sensitive?
Optional varSquigglyColor As Variant	Squiggly color (red is default)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional blnTransparent As Boolean	Transparent?
Optional strContents As String	
Optional blnHidden As Boolean	
Optional strName As String	
Optional blnPopupOpen As Boolean	
Optional blnNoView As Boolean	
Optional blnToggleNoView As Boolean	

Returns: Long (# of annotations created)

3.60 FindAndStrikeoutText - Method to find specified text in an Adobe Acrobat (PDF) file and strikeit out.

Arguments:

strFindText As String	Text to find
strText As String	Comment text
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnCaseSensitive As Boolean	Case sensitive?
Optional varStrikeoutColor As Variant	Strikeout color (red is default)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional blnTransparent As Boolean	Transparent?
Optional strContents As String	
Optional blnHidden As Boolean	
Optional strName As String	
Optional blnPopupOpen As Boolean	
Optional blnNoView As Boolean	
Optional blnToggleNoView As Boolean	

Returns: Long (# of annotations created)

3.61 FindAndUnderlineText - Method to find specified text in an Adobe Acrobat (PDF) file and underline it.

Arguments:

strFindText As String	Text to find
strText As String	Comment text
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional blnCaseSensitive As Boolean	Case sensitive?
Optional varUnderlineColor As Variant	Underline color (red is default)
Optional strAuthor As String	Author
Optional strSubject As String	Subject
Optional blnLock As Boolean	Locked?
Optional blnReadOnly As Boolean	Read only?
Optional blnTransparent As Boolean	Transparent?
Optional strContents As String	
Optional blnHidden As Boolean	
Optional strName As String	
Optional blnPopupOpen As Boolean	
Optional blnNoView As Boolean	
Optional blnToggleNoView As Boolean	

Returns: Long (# of annotations created)

3.62 FlattenPages - Method to convert all annotations within a document to page contents.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

3.63 GetAlignment - Method to get a valid alignment string (left, center, right).

Arguments:

i As Long	Alignment number

Returns: String (alignment)

3.64 GetAnnotations - Method to populate a one dimensional array with annot names within a document.

Arguments:

varAnnots As Variant	Array to store annotation names
Optional lngPage As Long	Page number

Returns: Long (# of annotations)

3.65 GetBookmarks - Method to retrieve all bookmarks in an Adobe Acrobat (PDF) file and store in an array.

Arguments:

varBookmarks As Variant	Array to store bookmark names

Returns: Long (total number of bookmarks)

3.66 GetDocumentTrigger - Method to get a valid document trigger string.

Arguments:

i As Long	Document trigger number

Returns: valid document trigger string (WillClose, WillSave, DidSave, WillPrint, DidPrint).

3.67 GetField - Method to instantiate the Field Object. Used to perform various operations on form fields.

Arguments:

strName As String	Field name

Returns: Field Object

3.68 GetFieldBorder - Method to get a valid field border string.

Arguments:

i As Long	Field border number

Returns: String (solid, beveled, dashed, inset, underline)

3.69 GetFields - Method that returns a collection object containing all the fields in the document.

Returns: Collection Object of field names

3.70 GetFieldsArray - Method that returns an array containing all the fields in the document.

Returns: a one dimensional array of field names

3.71 GetFileType - Method to get a valid file type string.

Arguments:

i As Long	File type number

Returns: String (File type)

3.72 GetFont - Method to get a valid font string.

Arguments:

i As Long	Font number

Returns: String (Font)

3.73 GetFreeTextAnnotProperties - Method to get the properties of a free text annotation within an Adobe Acrobat (PDF) document. Note: Use method SetFreeTextAnnotProperties to set properties.

Arguments:

strName As String	Annotation name
lngPage As Long	Page number

Returns: FreeTextAnnotType

3.74 GetGlyphStyle - Method to get a valid glyph style string for a check box or radio button field.

Arguments:

i As Long	Field glyph style number

Returns: String (check, cross, diamond, circle, star, square)

3.75 GetLayout - Method to get the document layout string.

Arguments:

i As Long — Layout number

Returns: String (SinglePage, OneColumn, TwoColumnLeft, TwoColumnRight, TwoPageLeft, TwoPageRight)

3.76 GetNoteIcon - Method to get a valid note icon string.

Arguments:

i As Long — Note icon number

Returns: String (valid note icon)

3.77 GetNoteState - Method to get a valid note status string.

Arguments:

i As Long — Note state number

Returns: String (valid note state)

3.78 GetNthFieldName - Method to get the nth field name in the document.

Arguments:

intIndex As Integer

Returns: String (the field name)

3.79 GetNthTemplate - Method to get the nth template in the document.

Arguments:

intIndex As Integer

Returns: String (the template name)

3.80 GetPageLabel - Method to get the page label information for the specified page.

Arguments:

Optional lngPage As Long

Returns: String

3.81 GetPageNthWord - Method to get the nth word on the page.

Arguments:

Optional lngPage As Long
Optional intWordIndex As Integer
Optional blnStrip As Boolean — Strip whitespace and punctuation?

Returns: String

3.82 GetPageNthWordQuads - Method to get the quads list for the nth word on the page.

Arguments:

Optional lngPage As Long
Optional intWordIndex As Integer

Returns: Variant (array of quadrant values)

3.83 GetPageRotation - Method to get the rotation of the specified page.

Arguments:

Optional lngPage As Long — Page number

Returns: Integer

3.84 GetPagesWordCount - Method to get the total number of words in a document/page.

Arguments:

Optional lngPage As Long
Optional varPages As Variant

Returns:

Populates varPages array with total # words, returns total words (Long Integer)

3.85 GetPageTrigger - Method to get the page trigger string.

i As Long — Trigger number

Returns: String (Open, Close)

3.86 GetStampIcon - Method to get the stamp icon string.

i As Long — Stamp icon number

Returns: String

3.87 GetTextAnnotProperties - Method to get the properties of a text annotation within an Adobe Acrobat (PDF) document. Note: Use method SetTextAnnotProperties to set properties.

Arguments:

strName As String — Annotation name
lngPage As Long

Returns: TextAnnotType

3.88 GetTextCoordinates - Method to get the coordinates of text.

Arguments:

strFindText As String
Optional lngStartPage As Long
Optional lngEndPage As Long
Optional blnCaseSensitive As Boolean

Returns: Variant (array of coordinates)

3.89 GetTrigger - Method to get a valid trigger string.

Arguments:

i As Long — Field trigger number

Returns: String (valid field trigger string)

3.90 GetURL – Method to get the specified URL over the internet using a GET and add to the document.

Arguments:

strURL As String
Optional blnAppend As Boolean = True

3.91 Icons – Property (variant) to get an array of document icons.

3.92 ImportAnFDF - Method to import an FDF file.

Arguments:

Optional strPath As String	File path

3.93 ImportAnXFDF - Method to import an XFDF file.

Arguments:

Optional strPath As String	File path

3.94 ImportTextData - Method that imports a row of data from a text file. Each row must be tab delimited.
Arguments:

Optional strPath As String	File path
Optional lngRow As Long	Row to use

3.95 ImportXFAData - Method to import XFA data.

Arguments:

Optional strPath As String	File path

3.96 InsertPages - Method to insert pages into an Adobe Acrobat (PDF) file.

Arguments:

strSourcePDF As String	Source PDF file to use
Optional lngInsertAfterPage As Long = -1	Page to insert after
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

3.97 InstanceID – Property (string) to get the document instance ID.

3.98 Keywords – Property (string) to get/set the document's keywords.

3.99 MetaData – Property (string) to get/set the XMP metadata embedded in a PDF document. Returns a string containing the XML text stored as metadata in a particular PDF document.

3.100 ModDate – Property (date) to get the date the document was last modified.

3.101 MovePage - Method to move a page in an Adobe Acrobat (PDF) file.

Arguments:

Optional lngPage As Long	Page number to move
Optional lngMoveAfterPage As Long = -1	Move after page

3.102 NewPage - Method to insert a new blank page into an Adobe Acrobat (PDF) file.

Arguments:

Optional lngInsertAfterPage As Long	Insert after page

3.103 NoAutoComplete – Property (boolean) to get/set auto-complete feature of forms.

3.104 NoCache – Property (boolean) to get/set forms data caching feature.

3.105 NumFields – Property (long integer) to get the total number of form fields in the document.

3.106 NumPages – Property (long integer) to get the total number of pages in the document.

3.107 NumTemplates – Property (long integer) to get the total number of templates in the document.

3.108 PageMode – Property (string) to get the page mode of the document.

3.109 Path – Property (string) to get the device-independent path of the document.

3.110 PermanentID – Property (string) to get the permanent ID of the document.

3.111 PermStatusReady – Property (string) to get the permissions ready status for the document. This can return false if the document is not available, for example when downloading over a network connection, and permissions are determined based on a signature that covers the entire document.

3.112 PrintDocument - Method to send the document to a printer.

Arguments:

Optional blnShowWindow As Boolean = True	Display user interface window
Optional lngStartPage As Long	Start page number to print
Optional lngEndPage As Long	End page number to print
Optional blnSilent As Boolean	Print silently?
Optional blnShrinkToFit As Boolean	Shrink to fit paper?
Optional blnPrintAsImage As Boolean	Print as image?
Optional blnReverse As Boolean	Reverse pages?
Optional blnAnnotations As Boolean = True	Print annotations?

3.113 Producer – Property (string) to get the producer of the document (for example, "Acrobat Distiller", "PDFWriter", etc).

3.114 RemoveAnnotations - Method to remove annotations from an Adobe Acrobat (PDF) file.

Arguments:

Optional lngPage As Long	Page number

Returns: Long (# of deleted annotations)

3.115 RemoveBookmark – Method to remove bookmarks in document.

Arguments:

Optional strBookmark As String
Optional blnCaseSensitive As Boolean
Optional blnWholeWord As Boolean = True
Optional blnSearchAll As Boolean

Returns: Long Integer (# of bookmarks removed)

3.116 RemoveDataObject – Method to remove data objects in document.

Arguments:

strName As String	Data object name to remove

3.117 RemoveField - Method to remove a field from a document.

Arguments:

strName As String	Field to remove

3.118 RemoveIcon - Method to remove an icon from a document.

Arguments:

strName As String	Icon to remove

3.119 RemoveScript - Method to remove a script from a document.

Arguments:

strName As String	Script to remove

3.120 RemoveTemplate - Method to remove a template from a document.

Arguments:

strName As String	Template to remove

3.121 RemoveThumbnails - Method to remove thumbnails from a document. See also method AddThumbnails.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

Returns: Number of deleted thumbnails (Long Integer)

3.122 RemoveWebLinks - Method to remove web links in a document. See also method AddWeblinks.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

Returns: Number of deleted web links (Long Integer)

3.123 ReplacePages - Method to replace pages in a document.

Arguments:

strSourcePDF As String	Source PDF file to use
Optional lngReplaceStartPage As Long	Replace start page number
Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number

3.124 ResetFormFields - Method to reset form fields to their default values.

Arguments:

Optional varFields As Variant	Array of fields to reset

3.125 Save - Method to save the current document.

Arguments:

ByVal lngSave As SaveType	Save changes only or full document
Optional strPDF As String	Optional path

3.126 SaveAs - Method to save document as a PDF or alternate file type including DOC, TXT, EPS, HTML, JPEG, XML, RTF, XDP, etc.

Arguments:

strPath As String	File path
Optional lngFileType As Long	File type

3.127 SecurityHandler – Property (string) to get the name of the security handler used to encrypt the document.

3.128 SetBookmarkAction - Method to set a bookmark's action in a document.

Arguments:

strAction As String	JavaScript code to execute
Optional strBookmark As String	Bookmark name to search for. If blank set action for all bookmarks
Optional blnCaseSensitive As Boolean	Case sensitive search?
Optional blnWholeWord As Boolean = True	Whole word search?
Optional blnSearchAll As Boolean	Search all bookmarks?

Returns: Long Integer (total number of bookmarks changed)

3.129 SetBookmarkColor - Method to set bookmark's color in a document.

Arguments:

varColor as Variant	Bookmark color (array)
Optional strBookmark As String	Bookmark name to search for. If blank set color for all bookmarks
Optional blnCaseSensitive As Boolean	Case sensitive search?
Optional blnWholeWord As Boolean = True	Whole word search?
Optional blnSearchAll As Boolean	Search all bookmarks?

Returns: Long Integer (total number of bookmarks changed)

3.130 SetBookmarkExpanded - Method to set bookmark as expanded or collapsed in a document.

Arguments:

blnExpand As Boolean	Values: True (expand), False (Collapse)

Returns: Long Integer (total number of bookmarks affected)

3.131 SetBookmarkName - Method to set a bookmark's name in a document.

Arguments:

strName As String	New bookmark name
Optional strBookmark As String	Bookmark name to search for. If blank set name for all bookmarks
Optional blnCaseSensitive As Boolean	Case sensitive search?
Optional blnWholeWord As Boolean = True	Whole word search?
Optional blnSearchAll As Boolean	Search all bookmarks?

Returns: Long Integer (total number of bookmarks changed)

3.132 SetBookmarkProperty - Method to set a bookmark's text style and/or color by level in a document.

Arguments:

varColor As Variant	Bookmark color (array)
ByVal lngStyle As Long	Bookmark style
intLevel As Integer	Bookmark level to change

Returns: Long Integer (total number of bookmarks changed)

3.133 SetBookmarkStyle - Method to set a bookmark's text style in a document.

Arguments:

ByVal lngStyle As Long	Bookmark style
Optional strBookmark As String	Bookmark name to search for. If blank set style for all bookmarks
Optional blnCaseSensitive As Boolean	Case sensitive search?
Optional blnWholeWord As Boolean = True	Whole word search?
Optional blnSearchAll As Boolean	Search all bookmarks?

Returns: Long Integer (total number of bookmarks changed)

3.134 SetColor - Method that sets color arrays for common colors.

Arguments:

lngColor As Long	Color type number
varColor As Variant	Color array to modify

3.135 SetDocumentAction - Method to set the action (JavaScript) of a specific document.

Arguments:

lngTrigger As Long	Document trigger type
strAction As String	JavaScript code to execute

3.136 SetFreeTextAnnotProperties - Method to set the properties of a free text annotation within a document. Note: Use method GetFreeTextAnnotProperties to get properties.

Arguments:

strName As String	Annotation name
lngPage As Long	Page number
fldt As FreeTextAnnotType	Free text annot properties

3.137 SetPageAction - Method to set the open/close actions (JavaScript) of a specific document's page.

Arguments:

lngPage As Long	Page number
lngTrigger As Long	Page trigger type
strAction As String	JavaScript code to execute

3.138 SetPageLabels - Method to establish the numbering scheme for the specified page and all pages following it until the next page with an attached label is encountered.

Arguments:

lngPage As Long	Page number
lngTrigger As Long	Page trigger type
strAction As String	JavaScript code to execute

3.139 SetPageRotations - Method to set the rotation of the specified page.

Arguments:

Optional lngStartPage As Long	Start page number
Optional lngEndPage As Long	End page number
Optional lngRotation As Long	Rotation type. Values: 0, 90, 180, 270

3.140 SetPageTabOrder - Method to set the tab order of the form fields on a page. The tab order can be set by row, by column, or by structure.

Arguments:

lngPage As Long
Optional lngPageTabOrder As Long

3.141 SetPageView - Method to set the initial page view of a document.

Arguments:

lngPageView As Long	Initial view style (show/hide bookmarks, show thumbnails, etc)

3.142 SetReadOnly - Method to set the specified form fields to read-only.

Arguments:

Optional blnReadOnly As Boolean
Optional varFields As Variant
Optional lngLock As Long

3.143 SetRequired - Method to set the specified form fields to required.

Arguments:

Optional blnRequired As Boolean
Optional varFields As Variant
Optional lngLock As Long

3.144 SetTextAnnotProperties - Method to set the properties of a text annotation within a document. Note: Use method GetTextAnnotProperties to get properties.

Arguments:

strName As String	Annotation name
lngPage As Long	Page number
fldt As TextAnnotType	Text annot properties

3.145 Sounds – Property (variant) to get an array containing all of the named Sound Objects in the document.

3.146 SpawnPageFromTemplate - Method to spawn a page in the document using the given template.

Arguments:

strTemplate As String	Template name to use
Optional lngPage As Long	Page number
Optional blnRenameFields As Boolean = True	Rename fields?
Optional blnOverlayPage As Boolean = True	Overlay page?

3.147 SpellCheckPDF - Method to spell check a document. Results are returned in a one dimensional array.

Arguments:

Optional strDelimiter As String = "^"	Word delimiter
Optional lngPage As Long	Page number

Returns: Variant (array of words and spelling suggestions)

3.148 Subject – Property (string) to get/set the document's subject.

3.149 SubmitFormToURL - Method to submit the current form to a specified URL.

Arguments:

URL As String	URL to submit form to
Optional blnFDF As Boolean = True	Submit as FDF? If false then submit as HTML

Optional blnEmpty As Boolean	Include empty fields?
Optional varFields As Variant	Field list (one dimensional array)
Optional blnGet As Boolean	Get?
Optional blnAnnotations As Boolean	Include annotations?
Optional blnXML As Boolean	XML?
Optional blnIncrementalChanges As Boolean	
Optional blnPDF As Boolean	
Optional blnCanonical As Boolean	
Optional blnExclNonUserAnnots As Boolean	
Optional blnExclFKey As Boolean	
Optional strPassword As String	
Optional blnEmbedForm As Boolean	

3.150 Templates – Property (variant) to get an array of all of the Template Objects in the document.

3.151 Title – Property (string) to get/set the document's title.

3.152 URL – Property (string) to get The document's URL.

4. Field Methods/Properties (Field Class)

4.1 Alignment - Property (string) to get/set how the text is laid out within a text field. Values: left, center, right.

4.2 BorderStyle - Property (string) to get/set border style for a field. Values: solid, beveled, dashed, inset, underline.

4.3 ButtonAlignX - Property (integer) to get/set space distributed from the left of the button face with respect to the icon. It is expressed as a percentage between 0 and 100, inclusive.

4.4 ButtonAlignY - Property (integer) to get/set space distributed from the bottom of the button face with respect to the icon. It is expressed as a percentage between 0 and 100, inclusive.

4.5 ButtonFitBounds - Property (boolean) to get/set button fit bounds. When true, the extent to which the icon may be scaled is set to the bounds of the button field; the additional icon placement properties are still used to scale/position the icon within the button face.

4.6 ButtonPosition - Property (long integer) to get/set button position. Controls how the text and the icon of the button are positioned with respect to each other within the button face.

4.7 ButtonScaleHow - Property (long integer) to get/set how the icon is scaled (if necessary) to fit inside the button face.

4.8 ButtonScaleWhen - Property (long integer) to get/set when an icon is scaled to fit inside the button face.

4.9 CalcOrderIndex - Property (integer) to get/set the calculation order of text and combo box fields in the document.

4.10 CharLimit - Property (integer) to get/set the number of characters that a user can type into a text field.

4.11 ClearItems - Method to clear all the values in a list box or combo box.

4.12 Comb - Property (boolean) to get/set comb property. If set to true, the field background is drawn as series of boxes (one for each character in the value of the field) and the each character of the content is drawn within those boxes. The number of boxes drawn is determined by the field's CharLimit property.

4.13 CommitSelOnChange - Property (integer) to get/set whether a combo box or list box field value is committed after a selection change. When true, the field value is committed immediately when the selection is made. When false, the user can change the selection multiple times.

4.14 CurrentValueIndices - Property (variant) to get/set single or multiple values of a list box or combo box.

4.15 DefaultValue - Property (string) to get/set the default value of all fields except button and signature.

4.16 DeleteItem - Method to delete an item in a check box or list box.

Arguments:

Optional intIndex As Integer = -1 Item index to remove

4.17 Display - Property (long integer) to get/set whether the field is hidden or visible on screen and in print.

4.18 DoNotScroll - Property (boolean). When true, the text field does not scroll and the user, therefore, is limited by the rectangular region designed for the field.

4.19 DoNotSpellCheck - Property (boolean). When true, spell checking is not performed on this editable text or combo box field.

4.20 Editable - Property (boolean) to get/set whether a combo box is editable. When true, the user can type in a selection. When false, the user must choose one of the provided selections.

4.21 ExportValues - Property (variant) to get/set export values defined for the field. For radio button fields, this is necessary to make the field work properly as a group with the one button checked at any given time giving its value to the field as a whole.

4.22 FieldPages - Method to get the page number or an array of page numbers of a specific field.

Arguments: None

Returns: Variant (array of page numbers)

4.23 FieldType - Property (string) to get the type of the field as a string. Values: button, checkbox, combobox, listbox, radiobutton, signature, text.

4.24 FileSelect - Property (boolean). When true, sets the file-select flag in for the text field. This indicates that the value of the field represents a pathname of a file whose contents may be submitted with the form.

4.25 FillColor - Property (variant) to get/set the background color for a field.

4.26 GetAlignment - Method to get text field alignment string. Values: left, center, right.

Arguments:

i As Long

Returns: String

4.27 GetBorderStyle - Method to get the field border style string. Values: solid, beveled, dashed, inset, underline.

Arguments:

i As Long

Returns: String

4.28 GetButtonCaption - Method to get the button field caption.

Arguments:

Optional ByVal lngButtonFace As Long

Returns: String

4.29 GetChildFields - Method to get an array of terminal child fields (that is, fields that can have a value) for this Field Object, the parent field.

Arguments:

varFields As Variant

Returns: Integer

4.30 GetFont - Method to get a valid font string.

Arguments:

i As Long	Font number

Returns: String (Font)

4.31 GetGlyphStyle - Method to get a valid glyph style string for a check box or radio button field.

Arguments:

i As Long	Field glyph style number

Returns: String (check, cross, diamond, circle, star, square)

4.32 GetHighlight - Method to get the highlight string used by a button.

Arguments:

i As Long	Highlight number

Returns: String (none, invert, push, outline)

4.33 GetItem - Method to get item export value/name in a combo box or list box.

Arguments:

intIndex As Integer	Item index
Optional blnExportValue As Boolean	Get export value?

Returns: Variant (value of field)

4.34 GetSignatureInfo - Method to get a SignatureInfo Object that contains the properties of the signature. The object is a snapshot of the signature that is taken at the time that this method is called.

4.35 GetTrigger - Method to get a valid trigger string.

Arguments:

i As Long	Field trigger number

Returns: String (valid field trigger string)

4.36 Hidden - Property (boolean) to get/set whether the field is hidden or visible to the user. If the value is false the field is visible, true the field is invisible.

4.37 Highlight - Property (string) to get/set how a button reacts when a user clicks it. Values: none, invert, push, outline.

4.38 InsertItem - Method to insert an item in a combo box or list box.

Arguments:

strItem As String	Item value to insert
Optional strExportValue As String	Export value to insert
Optional intIndex As Integer	Item insert index

4.39 IsBoxChecked - Method to determine if a radio button or check box is checked.

Arguments:

intIndex As Integer	Item index

Returns: Boolean (True if checked, False if unchecked)

4.40 IsDefaultChecked - Method to determine if a radio button or check box's default value is checked.

Arguments:

intIndex As Integer Item index

Returns: Boolean (True if checked, False if unchecked)

4.41 IsSigned - Method to determine whether the signature field is signed or not.

Arguments: None

Returns: Boolean (True if signed, False if unsigned)

4.42 LineWidth - Property (long integer) to get/set the thickness of the border when stroking the perimeter of a field’s rectangle. If the stroke color is transparent, this parameter has no effect except in the case of a beveled border.

4.43 MultiLine - Property (boolean) to get/set how the text is wrapped within a text field. When false, the default, the text field can be a single line only. When true, multiple lines are allowed and wrap to field boundaries.

4.44 MultipleSelection - Property (boolean) to get/set whether a list box allows multiple selection of the items.

4.45 Name - Property (string) to get the fully qualified field name of the field.

4.46 NumItems - Property (integer) to get the number of items in a combo box or list box.

4.47 Page - Property (variant) to get the page number or an array of page numbers of a field.

4.48 Password - Property (boolean) to get/set whether asterisks will be displayed for the data entered into the text field. Upon submission, the actual data entered is sent. Fields that have the password attribute set will not have the data in the field saved when the document is saved to disk.

4.49 PrintField - Property (boolean) to get/set whether a given field prints or not. Set the print property to true to allow the field to appear when the user prints the document, set it to false to prevent printing. This property can be used to hide control buttons and other fields.

4.50 RadiosInUnison - Property (boolean) to get/set radio button behavior. When false, even if a group of radio buttons have the same name and export value, they behave in a mutually exclusive fashion, like HTML radio buttons. The default for new radio buttons is false.

4.51 ReadOnly - Property (boolean) to get/set the read-only characteristic of a field. If a field is read-only, the user can see the field but cannot change it.

4.52 Rect - Property (variant) to get/set an array of four numbers in rotated user space that specifies the size and placement of the form field. These four numbers are the coordinates of the bounding rectangle and are listed in the following order: upper-left x, upper-left y, lower-right.

4.53 Required - Property (boolean) to get/set the required characteristic of a field (except button). When true, the field’s value must be non null when the user clicks a submit button that causes the value of the field to be posted. If the field value is null, the user receives a warning message when the form is submitted.

4.54 Rotation - Property (long integer) to get/set the rotation of a field in 90 degree counter-clockwise increments. Values: 0, 90, 180, 270

4.55 SetButtonCaption - Method to set the caption of a button field.

Arguments:

strCaption As String Button caption

4.56 SetCheckBox - Method to set the value of a check box field.

Arguments:

intIndex As Integer	Item index
Optional blnChecked As Boolean	Checked?

4.57 SetDefaultValue - Method to set the value of a check box or radio button.

Arguments:

intIndex As Integer	Item index
Optional blnChecked As Boolean	Checked?

4.58 SetFieldAction - Method to set the action of a specific field.

Arguments:

lngTrigger As Long	Field trigger type
strAction As String	JavaScript code to execute

4.59 SetItems - Method to set the items for a combo box or list box.

Arguments:

varItems As Variant	Array of items.

4.60 SetSignatureFieldLocks - Method to set the fields to be locked (set to ReadOnly) when this signature field is signed.

Arguments:

Optional varFields As Variant	Array of fields
Optional lngLock As Long	Lock type

4.61 SignDocument - Method to sign the document via this signature field.

Arguments:

strCertificateFile As String
strPassword As String

4.62 StrokeColor - Property (variant) to get/set the stroke color (as array) for a field which is used to stroke the rectangle of the field with a line as large as the line width.

4.63 Style - Property (string) to get/set the glyph style of a check box or radio button. The glyph style is the graphic used to indicate that the item has been selected. Values: check, cross, diamond, circle, star, square.

4.64 SubmitName - Property (string) to get/set name if submitting form in HTML format.

4.65 TextColor - Property (variant) to get/set the foreground color (array) of a field.

4.66 TextFont - Property (string) to get/set the font that is used when laying out text in a text field, combo box, list box or button. See method GetFont.

4.67 TextSize - Property (integer) to get/set the text size (in points) to be used in all controls. In check box and radio button fields, the text size determines the size of the check. Valid text sizes range from 0 to 32767 inclusive. A text size of zero means to use the largest point size.

4.68 UserName - Property (string) to get/set the user name (short description string) of the field. The user name is intended to be used as tooltip text whenever the mouse cursor enters a field.

4.69 Value - Property (variant) to get/set the value of the field data that the user has entered. Depending on the type of the field, may be a String, Date, or Number. Typically, the value is used to create calculated fields.

4.70 ValueAsString - Property (string) to get the value of a field as a JavaScript string.

5. Miscellaneous Methods/Properties (CreatePDF Class)

5.1 AcrobatVersion – Method to get the Acrobat version number as a string.

5.2 BookmarksToArray - Method to populate a two dimensional array with bookmark data from an Adobe Acrobat (PDF) file.

Arguments:

strPDFFile As String	PDF file to use
varBookmarks As Variant	Array to store bookmark names

Returns: Long Integer (total number of bookmarks)

5.3 BookmarksToPDF - Copy bookmarks from one Adobe Acrobat (PDF) file to another.

Arguments:

strSourcePDF As String	Source PDF file to get bookmarks from
strTargetPDF As String	Target PDF file to copy bookmarks to

Returns: Long Integer (total number of bookmarks copied)

5.4 CloseAcrobat - Method to close a hidden instance of Adobe Acrobat.

5.5 CreateBookmarkInPDF - Method to create a bookmark in an Adobe Acrobat (PDF) file.

Note: This function compatible with Acrobat 5 and 6 only. Please use newer bookmark functions.

Requirements: file PDFOSEND.EXE (see page 6 of this document) if opening a password protected PDF file

Arguments:

strPDFFile As String	PDF file to use
strBookmarkTitle As String	Bookmark title
Optional blnCaseSensitive As Boolean	Case sensitive search for bookmark
Optional blnWholeWordsOnly As Boolean	Whole word search for bookmark
Optional blnReset As Boolean	Reset search conditions
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim i As Integer

strPDF = App.Path & “\invoice.pdf”
‘create bookmark
oPDFmaker.CreateBookmarkInPDF strPDF, "Introduction", True, True, True
‘open PDF and move to bookmark
oPDFmaker.OpenPDF strPDF, "Introduction"
```

5.6 CreateBookmarksInPDF - Method to create multiple bookmarks in an Adobe Acrobat (PDF) file.

Note: If you need to create multiple bookmarks in a PDF file use this instead of function CreateBookmarkInPDF because it is much more efficient. This function compatible with Acrobat 5 and 6 only. Please use newer bookmark functions.

Requirements: file PDFOSEND.EXE (see page 6 of this document) if opening a password protected PDF file

Arguments:

strPDFFile As String	PDF file to use
varBookmarkTitles As Variant	Bookmark title
Optional blnCaseSensitive As Boolean	Case sensitive search for bookmark
Optional blnWholeWordsOnly As Boolean	Whole word search for bookmark

Optional blnReset As Boolean	Reset search conditions
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim i As Integer
Dim varBookmarks(2) As String

strPDF = App.Path & “\invoice.pdf”
varBookmarks(0) = “Introduction”
varBookmarks(1) = “Body”
varBookmarks(2) = “Conclusion”
‘create multiple bookmarks
oPDFmaker.CreateBookmarksInPDF strPDF, varBookmarks, True, True, True
‘open PDF and move to first bookmark
oPDFmaker.OpenPDF strPDF, "Introduction"
```

5.7 CreateLinksInPDF - Method to create hyperlinks in an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document)

Arguments:

strPDFFile As String	PDF file to use
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim i As Integer

strPDF = App.Path & “\invoice.pdf”
'create hyperlinks
oPDFmaker.CreateLinksInPDF strPDF
```

5.8 CreatePageHeaderInPDF - Method to create a page header (page no’s, custom text, date) in an Adobe Acrobat (PDF) file.

Note: From Adobe Acrobat’s Document Menu select ‘Add Headers & Footers…”

Requirements: file PDFOSEND.EXE (see page 6 of this document)

Arguments:

strPDFFile As String	PDF file to use
intFontName As Integer	Font name (Values: 0 through maximum # in list)
intFontSize As Integer	Font size (Values: 0 through maximum # in list, 0 = 8 pt, 1 = 9 pt, etc)
intPageStyle As Integer	Page style (-1 = don’t show page, 0 = #, 1 = 1 of n, 2 = 1/n, 3 = Page #)
ByVal lngPageAlign As Long	Page number alignment (0 = left, 1= center, 2 = right)
intDateStyle As Integer	Date style (-1 = don’t show date, 0 = m/d, 1 = m/d/yy, 2 = m/d/yyyy)
ByVal lngDateAlign As Long	Date alignment (0 = left, 1= center, 2 = right)
strCustomText As String	Custom text
ByVal lngCustomTextAlign As Long	Custom text alignment (0 = left, 1= center, 2 = right)
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim strTXT As String

On Error Resume Next

Screen.MousePointer = vbHourglass
```

```
strPDF = App.Path & "\file.pdf"
strTXT = App.Path & "\file.txt"

oPDFmaker.CreatePDFfromTextFile3 strPDF, strTXT

oPDFmaker.CreatePageHeaderInPDF strPDF, 1, 1, 4, pdoAlignRight, 4, pdoAlignLeft, "My PDF File", pdoAlignCenter

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open PDF file
  oPDFmaker.OpenPDF strPDF
End If
```

5.9 CreateTextfileFromPDF - Method to create a text file from an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document) if opening a password protected PDF file

Arguments:

strPDFFile As String	PDF file to use
strTextfile As String	Text file to create
Optional blnDoubleSpace As Boolean	Use single or double spacing. Values: True,False
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim strTXT As String
On Error Resume Next

Screen.MousePointer = vbHourglass

strPDF = App.Path & "\pdfmaker.pdf"
strTXT = App.Path & "\pdfmaker.txt"

'Create the text file!
oPDFmaker.CreateTextfileFromPDF strPDF, strTXT

Screen.MousePointer = vbDefault

If Err.Number Then
  MsgBox Err.Number & " : " & Err.Description, vbExclamation, "Error Message"
Else
  'open text file
  ShellExecute 0, vbNullString, strTXT, vbNullString, vbNullString, 1
End If
```

5.10 CreateThumbnailImage - Method to create a thumbnail image file from a page in a PDF file.

Arguments:

```
strPDF As String
Optional ByVal strFile As String
Optional ByVal strFileExtention As String = "bmp"
Optional ByVal intZoom As Integer = 10
Optional ByVal lngPage As Long = 1
```

Returns: Integer (1 = successful, -1 = unsuccessful)

5.11 CreateThumbnailImages - Method to create thumbnail image files from all pages in a PDF file.

Arguments:

strPDF As String
Optional ByVal strDir As String
Optional ByVal strFileExtention As String = "bmp"
Optional ByVal intZoom As Integer = 10

Returns: Integer (1 = successful, -1 = unsuccessful)

5.12 DatabaseTableToArray - Method to move items from an ODBC compliant database table to an array.

Arguments:

strDSN As String	ODBC Data source name
strSQL As String	SQL statement
varArray As Variant	Array to populate
Optional strUserID As String	User ID
Optional strPassword As String	User password
Optional blnIncludeHeadings As Boolean	Include field headings?

Returns: Integer

1 = Success
-1 = varArray is not an array
-2 = Unable to establish a connection to the database
-3 = Unable to open recordset
-4 = Unknown error

5.13 DatabaseTableToTextFile - Method to move items from an ODBC compliant database table to a text file.

Arguments:

strDSN As String	ODBC Data source name
strSQL As String	SQL statement
strTextFile As String	Text file to create
Optional strUserID As String	User ID
Optional strPassword As String	User password
Optional blnIncludeHeadings As Boolean	Include field headings?
Optional ByVal strDelimiter As String	Field delimiter
Optional blnQuotes As Boolean	Include quotes?

Returns: Integer

1 = Success
-1= Unable to establish a connection to the database
-2 = Unable to open recordset
-3 = Unknown error

5.14 DownloadFile - Function to download a file from a remote web server.

Requirements: MSINET.OCX

Note: An internet connection must be established before using this function.
Arguments:

inet As Object
strRemoteHost As String
strRemoteDirectory As String
strLocalFile As String
strRemoteFile As String

Optional strUsername As String
Optional strUserPassword As String

Example:

'Note: you must modify the remote host settings before this will work
oPDFmaker.DownloadFile Inet1, "microsoft.com", "/mydirectory", "c:\file.txt", "file.txt"

5.15 ExcelToOneDArray - Method to copy a an Excel workbook worksheet range to a one dimensional array.

Arguments:

strWorkbook As String	Excel workbook
varWorksheet As Variant	Worksheet name/number
lngStartRow As Long	Start row number
lngStartCol As Long	Start column number
lngEndRow As Long	End row number
lngEndCol As Long	End column number
varArray As Variant	Array to populate
Optional blnCellValue As Boolean	Use cell values?
Optional strPassword As String	Workbook password
Optional strWriteResPassword As String	Workbook write reserve password

Returns: Integer

1 = Success
-1 = Unable to create Excel object
-2 = varArray is not an array
-3 = Unable to open workbook
-4 = Unable to open worksheet
-5 = Unknown error

5.16 ExcelToTwoDArray - Method to copy a an Excel workbook worksheet range to a two dimensional array.

Arguments:

strWorkbook As String	Excel workbook
varWorksheet As Variant	Worksheet name/number
lngStartRow As Long	Start row number
lngStartCol As Long	Start column number
lngEndRow As Long	End row number
lngEndCol As Long	End column number
varArray As Variant	Array to populate
Optional blnCellValue As Boolean	Use cell values?
Optional strPassword As String	Workbook password
Optional strWriteResPassword As String	Workbook write reserve password

Returns: Integer

1 = Success
-1 = Unable to create Excel object
-2 = varArray is not an array
-3 = Unable to open workbook
-4 = Unable to open worksheet
-5 = Unknown error

5.17 FindTextInPDF - Method to find specified text in an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document) if opening a password protected PDF file

Arguments:

strPDFFile As String	PDF file to use

strText As String	Text to search for
Optional blnCaseSensitive As Boolean	Case sensitive search? Values: True,False
Optional blnWholeWordsOnly As Boolean	Search for whole words only? Values: True,False
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim i As Integer

strPDF = App.Path & "\file.pdf"

oPDFmaker.FindTextInPDF strPDF, "Hello World!", True, True
```

5.18 GetNumPagesPDF - Method to return the number of pages in an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document) if opening a password protected PDF file

Arguments:

strPDFFile As String	PDF file to use
Optional strOpenPassword As String	PDF open password

Returns: Long Integer

= Success, number of pages

Example:

```
Dim strPDF As String

strPDF = App.Path & "\file.pdf"

MsgBox oPDFmaker.GetNumPagesPDF(strPDF), vbInformation, "Number of Pages"
```

5.19 IsAcrobatHidden - Method to determine if an instance of Adobe Acrobat is loaded in memory.

Arguments: None

Returns: Boolean (True if found, False if not found)

5.20 OneDArrayToText - Method to move items in a one dimensional array to a string variable.

Arguments:

varArray As Variant	Array
strText As String	Text string to create
strEndOfRowChar As String	End of row character
Optional strEndOfArrayChar As String	End of array character
Optional blnQuotes As Boolean	Surround text with quotes?

Returns: Integer

1 = Success
-1 = varArray is not an array
-2 = Unknown error

5.21 OneDArrayToTextFile - Method to move items in a one dimensional array to a text file.

Arguments:

varArray As Variant	Array
strTextFile As String	Text file to create

Optional blnQuotes As Boolean	Surround text with quotes?

Returns: Integer

1 = Success
-1 = varArray is not an array
-2 = Unknown error

5.22 OpenDocument - Method to instantiate the Document Object. Used to perform various operations on a PDF file.

Arguments:

strPDF As String	PDF file to open

Returns: Document Object

5.23 OpenPDF - Method to open an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document)

Arguments:

strPDFFile As String	PDF file to open
Optional strBookmark As String	Bookmark to move to
Optional lngGoToPageNo As Long	Page number to jump to
Optional ByVal lngReadType As Long	Values: 0 = None, 1 = Current Page, 2 = Entire Document
Optional strOpenPassword As String	PDF open password

Example:

```
Dim strPDF As String
Dim strBookmark As String

strPDF = App.Path & "\file.pdf"
strBookmark = "Chapter One"

oPDFmaker.OpenPDF strPDF, , 6, 1
```

5.24 PrinterName - Method to get the Adobe Acrobat Distiller printer name.

5.25 PrintPDF - Method to print an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document) if opening a password protected PDF file

Arguments:

strPDFFile As String	PDF file to print
Optional strPrinter As String	Printer name to use
Optional blnSilent As Boolean	Show print window? Values: True,False
Optional strOpenPassword As String	PDF open password
Optional lngFirstPageNo As Long	First page number to print
Optional lngLastPageNo As Long	Last page number to print

Example:

```
Dim strPDF As String

strPDF = App.Path & "\file.pdf"

oPDFmaker.PrintPDF strPDF, "HP Laserjet III", True
```

5.26 PrintPDFinDir - Method to print all specified Adobe Acrobat (PDF) files in a specific directory.

Arguments:

strDirectory As String	Directory to use
Optional strFiles As String = "*.pdf"	PDF files to print (can use wildcard characters)
Optional strPrinter As String	Printer name to use
Optional blnSilent As Boolean	Show print window? Values: True,False
Optional lngSortType As Long	Sort field (0=None,1=File Name,2=File Size,3=File Date)
Optional lngSortOrder As Long	Sort order (0=Ascending, 1=Descending)

Example:

oPDFmaker.PrintPDFinDir App.Path, “a*.pdf”, “HP Laserjet III”, True, 1, 0

5.27 RecordsetToArray - Method to move items from an ADO recordset to an array.

Arguments:

objRecd As Object	ADO recordset object
varArray As Variant	Array to populate
Optional blnIncludeHeadings As Boolean	Include field headings?

Returns: Integer

1 = Success
-1 = varArray is not an array
-2 = Unknown error

5.28 RecordsetToTextFile - Method to move items from an ADO recordset to a text file.

Arguments:

objRecd As Object	ADO recordset object
strTextFile As String	Text file to create
Optional blnIncludeHeadings As Boolean	Include field headings?
Optional ByVal strDelimiter As String	Field delimiter
Optional blnQuotes As Boolean	Surround fields with quotes?

Returns: Integer

1 = Success
-1 = Unknown error

5.29 ReduceFileSize - Method to reduce the size of an Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document) , Acrobat 6 and greater

Arguments:

strPDFFile As String	PDF file to print
Optional strOpenPassword As String	PDF open password
Optional lngVersion As Long	Acrobat version (0=Version 4, 1=Version 5, 2=Version 6)

Example:

‘Reduce file size by making compatible with Acrobat 6 only
oPDFmaker.ReduceFileSize App.Path, “file.pdf”, , 2

5.30 SaveFileAs - Method to save an Adobe Acrobat (PDF) file using a different file format (DOC,TXT,HTM,etc).

Requirements: file PDFOSEND.EXE (see page 6 of this document) , Acrobat 6 and greater

Arguments:

strPDFFile As String	PDF file to use

strFile As String	File to create
Optional strOpenPassword As String	PDF open password
Optional ByVal lngFileType As Long	File format to use

Example:

```
Dim strPDF As String
Dim strDOC As String

strPDF = App.Path & "\sample.pdf"
strDOC = App.Path & "\sample.doc"

‘Save PDF file as a Word document
oPDFmaker.SaveFileAs strPDF, strDOC, , pdoDOC
```

5.31 SetDefPrinter - Method to set the default system printer.

Arguments:

strPrinter As String	Printer to use

Example:

```
oPDFmaker.SetDefPrinter “HP DeskJet 890C”
```

5.32 SetOpenPassword - Method to set the open password for a specific Adobe Acrobat (PDF) file.

Requirements: file PDFOSEND.EXE (see page 6 of this document)

Note: This function will not work with PDF files that already have open passwords

Arguments:

strPDFFile As String	PDF file to use
strOpenPassword As String	Open password to use

Example:

```
oPDFmaker.SetOpenPassword “c:\file.pdf”, “opensezme”
```

5.33 TempFilePath – Property (string) to get/set the path where temporary files will be created. Default is C:\Temp

5.34 TextFileToArray - Method to move the contents of a text file to an array.

Arguments:

strTextFile As String	Text file
varArray As Variant	Array to populate
Optional ByVal strDelimiter As String	Field delimiter
Optional blnRemoveQuotes As Boolean	Remove quotes surrounding fields?

Returns: Integer

```
 1 = Success
-1 = Unable to open text file
-2 = varArray is not an array
-3 = Unknown error
```

5.35 TextFileToOneDArray - Method to move the contents of a text file to a one dimensional array.

Arguments:

strTextFile As String	Text file

varArray As Variant	Array to populate

Returns: Integer

 1 = Success
-1 = Unable to open text file
-2 = varArray is not an array
-3 = Unknown error

5.36 TwoDArrayToText - Method to move items in a two dimensional array to a string variable.

Arguments:

varArray As Variant	Array
strText As String	Text string to create
strEndOfRowChar As String	End of row character
Optional strEndOfArrayChar As String	End of array character
Optional blnQuotes As Boolean	Surround text with quotes?

Returns: Integer

 1 = Success
-1 = varArray is not an array
-2 = Unknown error

5.37 TwoDArrayToTextFile - Method to move items in a two dimensional array to a text file.

Arguments:

varArray As Variant	Array
strTextFile As String	Text file to create
Optional blnQuotes As Boolean	Surround text with quotes?

Returns: Integer
 1 = Success
-1 = varArray is not an array
-2 = Unknown error

5.38 UploadFile - Function to copy a local file to a remote web server.

Requirements: MSINET.OCX

Note: An internet connection must be established before using this function.

Arguments:

inet As Object
strRemoteHost As String
strRemoteDirectory As String
strLocalFile As String
strRemoteFile As String
Optional strUsername As String
Optional strUserPassword As String

Example:

```
'Note: you must modify the remote host settings before this will work
oPDFmaker.UploadFile Inet1, "www.microsoft.com", "/mydirectory", App.Path & "\table.htm", "table.htm", "username", "password"
```

5.39 XMLFileToArray - Method to copy items from an XML table file to a two dimensional array.

Arguments:

strXMLFile As String	XML file
varArray As Variant	Array to populate
Optional blnIncludeHeadings As Boolean	Include field headings?

Returns: Integer
 1 = Success
-1 = varArray is not an array
-2 = Unable to establish connection
-3 = Unable to open recordset
-4 = Unknown error

5.40 XMLFileToRecordset - Method to copy items from an XML table file to an ADO recordset.

Arguments:

strXMLFile As String	XML file
objRecd As Object	ADO recordset object

Returns: Integer

 1 = Success
-1 = Unable to establish connection
-2 = Unknown error