ActiveX PDF Viewer OCX

1. Introduction

Author: SkySof Software Inc. at http://www.skysof.com
Email: kusluski@nc.rr.com
Download URL: http://www.getfilez.com/axpdfvw.zip
Updated: 7/15/14

ActiveX PDF Viewer OCX is an ActiveX control which enables your application to display and interact with PDF files. Simply place the control on your form, set the Path property, and you are all set! ActiveX PDF Viewer OCX does not require Adobe Reader or any other PDF reading program and is .NET compatible.

Features:

- Link and hyperlink support
- Form field support
- Print support
- Extract text from PDF files
- Open password-protected PDF files
- Save PDF file pages as image files
- Add comments, images, and text to PDF file pages
- Bookmark support
- Extract PDF file pages
- Extract images from PDF file
- Find text in PDF file
- Set PDF form field values
- Zoom in and Zoom out support
- Open PDF files from an ADO Data Stream Object

2. Using with Visual Basic 6

To add ActiveX PDF Viewer OCX to your Visual Basic 6.0 project:

- From the Project Menu select **Components…**
- Select control **SkySof ActiveX PDF Viewer**
- Click the OK Button
- The control will now appear in your toolbox
- Drag and drop the control on your form, resize it as desired

There are two methods of opening a PDF file with the control:

```
Dim strFile As String
Dim b As Boolean
strFile = "C:\Documents\file.pdf"
PDFViewer1.Path = strFile
b = PDFViewer1.OpenPDF
If b = False Then
   MsgBox "Unable to open file."
EndIf
```

OR

```
Dim strFile As String
Dim b As Boolean
strFile = "C:\Documents\file.pdf"
b = PDFViewer1.OpenPDF(strFile)
If b = False Then
   MsgBox "Unable to open file."
EndIf
```

PDF files located on web servers can also be opened. But, you must download them to a disk drive before opening them. For example:

```
Dim strWebFile As String
Dim strLocalFile As String
Dim b As Boolean
strWebFile = "http://www.getfilez.com/axpdfvw.pdf"
strLocalFile = "C:\Program Files\SkySof ActiveX PDF Viewer\axpdfvw.pdf"
PDFViewer1.BeginDownload strWebFile, strLocalFile
'Allow time for file to be downloaded. See VB Demo for example.
b = PDFViewer1.OpenPDF(strLocalFile)
If b = False Then
   MsgBox "Unable to open file."
EndIf
```

3. Using with VBA for Access

How to add ActiveX PDF Viewer OCX to your Access Form

- Open Access and create a new blank database
- Create a new form in Design View. You should now be in design mode
- From the Insert Menu select "ActiveX Control..."
- Select SkySof.PDFViewer and click the OK Button
- The control should appear on the form
- Resize and move the control on the form as desired
- Add a button to the form. If the wizard appears cancel it
- Right-click on the button to invoke the popup menu
- Select "Build Event..." then select Code Builder and click OK Button
- You should be inside the Visual Basic Development Environment
- Within the Command_Click enter the following code:
  PDFViewer0.OpenPDF "C:\Program Files\SkySof ActiveX PDF Viewer\axpdfvw.pdf"
- Save your changes and close Visual Basic
- Save the form and close it
- Double-click on the form to run it.
- Click the button. The PDF should appear!
- Experiment by changing the PDF path to a local file such as:
PDFViewer0.OpenPDF "c:\temp\myfile.pdf"

4. Using with .NET

How to add ActiveX PDF Viewer OCX to your .NET project

- Open Visual Studio .NET
- Right-click on the toolbox and select "Add/Remove Items…"
- Select the COM Components Tab
- Check SkySof.PDFViewer and click OK Button
- The control should appear in the toolbox as "SkySof.PDFViewer" or "PDFViewer"
- Double-click on the control to add to your form
- Resize and move the control on the form as desired
- Add a button to the form
- Double-click on the button to access code editor and enter the following code within the Click event: AxPDFViewer1.OpenPDF("C:\Program Files\SkySof ActiveX PDF Viewer\axpdfvw.pdf", "", 1, 100)
- Change the anchor property of AxPDFViewer1 to Top, Bottom, Left, Right
- Run the application and click on the button. The PDF should appear!

5. Purchase Information

The pricing for ActiveX PDF Viewer OCX through http://www.regnow.com is as follows:

$299.95 USD Individual Developer License
$499.95 USD Site Developer License (Unlimited number of developers per site)
$999.95 USD Site Developer License and Complete Visual Basic 6 Source Code

Go to this link to purchase using RegNow:

https://www.regnow.com/softsell/nph-softsell.cgi?item=4459-65

Save 10% by buying ActiveX PDF Viewer OCX through www.paypal.com!!!

$269.95 USD Individual Developer License
$449.95 USD Site Developer License (Unlimited number of developers per site)
$899.95 USD Site Developer License and Complete Visual Basic 6 Source Code

Go to this link to buy a Single Developer License using PayPal:

https://www.paypal.com/xclick/business=kusluski%40mail.ic.net&item_name=ActiveX+PDF+Viewer+OCX+Single+Developer+License&amount=269.95

Go to this link to buy a Site Developer License using PayPal:

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

Go to this link to buy a Site Developer License and Complete VB6 Source Code using PayPal:

https://www.paypal.com/xclick/business=kusluski%40mail.ic.net&item_name=ActiveX+PDF+Viewer+OCX+Site+and+VB6+Source&amount=899.95

ActiveX PDF Viewer OCX is compatible with all languages that support ActiveX controls and is guaranteed to work with any PDF file (including future releases)!

ActiveX PDF Viewer OCX may be distributed with your application royalty free and all future upgrades are free!

Please send emails to mailto:kusluski@nc.rr.com for additional ordering methods. Please visit http://www.skysof.com for other fine products.

6. Methods

AddComment - Method to add a comment to the active PDF file. Note: PDF must not be encrypted.

ByVal lngIcon As IconType	Icon type: Note,Comment,Help,Insert,Key,NewPara,Para
strText As String	Comment text
ByVal lngStartPage As Long	Start page number. If -1 then use last page
ByVal lngEndPage As Long	End page number. If -1 then use last page
ByVal dblX As Double	Comment X coordinate position
ByVal dblY As Double	Comment Y coordinate position
ByVal dblPopupX As Double	Comment popup X coordinate position
ByVal dblPopupY As Double	Comment popup Y coordinate position
ByVal dblPopupWidth As Double	Comment popup width
ByVal dblPopupHeight As Double	Comment popup height
ByVal strTitle As String	Comment title
ByVal dblRed As Double	Comment red color value. 0 through 1.0
ByVal dblGreen As Double	Comment green color value. 0 through 1.0
ByVal dblBlue As Double	Comment blue color value. 0 through 1.0
Optional ByVal blnOpen As Boolean	Show Comment popup automatically?

Returns: Integer (1 = success)

AddImageFile - Method to add an image to the active PDF file. Note: PDF must not be encrypted.

strImageFile As String	Image file to use
ByVal lngStartPage As Long	Start page number. If -1 then use last page
ByVal lngEndPage As Long	End page number. If -1 then use last page
ByVal dblX As Double	Image X coordinate position
ByVal dblY As Double	Image Y coordinate position
ByVal dblSize As Double	Image size. 0.5 = 50%
Optional ByVal intTransparency As Integer	Image transparency. 0 = none, 50 = 50%, 100 = invisible

Returns: Integer (1 = success)

AddText - Method to add text to the active PDF file. Note: PDF must not be encrypted.

strText As String	Text to use
ByVal lngStartPage As Long	Start page number. If -1 then use last page
ByVal lngEndPage As Long	End page number. If -1 then use last page
ByVal dblX As Double	Image X coordinate position
ByVal dblY As Double	Image Y coordinate position
ByVal dblSize As Double	Image size. 0.5 = 50%
ByVal dblRed As Double	Comment red color value. 0 through 1.0
ByVal dblGreen As Double	Comment green color value. 0 through 1.0
ByVal dblBlue As Double	Comment blue color value. 0 through 1.0
Optional ByVal intTransparency As Integer	Image transparency. 0 = none, 50 = 50%, 100 = invisible
Optional ByVal dblRotation As Double	A value 0 through 360
Optional ByVal lngFontID As FontType	A value 0 through 12

Returns: Integer (1 = success)

BeginDownload - Method to download a file from a web server.

| strURL As String | Download from location |
| strFile As String | Download to file |

Returns: Boolean

CollapseBookmarks - Method to collapse all bookmarks for the active PDF file. Note: PDF must not be encrypted.

Returns: Integer (1 = success)

ExpandBookmarks - Method to expand all bookmarks for the active PDF file. Note: PDF must not be encrypted.

Returns: Integer (1 = success)

ExtractHyperlinks – Method to extract hyperlinks from the active PDF file.

| Optional lngPage As Long | The page number to use |

Returns: String (A comma-delimited string of hyperlinks)

ExtractImages – Method to extract images from the active PDF file and save to image files.

| ByVal strFilePath As String | The file folder to save in |
| strFileName As String | The file name to save as |

Returns: Long Integer (The number of image files extracted)

ExtractPageImages – Method to extract images from the current page of the active PDF file.

| ByVal strFilePath As String | The file folder to save in |
| strFileName As String | The file name to save as |

Returns: Long Integer (The number of image files extracted)

ExtractPages – Method to extract specified pages from the active PDF file and save to PDF file.

| strOutputPDFfile As String | The PDF file to create |
| strPages As String | Range of pages to use as string. Example: 1-3,5,7 |

Returns: Integer (1 = success)

FindText – Method to find the specified text in the active PDF file.

ByVal strText As String	The text to search for
Optional lngPage As Long	The page number to use (If zero then all pages are used)
Optional blnCaseSensitive As Boolean	Case sensitive search?

Returns: String (a list of comma-delimited page numbers and number of occurrences)

FindTextAndHighlight – Method to find the specified text in the active PDF file and highlight in new PDF file.

| strOutputPDFfile As String | The PDF file to create with highlighted text |

ByVal strText As String	The text to search for
Optional lngPage As Long	The page number to use (If zero then all pages are used)
Optional blnCaseSensitive As Boolean	Case sensitive search?

Returns: String (a list of comma-delimited page numbers and number of occurrences)

GetChildOutlines – Method to get the child bookmarks for the active bookmark.

tv As Object	TreeView control to populate with bookmarks
NodParent As Object	The active tree node to expand

This method should be used in the TreeView's Expand subroutine. Sample VB code:

```
Private Sub TreeView1_Expand(ByVal Node As MSComctlLib.Node)
'expand the TreeView
Call PDFViewer1.GetChildOutLines(TreeView1, Node)
End Sub
```

GetFormFields – Method to get all the form fields for the active PDF file and store to text file.

strTextfile As String	Text file to save too

Returns: String (list of PDF form fields)

GetOutlines – Method to populate a Microsoft Windows TreeView control with bookmarks from the active PDF file. If no bookmarks exist the page numbers will be used.

tv As Object	TreeView control to populate with bookmarks

This method should follow the OpenPDF method. To move to the bookmark's page when the bookmark is clicked on in the TreeView enter the following VB code in the TreeView's NodeClick subroutine:

```
Private Sub TreeView1_NodeClick(ByVal Node As MSComctlLib.Node)
  Dim iPosit As Integer
  Dim Page As Long

  If Val(Node.Tag) = 0 Then
    Exit Sub
  End If
  iPosit = InStr(Node.Tag, "|")

  'retrieve the page and position inside the page
  Page = Val(Left(Node.Tag, iPosit - 1))

  Screen.MousePointer = vbHourglass
  PDFViewer1.Page = Page
  Screen.MousePointer = vbDefault
End Sub
```

GetScrollMaxValue - Method to return the maximum value of either the vertical or horizontal scrollbar.

eBar As ScrollBarConstants	Either Vertical or Horizontal

Returns: Long

GetScrollMinValue - Method to return the minimum value of either the vertical or horizontal scrollbar.

eBar As ScrollBarConstants   Either Vertical or Horizontal

Returns: Long

GetScrollValue - Method to return the value of either the vertical or horizontal scrollbar.

eBar As ScrollBarConstants   Either Vertical or Horizontal

Returns: Long

HasPassword - Method to determine if a PDF file is protected with a password.

Optional ByVal strPDF As String   PDF file to check for password

Returns: Boolean (True if PDF is encrypted)

OpenPDF - Method to open a PDF file and display it in the component.

Optional ByVal strPDF As String   PDF file to open
Optional ByVal strPassword As String   Password to use (if any)
Optional ByVal lngPage As Long   Page to display. Page 1 is the default
Optional ByVal lngDotsPerInch As Long   Dots per inch. 100 is the default

Returns: String ("OK" if PDF was opened successfully)

OpenPDFstream - Method to open a PDF file stored as an ADO data stream object.

ByVal objStream As Object   Stream object to open
Optional ByVal strPassword As String   Password to use (if any)
Optional ByVal lngPage As Long   Page to display. Page 1 is the default
Optional ByVal lngDotsPerInch As Long   Dots per inch. 100 is the default

Returns: String ("OK" if stream object was opened successfully)

PrintPDF - Method to print a PDF file.

Optional ByVal strPrinter As String   Printer to use. Use default printer if not specified
Optional ByVal lngStartPage As Long   Start page number. Default is page 1
Optional ByVal lngEndPage As Long   End page number. Use last page if not specified
Optional ByVal lngPageScaling As PageScalingType   Scale type: None,FitToPaper,ShrinkLargePages
Optional ByVal blnAutoRotateCenter As Boolean   Auto rotate center? Default is True
Optional ByVal strTitle As String   Document title

Returns: Boolean (True if PDF printed successfully)

RemovePassword - Method to remove the password for the active PDF file.

Returns: Integer (1 = success)

SavePageAsImage - Method to save the current page in the active PDF file as an image file.

strImageFile As String	Image file to create
lngImageType As ImageType	Image type: BMP,JPG,WMF,EMF,EPS,PNG,GIF,TIF
lngDotsPerInch As Long	Dots per inch. Default value is 100

Returns: Integer (1 = success)

SavePageAsTextFile - Method to save the current page in the active PDF file as a text file.

strTextfile As String	Text file to create
Optional ByVal lngStartPage As Long	Start page number. Default is page 1
Optional ByVal lngEndPage As Long	End page number. If not specified the last page is used

Returns: Integer (1 = success)

SavePDFAsTextFile - Method to save the active PDF file as a text file.

strTextfile As String	Text file to create

Returns: Integer (1 = success)

ScrollControl - Method to scroll the PDF control either vertically or horizontally.

eBar As ScrollBarConstants	Either Vertical or Horizontal
lngValue As Long	Value to use

SetFormFieldDefaultValue - Method to set the specified form field's default value. Note: PDF must not be encrypted.

strFieldName As String	Field name to use
strFieldValue As String	Field value to use
Optional blnFlatten As Boolean	Flatten the form field?

Returns: Integer (1 = success)

SetFormFields - Method to set multiple form field values and create many PDF files from an ODBC-compliant recordset. Note: PDF must not be encrypted and table field names must match form field names (field names are case-sensitive).

strConnection As String	Connection string to use
strSQL As String	SQL SELECT statement to use
ByVal strPath As String	File path where PDFs will be created
strFile As String	File name to use (appended with sequential number)
Optional blnFlattenFields As Boolean	Flatten all the form fields?
Optional blnReadOnlyFields As Boolean	Make all the fields read only?

Returns: String ("OK" = success, otherwise an error message is returned)

SetFormFieldValue - Method to set the specified form field's value. Note: PDF must not be encrypted.

strFieldName As String	Field name to use

strFieldValue As String	Field value to use
Optional blnFlatten As Boolean	Flatten the form field?
Optional blnReadOnly As Boolean	Make the field read only?

Returns: Integer (1 = success)

SetFormPDF - Method to set the PDF form file.

Returns: Integer (1 = success)

SetInitialView - Method to set the initial view for the active PDF file. Note: PDF must not be encrypted.

ByVal lngPageMode As PageModeType   Page mode: Normal,ShowOutlines,ShowThumbnails,FullScreen

Returns: Integer (1 = success)

SetPageLayout - Method to set the page layout for the active PDF file. Note: PDF must not be encrypted.

ByVal lngPageLayout As PageLayoutType   Page layout: Single,OneColumn,TwoColumnsLeft,etc.

Returns: Integer (1 = success)

UnloadPDF - Method to unload the active PDF file and release resources.

7. Properties

Border - Get/set the border style to use for the component.

ControlBackgroundColor - Get/set the background color of the control.

DotsPerInch - Get/set the dots per inch of the PDF page to be displayed.

FitHeight - Get/set value to determine if the PDF page should fit the height of the control window.

FitWidth - Get/set value to determine if the PDF page should fit the width of the control window.

Hwnd - Get the handle number (long integer) of the control.

IsPDFloaded - Get the status of active PDF (boolean) in the control.

LinkPage - Get the active link page number that was clicked.

Page – Get/set the current page number (long integer) of the active PDF file.

PageTotal – Get the total page count (long integer) of the active PDF file.

Password – Get/set the password for the active PDF file.

Path – Get/set the file path for the active PDF file.

StoreLinks – Get/set a Boolean value to determine if page links will stored and executed when clicked.

TempPath – Get/set the folder where temporary files will be created.

8. Events

Change - Runs when the control has been changed.

Click – Runs when a mouse click is performed on the control.

DoubleClick - Runs when a mouse double-click is performed on the control.

DownloadComplete - Runs after the file has downloaded.

DownloadError - Runs if an error occurs while the file is being downloaded.

DownloadProgress - Runs while the file is being downloaded.

DownloadStarted - Runs before the file is downloaded. See BeginDownload method.

KeyDown - Runs when a key is pressed down while the control has focus.

KeyPress - Runs when a key is pressed while the control has focus.

KeyUp - Runs when a key is released while the control has focus.

MouseDown – Runs when a mouse button is pressed down on the control.

MouseMove – Runs when the mouse cursor passes over the control.

MouseUp – Runs when a mouse button is released while on the control.

Paint – Runs when the control is repainted.

PDFopened(fileName As String) – Runs after the PDF file is loaded in the control.

Resize – Runs when the control is resized.

9. Other PDF Products

To download another ActiveX PDF viewer control with more functionality than ActiveX PDF viewer but requires Adobe Reader, Foxit Reader, or PDF-XChange PDF Viewer click here:

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

To download .NET PDF Viewer for WinForms, a .NET component (does not require Adobe Reader) for VB .NET and C# software developers for viewing/manipulating PDF files click here:

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

To download PDF Maker DLL, a <mark>FREE</mark> ActiveX dynamic link library (requires Adobe Acrobat Standard or Professional) for software developers which greatly simplifies creation of PDF files from multiple file types click here:

http://www.getfilez.com/pdfmaker.exe

To download PDF Viewer .NET, a .NET component (requires Adobe Reader) for VB .NET and C# software developers for viewing/manipulating PDF files click here:

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

To download PDF Bookmarks, a <mark>FREE</mark> tool for editing and creating bookmarks in PDF files click here:

http://www.getfilez.com/pdfbm.exe

To download PDF Builder, a <mark>FREE</mark> tool for creating PDF files from various file types click here:

http://www.getfilez.com/pdfb.exe

To download Ultra PDF Tool, a <mark>FREE</mark> tool for managing and editing PDF files click here:

http://www.getfilez.com/updf.exe