

RE: VBA with Adobe

Source: <http://www.tech-archive.net/Archive/Excel/microsoft.public.excel.programming/2005-01/8073.html>

From: quartz (quartz_at_discussions.microsoft.com)

Date: 01/28/05

Date: Fri, 28 Jan 2005 10:11:04 -0800

I hope this helps. I've broken down the code into functions, so I hope you can follow it. First, you need to create a reference to "Acrobat Distiller" and "Acrobat PDFMaker". Sorry, but my code is so customized it is hard to strip it down to just the PDF stuff, but hopefully this will give you something to play with.

First it converts the file to a postscript file, then it converts the postscript file to a PDF, then there is a function that combines all the files into one PDF document. I have a lot of variables for file names and paths, but I think you will be able to tell what they are, if not, post back.

This has been stripped away from a program that is designed to deal with hundreds of files at a time. So it uses arrays a lot.

1) Code that converts files into postscript files:

```
'Get original printer name
strOriginalPrinterName = Application.ActivePrinter

'Application.ActivePrinter = "Adobe PDF on Ne01:"
ActiveWindow.SelectedSheets.PrintOut _
    Copies:=1, _
    ActivePrinter:="Adobe PDF on Ne01:", _
    PrintToFile:=True, _
    Collate:=True, _
    PrToFileName:=pstrPath_PST & strNewFileName

'Reset printer
Application.ActivePrinter = strOriginalPrinterName
```

Function that converts a postscript file into a PDF (I had to build in a pause to give distiller time to run):

```
Public Function PDFConvertPSToPDF(argPSPPath As String, argPSFileName As
String, argPDFPath As String, argPDFFileName As String)
'FUNCTION CONVERTS ADOBE POSTSCRIPT FILES (PS) TO PDF FILES;
Dim strPSFullPath As String
strPSFullPath = argPSPPath & argPSFileName
```

```
Dim strPDFFullPath As String  
strPDFFullPath = argPDFPath & argPDFFileName
```

```
Application.Wait (Now + TimeValue("0:00:01"))
```

```
Dim objDistiller As PdfDistiller  
Set objDistiller = New PdfDistiller  
objDistiller.FileToPDF strPSFullPath, strPDFFullPath, ""  
Set objDistiller = Nothing  
DoEvents
```

```
End Function
```

Function to combine all the PDF files in a folder into one PDF document:

```
Public Function PDFCombineFiles(argPDFSourcePath As String,  
argDestinationPath As String, argDestinationFileName As String)  
'FUNCTION COMBINES ALL PDF FILES IN THE SOURCE PATH INTO A SINGLE PDF  
DOCUMENT;
```

```
Dim arrPDFFileList() As Variant  
Dim strFullPath As String  
Dim intX As Integer  
Dim intLastPage As Integer  
Dim intNumPagesToInsert As Integer
```

```
'Copy all PDF files in source path into an array  
arrPDFFileList = FileList(argPDFSourcePath, "*.pdf", False)  
If IsEmpty(arrPDFFileList) Then MsgBox "No PDF files were found!",  
vbCritical, "FILES NOT FOUND!": Call CommonEnd(True)
```

```
'Dimension required objects  
Dim objAcroExchApp As Object  
Dim objAcroExchPDDoc As Object  
Dim objAcroExchInsertPDDoc As Object
```

```
'Establish object references  
Set objAcroExchApp = CreateObject("AcroExch.App")  
Set objAcroExchPDDoc = CreateObject("AcroExch.PDDoc")
```

```
'Optionally show the Acrobat Exchange window – just to see if it works  
'oAcroExchApp.Show
```

```
'Open the first file in the list  
strFullPath = argPDFSourcePath & arrPDFFileList(1)  
objAcroExchPDDoc.Open strFullPath
```

```
'Initialize a loop through each file in the PDF folder  
For intX = 1 To UBound(arrPDFFileList)
```

```
    'Concatenate source path and file name into a single string  
    strFullPath = argPDFSourcePath & arrPDFFileList(intX)
```

```
'If intX > 0 Then
If intX > 1 Then

'Sequentially open each remaining file in the directory
objAcroExchPDDoc.Open strFullPath

'Get the total pages less one for the last page num [zero based]
intLastPage = objAcroExchPDDoc.GetNumPages - 1

'Obtain an object reference to the Exchange program in PDF
Set objAcroExchInsertPDDoc = CreateObject("AcroExch.PDDoc")

'Open the file to insert
objAcroExchInsertPDDoc.Open strFullPath

'Count the pages in the current document to insert
intNumPagesToInsert = objAcroExchInsertPDDoc.GetNumPages

'Insert the pages
objAcroExchPDDoc.InsertPages intLastPage, objAcroExchInsertPDDoc, 0,
intNumPagesToInsert, True

'Close the document
objAcroExchInsertPDDoc.Close
End If
Next intX

'Save the entire document using SaveFull [0x0001 = &H1]
objAcroExchPDDoc.Save &H1, argDestinationPath & argDestinationFileName

'Close the PDDoc
objAcroExchPDDoc.Close

'Close Acrobat Exchange
objAcroExchApp.Exit

'Toss objects and release memory
Set objAcroExchApp = Nothing
Set objAcroExchPDDoc = Nothing
Set objAcroExchInsertPDDoc = Nothing

End Function
```

Hope this helps you out.

"Tony_VBACoder" wrote:

> *Could you post some of your VBA code that you use to convert an Excel sheet
> to a PDF document using Version 6+ of Acrobat. I too have been trying to
> find code that will work with Version 6+ of Acrobat, as my Acrobat 5 code no*

> *longer works since they have removed the PDFWriter printer.*
>
> *Thanks.*
>
> *"Kou Vang" wrote:*
>
> > *Has anyone ever worked with converting multiple sheets in Excel into PDF? I*
> > *have done this on a number of occasions with good results, however I have*
> > *noticed that with the newer versions of Acrobat, it is becoming increasingly*
> > *more difficult to "cut" the fat of the PDF writing code. It keeps popping up*
> > *after every conversion, and timing out when I try to run my Macro. I am*
> > *running version 6 in Acrobat. Thanks.*
> >
> > *Kou*