[Print]

Welcome to PDF Programming

by: Nick Kowalewicz

ADO, ASP, FDF, PDF, VB

Overview

This book, “PDF Programming”, describes ways to web enable fillable Adobe Acrobat forms. Programming languages used in this book are ActiveX Data Objects (ADO), Active Server Pages (ASP), FDF, PDF, Visual Basic Script (VBScript), Java-Script, and Visual Basic (VB). Some Network Administration may be needed, as well as some Database Administration. Microsoft Access is required for creating the Databases for you Adobe Acrobat Forms.

What this book tells you how to do, the PDF Suite Program already does for you. So this can be used as a reference guide to modify the code created by PDF Suite, Educational Purposes, or any other purpose you deep fit.

Although this is not nearly a complete reference guide, it will help you to get started. It took me over a year to research, create, and modify the PDF Suite application. Mostly I received a lot of help from the planetpdf.com help forum. If you haven’t already done so, you should check it out, it’s free. Just beware of people trying to sell you their products, and always wait a week to do research, and testing on different products before you buy it.

A good source for Java-Script programming can be found if you do a web search for Ted Padova. He has a lot to say in his e-books available for a small price. His demonstrations are mostly for Adobe’s Java-Scripting programming Language with a PDF, so he doesn’t speak a whole lot about the server side. A good source for Active Server Pages and ADO programming can be found at ASP101.com. Also take a look at planetsoucecode.com for ASP, and ADO programming code snippets.

Please note if you create a program from my code that’s ok too. Just mention my web site, and a reference to my name would be nice in your application also. The web site link is http://www.nk-inc.com/PDFSuite/. My name is Nick Kowalewicz.

This material is not error free, so use at your own risk. If you do not want to be responsible for your own actions please do not read any further. Nk-Inc.com and owners of NK-Inc.com are not responsible for the damages that may result to your computer systems or anyone.

Table of Contents

Active Server Pages are scripting pages served on a web server. They can perform server-side scripting as well as client-side scripting. Also known as there acronym ASP, they can be useful for handling web content other than just html pages. ASP page scripts can be executed on Microsoft Internet Information servers, as well as other web servers supporting ASP. IIS is shipped with Windows 2000, Windows XP Pro, and Windows Server operating systems. The layout of an ASP page is as follows. The first line must include the opening statement. There must be a declaration to the language used, enclosed in the beginning and closing parameter brackets. Don't forget the @ symbol on the first line. They must also use the .asp extension.

```
<%@ Language=VBSCRIPT %>
```

Next, It is wise to include the option explicit parameter.

```
<% Option Explicit %>
```

Next, to use any script besides HTML it must be included in the opening and closing brackets.
Opening Statement is less-than sign and then the percent sign
Closing Statement is the percent sign and then the greater-than sign

```
<%
Sub XXX()
	' Do Something
end Sub

Function YYY(x) as Boolean
	' Do Something
End function
%>
```

Next to make a comment use the single quotation mark before the comment.
' This is a comment in VB Script

```
Dim intX as integer ' This is also another comment
```

Or you can also use the REM statement to comment a whole line

```
REM This is yet another comment using the "REM" line
```

So an ASP page might look like this.

```
<%@ Language=VBSCRIPT %>
<% Option Explicit %>
<%
Dim intX
intX = 2
Main(intX)

Sub Main(x)
	Response.Write(x)
End Sub
%>
```

FDF and ASP (Exporting Data from FDF/PDF)

There are a few types of ways to populate a PDF Form.
Two ways are:
1. Populate by requesting Data from PDF Submit Button
2. Populate by requesting Data to a desired PDF File

The following Code illustrates the Creating, and Outputting to an FDF

```
<%@ Language=VBSCRIPT %>
<% Option Explicit %>
<%
‘ Resume to next statement on error
On error resume next

‘ Next we add our declare and set our FDF Objects, and set MIME Type(Output).
Dim FDFAcx, FDFin

‘ Set the Output MIME Type to FDF before we write anything to buffer
Response.ContentType = "application/vnd.fdf"

‘ Create FDF Application
Set FdfAcx = Server.CreateObject("FdfApp.FdfApp")

‘ Create FDF output to PDF File
Set FDFout = FdfAcx.FDFCreate

‘ Set the PDF File URL to output the FDF to (Not needed for Importing FDF)
FDFout.FDFSetFile = “http://mydomain.com/My_PDF.pdf”

‘ Next we add values to the FDF Output file
FDFout.FDFSetValue “Fieldname_One”, "Value_Of_Field_Onw"
FDFout.FDFSetValue “Fieldname_Two”, "Value_Of_Field_Two"

 ‘ Write the Status to show user the values are updated
FDFout.FDFSetStatus “Page Updated with new Values”

‘ Next we write the response to the PDF File through FDF
Response.BinaryWrite FDFout.FDFSaveToBuf

‘ Close the FDFout Object
FdfAcx.Close

‘ End the Response
Respone.End
%>
```

FDF and ASP (Importing to FDF/PDF)

There are a few types of ways to import an FDF Form.
Two ways are:
1. Importing FDF by requesting Data from PDF Submit Button
2. Importing by requesting Data from PDF File
The following Code illustrates the Creating, and Importing from an FDF

```
<%@ Language=VBSCRIPT %>
<% Option Explicit %>
<%
' Resume to next statement on error
On error resume next

' Next we add our declare and set our FDF Objects, and set MIME Type(Output).
Dim FDFAcx, FDFin
' Set the Output MIME Type to FDF before we write anything to buffer
Response.ContentType = "application/vnd.fdf"
' Create FDF Application
Set FdfAcx = Server.CreateObject("FdfApp.FdfApp")

' Create FDF output to PDF File
Set FDFout = FdfAcx.FDFCreate

' Next, we create the FDF Input from PDF File (Uncomment One of them)
' 1. Open FDF From PDF File URL
' Set FDFin=FdfAcx.FDFOpenFromFile(Server.MapPath("/PDF/my_file.pdf"))
' 2. Or Open from Buffer (PDF Submit Button)
' Set FDFin = FdfAcx.FDFOpenFromBuf(Request.BinaryRead(Request.TotalBytes))
 ' Next we get the values from the FDF file
Dim FieldValues(2)
FieldValues(1) = FDFin.FDFGetValue("Fieldname_One")
FieldValues(2) = FDFin.FDFGetValue("Fieldname_Two")

 ' Write the Status to show user the values are updated
FDFout.FDFSetStatus "Page Imported to ASP: Successful"

' Next we write the response to the PDF File through FDF
Response.BinaryWrite FDFin.FDFSaveToBuf

' Close the FDF Object and End the Response
FdfAcx.Close
Respone.End
%>
```

Access Data Objects (ADO) 2.5 or Later

Access Data Objects are an object that links the server scripts to the database.
Within the active server script you must declare the object, set the parameters, and then close the object as soon as possible.
This coding example illustrates how to import data to FDF/PDF from a database

```
<%@ Language=VBSCRIPT %>
<% Option Explicit %>
<%
On error resume next
Dim RS ‘ Declare Recordset
Dim ConnStr ‘ Declare Connection String
Dim RecordCounter
Dim FDFAcx, FDFin
Set RS = Server.CreateObject(“ADODB.Recordset”)
ConnStr = “Provider=Microsoft.Jet.OLEDB.4.0; Data Source=” & _
        Server.MapPath(“/Databases/My_Database.mdb”)

‘ Open Updatable Recordset Syntax is as follows
‘ RS.Open SQL/Table, Connection, 3=OpenKeyset, 3=LockOptimistic
RS.Open “Select * From Table_One Where Unique_ID = 2”, ConnStr, 3,3

‘ Check if a record is found
RecordCounter = RS.RecordCount
If RecordCounter < 1 then
        Response.Write “No Records Found”
        Response.End
End if
‘ Call Subroutine to perform Data Population
Perform_Query

Sub Perform_Query()
Response.ContentType = "application/vnd.fdf"
Set FdfAcx = Server.CreateObject("FdfApp.FdfApp")
Set FDFout = FdfAcx.FDFCreate
FDFout.FDFSetFile = Server.MapPath(“PDFs/My_PDF.pdf”)

‘ 1. Open FDF From File Or Open from Buffer (Uncomment One of them)
‘ Set FDFin=FdfAcx.FDFOpenFromFile(Server.MapPath(“/PDF/my_file.pdf”))
‘ Set FDFin = FdfAcx.FDFOpenFromBuf (Request.BinaryRead(Request.TotalBytes))

‘ If the field type is a string then add a empty string to the end to not throw an error
FDFout.FDFSetValue “Fieldname_One”, RS(“Fieldname_One") & “”

‘ If the field type is numeric then add zero to not throw an error
FDFout.FDFSetValue “Fieldname_Two”, RS(“Fieldname_Two”) + 0
FDFout.FDFSetStatus “Page Updated with new Values”
Response.BinaryWrite FDFout.FDFSaveToBuf
FdfAcx.Close
Respone.End
End Sub
%>
```

PDF and Active Server Pages

Active Server Pages are objects that write a specific file formats (MIME) to the web page. Whether it’s FDF or PDF, you must specify a Content Type or MIME type for the ASP to write to.
FDF Content Type is as follows:

Response.ContentType = “application/vnd.fdf”

PDF Content Type is as follows:

Response.ContentType = “application/pdf”

Since we are writing the PDF Format we use the PDF MIME for this example.
We also need a component registered with the Reg32Srv.dll (Server) called BinReadWriteBuf.dll. This object can read a binary file (PDF) and return it into a variable in an Active Server Pages Document.

```
<%@ Language=VBSCRIPT %>
<% Option Explicit %>
<%
On Error Resume Next
‘ Declare Binary Object Variables
Dim vntStream3, binObj3

‘ Turn Buffer On (Must be enabled in Server Settings)
Response.Buffer = true

‘ Declare PDF Filename Variable
Dim strPdfFileName
strPdfFileName = Server.MapPath(“/PDFs/My_PDF.pdf”)

‘ Create Binary Read Buffer Object
Set binObj3 = Server.CreateObject("binReadWriteBuf.BinRead")

‘ Set the Response MIME or Content Type = PDF
Response.ContentType = "application/pdf"

‘ Load Binary File to Object Variable
vntStream3 = binObj3.readBinFile(cPdfFileName)

‘ Write the Variable to Buffer
Response.BinaryWrite vntStream3

‘ Close Object
Set binObj3 = Nothing

‘ Clear errors
err.Clear

‘ End Response
Response.End
%>
```

F/PDF, ASP, Self-Sign Signatures and ADO (Load Signatures Subroutine)

```
Sub LOAD_SIGNATURES(byVal Form_ID,byVal Primary_Value)
On Error Resume Next
Dim vntStream3, RS_Apr3, intAppr3
Response.Buffer = true
‘ Create Recordset Variable
Set RS_Apr3 = Server.CreateObject("ADODB.Recordset")
Dim aSQL, strDBPath

‘ Set the Database Path
strDBPath = server.MapPath("../Databases/PDFData.mdb")

‘ Set the Sql statement
aSQL = "Select * From Signature_Table Where Form_ID Like '" & Form_ID & "' AND Primary_Value LIKE
'" & Primary_Value & "';"

‘ Open Recordset Object
RS_Apr3.Open aSQL, "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & strDBPath & ";", 3, 3
Dim cPdfFileName
‘ Set PDF Filename
cPdfFileName = Server.MapPath(“PDFs/My_PDF.pdf”)

Dim binObj3
‘ Open Binary Read Buffer Object
Set binObj3 = Server.CreateObject("binReadWriteBuf.BinRead")

‘ Set MIME Type to PDF
Response.ContentType = "application/pdf"

‘ Load Blank PDF First to Buffer
vntStream3 = binObj3.readBinFile(cPdfFileName)
Response.BinaryWrite vntStream3
Set binObj3 = Nothing

if RS_Apr3.RecordCount > 0 then
  RS_Apr3.MoveFirst
  ‘ Send each signature to buffer
  For intAppr = 1 to RS_Apr3.RecordCount
     vntStream3 = RS_Apr3("FormObj").GetChunk(LenB(RS_Apr3("FormObj")))
     Response.BinaryWrite vntStream3
     If Not RS_Apr3.EOF Then
        RS_Apr3.MoveNext
     End if
  Next
end if

‘ Close RecordSet
RS_Apr3.Close
Set RS_Apr3 = Nothing
err.Clear
Response.End
End Sub
```

F/PDF, ASP, Self-Sign Signatures and ADO (Save Signatures Subroutine)

```
Sub SAVE_SIGNATURES(byVal Form_ID,byVal Primary_Value)
On error resume next
Dim FDFAcx2, FDFin2
Dim objFDF2
Set FdfAcX2 = Server.CreateObject("FdfApp.FdfApp")
Set objFdf2 = FDFAcx2.FDFCreate
Dim cTempFileSaveAppends, vntStream2, binObj2
Dim RS_Apr2
' Set to Session
cTempFileSaveAppends = Server.MapPath("Databases/”& Session.SessionID & ".pdf")

Set FDFin2 = FDFAcx2.FDFOpenFromBuf(Request.BinaryRead(Request.TotalBytes))
FDFin2.FDFExtractAppendSaves cTempFileSaveAppends
Set binObj2 = Server.CreateObject("binReadWriteBuf.BinRead")
vntStream2 = binObj2.readBinFile(cTempFileSaveAppends)
Dim SIG_NAME2
SIG_NAME2 = Fdfin2.FDFGetValue("NK_sigName") & ""

IF NOT SIG_NAME2 = "" THEN
Set RS_Apr2 = Server.CreateObject("ADODB.Recordset")
strDBPath = server.MapPath("Databases/PDF_Data.mdb")

        Dim sSQL, ConnStr
sSQL = "Select * From [Signature_Tble] Where [SIGNAME] LIKE '" & SIG_NAME2 & "' AND [FormID] Like
'" & Form_ID & "' AND [PrimaryValue] LIKE '" & Primary_Value & "';"

ConnStr = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & strDBPath & ";"

        RS_Apr2.Open sSQL, ConnStr, 3, 3
IF RS_Apr2.RECORDCOUNT < 1 THEN
                RS_Apr2.AddNew
        Else
                RS_Apr2.MoveFirst
        END IF
        Response.ContentType = "application/pdf"
        RS_Apr2("FormID") = Form_ID
        RS_Apr2("PrimaryValue") = Primary_Value
        RS_Apr2("FormObj").AppendChunk(vntStream2)
        RS_Apr2("SIGNAME") = SIG_NAME2
        RS_Apr2("FormSize") = LenB(vntStream2)
        RS_Apr2("ContentType") = "application/pdf"

        if RS_Apr2("FormObj").ActualSize > 3000 then
                RS_Apr2.Update
        else
                RS_Apr2.CancelUpdate
        end if
        Set RS_Apr2 = Nothing
END IF
Set binObj2 = Nothing
Response.End
End Sub
```

Setting up the Server Resolves Issues With The Web Server

Opening Internet Information Services (IIS)

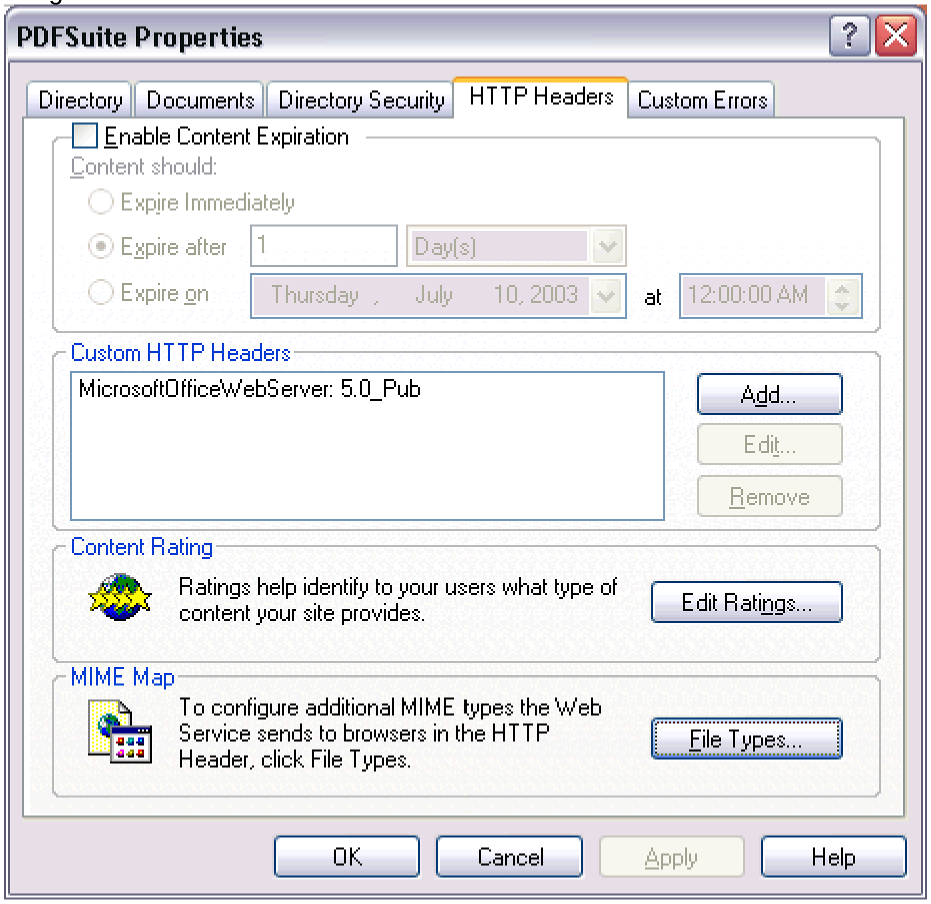
1. Click [Start, Settings, Control Panel]
2. Double Click [Administration Tools]
3. Double Click [Internet Information Services]

Setup Content Type Mime in Internet Information Services

1. Install the Server Setup Program Accompanied with PDF Suite ZIP File.
2. Run the Server Setup Program.
1. Open Internet Information Services (IIS)
2. Expand the Hierarchy Tree by clicking on [+] until you see
   the Root of the Default Web Site.
3. Right Click the Default Web Site, Select [Properties]
4. Click the Tab [HTTP Headers]
5. Click the [File Types...] Button in the Mime Map Section at the bottom (See Image A)
6. If you see a file type already there like .FDF make the following changes
   else create a new type. (See Image B)
   File Type Extension: .FDF (with a period at the beginning if it lets you)
   Content Type MIME: application/vnd.fdf

Click OK, until your at the main Interface of the Internet Information Services Panel.

Image A:


PDFSuite Properties

Directory | Documents | Directory Security | HTTP Headers | Custom Errors

☐ Enable Content Expiration
Content should:
○ Expire Immediately
● Expire after 1 Day(s)
○ Expire on Thursday , July 10, 2003 at 12:00:00 AM

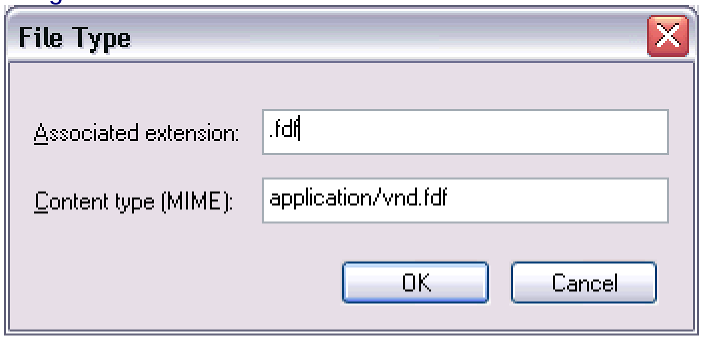
Custom HTTP Headers
MicrosoftOfficeWebServer: 5.0_Pub
Add... Edit... Remove

Content Rating
Ratings help identify to your users what type of content your site provides.
Edit Ratings...

MIME Map
To configure additional MIME types the Web Service sends to browsers in the HTTP Header, click File Types.
File Types...

OK Cancel Apply Help

Image B:



File Type

Associated extension: .fdf

Content type (MIME): application/vnd.fdf

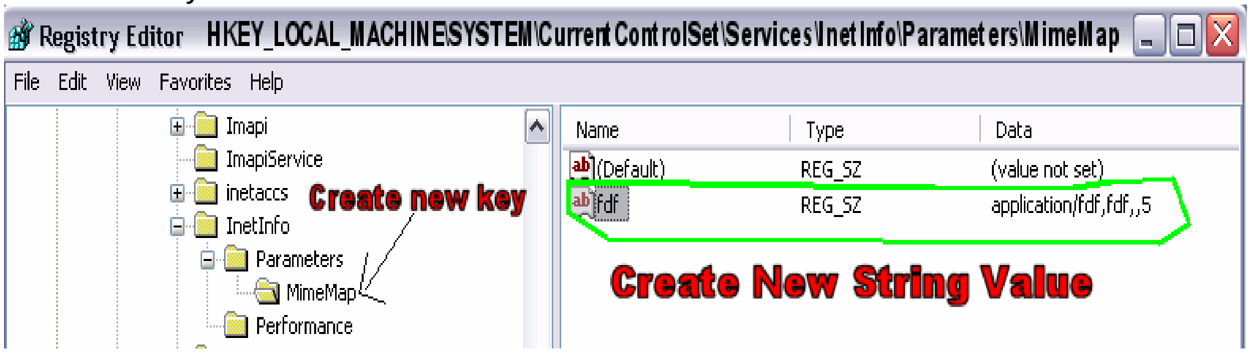
Enhancements to your registry on your server.
Open regedit.exe [Start > Run > "Regedit.exe"]

Navigate to the following sub-tree. And do the following.
HKEY_LOCAL_MACHINE\
SYSTEM\
CurrentControlSet\
Services\
InetInfo\
Parameters\
If The following key does not exist then create the key: MimeMap.
From the menu select Edit -> New -> String Value

Name: fdf

Value: application/fdf,fdf,,5

Exit Regedit.exe
Reboot the system.



Registry Editor HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\InetInfo\Parameters\MimeMap

Name	Type	Data
(Default)	REG_SZ	(value not set)
fdf	REG_SZ	application/fdf,fdf,,5

Make sure Directories have the correct IIS Permissions

1. Follow the steps for opening IIS
2. Expand the Hierarchy Tree by clicking on [+] until you see the
   main directory of you PDF Suite Web.
3. Make sure the PDF directory has read access with Scripts Only.
   (Right click on directory, select properties)
4. Make sure the Database directory has at least read & write access with <u>Scripts Only</u>.
(Right click on directory, select properties)
5. Make sure the ASP directory has at least read access with Scripts Only.
   (Right click on directory, select properties)
6. Click OK, until you close the Internet Information Services Panel

Make sure the Directories have the correct *NTFS Permissions

1. Open Windows Explorer, by Clicking [Start, Programs, Accessories],
   [Windows Explorer]
2. Explore to the Directory where the PDF Web site is.
3. Make sure Folders have correct permissions by Right Clicking on Folder,
   select [Properties]
4. Navigate to the Security Tab.
5. Set the **IUSR_MachineName Account to have at least the Following for each folder.

A. ASP Folder:	Read & Execute
B. Database Folder:	Read & Write
C. PDF Folder:	Read

Make sure the TEMP & TMP Directories have the correct *NTFS Permissions

1. Right Click on Windows Explorer Icon on your desktop, Select [Properties]
2. Look for [Environmental Variables] on one of the tabs.
3. Change the TEMP Directory, and TMP Directory reference to "C:\TEMP\"
1. Open Windows Explorer, by Clicking [Start, Programs, Accessories],
   [Windows Explorer]
2. Explore to the C:\TEMP Directory
3. Make sure Folders have correct permissions by Right Clicking on Folder
   select [Properties]
4. Navigate to the Security Tab.
5. Set the **IUSR_MachineName Account to have at least the Following for each folder.
   TEMP Folder: Modify, Read, Write, Read & Execute, List

Restart the server/workstation.

*NTFS is a format that may be used by Windows NT or Similar Computer Systems.
If your Server has FAT32 system, Then disregard the NTFS Permissions Section.
**IUSR_MachineName refers to your Internet User Guest Account,
and the Machine Name is the Name of your Machine.