Documaker

Using the PDF Print Driver

version 11.3

Skywire Software, L.L.C.
3000 Internet Boulevard
Suite 200
Frisco, Texas 75034
www.skywiresoftware.com

Phone:	**(U. S.)**	**972.377.1110**
	(EMEA)	**+44 (0) 1372 366 200**
FAX:	**(U. S.)**	**972.377.1109**
	(EMEA)	**+44 (0) 1372 366 201**
Support:	**(U. S.)**	**866.4SKYWIRE**
	(EMEA)	**+44 (0) 1372 366 222**
		support@skywiresoftware.com

PUBLICATION COPYRIGHT NOTICE

TRADEMARKS

SOFTWARE COPYRIGHT NOTICE AND COPY LIMITATIONS

PUBLICATION COPY LIMITATIONS

DISCLAIMER

The contents of this publication and the computer software it represents are subject to change without notice. Publication of this manual is not a commitment by Skywire Software or its subsidiaries to provide the features described. Neither Skywire Software nor it subsidiaries assume responsibility or liability for errors that may appear herein. Skywire Software and its subsidiaries reserve the right to revise this publication and to make changes in it from time to time without obligation of Skywire Software or its subsidiaries to notify any person or organization of such revision or changes.

The screens and other illustrations in this publication are meant to be representative, not exact duplicates, of those that appear on your monitor or printer.

Contents

Chapter 1, Setting Up the PDF Print Driver

Chapter 2, Customizing PDF Output

Chapter 3, Working With Fonts

Chapter 4, Adding Security to PDF Files

CHAPTER 1

Setting Up the PDF Print Driver

The PDF Print Driver creates Adobe Portable Document Format (PDF) files from output from Skywire Software applications such as the Policy Production System (PPS), Documaker Server, and Documaker Workstation.

PDF is a document language developed by Adobe Systems, Inc., that allows formatting and graphics information to be stored along with text. Using PDF files, you can make sure form sets viewed and printed match the originals.

A document stored in PDF format can be transmitted to and viewed on many types of systems. There are PDF viewer applications available for many platforms, both as stand-alone programs and as add-ons for existing applications (such as word processors and Internet web browsers). You can download Acrobat Reader from Adobe Systems' web site (www.adobe.com).

Print output directed to the PDF Print Driver is written to disk and stored in one or more files. You can then view these files using Acrobat Reader. This document discusses...

Installing the PDF Print Driver

First make sure you have the correct system requirements to run your Skywire Software applications. For instance, if you are using Policy Production System (PPS), see the Documaker Workstation Supervisor Guide for information on what you need to run the system. The PDF Print Driver runs on a variety of Windows 32-bit operating systems, such as Windows 2000 and Windows XP.

NOTE: The PDF Driver is also available on OS/390 platforms. Contact your sales representative for licensing and installation information.

Once you make sure your computer has the correct hardware and software, adding the ability to output PDF files from PPS requires these steps:

1. Insert the PDF Print Driver CD into your CD-ROM drive and use a text editor to view the readme file on the CD for detailed installation instructions.

   The installation routine installs the PDF DLL file into your Skywire Software application's executable directory, such as \ppswin\dll.

2. Make sure you have the following options in your FSISYS.INI file:

NOTE: *Before* making any changes to these files, back up your INI files.

```
< Control >
    LoadPrintOnly      = Yes
< PrtType:PDF >
    Device             = E:\TEST.PDF
    Bookmark           = Yes,Page
    DownloadFonts      = No,Enabled
    Module             = PDFW32
    PageNumbers        = Yes
    PrintFunc          = PDFPrint
    SendOverlays       = No,Enabled
    SendColor          = Yes,Enabled
    Class              = PDF
    DisplayMode        = UseOutlines
    PaperSize          = 0
    Linearize          = Yes
    SubsetAllEmbeddedFonts = Yes
    ForceColorBitmaps  =   No
    FontCompression    = 2
```

Option	Description
Control	
LoadPrintOnly	Enter Yes to tell the system to load print only forms.
PrtType:PDF	

Device	Enter the path and file name for the PDF output. Here is an example:
	`Device = F:\PDF\~KEYID ~DATE ; [%I-%M %p];.PDF`
	In this example, the system will write a file to the \PDF directory on the F: drive. The file name would include the policy number (KeyID), the time the file was created, and it will have an extension of *PDF*. For example, the file written to disk would have a name similar to:
	`F:\PDF\A100[3-47PM].PDF`

Bookmark	This option contains these values. Separate the values with commas (,). • The first value enables bookmarks. Enter Yes to tell the system to create bookmarks. The default is *No*. • The second value indicates the lowest level for which the system will create bookmarks. You can choose from *Formset*, *Group*, *Form*, *PageOnly*, or *Page*. For example, if you enter *Form*, the system creates bookmarks for each form set, for each group in all form sets, and for each form in all of the groups. The default is *Page*. • Use the third value to turn off generic bookmarks for all pages. For example, enter No if you do not want every page to have a bookmark like Page 01, Page 02, Page 03, and so on. If you only want TLE bookmarks to appear, enter No for the third value. • Use the fourth value to indicate the lowest level to which bookmarks can be expanded. It takes the same values as The second value and also defaults to *Page*. In the bookmark pane in Adobe Reader, only levels of bookmark above the one you specify here are expanded. If this parameter is assigned a value lower than that of the second parameter, the system automatically sets it to the value in the second parameter. If you chose PageOnly in the second parameter, this parameter is ignored. For example, this entry: `Bookmark = Yes,Page,Yes,Page` Tells the PDF Print Driver to... • Create bookmarks for each form set, group, and page • Create anonymous bookmarks • Expand the form set and group bookmarks in the Bookmark pane, but hide the page bookmarks.

Option	Description
DownloadFonts	Set to *No, Enabled*. Set to *Yes* if you want to embed fonts. See Embedding Fonts on page 45 for more information.
Module	The name of the program module which contains the system's PDF print driver. See also the Class option. See also Using defaults for the Module and PrintFunc options on page 5.
PageNumbers	Set to *No* if you do not want page numbers. Defaults to *Yes*.
PrintFunc	The name of the program function that is the main entry point into the system's PDF print driver. See also Using defaults for the Module and PrintFunc options on page 5.
SendOverlays	Set to *No, Enabled*
SendColor	Set to *Yes, Enabled*
Class	Specifies the printer classification, such as AFP, PCL, XER, PST, or GDI. If you omit this option, the system defaults to the first three letters from the Module option. Some internal functions expect a certain type of printer. For instance, all 2-up functions require an AFP printer. The internal functions check the Class option to make sure the correct printer is available before continuing.
DisplayMode	This option lets you control how the PDF file is initially displayed. To have the PDF file open... • With bookmarks (document outline) visible, enter *UseOutlines* • With thumbnail images visible, enter *UseThumbs* • In full-screen mode, with no menu bar or other window controls visible, enter *FullScreen* • In default mode, with neither bookmarks or thumbnails visible, enter *UseNone*

Option	Description
PaperSize	This option selects the paper size. The most commonly chose options are: 0=Letter (default) 1=Legal 2=A4 3=Executive 98=Custom For a list of all options, see Paper Sizes on page 15. When deciding the size the system first sets the page size to the size of the first image on the page. If the page size is *Custom*, the page size will be set to the form size. If the page size is now *Letter* (the default), the PDF Print Driver checks the PaperSize option. If the PaperSize is specified, the system uses it to determine the size. If the PaperSize option is set to *Custom* and page size is less than *Letter*, the page size will set to *Letter*. Otherwise, the system uses the custom width and height.
Linearize	Enter Yes to create linearized PDF files. See Creating Linearized PDF Files on page 28 for more information.
SubsetAllEmbeddedFonts	The default is Yes, which tells the PDF Printer Driver to embed only the font glyphs used in the document when it creates a PDF file. This is true for both normal PDF files and PDF/A-1b-compliant files. The result is smaller PDF files. If you enter No, the PDF Printer Driver embeds all of the glyphs for the fonts if embedding is also enabled. See Subsetting Fonts on page 48 for more information.
ForceColorBitmaps	Enter Yes if you want the PDF Driver to print images in color even if the images are not set to print in color (the image does have to be a color image, of course). The default is No. See Forcing the PDF Driver to Print Color Images on page 26 for more information.
FontCompression	Use this option to compress embedded fonts. Enter zero (0), 1, 2, or 3 to indicate the level of compression. Zero indicates no compression and three indicates the highest level of compression. The default is two (2).

Using defaults for the Module and PrintFunc options

Default values for the Module and PrintFunc options in the PrtType:xxx control group are provided when you use a standard print type name or print class, such as AFP, PCL, PDF, PST, VPP, XER, XMP, or GDI.

These defaults keep you from having to enter the Module and PrintFunc names in your INI file. For example, if you want to generate PDF print files, you can specify these INI options:

```
< Printer >
    PrtType     = MYPDF
< PrtType:MYAFP >
    Class       = PDF
```

And the system will default these options for you:

```
< PrtType:MYAFP >
    Module      = PDFPRT
    PrintFunc   = PDFPrint
```

3 If you have a Printers control group, simply add this option to that control group:

```
PrtType = PDF
```

If you do not have a Printers control group, add this control group and option. For example, your Printers control group might look like this:

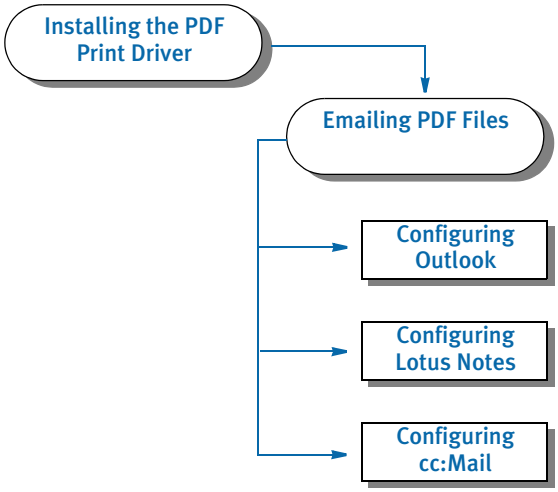
```
< Printers >
    PrtType = PDF
```

Your system can now use the PDF Print Driver. The following topic describes some additional features you can set up to work with the PDF Print Driver.

NOTE: If you are using strong encryption, you must use Acrobat 5.x or higher. Otherwise, you can use Acrobat 4.x or higher.

Additional Feature Setup

This illustration shows the various steps involved in setting up the PDF Print Driver, setting up your system to email PDF files, and configuring various email systems. It also shows how to set up the PDF Print Driver and then configure your system to send recipient output to multiple printers.



This table shows your options:

For information on	See	You must perform this task
Installing and configuring the PDF Print Driver	Installing the PDF Print Driver on page 2	Before you can email PDF files or print recipient output to multiple devices
Emailing PDF files	Emailing PDF Files on page 8	After setting up the PDF Print Driver
Configuring Microsoft Outlook	Configuring Outlook on page 9	After setting up the PDF Print Driver and setting up your system to email PDF files
Configuring Lotus Notes	Configuring Lotus Notes on page 9	After setting up the PDF Print Driver and setting up your system to email PDF files.
Configuring cc:Mail	Configuring cc:Mail on page 9	After setting up the PDF Print Driver and setting up your system to email PDF files

EMAILING PDF FILES

Once you set up the PDF Print Driver, make the following changes in your INI files to set up your system to email files.

NOTE: You can find additional information on email support in the Documaker Workstation Supervisor Guide.

Your INI files should have these settings:

```
< Printers >
    PrtType     = EPT
< PrtType:EPT >
    FileName    = F:\PDF\~KEYID ~DATE ; [%I-%M %p];.PDF
    InitFunc    = EPTInit
    KeepFile    = Yes
    Message     = PDF File attached...
    Module      = EPTW32
    PrintFunc   = EPTPrint
    PrtType     = PDF
    RecipFunc   = CSTSetMailRecip
    RecipMod    = CSTW32
    Subject     = PDF File from PPS
    TermFunc    = EPTTerm
```

The FileName option is where you specify the path and file name for the PDF file. Be sure to include the spaces and characters as noted.

In this example, the system writes a file to the \PDF directory on the F: drive. The file name includes the policy number (KeyID), the time the file was created, and an extension of *PDF*.

For example, the file written to disk might look like the one shown here:

```
F:\PDF\A100[3-47PM].PDF
```

When you finish with these changes, you can then perform one of the following tasks to configure your email system.

- Configuring Outlook on page 9
- Configuring Lotus Notes on page 9
- Configuring cc:Mail on page 9

Configuring Outlook

Once you set up the PDF Print Driver, configure your system to email PDF files, and follow the steps below, you can use the system to create a PDF file of a form set and then email that file via Outlook to a recipient.

Add these settings to your FSIUSER.INI file using a text editor:

```
< Mail >
    MailType    = MSM
< MailType:MSM >
    MailFunc    = MSMMail
    Module      = MSMW32
    Name        = MSMail
    Password    = (intentionally left blank)
    UserID      = (user ID to access mail)
```

If you are unsure of the user ID, check your control panel for mail setting or with your Administrator.

Configuring Lotus Notes

Once you set up the PDF Print Driver, configure your system to email PDF files, and follow the steps below, you can use the system to create a PDF file of a form set and then email that file via Lotus Notes to a recipient.

Add these settings to your FSIUSER.INI file using a text editor:

```
< Mail >
    MailType    = VIM
< MailType:VIM >
    MailFunc    = VMMail
    Module      = VIMW32
    Name        = Notes (Name of the Notes Server)
    Password    =
    UserID      =
```

Configuring cc:Mail

Once you set up the PDF Print Driver, configure your system to email PDF files, and follow the steps below, you can use the system to create a PDF file of a form set and then email that file via cc:Mail to a recipient.

Add these settings to your FSIUSER.INI file using a text editor:

```
< Mail >
    MailType    = CCM
< MailType:CCM >
    MailFunc    = VMMail
    Module      = VIMW32
    Name        = CCMail (Name of the mail server)
    Password    =
    UserID      =
    DataPath    =  V:\ccdata (Default path for the email system)
```

Using the PDF Print Driver with GenPrint

The GenPrint program creates the print stream for each recipient batch and sends it to a printer output file. A *batch active* flag tells PRTLIB's installed output function to keep the current file open and to append each output group to the active stream, only closing the file at the end of the batch.

In most GenPrint processing situations, this is how you want it to work. For PDF, however, you need to break each recipient record into a separate file. The following topics describe how to send each form set to a separate file.

Changing the GenPrint Program

You can use the MultiFilePrint() callback function when running the Documaker Server (GenPrint) in multi-step mode. This callback function is included with the GenPrint program and lets you create a separate file for each transaction.

NOTE: To print multiple files when you are using Documaker Server in single-step mode, use the PrintFormset rule. For more information on this rule, see the Rules Reference.

To use this callback function, you must change the FSISYS.INI file as shown below:

```
< Print >
    CallbackFunc    = MultiFilePrint
    MultiFileLog    = {full path file name of a log file (optional)}
< RunMode >
    DownloadFAP     = Yes
    LoadFAPBitmap   = Yes
    CheckNextRecip = No
```

Here is an example setup:

```
< Printer >
    PrtType         = PDF
< PrtType:PDF >
    Device          = e:\test.pdf
    DownloadFonts   = No,Disabled
    Module          = PDFW32
    PageNumbers     = Yes
    PrintFunc       = PDFPrint
    SendOverlays    = No,Disabled
    SendColor       = Yes,Enabled
< Printer1 >
    Port            = ..\RPEX1\DATA\BAT10001.PDF
< Printer2 >
    Port            = ..\RPEX1\DATA\BAT20001.PDF
... (and so on)
```

Setting the CheckNextRecip option

When you use the PDF Print Driver, make sure the CheckNextRecip INI option is set to No (the default is No.) The GenPrint program uses this option to look ahead to subsequent recipient records and queue up recipient records that match the same form set.

This improves system performance when many recipient batch records are placed in the same print batch and sorted together. However, it is essential that each recipient record be viewed as a separate print transaction for this to work. Without this option disabled, each file will contain multiple recipients for the same form set, which is probably not what you want.

NOTE: With the release of version 11.2, the default for this option is No. In prior releases, the default was Yes.

Using overlays

You cannot use overlays with the PDF Print Driver. There is no way to generate PDF overlays or use them at print time. Because of this, the GenPrint program ignores the SendOverlays option when it prints to the PDF Print Driver. FAP files and bitmaps must be loaded, which is indicated by setting these INI options:

```
DownloadFAP     = Yes
LoadFAPBitmap   = Yes
```

Using the MultiFilePrint Callback function

If you specify the MultiFileLog option in the Print control group, the specified file is created at the start of the GenPrint program when the callback is installed. The file is closed at the end of the GenPrint program when the callback is uninstalled.

At the end of each transaction, a new output file name is constructed and the GenPrint program's normal behavior of only outputting to one file is overridden. MultiFilePrint makes the following assumption about the output file name:

```
XXXX####

XXXX = four characters that are preserved.
#### = four characters set to a zero-filled sequence number.
```

MultiFilePrint assumes that the original print batch name ends in 0001. The second file opened will be 0002, and so on, up to 9999. MultiFilePrint assumes that no single recipient batch contains more than 9999 recipient batch records. If this is the case, a custom version of MultiFilePrint is required.

Avoid this approach, however, since this is a large number of output files to attempt to track and manage. MultiFilePrint does work with multiple print batches, and each batch can contain up to 9999 recipient records.

If you turned on logging, as each file is completed the system creates a log record in the log file you specified.

Using the log file

The log record has the following format:

```
;FIELD1;FIELD2;FIELD3;FIELD4;FIELD5;FIELD6;FIELD7;FIELD8;FIELD9;

FIELD1 = Logical recipient batch file name
FIELD2 = Physical (full file name) recipient batch file
FIELD3 = Group name 1 (such as Company)
FIELD4 = Group name 2 (such as LOB)
FIELD5 = Group name 3 (usually empty)
FIELD6 = TransactionId (such as Policy No.)
FIELD7 = Transaction type
FIELD8 = Recipient type (specified in the POLFILE or FORM.DAT files)
FIELD9 = Print output file (full file name)
```

The log file is provided for use by a custom application and implementation to handle the management and distribution of the many individual output files.

Caching fonts

Caching fonts lets you load commonly used fonts into memory. To enable font caching, make sure these options are in your INI file:

```
< PrtType:PDF >
    InitFunc    = PDFInit
    TermFunc    = PDFTerm
    CacheFiles  = 16
```

The default for the CacheFiles option is 16, meaning 16 fonts are kept in memory. If you set this option to a higher value, more fonts are kept in memory and more memory is devoted to storing fonts, which may slow performance.

Generating Separate Files

You can generate separate files for each transaction when you choose PDF (or RTF) from WIP or batch print.

The name of the files will have a rolling number appended to the end of the name that starts the process and is filled in on the Print window. This is automatically handled and you do not have to set INI options to get the WIP or batch print to work as long as your PrtType name is PrtType:PDF.

There are several INI options you can use to override the naming process and also name other print drivers that require this unique handling.

```
< BatchPrint >
    NoBatchSupport = PDF
    PreLoadRequired= PDF
```

These are the default settings and cannot be overridden. However, you can specify other PrtType print driver definitions you want to fall into these same categories.

Option	Description
NoBatchSupport	Indicates that the named PrtType items, separated by semicolon, do not really support batch transactions and require special handling.
PreLoadRequired	Lets you specify all the PrtType items, separated by semicolon, that should be forced to load the form set prior to the starting print. Most print drivers don't require this special requirement, but some, such as PDF do.

Also, you can name PrtType specific items under the BatchPrint control group to override the normal Device naming option. Here is an example:

```
< BatchPrint >
    PDF = ~HEXTIME .PDF
    RTF = ~HEXTIME -~KeyID .RTF
```

Any batch print sent to PrtType:PDF (picking PDF on the Print window) will override the name and store the current hexidecimal date and time, such as BCF09CA4.PDF, which is an eight-character name, as the name of each transaction's output.

Also, you can combine INI built-in calls as shown in the RTF example. Here any WIP or batch print sent to RTF will name the files using the HEXTIME and the KeyID from the WIP transaction. This will result in names similar to this: BCF099A4-123456.RTF

Note that you must leave a space after the built-in INI function name for it to work properly. That space will not appear in the resulting output name.

LIMITATIONS

The PDF Print Driver does not currently support the full set of Adobe Acrobat PDF capabilities. The following are a some of the product's limitations.

Type 1 fonts

The PDF Print Driver checks to see if the beginning of the setup data matches Courier, Dingbats, DocuDings, Helvetica, Symbol, Times, Univers, Wingdings, or ZapfDingbats. If the text matches one of the fonts, the print driver checks the font attributes for weight and style (italic or not), and uses that information to complete the mapping.

For example, if the setup data starts with *Univers*, the font has a weight of zero, and it has the italic style set, the PDF Print Driver will map it to Helvetica-Oblique.

PDF objects

Although Acrobat Reader supports variable fields, radio buttons, push buttons, and list boxes, the PDF Print Driver does not support the creation of these objects within a PDF file.

The PDF Print Driver does support hypertext links created in text labels, variable fields, and logos. You create these links in Documaker Studio as you build the sections that comprise your forms.

NOTE: Keep in mind the PDF Print Driver does support Unicode with TrueType fonts and it supports all paper sizes.

Acrobat and PDF versions

Since some settings require specific PDF versions, you should consider the version of Acrobat or Adobe Reader needed to fully support each PDF version.

This version of Acrobat	Supports
4.0 and higher	PDF version 1.3
5.0 and higher	PDF version 1.4
6.0 and higher	PDF version 1.5
7.0 and higher	PDF version 1.6
8.0 and higher	PDF version 1.7

If you are creating documents that will be distributed widely, consider only using features that only require Acrobat 5 (PDF 1.4) or Acrobat 6 (PDF 1.5). This will help make sure all users can view and print your documents correctly.

Paper Sizes

Here is a complete list of all the paper sizes you can choose from in the PaperSize INI option:

For	Enter
US letter	zero (o). This is the default.
US legal	1
ISO A4	2
US executive	3
US ledger	4
US tabloid	5
US statement	6
US folio	7
US fanfold	8
ISO A0	20
ISO A1	21
ISO A2	22
ISO A3	23
ISO A5	25
ISO A6	26
ISO A7	27
ISO A8	28
ISO A9	29
ISO A10	30
ISO 2A	32
ISO 4A	34
ISO B0	40
ISO B1	41
ISO B2	42
ISO B3	43
ISO B4	44

For	Enter
ISO B5	45
ISO B6	46
ISO B7	47
ISO B8	48
ISO B9	49
ISO B10	50
ISO 2B	52
ISO 4B	54
ISO C0	60
ISO C1	61
ISO C2	62
ISO C3	63
ISO C4	64
ISO C5	65
ISO C6	66
ISO C7	67
ISO C8	68
ISO C9	69
ISO C10	70
ISO DL	71
JIS B0	80
JIS B1	81
JIS B2	82
JIS B3	83
JIS B4	84
JIS B5	85
JIS B6	86
JIS B7	87

For	Enter
JIS B8	88
JIS B	89
JIS B10	90
custom	98

CHAPTER 2

Customizing PDF Output

There are a number of ways you customize the PDF files you produce with the PDF Print Driver.

This chapter discusses your options:

Setting PDF Compression Options

You can choose from these PDF compression methods:

Choose	For
0 (zero)	no compression
1	best speed
2	default compression
3	best compression

To override the default, add the Compression option in the PrtType:PDF control group in the DAP.INI file.

```
< PrtType:PDF >
    Compression    = 3
```

You can test the various compression options to see what works best for your implementation by comparing...

- The time difference between the request to view the transaction in Acrobat Reader and when it is displayed.
- The size of the PDF file after it is retrieved.

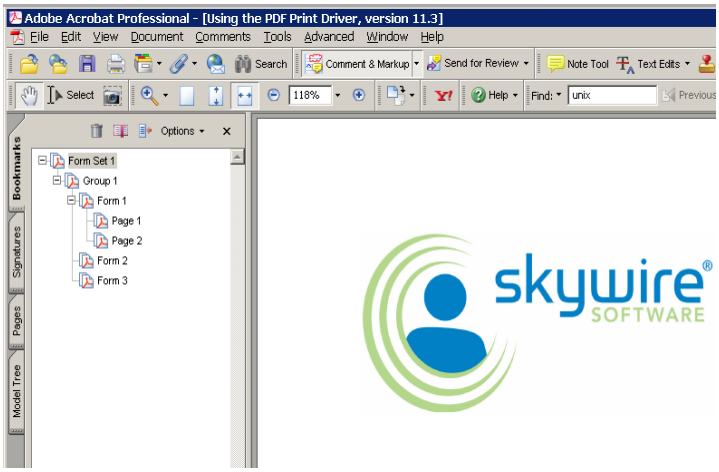
You can also control how much compression is used for fonts embedded into your PDF files. To do this, see the FontCompression option, discussed on page 5.

Setting Up Bookmarks

The PDF Print Driver sets up bookmarks at these levels by default:

- Form set - The text for this bookmark is the recipient name if applicable, otherwise it is the name of the form set.
- Group - The text for this bookmark is the name of the group.
- Form - The text for this bookmark is the description of the form. If there is no form description, the PDF Print Driver uses the name of the form.
- Page - The text for this bookmark is the name of the page.

If the text for any of the bookmarks is blank, the PDF Print Driver inserts text to describe the bookmark level, such as Form Set, Group, Form, or Page, and the index for that level. Here is an example:



Creating custom bookmarks

The PDF Print Driver uses now lets you use custom rules to create custom bookmarks in the PDF file. To do this, you must create a custom rule.

With a custom rule, you can use the *extra info* in the FAP objects to store custom bookmark titles. Currently, the system uses *extra 1* and *extra 2*, leaving *extra 3* for bookmark titles.

If you choose do develop a custom rule to use *extra 3* for this purpose, keep these things in mind:

- The setting for the Bookmark option remains the same.
- The maximum length for a custom bookmark title is 128 characters.
- This feature is not a callback, so all bookmark settings using *extra 3* must be finished before you send them to the PDF Print Driver.
- The PDF Print Driver expects to receive a character string for extra 3 and the first eight characters must be *BOOKMARK*. The actual bookmark text begins with the ninth character and is NULL terminated.
- If you do not set *extra 3* (NULL handle) or the first eight characters are not *BOOKMARK*, the PDF Print Driver ignores *extra 3* and uses its original logic to create bookmarks.

Refer to the discussion of the FAPGetExtraInfo and FAPPutExtraInfo functions in the API documentation, available from the Support site (www.skywiresoftware.com/Support), for information on getting and setting *extra3*.

Here's how the system determines the text for a bookmark in a form set:

If you are filtering by recipient, the system...

1. Checks extra3 of the recipient (128 character limit).
2. Checks extra3 of the form set (128 character limit).
3. Check the recipient name (15 character limit).

If you are not filtering by recipient, the system...

1. Checks extra3 of the form set (128 character limit).
2. Checks the form set name, which cannot exceed eight characters.

For group level bookmarks, the PDF Print Driver first checks the extra3 of the group. If there is no information in extra3, it tries to use a string with this format:

```
GROUPNAME1, GROUPNAME2, GROUPNAME3
```

For form level bookmarks, the PDF Print Driver first checks the form's extra3. If there is no information in extra3, the driver uses the form description. If there is no form description, it uses the form name.

For page level bookmarks, the PDF Print Driver checks extra3. If there is no information in extra3, the driver uses information from the page name, unless the page name is...

- Of the format "Page #"

  or

- The same as the name of the first image on the page

Interfacing with Imaging Systems

The PDF Print Driver can add free form text or data at the beginning of a batch or each form set within the batch. This can help you interface with imaging systems such as RightFax.

Use the TEXTScript option to specify the DAL script you want to run. This DAL script creates a free form data or text buffer and adds it to the print stream.

Here is an example of the DAL script:

```
* Populate the PDF stream comment with these values from RCBDFD
faxnum = trim(GVM('FaxNumber'))
faxname = trim(GVM('FaxName'))

AddComment('<TOFAXNUM:' & faxnum & '>',1)
AddComment('<TONAME:' & faxname & '>',1)

Return
```

Notice the use of the second parameter to the AddComment DAL function. The 1 indicates the string should be an ASCII string. If you omit this parameter, the system converts the string into an EBCDIC string. You can also use the TEXTCommentOn option to tell the system to add free form text or data to the beginning of every form set or print batch.

Here is an example:

```
< PrtType:PDF >
    TEXTScript      = imaging.DAL
    TEXTCommentOn   = formset
```

Producing Optimal PDF Output

To produce optimal PDF output, make the following changes in your font cross-reference (FXR) file.

NOTE: The tools used to customize a font cross-reference file are included with the full PPS system and in Documaker Server. For PPS runtime users, the people who create the libraries of form sets you use will have these tools and are responsible for setting up the FXR correctly.

- Remove font IDs built from the FORMSX font.

  This is not needed in Metacode conversions and there is no equivalent built-in Acrobat font. Furthermore, the original entry does not contain character width information.

- Fix point size settings for font IDs.

  Metacode fonts do not contain point size information. Therefore, a point size is approximated when a Metacode font is inserted into the FXR file. Sometimes, this approximation is incorrect. Many Metacode fonts include the point size as a part of its file name. For example, font ID 5 was built from the Metacode font AR07BP but is listed as having a point size of 8. However, AR07BP is really a 7-point font and the point size for this font ID should be changed to 7.00.

- Remove font IDs built from landscape fonts.

  For landscape fonts, where the equivalent portrait fonts are included in the FXR, the font IDs for the landscape fonts should be deleted and the landscape font name should be in the Rotated Fonts field of the portrait font. For example, font ID 4 was built from the landscape font AR07BL (Arial Bold 7-point). Font ID 5 was built from the portrait font AR07BP (Arial Bold 7-point). Therefore, font ID 4 was deleted and *;;AR07BL* was added to the Rotated Font Files field in the Metacode section of the Printers tab of the FXR file.

- Remove non-text fonts from the FXR file.

  You may decide to leave a non-text font in the FXR if you have an equivalent TrueType or PostScript font to embed and you have enabled font downloading. If not, remove the font IDs for non-text fonts.

NOTE: Two non-text fonts are included in the base Acrobat fonts: Symbol and Zapf Dingbats.

  If the non-text font contains a signature or graphic, such as a company logo, convert the font to a logo (LOG) file. Fonts of this style have contiguous characters and are always referenced one way. For example, the font JOHNDO might contain only the letters *A*, *B*, and *C*. When the letters *ABC* are printed together using the JOHNDO font, the signature *John Doe* is printed. When a font is always used with a single contiguous set of characters, convert the font to a logo.

For non-text fonts that contain characters which will be printed using a variety of combinations, you cannot use a logo. In this case, make the Xerox font available in the directory specified by the FontLib option in the MasterResource control group.

Examples of these types of non-text fonts include OCR, MICR, and barcode fonts. For example, a ZIP code (barcode) font produces a different picture when different ZIP codes are used. The ZIP code for 30309 looks different than the ZIP code for 49501. Using this approach, a bitmap image is created from the Xerox font using the specified characters (30309, 49501, and so on.) in the print stream. Only use this approach for fonts that are used infrequently because it results in larger PDF files which affects the time it takes to create and download a PDF file.

- Make sure color bitmaps are not saved as Comp TIFF or Comp Pack. These compression methods are not supported by PDF.

  The Comp TIFF format is designed to compress monochrome bitmaps, not color ones. The Comp Pack format is only useful when the color bitmap is 16 or 256 colors. There is no reason to use Comp Pack on a full-color (24-bit) bitmap.

  In most cases, you can simply leave full color bitmaps as JPEG or bitmap files and not convert them at all. The only reason to convert these files to the LOG format is to move them to a platform that does not support those file types, like MVS.

NOTE: The PDF driver supports monochrome, monocolor, 8-bit color (256 color) and 24-bit color logos.

- Run the FXRVALID utility to check the FXR file.

  The FXRVALID utility reports missing font files, incorrect built-in Acrobat font names, and so on. Correct any errors reported by the FXRVALID utility.

- Avoid specialty fonts when mapping to built-in Acrobat fonts.

  The built-in Acrobat fonts include Courier, Helvetica, and Times plus a couple of fonts containing non-text characters (Symbol and ZapfDingbats). Fonts such as Arial and Univers are pretty similar to Helvetica in terms of appearance and size and the built-in Acrobat font can often be used without any noticeable effect. Some font vendors also supply versions of the fonts where the characters are condensed (narrow) or expanded (wide). Although these fonts may have a similar character appearance, their size has been altered in a way that makes mapping to a built-in Acrobat fonts problematic.

Forcing the PDF Driver to Print Color Images

You can use the ForceColorBitmaps option to make sure the PDF Print Driver prints images in color, as shown here:

```
< PrtType:PDF >
    ForceColorBitmaps = Yes
```

Option	Description
ForceColorBitmaps	Enter Yes if you want the PDF Driver to print images in color even if the images are not set to print in color (the image does have to be a color image, of course). The default is No.

Before version 11.1, the PDF Print Driver always printed logos in color, regardless of how the Print in Color option is set for the logo. In version 11.1, this was changed so the PDF Print Driver now honors the Print in Color setting for logos.

So you can continue to produce PDF files with color logos without having to update your FAP files to turn on the Print in Color option for logos, this INI option was added. Essentially, this option lets you produce PDF files just as before, without having to check or reset the Print in Color option.

Reducing PDF File Sizes

You can customize the PDF Print Driver so it produces PDF files more efficiently and makes JPEG compression the default method for compressing logos (LOG files).

Depending upon the options you choose, you can see a very significant size reduction in the PDF files you produce. For instance, if you are using 24-bit color logos, you will see the greatest reduction. Also, if you are using embedded fonts and you are sub-setting the fonts instead of embedding all of them, you will see the file size decrease.

NOTE: For more information on embedding and subsetting fonts, see Working With Fonts on page 43.

If compression is turned on, and by default the PDF Print Driver uses compression method 2, 24-bit color images are compressed using the JPEG compression method. This is a *lossy* compression method. You can, however, use the JPEGCompression option to disable JPEG compression, as shown here:

```
< PrtType:PDF >
    JPEGCompression = No
```

Option	Description
JPEGCompression	Enter No to disable the default compression method (JPEG Compression) and instead use a *lossless* compression method. The default is Yes.

A lossy compression method is one where compressing data and then decompressing it retrieves data that may differ from the original, but is close enough to be useful. Lossy compression is typically used to compress multimedia data (audio, video, still images), especially in applications such as streaming media and internet telephony.

On the other hand, lossless compression is preferred for text and data files, such as bank records, text articles, and so on. Most lossy compression formats suffer from generation loss: repeatedly compressing and decompressing the file will cause it to progressively lose quality. This is in contrast with lossless data compression.

Lossless data compression allows the exact original data to be reconstructed from the compressed data. Use lossless compression when it is important that the original and the decompressed data be identical. For instance, you would use lossless compression for executable programs and source code. Some image file formats, like PNG or GIF, use only lossless compression, while others like TIFF and MNG use either lossless or lossy methods.

Creating Linearized PDF Files

Use the Linearize INI option to specify whether you want linearized or non-linearized PDF files.

With a linearized PDF file, a browser can display the first page of the PDF file before it finishes loading the entire file. This lets those who are using the PDF file start working with it sooner.

Also, if you are making the PDF files available via byte-serving web servers (servers that allow a user to request a specific range of bytes), linearization lets the user jump to any page in the PDF without having to first load all of the pages in between.

So if you are creating large PDF files which may be accessed via the internet, you probably want to linearize them.

NOTE: Sometimes this is called *optimizing a PDF file*.

```
Linearize = Yes
```

Set this option to Yes to create a linearized PDF file. Setting this option to No tells the PDF Driver to produce a non-linearized PDF file.

Adding Hypertext Links

Using Documaker Studio, you can create forms which, when output from the PDF Print Driver, contains hypertext links. These links can, for instance, launch a web browser and open a specified web site. You can create hypertext links in these types of objects in Studio:

- Text labels
- Variable fields
- Logos

For more information, see the Documaker Studio User Guide.

Meeting the PDF for Archive Specification

The PDF Print Driver complies with the requirements of the PDF for Archive (PDF/A) specification. This specification was designed by the International Standards Organization (ISO) and is intended to facilitate the long-term storage of electronic documents. The specification (ISO 19005-1) outlines the restrictions to which conforming files must adhere.

There are two sets of requirements. The PDF/A-1a set of requirements is more rigorous than the PDF/A-1b set. The Skywire Software PDF Print Driver implements PDF/A-1b compliance.

To set up your system to generate PDF/A-1b compliant documents, add the following INI option:

```
< PrtType:PDF >
    PDFAOptions = PDFA
```

Option	Description
PDFAOptions	Include this option if you want to produce PDF/A-1b compliant PDF files. Your entry identifies the INI control group that contains PDF/A-specific options. The default is PDFA.

If you use the default, the system automatically assumes the PDFA control group has the following options and values, so you do not have to actually include it — unless you want to use different values:

```
< PDFA >
    ICCProfile         = srgb.icc
    OutputCondition    = Generic CRT Monitor
    OutputConditionID  = sRGB IEC61966-2.1
```

Option	Description
ICCProfile	This specifies the name of the International Color Consortium (ICC) profile. This file describes the color attributes of a device or viewing requirement by defining a mapping between the source or target color space and a profile connection space (PCS). The file you specify must reside in the directory defined by the DefLib option in the MasterResource control group. The default is srg.icc. This file is embedded in the source code and will be used unless you include this control group and option and enter a different value in this field.
OutputCondition	A text string that describes the intended output device or production condition in human-readable form.
OutputConditionID	A text string identifier for the output condition.

NOTE: The PDFAOptions option tells the system you want to produce PDF/A-1b compliant PDF files. The system looks at the value you enter for that option and tries to find a control group by that name. If it cannot find a control group that matches, it uses the default PDFA values.

Keep in mind...

- The PDF file header must begin at byte offset zero (0) in the file. This is not a concern unless DAL scripts are used to place comments at the start of the file. If you enable PDF/A in a system that does emit comments at the beginning of the PDF file, the PDF driver will issue a warning but will still emit the comments.
- The document cannot be encrypted. If encryption and PDF/A support are both enabled, a warning appears and the document will not be encrypted.
- Colors must be specified in a device-independent manner. Skywire Software supports the RGB scheme of specifying colors, which is device-dependent.

  To implement device-independence, the PDF driver must embed an ICC profile in the file. ICC profiles are widely available for an extensive array of devices. There may be licensing requirements if color profiles are to be embedded in PDF files. Adobe offers a royalty-free license for embedding the color profiles located at:

  www.adobe.com/digitalimag/adobergb.html

  The embedded color profile must be an RGB-based profile. The Skywire Software PDF Print Driver defaults to using a file called srgb.icc, which is a generic color profile with a wide range of colors (gamut).

NOTE: ICC profiles vary in size from a few hundred bytes to a megabyte or more. If the system is configured to produce compressed PDF files, these are compressed as well. Be aware, however, that they add to the size of the file.

- To be PDF/A compliant, all fonts, TrueType or Postscript, must be embedded into the PDF file. This means the font files for every font used must be available on disk. Further, this requirement demands that all fonts used in a document be legally embeddable. It is up to you to handle any font licensing issues that may arise.

  Keep in mind that embedding fonts increases the size of the PDF file. Enabling font compression in the PDF driver will minimize this to an extent.
- The document catalog must contain a metadata key referencing a valid xmp metadata stream. The metadata stream must contain 2-4 KB of padding at the end to allow for in-place updating by applications that may not be PDF-aware. The Skywire Software driver uses 3KB. Additionally, the metadata stream must be unfiltered — specifically, uncompressed. Therefore, metadata will increase the size of PDF files by a minimum of 3kB.

NOTE: Metadata is generated by the driver and is not configurable.

Important Notice

1. Skywire Software applications may include an ICC Profile for use in producing PDF/A-compliant output. Skywire Software obtained the ICC Profile from Adobe Systems Incorporated (“Adobe”), but this ICC Profile and others can also be obtained directly from Adobe's Support Web site and then used with Skywire Software applications.

2. Skywire Software has not modified the Adobe ICC Profile, however, Customer acknowledges that (a) Adobe disclaims all warranties and conditions, express or implied; (b) Adobe excludes any and all liability for damages related to the use of the ICC Profile as provided by Skywire Software; and (c) the terms and conditions with respect to the license granted herein for the Adobe ICC Profile are offered by Skywire Software alone and not by Adobe. The ICC Profile is Copyrighted 2006 by Adobe Systems Incorporated.

Adding Digital Signature Placeholders

On June 30, 2000, the Electronic Signatures in Global and National Commerce (E-SIGN) Act was signed into law. This law provides for digital signatures to carry the same weight as their written counterparts.

Adobe's PDF format lets you add digital signatures to documents. Some vendors and companies are using PDF-based software solutions as a way of implementing electronic document signatures.

Skywire Software adds support for placing empty signature fields in PDF documents to its Documaker line of products beginning with version 11.3. In Documaker software, these empty signature fields are called *PDF signature placeholders*.

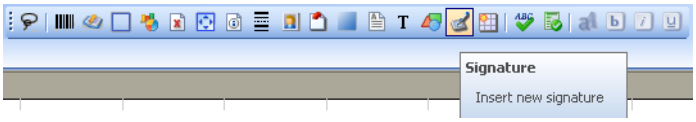
When you open a PDF file that contains a signature placeholder in Adobe Acrobat, the signature field appears as shown below. When you click on the signature field, Acrobat displays a window that lets you sign the document electronically.

While Documaker and the PDF standard allow for any number of signatures to be placed in a document, the PDF standard does not provide for signing only sections of a document. That means the digital signatures will apply to the entire document.

NOTE: Other, third-party signing tools, typically sold as plug-in extensions to Acrobat, may support multiple signatures in a document. The Documaker solution can be used with third-party signing tools, but Documaker itself does not manage the signing process for this type of signature.

Inserting a signature placeholder

To insert a signature placeholder into an image, first open the Studio's Image manager. Then select Signature from the Insert menu or click the Signature icon on the toolbar.



Your cursor changes to look like the one shown below. Use the cursor to draw a rectangle where you want the signature object to be located.

After drawing the signature, you can choose from these options for the signature object:

Option	Description
Name	(Optional) Enter up to 64 characters as a name for the signature object.
Type	The type of the signature object. The only type currently supported is PDF Placeholder, which is an empty signature field placed in PDF files.
Font ID	Specify the font with which the signed field will be displayed by Acrobat.
Color	Specify the color in which the signature text will be displayed by Acrobat. Make sure the Print in Color option has been checked.

No additional options are required for the PDF driver. The PDF driver automatically inserts the signature placeholder into the PDF file.

NOTE: This feature is supported by the PDF driver on all supported platforms. Signature objects are ignored by print drivers other than PDF.

Emulating Duplex Printing from the PDF Print Driver

You can tell the PDF Print Driver to add blank pages so it will produce output that emulates how a form set that contains both simplex and duplex forms would print on a duplex printer. You can print this PDF file from Adobe Acrobat to a Windows print driver which has duplex printing enabled. The printed document will appear to have been printed on a duplex printer. In addition, you can tell the PDF Print Driver to add FAP files onto the pages it adds.

NOTE: You must have a printer capable of duplex printing.

Here is an example of how you would set up the INI options in your MRL INI file for the PDF Print Driver:

```
< Printer >
    PrtType             = PDF
< PrtType:PDF >
    BlankPageImage      = LeftBlank
    BookMark            = Yes, Page
    Device              = data\pdfout.pdf
    DownloadFonts       = Yes,Enabled
    EmulateDuplexPrinter= Yes
    ForceColorBitmaps   = Yes
    LanguageLevel       = Level1
    Module              = PDFW32
    PageNumbers         = Yes
    PrintFunc           = PDFPrint
    SendColor           = Yes,Enabled
```

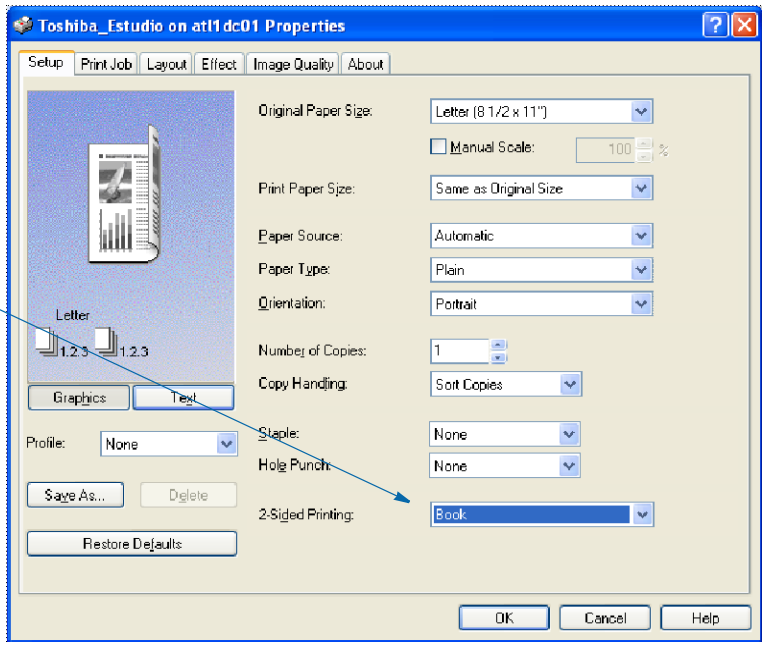
The options for this feature are discussed here:

Option	Description
BlankPageImage	(Optional) Enter the name of the FAP file you want the PDF Print Driver to insert when it needs to insert a page to emulate duplex printing. For instance, you could use this to have the PDF Print Driver insert a FAP file named LeftBlank which contained a text label that said: `This page intentionally left blank.`
EmulateDuplexPrinter	(Optional) Enter Yes if you want the PDF Print Driver to emulate a duplex printer. This means the driver will insert pages as necessary in the form set. The pages are blank by default, but you can use the BlankPageImage option to specify a FAP file to insert. The default is No.

NOTE: Do not use this feature with the AddBlankPages DAL function or the DPRAddBlankPages Docupresentment Bridge rule. This feature is used instead of these rules.

You can print the PDF file the print driver produces from Acrobat to a Windows print driver (printer) with duplex printing enabled. The printed document will appear to contain the same mixture of simplex and duplex forms. When you print the PDF file through Adobe, be sure to turn on 2-sided printing. This option will appear on the Properties tab of the Print window for your printer. Here is an example:

Be sure to turn on 2-sided printing



In this example the options for 2-sided printing include none, book (long binding), or tablet (short binding). These options will vary depending on your printer. Check your printer manual for more information.

Keep in mind, the PDF Print Driver...

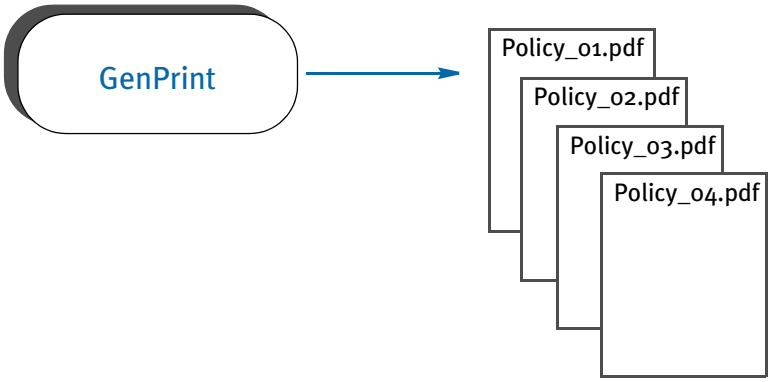
- Does not rotate pages (turn upside down) that would have printed on the back of a short bind duplex print job. PDF files are meant to be viewed.
- Does not support a mixture of long and short bind duplex for the PDF files with the blank back pages that are added. If the entire print job is a mixture of simplex and long bind duplex, you can select the long bind duplex setting for your Windows print driver. If the entire print job is a mixture of simplex and short bind duplex, you can select the short bind duplex setting for your Windows print driver.

EXAMPLES

Here are some examples of how you can use the PDF Print Driver.

Using the KeyID field to create one PDF file per transaction

This scenario shows how to set up your system so it will produce a single PDF file for each transaction and the PDF file will be named using the KeyID field and the PDF extension, such as: *policynumber.pdf*.



This scenario assumes:

- You are running Documaker Server in multi-step mode, with separate the GenTrn, GenData, and GenPrint programs executing separately
- The KeyID field (policy number) is unique throughout the entire batch run
- You have one recipient per transaction

To accomplish this, set up your system as described here:

1. Create a DAL library that contains the function to use to create the unique file name. Then copy the DAL library into DEFLIB. Assume the library illustrated below is named *SETPDFNM.DAL*.

   Here is a sample DAL library that contains two DAL scripts, one to illustrate a simple way to accomplish this, and a more complex example that can handle more possible name collisions.

```
BEGINSUB SIMPLE_NAME
* This example presumes:
*   One occurance per run of a given policy number
*   Only one recipient per transaction
*
    IF HAVEGVM("PolicyNum")
        RETURN("data\" & GVM("PolicyNum") & ".pdf")
    END
    COUNTER += 1
    RETURN("data\Emptyfile" & COUNTER & ".pdf")
ENDSUB

BEGINSUB COMPLEX_NAME
* This example is more complex and assumes
*   many possible factors should be considered
```

```
*   to prevent a name collision
*
COUNTER += 1
IF HAVEGVM("Company")
    FILENAME = GVM("Company") & \
               GVM("Lob") & \
               GVM("PolicyNum") & \
               GVM("TransactionType") & \
               GVM("RCBRcpCode") & \
               GVM("RunDate") & \
               COUNTER
ELSE
    FILENAME = "emptyfile" & COUNTER
END
RETURN("data\" & FILENAME & ".pdf")
ENDSUB
```

2 Enable the DAL library to be loaded. In your FSISYS.INI file, add the following option:

```
< DALLibraries >
    Lib = SETPDFNM
```

3 Enable the multi-file print callback function and log file. In your FSISYS.INI file, include options similar to these:

```
< Print >
    CallbackFunc = CUSMultiFilePrint
    MultiFileLog = data\pdflog.txt
```

4 Make sure PDF print is enabled. In your FSISYS.INI file, include options similar to these:

```
< PrtType:PDF >
    Module = PDFW32
    PrintFunc = PDFPrint
< Printer >
    PrtType = PDF
```

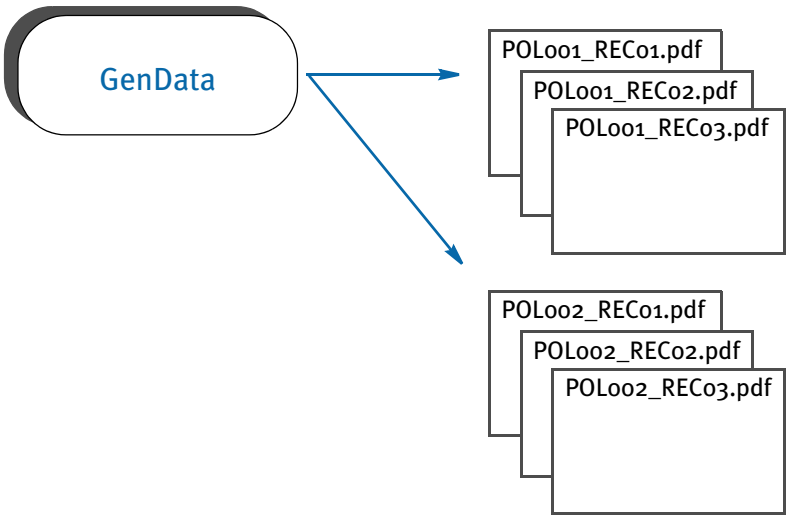
5 Make the Port option call the DAL function to get the name. In your FSISYS.INI file, change all of the Port options to something like this:

```
< Printer1 >
    Port = ~DALRUN SIMPLE_NAME
< Printer2 >
    Port = ~DALRUN SIMPLE_NAME
< Printer3 >
    Port = ~DALRUN SIMPLE_NAME
< Printer4 >
    Port = ~DALRUN SIMPLE_NAME
< Printer5 >
    Port = ~DALRUN SIMPLE_NAME
    and so on ...
```

When the GenPrint program runs, it will create a name using the KeyID (policy number) field.

Naming recipient print streams in using single-step mode

This scenario shows how to set up a master resource library (MRL) so it will produce a unique file name for each recipient of each transaction when running Documaker Server in single-step mode.



This scenario assumes:

- You have multiple recipients per transaction.
- You are using the XDB for its token lookup ability. (Using XDB is not required, it was just used in this example.)

To accomplish this, set up your MRL as described here:

1 Create a DAL library file in your MRL DefLib directory that contains the user-defined function to create unique file names. Assume the library file illustrated below is named: *unique_print_names.dal*.

This file contains two DAL user-defined functions; one illustrates using an XML file extract file, and a more complex example that can handle possible name collisions using a standard extract file. The DAL scripts are not extract specific.

```
* The xml_prt_names script assumes:
* Three unique global variables are defined in the RCBDFDFL.DAT file.
* The XDB has an entry called policy_number, whose xpath points to the
* policy number and each policy number is unique.
* Only three recipients exist.
* The PrintFormSet control group has the correct INI options defined.

BeginSub xml_prt_names
   prt_name = Lower(GetINIString(, "Printer", "PrtType"))
   pol_num1 = Trim(?("policy_number")) & ".insured." & prt_name
   pol_num2 = Trim(?("policy_number")) & ".agent."   & prt_name
   pol_num3 = Trim(?("policy_number")) & ".company." & prt_name
   SetGVM("PrtName001" ,pol_num1 ,,"C",25)
   SetGVM("PrtName002" ,pol_num2 ,,"C",25)
   SetGVM("PrtName003" ,pol_num3 ,,"C",25)
EndSub
```

```
* The std_prt_names script assumes:
* Three unique global variables are defined in the RCBDFDFL.DAT file.
* Only three recipients exist.
* The PrintFormSet control group has the correct INI options defined.

BeginSub std_prt_names
   cnt += 1
   prt_name = Lower(GetINIString(, "Printer", "PrtType"))
   pol_num1 = "insured." & #cnt & prt_name
   pol_num2 = "agent."   & #cnt & prt_name
   pol_num3 = "company." & #cnt & prt_name
   SetGVM("PrtName001" ,pol_num1 ,,"C",25)
   SetGVM("PrtName002" ,pol_num2 ,,"C",25)
   SetGVM("PrtName003" ,pol_num3 ,,"C",25)
EndSub
```

2 Make sure your FSISYS.INI or FSIUSER INI file has the following control groups and options defined. These examples assume you want to create PDF output.

Enable the DAL sub-routine library to be loaded:

```
< DALLibraries >
    Lib            = unique_print_names.dal
```

Enable the multiple file print function:

```
< PrintFormSet >
    MultiFilePrint = Yes
    LogFile        = .\data\pdflog.dat
    RCBDFDField    = PrtName
```

Make sure PostScript print is enabled:

```
< Printer >
    PrtType        = PDF
< PrtType:PDF >
    Module         = PDFW32
    PrintFunc      = PDFPrint
```

Define the Port options as follows:

```
< Insured >
    Printer        = Insured
    Port           = .\print\.pdf
< Agent >
    Printer        = Agent
    Port           = .\print\.pdf
< Company >
    Printer        = Company
    Port           = .\print\.pdf
```

3 Make sure the RCBDFDFL.DFD file has the following global variables defined.

```
FieldName = PRTName001
FieldName = PRTName002
FieldName = PRTName003

< Field:PRTName001 >
    Int_Type    = Char_Array
    Int_Length = 26
    Ext_Type    = Char_Array_No_Null_Term
    Ext_Length = 25
    Key         = N
    Required    = N
< Field:PRTName002 >
    Int_Type    = Char_Array
    Int_Length = 26
    Ext_Type    = Char_Array_No_Null_Term
    Ext_Length = 25
    Key         = N
    Required    = N
< Field:PRTName003 >
    Int_Type    = Char_Array
    Int_Length = 26
    Ext_Type    = Char_Array_No_Null_Term
    Ext_Length = 25
    Key         = N
    Required    = N
```

4 Include the following rule in your AFGJOB.DAT file to call the DAL user-defined function before each transaction is executed.

```
/* Every form set in this base uses these rules. */
<Base Form Set Rules>
;NoGenTrnTransactionProc;;;
;PrintFormset;;;
;UseXMLExtract;;;
;ResetOvFlw;;;
;BuildFormList;;;
;PreTransDAL;;xml_prt_names();
```

Keep in mind you could also use this PreTransDAL rule:

```
;PreTransDAL;;std_prt_names();
```

If you want to use the *std_prt_names* DAL script.

When the GenData program runs, it creates print output files as follows. Assume the policy numbers are: *MVF 01-12-03* and *GRA 06-22-03*.

- When using the *xml_prt_names* DAL library function:

```
MVF 01-12-03.insured.pdf
MVF 01-12-03.company.pdf
GRA 06-22-03.company.pdf
...
```

- When using the *std_prt_names* DAL library function:

```
Agent.1.pdf
Insured.1.pdf
Company.1.pdf
Agent.2.pdf
...
```

CHAPTER 3

Working With Fonts

The fonts you use determine how the text on the page looks. With PDF files, you can choose to simply use the base fonts distributed with Acrobat Reader or you can embed the actual fonts used into the PDF file. The latter approach makes sure the document represented by the PDF file looks just like the original.

Embedding fonts also makes for larger PDF files, so if file size is a greater consideration than fidelity, you may want to choose not to embed fonts or to design the document with fonts similar to those distributed with Acrobat Reader.

This chapter discusses...

Using the Base Fonts

Adobe includes the following fonts with Acrobat Reader. You do not have to embed these fonts in PDF files.

Fixed Pitch Fonts	Proportional Fonts
Courier	Helvetica
Courier-Bold	Helvetica-Bold
Courier-Oblique	Helvetica-Oblique
Courier-BoldOblique	Helvetica-BoldOblique
	Times-Roman
	Times-Bold
	Times-Italic
	Times-BoldItalic
	Symbol
	ZapfDingbats

Embedding Fonts

The PDF Print Driver lets you embed fonts into the PDF print stream. This topic discusses when to embed fonts and when not to. It also describes how the system determines which base font to use or which custom font to embed.

Generally, you want the PDF Print Driver to reproduce your document so that it looks exactly as it did when you created it. Embedding fonts lets you accomplish this if you are using fonts that do not match the criteria discussed below.

When not to embed fonts

If you do not need to reproduce the exact look of the original document, you do not need to embed fonts. If you use the base Monotype (formerly Agfa) fonts and the FXR file Skywire Software distributes, you do not need to embed fonts. The Monotype fonts are scalable and the Monotype Courier, Times, and Univers fonts are mapped to the names of the 14 base fonts Adobe distributes with Acrobat Reader.

NOTE: For a list of the base fonts, see Using the Base Fonts on page 44.

The Monotype Letter Gothic (fixed pitch, sans-serif) font is mapped to the base Adobe Courier (fixed pitch, serif) font. If you prefer the sans-serif look of Letter Gothic, you should embed that font.

Adobe also uses a standard scaling algorithm. If your implementation uses fonts that scale exactly as Adobe expects, you do not need to embed fonts. The PDF Print Driver determines the fonts to use and scales them for you. See Not Embedding Fonts on page 46 for more information.

In summary, you do not need to embed fonts if the fonts you are using ...

- Are already scalable
- Closely match the PDF base fonts

When to embed fonts

Embedding fonts lets you control the appearance of the document by letting you specify which fonts Acrobat Reader should use and what the font width will be. When you need to reproduce the exact look of the original document and you use custom fonts that do not scale exactly as Adobe expects, you should embed fonts.

To embed fonts, you need a set of PostScript Type 1 fonts or TrueType fonts, and you need to set the DownloadFonts INI option to *Yes*. You also need to run the FXRVALID utility to prepare your FXR file. For detailed instructions, see Using Embedded Fonts on page 47.

Not Embedding Fonts

If you are not going to embed fonts, you must set the following INI option to *No*, as shown here:

```
DownloadFonts = No
```

When you use a font that is not included in the 14 base fonts distributed with Acrobat Reader, the PDF Print Driver uses the information in the following fields, in this order, to determine what to do with the font:

- The PDF Print Driver checks to see if the beginning of the setup data matches Courier, Dingbats, DocuDings, Helvetica, Symbol, Times, Univers, Wingdings, or ZapfDingbats. If the text matches one of the fonts, the print driver checks the font attributes for weight and style (italic or not), and uses that information to complete the mapping.
- The PDF Print Driver checks the Setup Data field under PostScript Properties.
- The PDF Print Driver checks the TypeFace field under Screen Properties. The system checks this field to see if its contents matches one of the 14 base Adobe fonts or an equivalent Monotype font name.

Note: For a list of the base fonts, see Using the Base Fonts on page 44.

When the system matches a criteria, it then stops. If, after checking these fields, the system does not find information that matches one of the 14 base Adobe fonts or an equivalent Monotype font, it then maps...

- Proportional fonts to the Adobe Helvetica font (normal, bold, italic, or bolditalic)
- Fixed pitch or non-proportional fonts to the Adobe Courier font (normal, bold, italic, or bolditalic)

The system then checks these fields to determine additional font attributes:

- Spacing (fixed pitch or proportional)
- Style (italic or upright)
- Stroke Weight (bold or normal)

Using Embedded Fonts

If you are going to embed fonts, you must have either PostScript Type 1 or TrueType fonts. In addition, you must set the following INI option to Yes, as shown here:

```
DownloadFonts = Yes
```

You must also have the following information set up correctly for the PDF Print Driver to embed the font. If there is an error embedding a font, the PDF file is not created.

- The Embed Font field under PDF Properties is set to Yes if the field should be embedded or No if it should not be embedded.
- The Font Name field under PDF Properties must be defined. If PDF Print Driver cannot find the file name here, it checks the Font Name field under PostScript Properties. (Postscript => .PFB TrueType => .TTF). This field contains the file name of the PostScript or TrueType font you want to embed. This file should exist in the directory specified by the FontLib setting in your master resource library.

Note: The information stored in the A,R3 OTH record in the FXR appears in the font property fields in Studio's Font manager. You can edit the information there. The FXRVALID utility can also create this record. For more information about the FXRVALID utility, see the Docutoolbox Reference. You can access this manual from Skywire Software's internet site (www.skywiresoftware.com/Support/).

Subsetting Fonts

By default, the PDF Print Driver embeds only those glyphs that appear in the document when it embeds fonts. The glyphs are typographical symbols, such as letters of the alphabet and punctuation symbols. Embedding only the glyphs used in the document results in smaller PDF files.

You can, however, tell the PDF Print Driver to include all of the glyphs — not just those used in the document — by adding the SubsetAllEmbeddedFonts option to your PDF printer control group and setting it to No, as shown in this example:

```
< PrtType:PDF >
    SubsetAllEmbeddedFonts = No
```

Handling Fonts with Multiple Width Tables

When embedding resources for multiple fonts, the PDF Print Driver must decide if the fonts are related and whether the various point sizes of related fonts are scalable. If the fonts are not embedded, The PDF Print Driver compares the base Adobe font names being used. If the fonts are embedded, the PDF Print Driver must consider the source of the font.

If an FXR entry is created from a bitmap font — anything except files with one of the extensions *TTF* or *PFB* — the driver assumes the font is not scalable with respect to any other font and it will get its own width table. Furthermore, when it generates PDF/A-compliant output, the width table must match the widths embedded in the font program. If there are multiple width tables, the PDF/A standard requires that there also be multiple embedded font programs.

When importing TrueType or PostScript Type 1 font files, the PDF Print Driver recognizes that all FXR entries created from a given file are scalable and only embeds a single width table.

To optimize PDF file size, it is a good idea to create forms using FXR entries generated from TrueType or PostScript fonts.

Note: Because the PDF Print Driver can discriminate between scalable and bitmap fonts as well as scale text both horizontally and vertically, the SplitText option is no longer necessary or supported. Also, the Font Index field is no longer needed and is ignored.

Using Font Cross Reference Files

When not embedding fonts, the quality of the PDF files you create is in large part influenced by the setup information contained in the font cross-reference (FXR) file. Keep the following tips in mind when looking at your FXR file to optimize the quality of your PDF output.

PostScript font names should be present in your FXR, and all font IDs should contain one of the following PostScript font names in the Setup Data field for PostScript printing. The names of the PostScript fonts are case sensitive.

- Courier (four versions)
- Helvetica (four versions)
- Times (four versions)
- Symbol
- ZapfDingbats

The point size value should be present and should be within 33% of the font height. Font heights are measured in 2400 dots per inch while point sizes are measured in 72 dots per inch, so some conversions to equivalent units will be necessary to determine their relative values.

Note: All of the fonts listed above, except for the Univers fonts, are included in Adobe's Acrobat Reader and do not have to be embedded in PDF files.

The spacing value (either fixed or proportional) should be present and accurate. Here is a list of PostScript fonts sorted by spacing value.

Fixed pitch fonts	Proportional fonts
Courier	Helvetica
Courier-Bold	Helvetica-Bold
Courier-Oblique	Helvetica-Oblique
Courier-BoldOblique	Helvetica-BoldOblique
Courier-Italic	Times-Roman
Courier-BoldItalic	Times-Bold
	Times-Italic
	Times-BoldItalic
	Univers-Medium
	Univers-Bold
	Univers-MediumItalic
	Univers-BoldItalic
	Symbol
	ZapfDingbats

The font style value (upright or italic) should be present and accurate; it should match a PostScript font with an equivalent font style.

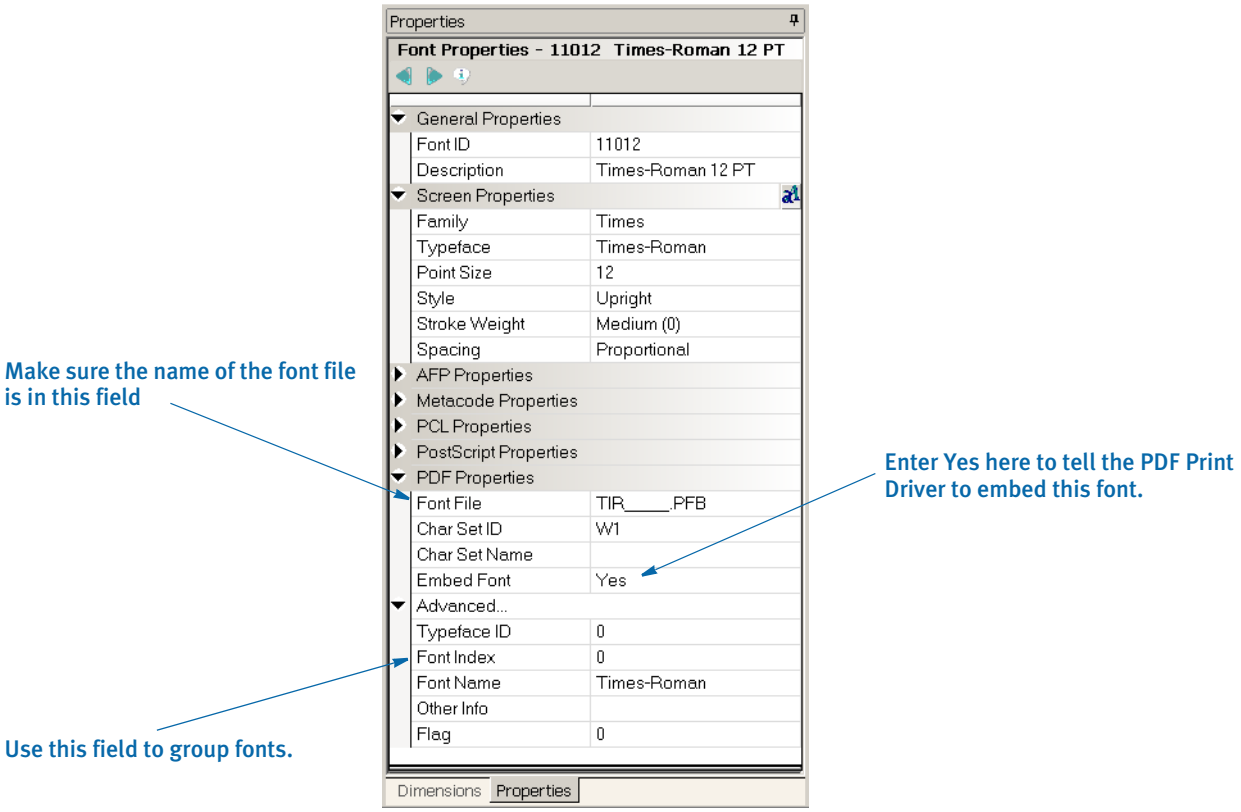
The font weight (bold or normal) should be present and accurate; it should match a PostScript font with an equivalent font weight.

How to embed fonts

Follow these instructions to embed fonts.

NOTE: You can embed TrueType or PostScript fonts. The PDF Print Driver uses the font file extensions to distinquish between the two types of fonts. TrueType fonts have a *TTF* extension. PostScript fonts have a *PFB* extension.

1. Use a text editor to open the INI file. Set the DownloadFonts option in the PrtType:PDF control group to Yes.
2. Next, use Font manager to set up your font cross reference file (FXR). When you select Fonts in Studio the Font manager opens the font cross reference file associated with your workspace. Select the font you want to embed and click Edit.
3. On the Properties tab, make the following changes:



- Enter Yes in the Embed Font field to indicate that the font should be downloaded.
- Use the Font Index field to group fonts.
- Enter the name of the font file in the Font File field.

4 Set up the remaining fields as you would for a PostScript font. Keep in mind...

- If you set the DownloadFonts option to No, all fonts are mapped to one of the 14 Type 1 base fonts and no fonts are downloaded.
- If you set the DownloadFonts option to Yes, the system downloads fonts used in the document as long as the Embed Fonts field is set to Yes.

  If you set the Embed Fonts field to No, the system maps the fonts used in the document to one of the 14 Type 1 base fonts.
- If you do not define the fields on the Properties tab, the system will not download the font — even if you set DownloadFonts option to Yes. Instead, the system maps the font to one of the 14 Type 1 fonts.
- For symbol fonts, such as DocuDings, make sure the Char Set ID field is set to *WD*.

NOTE: Each font you embed increases the size of the PDF file. See FontCompression on page 5 for information on compressing fonts.

CHAPTER 4

Adding Security to PDF Files

You can make your PDF documents more secure by encrypting them and by adding security settings. Secure PDF documents may be required for electronic bills, confidential documents (like medical records), and other documents containing sensitive material (bank statements, loan applications, and so on).

Security settings let you assign passwords for opening and modifying the document and control access to printing, editing, and annotating the document.

A document can have both an open password and an owner password and these passwords can consist of up to 32 characters. You can view a document with either type of password, but you must enter the owner password to change the document or its security settings.

If you use the security settings to restrict access to certain features, all toolbar and menu items related to those features are dimmed in Adobe Reader or Acrobat.

To enforce these restrictions, the content of the document is encrypted using a 40- or 128-bit algorithm specified by Adobe.

This chapter provides information on the following:

Configuring the Security Features

The PDF security features are built into the PDFLIB library so there are no files to install.

Configuring the INI Files

In each PrtType:XXX control group for which you want security features, add this INI option:

```
< PrtType:PDF >
    Encrypt = Yes
```

This option tells the PDF library to encrypt PDF files. Encrypting the PDF file changes the file so it is no longer easy to read when transmitted over the network. It also means you cannot use a text editor to alter the file without destroying it.

Since, however, no passwords or permissions have been specified, this option provides only a minimal amount of security. If someone gets a copy of the file, there is no way to prevent that person from viewing the file in Acrobat Reader or altering it with the full Acrobat product.

You can, however, create more secure documents by including additional options in the PDF printer control group of your INI file.

In the same control group as the Encrypt option, you can add the SecurityGroup option to specify a control group which will contain the permissions, passwords, and encryption strength. Here is an example:

```
< PrtType:PDF >
    SecurityGroup  = PDF_Encryption
    Encrypt        = Yes
< PDF_Encryption >
    .
    .
```

NOTE: While the only way to specify passwords is through INI options, the PDFKEYGEN built-in function lets you create a custom built-in that supplies the actual passwords. This way, the only thing that the INI file contains is

```
OwnerKey = ~PDFKEYGEN ~CUSTOMPASSWORDRULE
```

Keep in mind the INI files are generally inside a firewall and the passwords are stored in encrypted form.

If you have a custom password rule, you can just specify:

```
Owner = ~CUSTOMPASSWORDRULE
```

Setting Up a Security Control Group

The SecurityGroup option specifies the control group where permissions, passwords, and encryption strength are set. The permissions let you control if users can...

- Print the document. (You can also control the print quality.)
- Modify the document.
- Copy text or graphics to the clipboard.
- Add or update annotations.
- Fill in form fields.
- Access the document with accessibility tools, such as text to speech applications.
- Add navigational aids to the document.

This table shows the options you can set. All AllowXXX options default to Yes, meaning the permission is granted. See also Understanding permissions on page 60.

Option	Description
AllowPrinting	Lets the user print the document.
AllowModify	Lets the user modify the document.
AllowCopy	Lets the user copy data to the clipboard.
AllowAnnotate	Lets the user add or update annotations.
AllowFormFields	Lets the user fill in form fields. You must set the KeyLength to 128 to use this option.
AllowAccessibility	Lets accessibility tools, such as text to speech applications, access the document. You must set the KeyLength to 128 to use this option.
AllowAssembly	Lets the user add navigational elements to the document. You must set the KeyLength to 128 to use this option.
AllowHighQualityPrinting	If you set AllowPrinting to Yes, set this option to Yes to let the user generate a high quality hard copy of the document. If you set AllowPrinting to Yes and this option to No, users can only print draft quality hard copy. You must set the KeyLength to 128 to use this option.

Option	Description

OwnerKey	Specifies the password required to change the document or its security settings. The password takes the form of an encrypted 64-byte hexadecimal encoded string. See Choosing passwords on page 58 for more information.
	For additional security, instead of specifying encrypted data in the INI file, you can use custom rules to provide the passwords directly. Here is an example:
	`Owner = ~CUSTOMPASSWORDRULE`
	`User = ~CUSTOMERUSERPASSRULE`
	For less security, you can use settings similar to these to specify plain text passwords:
	`Owner = pdfowner`
	`User = pdfuser`

UserKey	Specifies the password required to open the document. The password takes the form of an encrypted 64-byte hexadecimal encoded string. See Choosing passwords on page 58 for more information.
KeyLength	Specifies the encryption strength, in bits, either 40 or 128. The default is 128. The key length must match the length provided to the PDFKey tool. The choice of key length is a primary factor in determining how secure the PDF file will be. Computers are fast enough that 40-bit encryption is susceptible to attacks in which every possible encryption key is tried. The 128-bit encryption is much more secure and allows finer control of permissions, but PDF files encrypted with a key length of 128 bits can only be viewed using Adobe Acrobat and Acrobat Reader 5.0 or later. Because Acrobat Reader is available at no cost, this restriction is generally no more than a minor inconvenience.

Choosing passwords

An encrypted version of the open and owner passwords as well as permission information is kept in the security control group. These take the form of 64-byte hexadecimal encoded strings. Here is an example:

```
6d6f62d768bc143cefa30fc4fd3cc00eb1f638157f1d985dd5fe8ebfaa7c8317
```

There are two ways to generate these keys:

- The PDFKey tool generates these keys (see Using the PDFKey Tool on page 65 for more information). Because the permissions are encoded in the keys, the permissions in the Security control group *must* match those provided to the PDFKey tool.
- You can also use the PDFKEYGEN built-in function to generate the OwnerKey and UserKey information. See Using the PDFKEYGEN Function on page 66 for more information.

You can set passwords any way you like and they can consist of up to 32 characters. Choosing the password is up to you. You can use special characters in passwords but, when using the PDFKey utility, you have to enclose the entire password in quotation marks (“) just as you would if the password contains spaces. Here is an example:

```
pdfkw32 /U="~!@#$%^&*()_+"      /O="?<>{}[]-/\|"
```

This table describes the levels of security you get based on the passwords you set up:

Add this password		
Owner	**Open**	**To provide this level of security**
Yes	Yes	You must enter the owner or open password to view the document. Users are bound by the assigned permissions You cannot change the security settings without the full Acrobat product and the owner password.
Yes	No	No password is required to view the document. Users are bound by the assigned permissions. You cannot change the security settings without the full Acrobat product and the owner password.
No	Yes	You must enter the open password to view the document Anyone with the full Acrobat product and the open password can view and change the PDF file, including the file’s security settings.
No	No	No password is required to view the document. Users are bound by the assigned permissions. Anyone with the full Acrobat product can change the security settings — so it makes little sense to set passwords in this manner.

NOTE: If you forget a password, there is no way to retrieve it from the document. Be sure to store passwords in a secure location in case you forget them.

Understanding permissions

This table provides a general overview of how permissions are related.

If this is set to True	Attempts to set the following to False are ignored
AllowModify	AllowAnnotate, AllowFormFields, and AllowAssembly
AllowAnnotate	AllowFormFields and AllowAssembly
AllowCopy	AllowAccessibility
AllowFormFields	AllowAssembly

NOTE: PDF files with security settings are called *secure* files. You cannot insert a secure PDF file into another PDF file. You can, however, insert a nonsecure PDF file into a secure PDF file. In this case, the nonsecure PDF file inherits the secure PDF file's security settings.

These tables show all permitted combinations of permissions for Adobe PDF files. If you set permissions differently than those shown below, Adobe changes them based on the information shown in these tables.

KeyLength	Modify	Annotations	FormFields	Assembly
40	Allowed	Allowed	Allowed	Allowed
40	Not allowed	Allowed	Allowed	Allowed
40	Not allowed	Not allowed	Allowed	Allowed
40	Not allowed	Not allowed	Not allowed	Allowed
128	Not allowed	Not allowed	Not allowed	Not allowed
128	Allowed	Allowed	Allowed	Allowed
128	Not allowed	Allowed	Allowed	Allowed
128	Not allowed	Not allowed	Allowed	Allowed
128	Not allowed	Not allowed	Not allowed	Allowed
128	Not allowed	Not allowed	Not allowed	Not allowed

KeyLength	Copy	Read Access
40	Allowed	Allowed
40	Not allowed	Allowed
40	Not allowed	Not allowed
128	Allowed	Allowed
128	Not allowed	Allowed
128	Not allowed	Not allowed

KeyLength	Print
40	High resolution
40	None
128	High resolution
128	Low resolution
128	None

While testing the security feature of the PDF Print Driver, Skywire Software has noted an issue with Adobe's products in which permissions set correctly in the PDF Print Driver are displayed incorrectly on the Adobe security settings window. It appears the permissions work as expected, regardless of whether those settings display correctly. This problem may be corrected in later updates to Acrobat and Reader.

There are, however, a couple of caveats that do not follow these permitted combinations:

For a 40-bit encryption

Adobe lets you set Annotation to *Not allowed*, even though Modify was set to *Allowed*. Below are the permission settings for this caveat.

```
Encryption     = Yes
KeyLength      = 40
Print          = High resolution
Modify         = Allowed
Annotation     = Not allowed
Form Fields    = Allowed
Assembly       = Allowed
Copy           = Not allowed
Accessibility  = Not allowed
```

In addition...

- When you set AllowModify to *Allowed*, AllowAnnotations remains *Not allowed*, although it should be *Allowed*.

- When you set AllowAnnotations to *Allowed*, AllowAssembly remains *Not allowed*, although it should be *Allowed*.
- When you set AllowFormFields to *Allowed*, AllowFormFields and AllowAssembly both remain *Not allowed*, although they should be *Allowed*.
- When you set AllowAccessibility to *Allowed*, it remains *Not allowed*.
- When you set AllowAssembly to *Allowed*, it remains *Not allowed*.

For a 128-bit encryption

Adobe lets you set Modify to *Allowed*, even though Annotation and Form Fields were set to *Not allowed*. Below are the permission settings for this caveat.

```
Encryption     = Yes
KeyLength      = 128
Print          = Low resolution
Modify         = Allowed
Annotation     = Not allowed
Form Fields    = Not allowed
Assembly       = Allowed
Copy           = Not allowed
Accessibility  = Not allowed
```

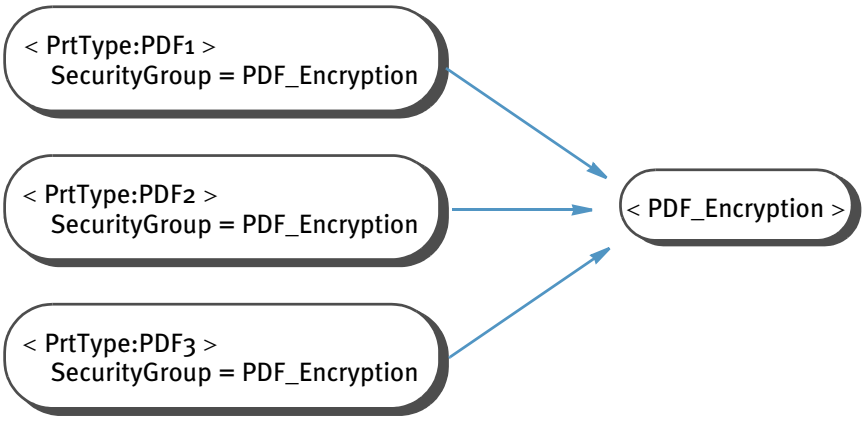
In addition...

- When you set AllowModify to *Allowed*, AllowAnnotations and AllowFormFields both remain *Not allowed*, although they be *Allowed*.
- When you set AllowCopy to *Allowed*, AllowAccessibility remains *Not allowed*.
- When you set AllowAnnotations to *Allowed*, AllowAssembly remains *Not allowed*, although it should be *Allowed*.
- When you set AllowFormFields to *Allowed*, AllowAssembly remains *Not allowed*, although it should be *Allowed*.

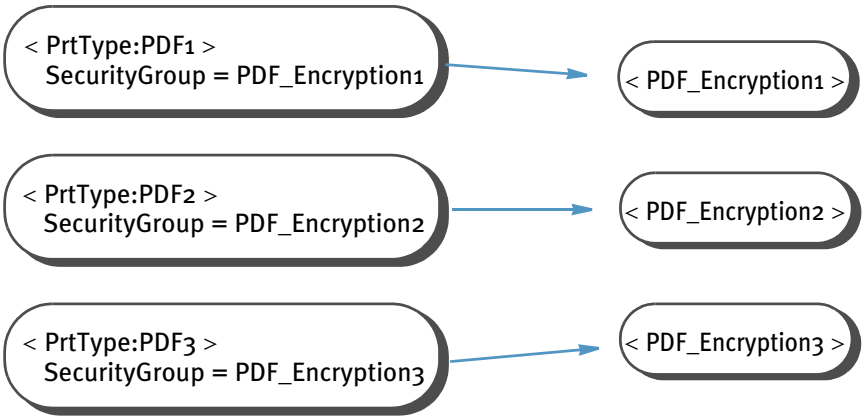
NOTE: If you encounter any other unusual combinations, you may want to contact Adobe.

Setting up multiple security groups

You can set up your PDF PrtType:XXX control groups to share a common set of security settings or have security settings tailored to each control group. This illustration shows how you would set your printer options to use the same security settings:



Or, you can have custom security settings for each printer control group:



Choose the approach that works best in your situation.

Setting up document-level security

You can implement unique passwords at the document level. All you have to do is provide a custom *~PDFPASSWORD* function (the name can vary).

For each recipient in each transaction in each batch, the GenPrint program calls the PDFPrint function in the PDFW32.DLL file. The PDFPrint function checks the Encrypt option in the PrtType:PDF control group to see if encryption is desired.

If the Encrypt option is set to Yes, the PDFPrint function calls the PDFCryptInit function to create an encryption *context*. If no passwords are passed to the PDFCryptInit function, as is the case with the PDF Print Driver, it checks the OwnerKey option:

```
< PDF_Encryption >
    OwnerKey  =
```

NOTE: Owner and User can also be used to specify passwords directly rather than specifying keys.

Keep in mind that the SecurityGroup control group must be specified in this option:

```
< PrtType:PDF>
    SecurityGroup = PDF_Encryption
```

If the OwnerKey option is set to *~PDFKEYGEN ~PDFPASSWORD*, the INI built-in function calls the ~PDFPASSWORD function, then passes the result to the ~PDFKEYGEN function.

NOTE: The ~PDFPASSWORD function varies for each implementation and is usually part of CUSLIB.

Using the PDFKey Tool

Use the PDFKey tool to generate the encrypted passwords used in the security control group.

Program names

Windows	PDFKW32.EXE
MVS	See the Documaker Server Installation Guide

Place the executable file (PDFKW32.EXE) in the directory which contains the other Skywire Software binary files.

Syntax

```
pdfkw32 /U /O /K /P /F /M /C /N /R /A /? /H
```

Parameter	Description
/U	Enter the password required to open the document. You can enter up to 32 characters. Passwords are case sensitive.
/O	Enter the password required to modify the document or its security settings. You can enter up to 32 characters. Passwords are case sensitive.
/K	Enter either 40 or 128 to specify the length of the encryption key. The default is 128.
/P	Enter No to prevent users from printing the file, L to permit low quality printing, or H to permit high quality printing.
/F	Enter No to prevent users from entering form fields. The default is Yes.
/M	Enter No to prevent users from modifying the document. The default is Yes.
/C	Enter No to prevent users from copying text from the document to the clipboard. The default is Yes.
/N	Enter No to prevent users from annotating the document. The default is Yes.
/R	Enter No to prevent users from using reader accessibility tools to view the document. The default is Yes.
/A	Enter No to prevent users from adding navigation aids, such as bookmarks. The default is Yes.
/? or /H	Enter either of these parameters to display information about the syntax and parameters you can use with this utility.

All parameters are case insensitive and can be preceded with either a backslash (\) or a dash (-). Omit spaces between an argument and its value. Passwords can, however, contain spaces. Simply enclose the entire password parameter in quotation marks, as shown here:

```
pdfkw32 "/O=Password With Spaces"
```

For more information about passwords see Choosing passwords on page 58.

Using the PDFKEYGEN Function

You can use the PDFKEYGEN built-in function to generate the OwnerKey and UserKey information.When you include a string in this form...

```
OWNERPASS/USERPASS
```

this built-in function generates the OwnerKey and the UserKey values. The OwnerKey is returned via the function call. The UserKey is added to the INI context. To omit one or both of the passwords, modify the string as shown here:

```
"/USERPASS"
 "OWNERPASS/"
 "/"
```

To use the PDFKEYGEN built-in, first create a custom built-in that returns the password string in the format described above.

To call PDFKEYGEN (assuming the custom built-in function is registered as PDFPASSWORD), specify the OwnerKey INI option as shown here:

```
OwnerKey = ~PDFKEYGEN ~PDFPASSWORD
```

You can omit the UserKey option since it will be supplied via the above call.

You can avoid using the PDFKEYGEN built-in by specifying

```
Owner =
User  =
```

But keep in mind that the custom password functions must return only the single password rather than the combination owner/user the PDFKEYGEN built-in expects.

NOTE: You can find information on writing and registering built-in functions in the API documentation at:

https://pd.docucorp.com/support/doc/rel102/api/coralib/INISyste.htm#INIRegisterFunction

Example Security Settings

Here are some examples of how you can set up your PDF security configuration. The examples include:

- Choosing permissions, passwords, and encryption strengths
- Generating keys with the PDFKey tool
- Setting the appropriate INI options

Example 1

This example shows how to set up:

- 128-bit encryption security
- An owner password of *Skywire*
- No password required to open the document
- Permissions that prevent readers from modifying information or copying information to the clipboard

Here is how you would run the PDFKey tool:

```
PDFKW32 /O=Skywire /K=128 /M=N /C=N
```

The output of the PDFKey tool is:

```
OwnerKey: e551db056412e608eeab3e5f96619ac9296b49e419467fdb8879f75f95c2a816
UserKey: 605e0bb040c9ce39d650e718b3127f9b10000000fc00000000000000000000000
```

Here are the additions you would make to your INI file, assuming that the printer type control group is called PrtType:PDF.

```
< PrtType:PDF >
  Encrypt = Y
  SecurityGroup = PDF_Encryption_Example_1
< PDF_Encryption_Example_1 >
  KeyLength = 128
  OwnerKey = e551db056412e608eeab3e5f96619ac9296b49e419467fdb8879f75f95c2a816
  UserKey = 605e0bb040c9ce39d650e718b3127f9b10000000fc00000000000000000000000
  AllowModify = No
  AllowCopy = No
```

You do not have to enter a password to view documents generated with these INI options. To modify a document, however, or to copy information from it to the clipboard, you must enter the owner password. Also, because 128-bit encryption option is used, you must have Acrobat 5.0 or later to view these documents.

Example 2

This example shows how to set up:

- 40-bit encryption security
- An owner password of *Skywire*
- A open password of *EncryptionIsFun*
- No permission restrictions

Here is how you would run the PDFKey tool:

```
pdfkw32 /o=Skywire /u=EncryptionIsFun /k=40
```

The output of the PDFKey tool is:

```
OwnerKey: 90ed4c70598767459de9523a9ce7e77ac51f4459257401fbae1936b6b1bbc7fd
UserKey: f6ac1d4b2600000000000000000000000200000058000000000000000000000000
```

Here are the additions you would make to your INI file:

```
< PrtType:PDF >
  Encrypt = Yes
  SecurityGroup = PDF_Encryption_Example_2
< PDF_Encryption_Example_2 >
  KeyLength = 40
  OwnerKey = 90ed4c70598767459de9523a9ce7e77ac51f4459257401fbae1936b6b1bbc7fd
  UserKey = f6ac1d4b2600000000000000000000000200000058000000000000000000000000
```

To view documents generated with these settings, you must enter the password *EncryptionIsFun* when prompted by Acrobat. Once the document opens, there are no restrictions.

To change the security settings, you must enter the owner password *Skywire*. Because 40-bit encryption is used, you only need Acrobat 4.0 or later to view documents created with these settings.

Example 3

This example shows how to set up:

- 128-bit encryption security
- No owner password
- A open password of *AnythingGoes*
- No permission restrictions

Here is how you would run the PDFKey tool:

```
PDFKW32 /u=AnythingGoes
```

Since the key length defaults to 128, the keys are:

```
OwnerKey: 87958ea2053c6e89302ba926dfd78a2b3d0213d8e9569734d3045a8be297370e
UserKey: 75a73844c09078ad1a587957d4328e3410000000830000000000000000000000
```

Here are the additions you would make to your INI file:

```
< PrtType:PDF >
  Encrypt = Yes
SecurityGroup = PDF_Encryption_Example_3
< PDF_Encryption_Example_3 >
  KeyLength = 128
  OwnerKey = 87958ea2053c6e89302ba926dfd78a2b3d0213d8e9569734d3045a8be297370e
  UserKey = 75a73844c09078ad1a587957d4328e3410000000830000000000000000000000
```

To view documents created with these settings, you must enter the password *AnythingGoes* when prompted. You then have unrestricted use of the document. Since there is no owner password, you can even modify security options. Because you are using 128-bit encryption, you must have Acrobat 5.0 or later to open the document.

Tips

In case you run into problems, keep in mind...

- If you cannot open a password-protected PDF file after supplying the *correct* password, this probably indicates you have errors in your setup for producing secure PDF files.

  The command line parameters used with the PDFKey tool, which produces the OwnerKey and UserKey hex strings, must match the secure PDF INI settings used when the PDF file is produced. Check your secure PDF INI settings for problems such as misspellings and INI settings that do not match the parameters used when running PDFKey. For example, specifying a 40-bit key length to PDFKey (/K=40) and using a KeyLength=128 INI setting will produce a PDF file that will not open in Acrobat.

- Using passwords instead of keys lessens the possibility errors.

- If you set the Print option (/P) to No in the PDFKey tool, you cannot set the AllowHighQualityPrinting option to No in the INI file. If you do, the result is an error, such as the one described above.

- When generating the hex key using the PDFKey tool, the system prints the output in INI format, as shown here:

  ```
  D:\rel103>pdfkw32
  < PDF_Encryption >
  KeyLength = 128
  OwnerKey = 36451bd39d753b7c1d10922c28e6665aa4f3353fb0348b536893e3b1db5c579b
  UserKey = 7880927481fd184b32c0efb547ef5adb100000006c0000000000000000000000
  ```

  The output is formatted this way so you can copy and paste these settings into your INI file.

- When configuring PDF encryption for use with IDS, you must modify two INI files: DAP.INI and the INI file for the MRL. The DAP.INI file contains all of the SecurityGroup names and SecurityGroups for all of the MRLs for which encryption is needed. The MRL's INI file contains only the SecurityGroup name and settings used by that MRL.

  You can also set up multiple SecurityGroups within one MRL. Both the DAP.INI and MRL's INI file would contain the names of all SecurityGroups in the PrtType:PDF control group as well as the individual SecurityGroups.

- If you add this INI option

  ```
  < PrtType:PDF >
      FieldTemplateAnnotations = Yes
  ```

  when templating fields, the PDF Print Driver creates annotation objects that display information about the field.

- The PDF Print Driver can embed its own INI control group within the PDF metadata for support purposes. This is enabled by adding the EmbeddedINI option, as shown here:

  ```
  < PrtType:PDF >
      EmbeddedINI = Yes
  ```

Index

A

B

C

D

E

F

G

I

J

K

L

M

N

O

P

R

S

T

U