ORACLE
INSURANCE

Oracle® Documaker

Using the Printstream Bridge

with Docupresentment version 2.2

Part number: E14902-01

May 2009

THIRD PARTY SOFTWARE NOTICES

This product includes software developed by Apache Software Foundation (http://www.apache.org/).
THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software distributed via the Berkeley Software Distribution (BSD) and licensed for binary distribution under the Generic BSD license.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the JDOM Project (http://www.jdom.org/).

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE JDOM AUTHORS OR THE PROJECT CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by the Massachusetts Institute of Technology (MIT).
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

This product includes software developed by Jean-loup Gailly and Mark Adler. This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

This software is based in part on the work of the Independent JPEG Group (http://www.ijg.org/).

This product includes software developed by Erwin Tratar. This source code and all accompanying material is copyright (c) 1998-1999 Erwin Tratar. All rights reserved.

THIS SOFTWARE IS PROVIDED "AS IS" WITHOUT EXPRESS OR IMPLIED WARRANTY. USE IT AT YOUR OWN RISK! THE AUTHOR ACCEPTS NO LIABILITY FOR ANY DAMAGE/LOSS OF BUSINESS THAT THIS PRODUCT MAY CAUSE.

This product includes software developed by Sam Leffler of Silicon Graphics.

THE SOFTWARE IS PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EXPRESS, IMPLIED OR OTHERWISE, INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IN NO EVENT SHALL SAM LEFFLER OR SILICON GRAPHICS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT OR CONSEQUENTIAL DAMAGES OF ANY KIND, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER OR NOT ADVISED OF THE POSSIBILITY OF DAMAGE, AND ON ANY THEORY OF LIABILITY, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE

Copyright (c) 1988-1997 Sam Leffler
Copyright (c) 1991-1997 Silicon Graphics, Inc.

This product includes software developed by Guy Eric Schalnat, Andreas Dilger, Glenn Randers-Pehrson (current maintainer), and others. (http://www.libpng.org)

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

This product includes software components distributed by the Cryptix Foundation.

THIS SOFTWARE IS PROVIDED BY THE CRYPTIX FOUNDATION LIMITED AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE CRYPTIX FOUNDATION LIMITED OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Copyright © 1995-2005 The Cryptix Foundation Limited. All rights reserved.

This product includes software components distributed by Sun Microsystems.

This software is provided "AS IS," without a warranty of any kind. ALLEXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANYIMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN AND ITS LICENSORS SHALL NOT BELIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Copyright (c) 1998 Sun Microsystems, Inc. All Rights Reserved.

This product includes software components distributed by Dennis M. Sosnoski.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS'" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2003-2007 Dennis M. Sosnoski. All Rights Reserved

It also includes materials licensed under Apache 1.1 and the following XPP3 license

THIS SOFTWARE IS PROVIDED "AS IS'" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Copyright © 2002 Extreme! Lab, Indiana University. All Rights Reserved

---

This product includes software components distributed by CodeProject. This software contains material that is © 1994-2005 The Ultimate Toolbox, all rights reserved.

This product includes software components distributed by Geir Landro.

Copyright © 2001-2003 Geir Landro (drop@destroydrop.com) JavaScript Tree - www.destroydrop.com/hjavascripts/tree/version 0.96

This product includes software components distributed by the Hypersonic SQL Group.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS'" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

Copyright © 1995-2000 by the Hypersonic SQL Group. All Rights Reserved

---

This product includes software components distributed by the International Business Machines Corporation and others.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) 1995-2009 International Business Machines Corporation and others. All rights reserved.

Contents

Chapter 1, Using the Printstream Bridge

Chapter 2, Printstream Bridge Rules

Appendix A, Choosing a Paper Size

Chapter 1

Using the Printstream Bridge

Docupresentment lets users connect to the server via the Internet; however, executing back-end applications requires additional components, called *bridges*. The bridge components provide software rules, document templates, and other files necessary to process documents.

The Printstream Bridge provides a link to the Documerge system which creates print-ready documentation sets. This chapter provides an overview of how the bridge works and discusses...

NOTE: Some changes may be required to your Documerge software before you can use the Printstream Bridge. These potential changes make sure output produced by the Documerge system can be read by the Printstream Bridge. Check with your sales representative for more information.

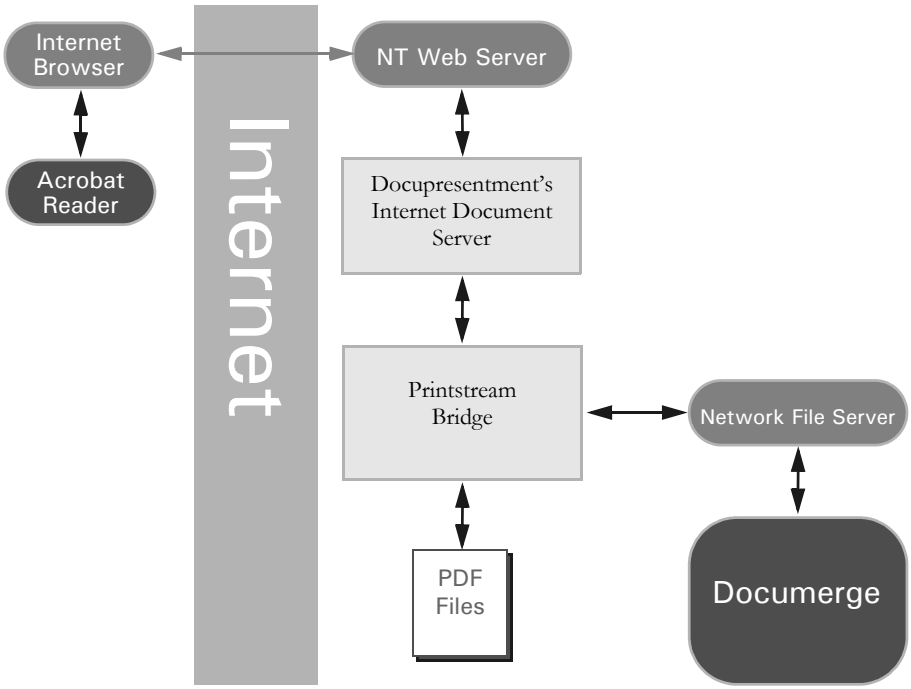
OVERVIEW

This chapter tells you how to prepare Docupresentment's Internet Document Server produce Adobe™ Portable Document Format (PDF) output from resources and then build the necessary system resources. The topics in this chapter provide the information you need to complete these tasks.

The Printstream Bridge lets users view document sets via the Internet using a web browser. Viewing the document sets requires Adobe's Acrobat Reader. The Acrobat Reader lets users view documents on-screen, just as the documents would look if they were printed.

Printstream Bridge workflow

Here is a illustration which shows how the components work together:



When a user starts his or her internet browser and connects to the Internet Document Server, a login screen appears. After logging in, the user can search for document sets using the search screen. The Internet Document Server displays the results and the user then selects a specific document set for retrieval. The selected document set is then displayed via the Acrobat Reader.

Here is a more detailed, step-by-step discussion of how it all happens:

1. The user's browser connects to the login dialog.
2. The user submits the user ID and password. Once logged in, the user's client module communicates with the Internet Document Server.
3. The client module submits the request to the server.
4. The server processes, and accepts or rejects the log-in request. The results are posted for the client module to respond to the user.

5. If the log-in request is unsuccessful, the client module sends an error to the user. If the log-in request is successful, the client module sends the first query dialog to the user.
6. The user completes and submits the first query dialog. This dialog supplies key information used in searching for the document set.
7. The client module submits the request to the server.
8. The server processes the query request, finding and returning a set of matching records.
9. The client module builds a query results dialog and returns it to the user.
10. The user either requests additional records, or selects a record.
11. Via the client module, the server receives the request and fulfills it. When a request is for a specific record, the server uses the Printstream Bridge to retrieve the document set and examine it. A list of eligible recipients for that document set is returned to the user, via the client module.
12. The user selects a specific recipient.
13. Once the request is received by the server, via the client module, the Printstream Bridge retrieves the document set and generates a PDF file. The URL reference to the temporary PDF file is returned to the user, via the server and the client module.
14. The user selects the URL of the PDF document, and the user's browser starts Acrobat Reader, which communicates to the web server, loads, and displays the PDF document set.
15. The user can view or print the displayed document using Acrobat Reader.

Checking Your Installation

To make sure the Printstream Bridge has been installed and is operating properly, this topic describes how to connect to the server using a web browser. Once connected, you can use of the sample environment to retrieve and view a form set. This will confirm that the Printstream Bridge is operating properly.

1. Make sure your NT server and web server are running. If the Internet Document Server is already running, restart it by clicking on the Restart Server Now button on the Configuration page to make sure your installation of the bridge has taken affect. If the server is not running, start the Internet Document Server by opening a DOS window on the server and entering...

   ```
   cd\docserv
   docserver.bat
   ```

   The following message appears:

   ```
   Docucorp International Internet Document Server ready!
   ```

2. Next, run the DSIEXW32.EXE program to test the Internet Document Server. You should see 30+ lines of server statistics and library data written to sysout. If you get a missing INI file diagnostic, ignore it—the defaults will suffice for this test.

   If you do not see any output or the program stalls, the connection to the server is probably not being completed. If the DSIEXW32.EXE program fails to work at all, the most cause is one or more missing DSOs. Check the files in the directory to make sure you have everything you need, then try again.

   Once you get the DOCSERVER.BAT program running, leave it running while you complete the remaining steps.

3. Start your web browser and enter the following URL:

   ```
   http://xxx.xxxxxxx.xxx/doc-html/loginmtc.htm
   ```

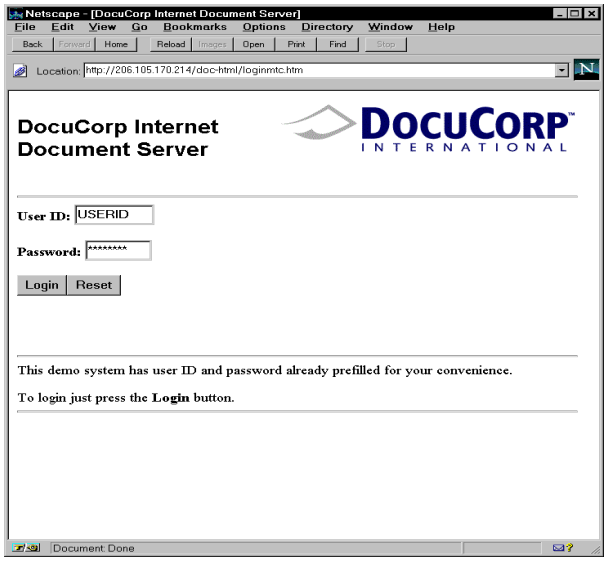
   The actual URL you enter will be reflect your web server's domain name or IP address. For example, if your web server has a domain name of www.anycompany.com, you would enter:

   ```
   http://www.anycompany.com/doc-html/loginmtc.htm
   ```

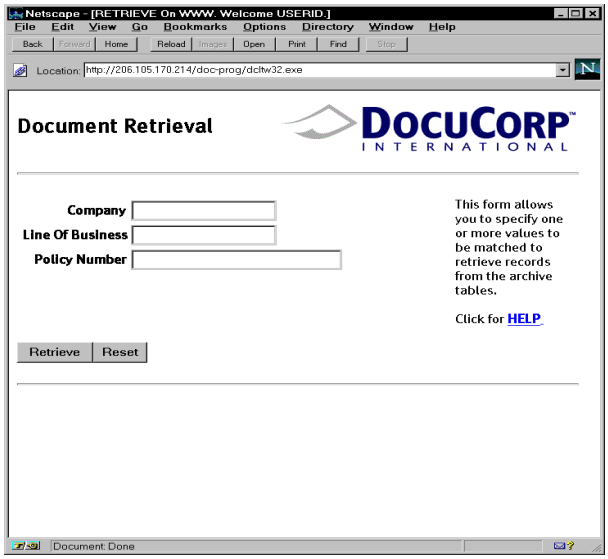
   You might also have a numeric IP address rather than a domain name. In that case, enter the numeric IP address. For example, if your IP address is 255.100.101.102, you would enter:

   ```
   http://255.100.101.102/doc-html/loginmtc.htm
   ```

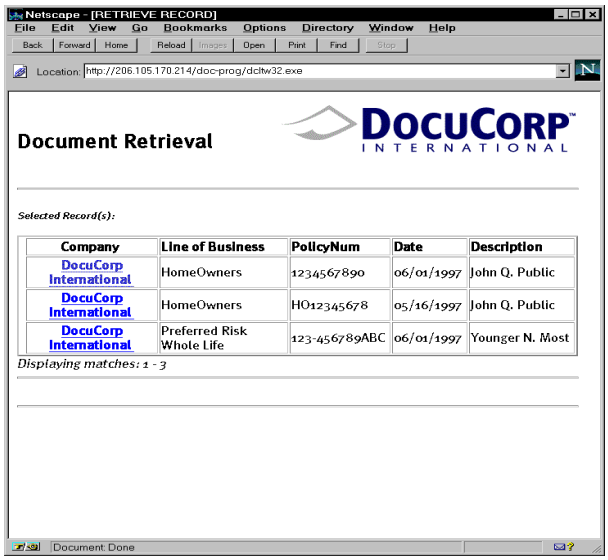
   You should now see a window similar to the following window:



This is the example login screen. The user ID and password are filled in for you. Click the Login button. A window similar to the following window appears.

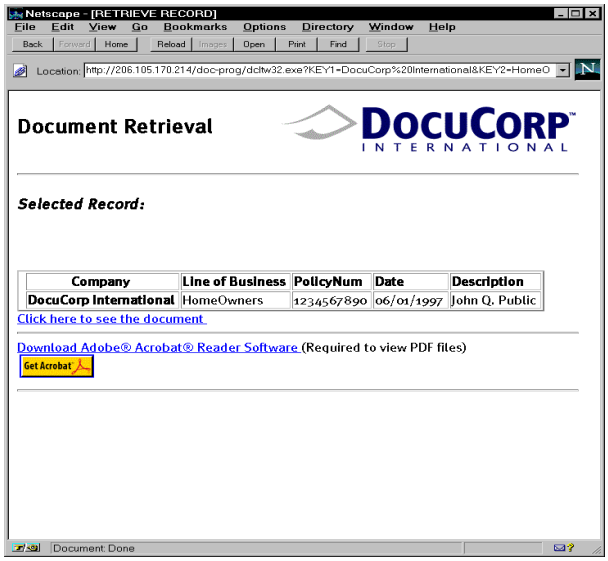


This window lets you look up specific transactions. Click the Retrieve button to see all of the examples. A window similar to the following window appears.

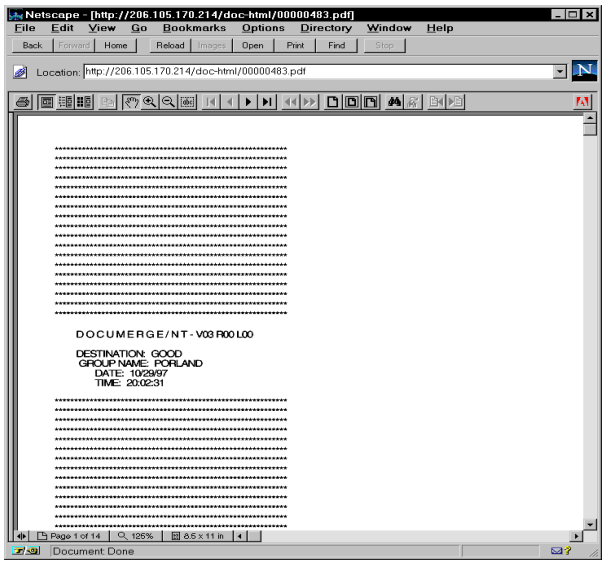


This window shows you the archived records that matched the search criteria you entered.

4 Click on the first account. You should now see a window similar to the following window.



5 Click the text *Click here to see document*. You should now see a window similar to the following window:



When this window appears, the Printstream Bridge has created an Adobe Acrobat file (PDF) of the form set you selected.

Once you perform these steps, you have verified the proper operation of your Printstream Bridge.

Getting the AFP Resources

The first step to prepare the Internet Document Server for producing Adobe™ Portable Document Format (PDF) output from AFP archives is to get a copy of the AFP resources used to produce these archives. You will need these AFP resources:

- AFP fonts (coded font, character set, and code page files)
- AFP overlays
- AFP page segments
- Archived AFP print streams

AFP Fonts

AFP fonts are designed solely for AFP printers. In IBM AFP terminology, a font is described by three components:

Coded font A coded font file contains references to specific character set and specific code page. Coded font files always begin with the letter X, such as X0DATIN8.

Code page In IBM AFP terminology, a code page file maps code points to an AFP character name in a character set file. Code page files always begin with the letter T, such as T1DOC037.

Character set A character set file contains the bitmap image of each character in the character set. Character set files always begin with the letter C (such as C0FATIN8.240 or C0FATIN8.300). The character set file name extension (240 or 300) indicates whether the bitmap images are drawn at 240 or 300 dots per inch.

You will use these AFP fonts to create a font cross-reference (FXR) file. You must have these system resources available to produce PDF files from the archived AFP print streams.

Having these AFP fonts also lets you print the AFP archives so you can establish baselines for testing. If the AFP fonts are not installed for the AFP printer you are testing with, you will need to install the AFP fonts for that printer.

NOTE: Sampling the AFP fonts will also help you fine-tune the font cross-reference (FXR) file.

AFP Overlays

You must have a copy of the AFP overlays used by the archived AFP print streams. You will use these AFP overlays to produce PDF files from the archived AFP print streams.

If the AFP overlays are not installed for the AFP printer you are testing with, you will need to install the AFP overlays for that printer.

AFP Page Segments

You must have a copy of the AFP page segments used by the archived AFP print streams. Page segments are graphics files. You will use these AFP page segments to produce PDF files from the archived AFP print streams.

If the AFP page segments are not installed for the AFP printer you are testing with, you will need to install the AFP page segments for that printer.

Archived AFP Print Streams

You must also have a copy of the archived AFP print streams to produce PDF files using the Internet Server. You can print the archived AFP print streams beforehand to establish baselines as you test.

Your next step is to build system resources, turn to Building AFP System Resources on page 12.

Getting the Metacode Resources

The first step necessary to prepare the Internet Document Server to produce Adobe Portable Document Format (PDF) output from Metacode archives is to get a copy of the Metacode resources used to produce these archives. You will need these Metacode resources:

- Xerox JSL
- Metacode fonts and images
- Archived Metacode print streams

Xerox JSL

First get a copy of the Xerox JSLs used to print the Metacode print streams before they were archived. You will use these Xerox JSLs as you configure the Internet Server's INI settings to read Metacode archives. If the JSL is not installed on the Metacode printer you are testing with, install the JSL on the printer and compile the JSL into a JDL on that printer.

You will also need to know which JDE was used within the JSL file to produce the archived Metacode print stream. Metacode print streams can switch to a different JDL/JDE than the JDL/JDE the printer was started with. If the archived Metacode print streams switch to a different JDL/JDE, the Internet Document Server's INI settings will be based on the JDL/JDE which is switched to by the archive Metacode print streams.

Once the Xerox JSL files are installed and compiled on the printer, you can print the Metacode archives to establish baselines for testing.

Metacode Fonts and Images

Next, get a copy of the Xerox fonts (FNT files) and images (IMG files) used by the archived Metacode print streams. You will use these Xerox fonts and images to create a font cross-reference (FXR) file and logo (LOG) files. You must have these system resources available to produce PDF files from the archived Metacode print streams.

The Printstream Bridge and the MRG2FAP utility let you load FRM files and IMG files referenced in the Metacode print stream being converted. The system looks for the FRM and IMG files in the directory specified by the FormLib option in the MasterResource control group. If you omit this option, the system looks in the current directory.

Having these Xerox fonts and images also lets you print the Metacode archives so you can establish baselines for testing. If the Xerox fonts and images are not installed on the Metacode printer you are testing with, you will need to install these files on that printer.

NOTE: Sampling the Xerox fonts and images will also help you fine-tune the font cross-reference (FXR) file.

Loading fonts directly

To handle Xerox fonts that contain multiple signatures or characters that must be printed vertically for the bitmap image to print correctly, the Documerge Metacode loader lets you load Xerox fonts directly.

The system loads a Xerox font when the print stream references a font that is not listed in the FXR and it cannot find a logo with the same name as the Xerox font. The system loads the Xerox font from the master resource's FontLib directory, as specified in the INI file. In addition to signature fonts, you can also use this feature to include barcode, MICR, or symbol fonts.

Using this capability slows performance and increases the size of PDF files. Do not use this capability to load all fonts if you are making PDF files—doing so causes the PDF driver to crash.

Archived Metacode Print Streams

You must also have a copy of the archived Metacode print streams to produce PDF files using the Internet Server. You can print the archived Metacode print streams beforehand to establish baselines as you test.

Your next step is to build Metacode system resources, turn to Building Metacode System Resources on page 14.

Building AFP System Resources

To build system resources, you must modify the system initialization (INI) files used by the various Skywire Software applications. The INI files you will modify are listed below:

- FSISYS.INI
- FAPCOMP.INI

System Initialization (INI) Files

You must add a PrtType:AFP control group to these INI files. This control group contains the AFP options used for the archived AFP print streams.

PrtType Control Group

Below is an example of the PrtType:AFP control group, which contains these new INI options:

```
< PrtType:AFP >
    OverlayExt = .ovr
    PageSegExt = .psg
    PaperSize  = 0
```

Option	Description
OverlayExt	Use this option to tell the system what file extension is used by the AFP overlay file names. The default is *OVL*.
PageSegExt	Use this option to tell the system what file extension is used by the AFP page segment file names. The default is *PSG*.
PaperSize	Use this option to specify the size of the paper. Here are the most commonly-used sizes: zero (0) for US letter size (default) 1 for US legal size 2 for A4 size 3 for US executive size 4 for US ledger 98 for a custom size To see a listing of all sizes, see Choosing a Paper Size on page 257.

AFP2PDF Control Group

Below is an example of the AFP2PDF control group which contains INI options used by the system

```
< AFP2PDF >
    AFPPath =
```

Option	Description
AFPPath	Defines the location of input AFP files.

Master Resource Control Group

Below is an example of the MasterResource control group that contains the INI options used by the system:

```
< CONFIG:AFPFiles >
    FormLib         = .\
< CONFIGURATIONS >
    CONFIG          = AFPFiles
< MasterResource >
    FormLib         = [CONFIG:AFPFiles] FormLib =
```

Option	Description
FormLib	Use this option to tell the system where the resource files (AFP overlays, AFP page segments, and the IBMXREF.TBL file) are stored. If not found, the system looks in the location specified in the DEFLIB option)

Building Metacode System Resources

The FSISYS.INI and FAPCOMP.INI files are system initialization files used by various Skywire Software applications. You must add a PrtType:XER control group to these INI files. This control group will contain the Xerox Metacode options used for the archived Metacode print streams.

PrtType Control Group

Below is an example of the PrtType:XER control group, which contains these options:

```
< PrtType:XER >
    DJDEIden   = A'@@@DJDE'
    DJDEOffset = 0
    DJDESkip   = 8
    OutMode    = BARR
    ImageOpt   = No
    JDEName    = DFLT
    JDLCode    = NONE
    JDLData    = 0,255
    JDLHost    = IBMONL
    JDLName    = CBA
    JDLRStack  = 0,10,EQ,X'13131313131313131313' (optional)
    JDLRPage   = 1,5,EQ,X'FFFF26FFFF'            (optional)
    PrinterInk = Blue
    PaperSize  = 0
    DefaultFont= 11010
```

Several of these INI settings are based on comparable options and values in the settings of the printer's JSL. A JSL may contain many JDLs from which to choose, or there may be multiple JSLs compiled into multiple JDLs.

An excerpt of a JDL follows, along with an explanation of each of the PrtType:XER control group options.

JDL Example

Here is an excerpt of a JDL. This excerpt is referenced in the control group options discussion.

```
CBA:    JDL;

T1:     TABLE    CONSTANT=X'121212121212121212';
T2:     TABLE    CONSTANT=X'13131313131313131313';
T3:     TABLE    CONSTANT=X'FFFF26FFFF';
C1:     CRITERIA CONSTANT=(0,9,EQ,T1);
C2:     CRITERIA CONSTANT=(0,10,EQ,T2);
C3:     CRITERIA CONSTANT=(1,5,EQ,T3);
VOLUME            HOST=IBMONL;
LINE              DATA=(0,255);
IDEN              PRE=A'@@@DJDE',
                  OFF=0,
                  SKIP=8;
ROFFSET           TEST=C1;
RSTACK            TEST=C2,DELIMITER=YES,PRINT=NONE;
RPAGETEST=C3,SIDE=NUFRONT;

/* 8.5 x 11 job */
USA1: JDE;        /* JOB can be used in place of JDE  */
OUTPUT             PAPERSIZE=USLETTER;
```

```
/* 8.5 x 14 job */
META: JOB;
VOLUME            CODE=NONE

/* Default job */
DFLT: JDE;
VOLUME            CODE=EBCDIC

END;
```

DJDEIden, DJDEOffset, and DJDESkip

These options represent the IDEN statement of the JDL. The value of the DJDEIden setting is a string constant. The types of supported string constants are ASCII (A'string'), EBCDIC (E'string'), Character ('string'), and Hex (X'string').

These types of strings are not supported: Octal, H2, and H6. Strings containing repeat counts, embedded hex values, and upper/lower case toggles are not supported. Using the JDL sample listed earlier, the INI options should be:

```
DJDEIden      = A'@@@DJDE'
DJDEOffset    = 0
DJDESkip      = 8
```

OutMode

This option indicates the output format for the Metacode data stream generated by your Documerge system. You have these options:

Enter **BARR**, if you generate output using a Windows system and then transmit that output to a Xerox printer using BARR SPOOL hardware and software. If you choose BARR, a length byte is placed at the start and end of each Metacode record.

Enter **BARRWORD** only if records longer than 255 characters can be handled by your Xerox printer.

Enter **ELIXIR** to convert Elixir-formatted Metacode print files into FAP files.

For normalized Metacode, the system supports the standard Documerge 4-byte ISI format and the 2-byte variable (ISI 2-byte) format. Enter **MRG4** to use the Documerge 4-byte ISI format. Enter **MRG2** to indicate you want to use the 2-byte variable (ISI 2-byte) format.

Enter **PCO** if you generate output using a Windows system and then transmit that output to a Xerox printer using PCO hardware and software (from Prism). When you select PCO, a 4-byte length field is placed at the start of each Metacode record.

NOTE: Skywire Software has not completely tested the PCO interface.

Enter **JES2** for MVS environments. If you will upload output generated on a Windows system to an MVS system and then transmit the output to your printer via JES2, use OutMode = JES2.

Enter **ENTIRE** if you will transmit output generated by a Windows or UNIX system to a Xerox printer via a Sun workstation using ENTIRE/FIBER GATEWAY hardware and software (from Entire, Inc.). When you choose ENTIRE, a 2-byte length field is placed at the start of each Metacode record.

Enter **LAN4235**, if you generate output for a Xerox 4235 printer attached to a network.

Here is an example of this INI option:

```
OutMode        = BARR
```

NOTE: This version assumes Metacode output produced by Documerge which does not correspond to any of the outmodes listed above. You must, however, still choose an outmode from those options listed above.

ImageOpt

Use this option to specify if the logos are saved on the Xerox printer as IMG files or as FNT files. To use IMG files, your printer must have GVG or GHO hardware installed. Also, in the JSL, you must set the Graphics option to Yes.

If you are using IMG files, set this option to Yes; otherwise set it to No. Metacode printers have a limit of 16 images on a page. Here is an example of this option:

```
ImageOpt       = No
```

JDEName

Use this option to represent the name of the job. A JDL may contain many jobs (JDEs) from which to choose. Using the JDL sample listed earlier, the Metacode job is selected using this INI setting: (This JDE must contain VOLUME CODE=NONE)

```
JDEName        = META
```

JDLCode

Use this option to represent the type of input format expected by the Xerox printer during normal operation (that is, the JDL/JDE setting used to start the printer). Character translation is performed as necessary.

The system supports EBCDIC, ASCII, or NONE, which is the same as ASCII. These formats are not supported: BCD, H2BCD, H6BCD, IBMBCD, PEBCDIC, and user-defined code translation.

Referring to the sample JSL, if the printer is normally started with STA DLFT,CBA then the JDLCode parameter must be set to CODE = EBCDIC. The INI setting must contain the value of the CODE= statement for the printer's normal operation. Here is an example of this INI option:

```
JDLCode        = EBCDIC
```

JDLData

Use this option to represent the starting position and length of the print line data within an input data record. The LINE statement contains a DATA entry which holds these values. Here is an example of this INI setting:

```
JDLData = 0,255
```

JDLHost

Use this option to tell the system whether the printer is normally on-line or off-line. You can choose from **IBMONL** (on-line) and **IBMOS** (off-line). Using the JDL example listed earlier, this INI option should be set to:

```
JDLHost = IBMONL
```

Additional settings for Xerox printers

For a Xerox print driver, specify these functions in the PrtType:XER control group:

```
OutputFunc = XEROutput
OutMetFunc = XEROutMet
InitFunc   = XERInit
```

```
TermFunc    = XERTerm
Module      = XERW32
```

JDLName

Use this option to represent the name of the JDL to use. Using the JDL sample listed earlier, this option should be set to:

```
JDLName = CBA
```

JDLRStack

Use this optional INI option to represent criteria which tells the system to send an end of report condition to the printer. In the JDL example, the RSTACK statement performed a criteria test named C2. The C2 test checks a specific part of each input line against the string named T2. If the string T2 matches an input data record at position 0 for length of 10 bytes, an end of report condition is signaled. Only CONSTANT criteria using an EQ operator is supported.

NOTE: If the printer is alternately used for Metacode and text file print jobs, you must include the JDLRStack option. We recommend always using JDLRStack.

Using the JDL sample listed earlier, this option should be set to:

```
JDLRStack      = 0,10,EQ,X'13131313131313131313'
```

JDLRPage

Use this optional INI option to represent the criteria which signals a jump to the front side of a new sheet to the printer. In the JDL sample listed earlier, the RPAGE statement performed a criteria test named C3. The C3 test checks a specific part of each input line against the string named T3. If the string T3 matches an input data record at position zero (0) for a length of 5 bytes, a *jump to new sheet* condition is signaled because of the SIDE=NUFRONT setting. Only CONSTANT criteria using an EQ operator is supported. The SIDE=NUFRONT setting in the JSL is required for JDLRPage to work properly.

NOTE: If the print job is likely to contain duplex pages alternating with simplex (one sided) pages, JDLRPAGE provides a way to leave the back sides of certain pages blank.

Using the JDL sample listed earlier, this option should be set to:

```
JDLRPage       = 1,5,EQ,X'FFFF26FFFF'
```

PrinterInk

Use this option to specify the color of ink loaded on a Xerox highlight color printer. You can set the PrinterInk option to either Blue, Red, or Green (blue is the default). This option is used with the SendColor INI option. If you set the SendColor option to Yes, you should also set the PrinterInk option. Here is an example of this INI option:

```
PrinterInk    = Blue
```

PaperSize

Use this option to specify the size of the paper. Here is a list of the most commonly-chose options. For a complete listing of all options, see Choosing a Paper Size on page 257.

For	Enter
letter	zero (0). This is the default.
legal	1
A4	2
executive	3
custom	98

DefaultFont

Use this option when displaying the names of fonts which are not found in the font cross reference (FXR) or LOGO.DAT files. The value for the DefaultFont option is a font ID which is contained in the font cross reference (FXR) file being used.

```
< PrtType:XER >
    DefaultFont = 11010
```

Creating Font Cross-reference Files

You will need to create a font cross-reference (FXR) file using the AFP Fonts referenced by the archived AFP or Metacode print streams. You can create and update font cross-reference (FXR) files using the FNTEDW32 utility.

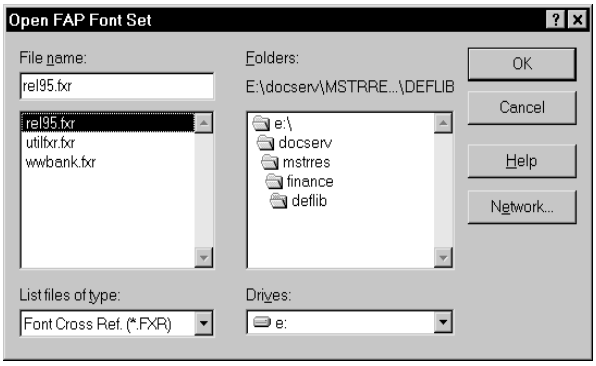
Adding Fonts to the Font Cross-reference File

First, start the FNTEDW32 utility by entering this command, in the directory in which you installed the Internet Document Server:

```
fntedw32
```

The FNTEDW32 utility's Insert option lets you add font information to your font set (FXR file). Follow these steps to add font information:

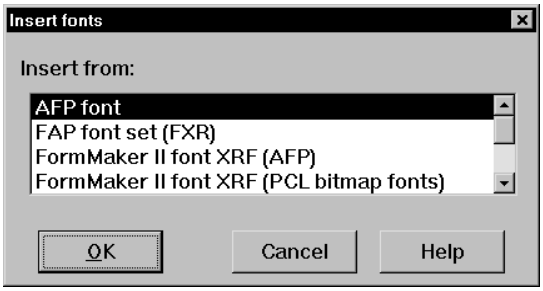
1 When the FNTEDW32 utility starts, you see the following window:



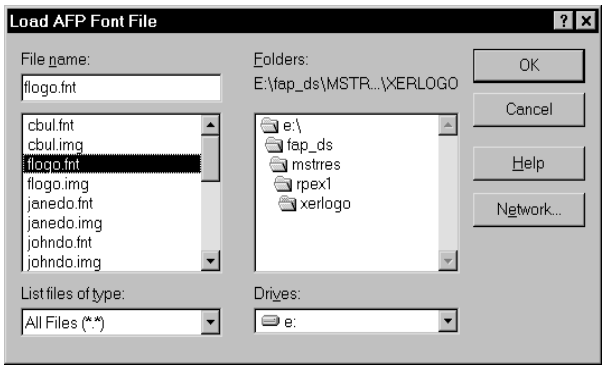
Select the font cross-reference file to which you want to add fonts and click Ok.The Font List window for the font cross-reference file appears.

2 To insert fonts, click Insert in the Font List window. The Insert Fonts window appears.

NOTE: The Insert button is active only if no font is selected. If the Insert button is not active, click the Deselect All button. This activates the Insert button.



3 Select *AFP font* or *Xerox Metacode fonts* as the font type you want to insert and click Ok. The Load AFP Font File window appears.



4 Select the font files you want to add. You can insert multiple fonts. If the file is in a different directory or folder, use the Drives and Folders fields to find the file. Once you select the file you want, click Ok. The selected font set is inserted in your font set.

NOTE: Remember that AFP coded font files begin with the letter X, such as X0DATIN8.FNT. A coded font file contains references to specific character set and specific code page. The corresponding character set and code page files should be in the same directory as the coded font file to import it. The font information imported from the AFP font will be assigned a font ID which is one greater than the largest font ID contained in the font cross-reference file (FXR).

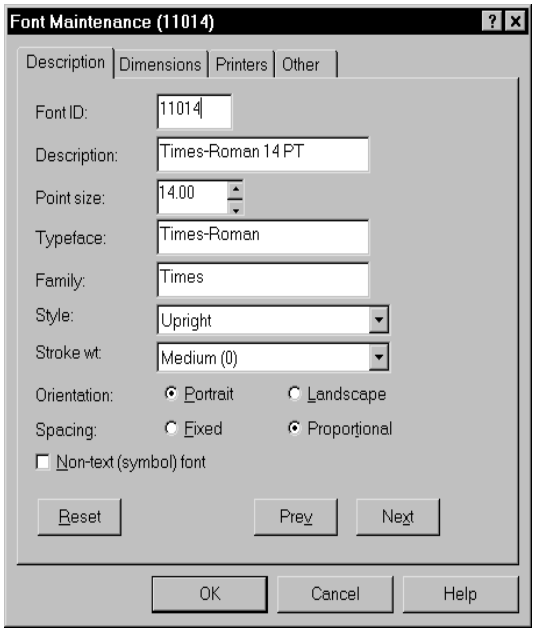
If you get an error while inserting AFP fonts, the error may have occurred because your AFP fonts are corrupt or the files do not contain AFP fonts.

If you get an error while inserting Xerox fonts, the error may have occurred because your…

- Xerox fonts are encrypted. The Printstream Bridge cannot use encrypted fonts. Xerox can take an encrypted font and provide a non-encrypted equivalent.
- Xerox fonts are corrupt or the files do not contain Xerox fonts.

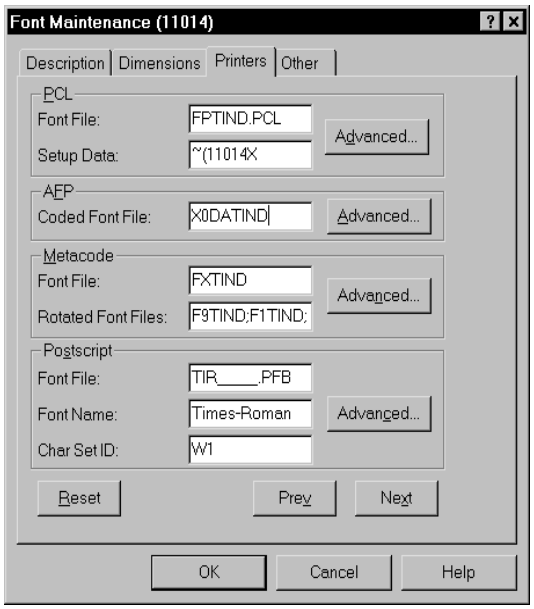
CUSTOMIZING A FONT CROSS-REFERENCE FILE

While using the FNTEDW32 utility, you can use the Edit option to edit the information about fonts and printers. Highlight the font you for which you want to edit information and click Edit. This window appears:



On the Description tab, make sure the settings for the Stroke wt., Style, and Spacing fields are correct. Italic fonts should have a style of *Italic*. Bold fonts should have a stroke weight that's greater than zero. Fixed pitch fonts should have a spacing of *fixed*. And, proportional fonts should have a spacing of *proportional*.

When you click the Printers tab, the following window appears:



The Printers tab lets you enter printer-specific information for PCL, AFP, Metacode, and PostScript printers. In the AFP (or Metacode) section of the Printers page, the Font File field should contain the name of the font file.

AFP font file names

For AFP font files, the names begin with an *X*. AFP font file names are limited to eight characters. AFP font file names must be in uppercase letters and *should not* include an extension.

Metacode font file names

Xerox Metacode font file names are limited to six characters. *Do not* enter an extension. The Rotated Font Files field should include the 90, 180, and 270 degree versions of the Portrait Xerox font file separated by semicolons, such as:

```
FNT90;FNT180;FNT270
```

Acrobat fonts

The Adobe Acrobat Reader uses PostScript fonts instead of AFP or Metacode fonts. To make the PDF look as much like the original printed output, the AFP or Metacode fonts must be mapped to one of the standard base fonts which are always available to the Adobe Acrobat Reader. The system uses the PostScript Font Name (also called Setup Data) setting in the font cross-reference file (FXR) to specify which base font to use. The standard base fonts for Acrobat Reader are:

- Courier, Courier-Bold, ***Courier-Oblique***, ***Courier-BoldOblique***
- Helvetica, Helvetica-Bold, ***Helvetica-Oblique***, ***Helvetica-BoldOblique***
- Times-Roman, Times-Bold, ***Times-Italic***, ***Times-BoldItalic***
- Symbol, ZapfDingbats

Checking Your Font Cross-reference File

Once you finish making changes to the font cross reference file, you can use the FXRVALID utility to check a font cross reference (FXR) file for settings which would cause problems when creating PDF files.

Note: For more information on FXRVALID and other utilities, see the Docutoolbox Reference. You can access this manual from Skywire Software's internet site (www.skywiresoftware.com\doss).

The FXRVALID utility performs several checks on font IDs in the font cross reference (FXR) file, including…

Checking typefaces

This check makes sure all font IDs contain one of the following PostScript font names in the Setup Data field for PostScript printing:

Courier	Helvetica-Bold	Symbol
Courier-Bold	Helvetica-Oblique	Univers-Medium
Courier-BoldItalic	Helvetica-BoldOblique	Univers-Bold
Courier-Oblique	Times-Roman	Univers-MediumItalic
Courier-BoldOblique	Times-Bold	Univers-BoldItalic

Courier-Italic	Times-Italic	ZapfDingbats
Helvetica	Times-BoldItalic	

The FXRVALID utility tells you via an error message if the FXR file contains an invalid PostScript font name or does not contain a PostScript font. The message also tells you whether a fixed or proportional font will be used in place of the invalid typeface. The PDF printer driver will make the font substitution.

NOTE: Font IDs have either a fixed pitch or a proportional spacing value. If font substitution is required for fixed pitch fonts, Courier is typically used. If font substitution is required for proportional fonts, Helvetica is typically used. In addition, the stroke weight and style settings of the font ID are checked to see if bold and/or italic versions of these fonts should be used.

Checking point sizes

This check compares the font height to the point size for each PostScript font in the FXR. A warning message appears for every font ID whose font height differs from the point size by a factor of 1/3 or greater. The utility uses the font height to determine the point size. A warning also appears if the font height equals zero.

NOTE: If the font height and point size do differ by the factor of 1/3, the printer driver will use font height to determine point size. The FXRVALID utility does not determine point size in these situations.

Checking the code page

A warning appears for any font IDs whose code page field is not empty or is not set to 1004.

NOTE: The PDF printer driver uses the ANSI code page for text. Code page 1004 is the OS/2 code page which is equivalent to the ANSI code page. Code page 1004 is the value in the FXR used to make the system display forms under OS/2 using the same code page as the ANSI code page.

Checking spacing

This check makes sure the spacing value (fixed or proportional) of the font ID matches a PostScript font with an equivalent spacing style. If the spacing value does not match, a warning appears.

Checking the style

This check makes sure the font style (upright or italic) of the font ID matches a PostScript font with an equivalent font style. If a font ID specifies an italic style, a warning appears if the Setup Data field does not contain a PostScript font name containing the word *Italic* or *Oblique*. If a font ID specifies an upright style, a warning appears if the Setup Data field contains Italic or Oblique.

Checking the weight

This check makes sure the font weight (bold or normal) of the font ID matches a PostScript font with an equivalent font weight. If a font ID specifies a bold style, a message appears if the Setup Data field does not contain a PostScript font name which includes the word *Bold* and vice versa.

Using the FXRVALID Utility

To use this utility, enter this command:

fxrvaldw32 /I /E /G /O /R /D?

Parameter	Description
/I	The name of the font cross-reference (FXR) file, omit the extension.
/E	(Optional) An error file name, omit the extension.
/G	Turns on the adding of "OTH" entry and the grouping of fonts. You can specify the grouping threshold as an error percentage. The default is zero (0). The default range is 32,127.
/O	(Optional) An output file name. The new FXR file contains "OTH" entries and grouping. If you omit the file name, the utility uses the input file name with an *FXX* extension. If you include a file name without an extension, the utility defaults to *FXX*.
/R	(Optional) Use this parameter (startchar,endchar) to specify the range of characters in width table to be checked for grouping. You can enter any integer from 0 to 255. The default value for *startchar* is 32 and the default value for *endchar* is 127. If *endchar* is less than *startchar*, the value of *endchar* is set to that for *startchar*.
/D?	Turns on the DownloadFont option, known as the Option field in the "OTH" entry, in every "OTH" entry. The DownloadFont option in every "OTH" entry is turned off if you omit this parameter.

For example, if you enter:

```
fxrvaldw /I=rel95sm
```

The utility checks the font cross-reference file named REL95SM.FXR and creates an error file named REL95SM.ERR which you can open in any ASCII text editor.

Creating Logo Files

To optimize performance, you should create logo (LOG) files for signature fonts, images, and logos referenced by the archived Metacode print streams. While the system can load Xerox fonts directly for signature fonts, doing so makes processing slower than if you had used logos.

You can convert a Xerox font, image, or logo into a logo using the XER2LOGW utility. For example, if you are running on a Windows 32-bit computer, you would enter…

```
xer2logw /I=xfont.fnt
```

to create a logo named XFONT.LOG. You could also enter…

```
xer2logw /I=ximage.img
```

to create a logo named XIMAGE.LOG. You could also enter…

```
xer2logw /I=xlogo.lgo
```

to create a logo named XLOGO.LOG.

Creating a LOGO.DAT File

You do not need to create logos for Xerox fonts and images which are rotated versions of the Xerox fonts and images you previously converted into system logos. Instead, you will need to create a LOGO.DAT text file for these non-portrait signature fonts or images referenced by the archived Metacode print streams.

The LOGO.DAT file should be placed in the FormLib directory. The LOGO.DAT file, which is a semicolon-delimited file, should look similar to...

```
FNT0;FNT90;FNT180;FNT270;
```

...where *FNT0* is the file name for zero (0°) rotation, *FNT90* is the file name for 90° rotation, *FNT180* is the file name for 180° rotation, and *FNT270* is the file name for 270° rotation.

Removing Unwanted Text and Logos

If you see text or logos when viewing a form using the Acrobat Reader that do not appear in the printed Metacode output, it is because your Metacode output contains characters not defined in the Xerox font.

To prevent this, add an INI control group whose name is the Xerox font name and specify the first and last characters defined in that font. Typically, this is only affects signature and other non-text fonts.

For example, if your signature font is named QFLOGO.FNT, you would set up an INI control group with the following options:

```
< QFLogo >
    FirstChar = 65
    LastChar  = 68
```

In this example, the first and last character code points are 65 (*A*) and 68 (*D*) respectively. Do not include the file extension (*.FNT*) in the group name. Do not use letters like *A* and *D* or hexadecimal numbers for the FirstChar and LastChar option settings. You must use decimal numbers.

NOTE: To determine the first and last characters used in a font, sample the font on the Xerox printer.

These settings only affect signature and other non-text fonts that have been converted to DAP logos.

USING THE MRG2FAP UTILITY

You can use the MRG2FAP utility to convert a Documerge AFP or Metacode file into a FAP file. This utility also converts AFP print files created by the DAP system into FAP files. You can then view and edit the FAP file using the Image Editor.

The MRG2FAP utility lets you load Xerox FRM files and IMG files that are referenced in the Metacode print stream being converted. In addition, the MRG2FAP utility can produce a BPSD / Field cross-reference listing.

The system looks for the FRM and IMG files in the directory specified by the FormLib option in the MasterResource control group. If you omit this option, the system looks in the current directory.

Use the KeepBlankPages option when you are converting AFP and Xerox Documerge files into FAP files (MRG2FAP) to retain blank pages. Here is an example:

```
< PrtType:AFP > or < PrtType:XER >
    KeepBlankPages = Yes
```

Normally blank pages are removed because the system assumes they are duplex back pages that are not needed. If, however, you want to retain these pages, add this option and set it to Yes. The default is No which indicates you do want to remove blank pages during a conversion.

For more information, see the Docutoolbox Reference. You can access this manual from Skywire Software's internet site (www.skywiresoftware.com\doss).

OVERLAYS AND PAGE SEGMENTS

If the AFP print file contains references to overlays or page segments, copy the overlay or page segment files into the directory in which the AFP print file resides. Add the following options in the PrtType:AFP control group in the FSISYS.INI file to specify the file extension for overlay and page segment files.

```
< PrtType:AFP >
    OverlayExt=
    PageSegExt=
```

Customizing Your Installation

The example HTML templates provided with the Printstream Bridge are designed to work using page-at-a-time downloading of PDF files. To take advantage of this, you need an Acrobat 3.0 or higher viewer with an appropriate web browser and web server.

The web browser must support a proposed HTTP extension for specifying byte ranges of a file to be downloaded, and the web server must support byte range downloading using a CGI script or other mechanism.

Implementing the Printstream Bridge with an existing system requires writing rules to retrieve Documerge files processed by that system. Also, you must either modify the sample HTML files or create HTML files for your system.

This topic discusses the modifications typically necessary to use the example HTML files with another archive. Even if it is necessary to create new HTML files, rather than modify the example files, this topic will point out the archive-specific components of the HTML files.

The architecture of the archive retrieval process was demonstrated when you verified the installation of the Printstream Bridge earlier in this chapter. In summary:

1. The first HTML display is the Login page.
2. Once the user is verified, the bridge presents a search page which contains fields for the Company, Line of Business, and Policy Number (Key1, Key2, and Key3) fields. When this page is submitted, the bridge searches the archive based on the data entered in these fields.
3. The next HTML page sent by the bridge returns the search results in table format. Each table entry has an HTML link for that particular form set.
4. When a user selects one of the table entries, the bridge generates the PDF file and sends the next HTML page. This HTML page again displays the form set information and contains an HTML link to the PDF file.
5. The user can then click the file link and view the PDF file using Acrobat Reader.

The key to interfacing the bridge to a particular archive are the archive-specific components in the various HTML files. There are several example HTML files included with the Printstream Bridge:

- LOGINMTC.HTM
- SEARCH.HTM
- RECORDS.HTM
- PRINTOUT.HTM

Assuming you installed the Printstream Bridge without changing the directory structure, the LOGINMTC.HTM file will be located in the DOCSERV\HTML directory, and the other HTML files will be in the DOCSERV\MSTRRES\METACODE\HTML directory.

We'll now look at each of these HTML files to explain how they work and what will need to be changed to make these example files work with another archive.

Setting Up the Login Page

The login HTML page is the first page that the browser loads when it connects to the Printstream Bridge. The login page can contain any information you want, including links to other pages, links to email, and so on. This page is not processed by the Internet Document Server and is not an HTML template, rather is it a content page that is automatically sent to the browser. The only part of the login page the Printstream Bridge needs is the HTML form with these required fields:

- USERID
- PASSWORD
- REQTYPE

and the optional value:

- CONFIG

The Printstream Bridge uses these fields to verify the user, using the USERINFO table, and to start the retrieval process.

The following is part of the LOGINMTC.HTM HTML page showing the form that lets you enter your user ID and password:

```
<FORM METHOD=POST ACTION="/doc-prog/dcltw32.exe">
<INPUT NAME="REQTYPE" value="LGN" TYPE="HIDDEN"> <BR><BR>
<INPUT NAME="CONFIG" value="DOCUMERGE" TYPE="HIDDEN"> <BR><BR>
<tr>
<b>User ID: </b> <INPUT SIZE=10 MAXLENGTH=8 NAME="USERID"
value="USERID"> <BR><BR>
</tr>
<tr>
<b>Password: </b> <INPUT TYPE=PASSWORD SIZE=8 MAXLENGTH=8
NAME="PASSWORD" VALUE=PASSWORD><P>
</tr>
<P>
<INPUT TYPE="submit" VALUE="Login"><INPUT TYPE="reset"
VALUE="Reset"><P>
</FORM>
```

The name of executable in ACTION="/doc-prog/dcltw32.exe" is the name and location of client program. Note that this name is relative to web server root directory. For information on web server root directory location and setup consult your web server manuals.

This part of the login page defines these HTML form variables:

Variable	Description
USERID	The entry field for the user ID
PASSWORD	The entry field for the user's password. Asterisks (*) appear as the user types the password.
REQTYPE	Hidden, the value is LGN.
CONFIG	Hidden, the value is DOCUMERGE.

and two buttons:

Button	Description
LOGIN	The user presses this button after entering values in the USERID and PASSWORD fields and selecting the CONFIG value.
RESET	The user can press this button to reset the values in the fields to their original values—blank in this example.

When the user enters the values and clicks on LOGIN button, the web server invokes the client program specified in the ACTION attribute of the HTML FORM. The client program receives the values of all the form variables.

The client program receives the request of type REQTYPE, LGN in this case, and runs rules registered for this REQTYPE. The rules needed to process the LGN are contained in the DOCSERV.INI file:

```
< REQTYPE:LGN >
    function = atcw32->ATCLogTransaction
    function = atcw32->ATCLoadAttachment
    function = dprw32->DPRSetConfig
    function = atcw32->ATCUnloadAttachment
    function = dprw32->DPRLogin
```

The processing of the LGN request on the server creates following attachment variables:

- RESULTS - SUCCESS or error code
- USERID
- PASSWORD
- RIGHTS
- REPORTTO
- SECURITY
- USRMESSAGE

These fields are part of USERINFO record, and the values are set when the record matching USERID is found in the file.

NOTE: An attachment is a block of information accessed in the form of name/value pairs. Attachments are used to pass information between the client and the server rules, as well as the API.

If you are using Websphere MQ or JMS queues, the size of all variables and attachments combined is limited to available memory or to the limit set by the messaging system.

Setting Up the Search Page

The search HTML page is returned by the client program after it processes the LGN request. The client creates this page by processing the SEARCH.HTM template. Generally, the search page can contain any information you want, including links to other pages, links to email, and so on. The only part of it the Printstream Bridge needs is the HTML form with these required fields:

- USERID
- REQTYPE
- FIELDS

and optional values:

- CONFIG
- MAXRECORDS (the default is 20)
- PARTIALMATCH
- TABLEINIGROUP (the default is ArcRet)
- TABLEINIOPTION (the default is AppIdx)

NOTE: TABLEINIGROUP and TABLEINIOPTION are advanced values that should not be changed in most situations.

The values for each field in the FIELDS variable are required. For example, if the FIELDS value is Key1,Key2,Key3, then the values for the Key1, Key2, and Key3 are required.

These FIELD variables are archive-specific and must match the archive keys for your archive. For example, if your archive used the keys ACCOUNT, NAME, and POLNO instead of Key1, Key2, and Key3, the HTML template must be modified to use those key names to work with that archive. For this Printstream Bridge sample the fields KEY1, KEY2, KEYID, are used. Here is part of the SEARCH.HTM file that shows the relevant HTML form:

```
<FORM METHOD=POST ACTION="#EXENAME,#">
     <INPUT NAME="REQTYPE" value="SCH" TYPE="HIDDEN">
     <INPUT NAME="USERID" value="#USERID,#" TYPE="HIDDEN">
     <INPUT NAME="DOCTYPE" value="#DOCTYPE,#" TYPE="HIDDEN">
     <INPUT NAME="CONFIG" value="#CONFIG,#" TYPE="HIDDEN">
     <INPUT NAME="FIELDS" value="KEY1,KEY2,KEYID" TYPE="HIDDEN">
     <INPUT NAME="PARTIALMATCH" value="YES" TYPE="HIDDEN">
     <INPUT NAME="MAXRECORDS" value="15" TYPE="HIDDEN">
     <table cellpadding=0>
     <tr><td align=right>
     Key1</td>
     <td>
     <td> <INPUT SIZE=20 MAXLENGTH=20 NAME="KEY1">
     </td></tr>
     <tr><td align=right>
     Line Of Business</td>
```

```
<td>
<td> <INPUT SIZE=20 MAXLENGTH=20 NAME="KEY2">
</td></tr>
<tr><td align=right>
Policy Number</td>
<td> <INPUT SIZE=30 MAXLENGTH=30 NAME="KEYID">
</td> </tr>
</table>
</TD>

<TD WIDTH=30% VALIGN=TOP>
This form allows you to specify one or more values to be matched
to retrieve records from the archive tables.

Click for <B>HELP</B>
</TD>
</TABLE>

<INPUT TYPE="submit" VALUE="Retrieve"><INPUT TYPE="reset"
VALUE="Reset">
</FORM
```

The name of executable in ACTION=”#EXENAME,#” is the name and location of client program. This value is replaced with the actual executable name by the rules on the LGN request. Note that this name is relative to web server root directory. For information on web server root directory location and setup, consult your web server documentation.

This part of the search page defines these HTML form variables:

Variable	Description
USERID	An entry field for user input
REQTYPE	A hidden field, invisible to the user (the value is SCH)
CONFIG	A hidden field, invisible to the user (the value is #CONFIG,#)
FIELDS	A hidden field, invisible to the user (the value is Key1,Key2,Key3)
Key1	An archive field containing the Customer Account Number
Key2	An archive field containing the Customer Name
Key3	An archive field containing the Customer Location

and two buttons:

Button	Description
RETRIEVE	The user presses this button after entering search criteria to retrieve the information.
RESET	The user can use this button to reset the values of the fields to their original values.

Remember that to make this template work with another archive, the archive-specific values must be changed to match those used by the archive. These values include the archive keys (as discussed earlier) and possibly drop-down selections.

When the user enters the values and clicks on the Retrieve button, the web server invokes the client program specified in the ACTION attribute of the HTML FORM. The client program receives the values of all the form variables.

The client program receives the REQTYPE request, SCH in this case, and runs rules registered for this REQTYPE. The query executed on the DOC Server is presented as…

```
SELECT FROM IndexTable WHERE Key1='Key1Value' .AND. Key2 =
'Key2Value' .AND. Key3 = 'Key3Value'.
```

…where Key1Value, Key2Value, and Key3Value are the values entered into the entry fields. You have these optional values:

Value	Description
PARTIALMATCH	If this variable is on the HTML form, a partial match search is executed criteria, if omitted, an exact match search occurs
TABLEINIGROUP	Name of the group in the INI file, that specifies the table name. If this value is omitted, the rule uses ArcRet INI control group. This is an advanced setting and should not be used in most situations.
TABLEINIOPTION	Name of the INI option specifying the actual table name. If this value is omitted, the system defaults to AppIdx. This is an advanced setting and should not be used in most situations.
MAXRECORDS	Maximum number of matching records to return. This value defaults to 20. If this value is too big, the waiting time for getting the query results might be too long. Adjust this value carefully.

The processing of the SCH request on the server creates these attachment variables:

- RESULTS - SUCCESS or reason for the error
- MORERECORDS - set to YES if there are more records than returned
- RECORDS - number of records in the result set
- RECORD##.FIELD1 - values for each of the columns in the table and for each returned record. FIELD1 is the actual column name.

Performance considerations

This search rule can be used on virtually any table in the supported database formats. Complex queries are not executed quickly and could result in the client program timing out or unacceptable wait times. The exact match query (omitted PARTIALMATCH HTML variable) is always faster than partial match. Consider using it, where possible. Some of the search values (such as state) might have limited range of values, so try creating the drop-down on the HTML page, for user to pick from, and use exact matches.

Not all the databases are equal in performance. For example, for faster performance codebase requires tags (indexes) created on all the search columns. If exact match is used, it will run faster if the combined index is created, for example, if the exact match search is specified on Key1, Key2, and Key3 columns, the composite index would have to have an expression as Key1+Key2+Key3.

Setting Up the Records Page

The records HTML page is returned by client program after it processes the SCH request. This page shows the search results in tabular format. The client creates this page by processing the RECORDS.HTM template. The records page can contain any information you want, including links to other pages, links to email, and so on. The only part the Printstream Bridge needs is the HTML form with these required fields:

- USERID
- REQTYPE
- ARCKEY

and this optional value:

- CONFIG

Here is an excerpt of the RECORDS.HTM file:

```
<html>
<head>
<base href="#BASELOCATION,#">
<meta http-equiv="keywords" content="Docucorp Retrieve">
<title> RETRIEVE RECORD </title>
</head>
<body bgcolor = "#FFFFFF" link=#0000ff vlink=#2525b5
alink=#ff0000 text="#000000" >
<IMG ALIGN=RIGHT TOP SRC="nlogo30s.gif" alt=fsilogo><BR>
<H2><B> Document Retrieval  </B> </H2><BR>
<hr>
<H5>  Selected Record(s):  </H5>
<TABLE BORDER=2 CELLPADDING=1 COLOR="blue">
<TR><TD><TH>  Company </TH></TD>
    <TD>   Line of Business </TD>
    <TD>   PolicyNum </TD>
    <TD>   Date </TD>
    <TD>   Description </TD>
</TR>
<!-- DCL BEGIN SECTION;NAME=MORERECORDS;IF MORERECORDS=YES; -->
<FORM METHOD=POST ACTION="#EXENAME,#" >
<INPUT VALUE="Next #MAXRECORDS,# matches" TYPE="submit">
<INPUT NAME="REQTYPE" value="SCH" TYPE="HIDDEN">
<INPUT NAME="USERID" value="#USERID,%s#" TYPE="HIDDEN">
<INPUT NAME="CONFIG" value="#CONFIG,%s#" TYPE="HIDDEN">
<INPUT NAME="FIELDS" value="KEY1,KEY2,KEYID" TYPE="HIDDEN">
<INPUT NAME="PARTIALMATCH" value="YES" TYPE="HIDDEN">
<INPUT NAME="RESTART" value="ARCKEY" TYPE="HIDDEN">
<INPUT NAME="ARCKEY" value="#RECORDS15.ARCKEY,#" TYPE="HIDDEN">
<INPUT NAME="KEY1" value="#.KEY1,#" TYPE="HIDDEN">
<INPUT NAME="KEY2" value="#.KEY2,#" TYPE="HIDDEN">
<INPUT NAME="KEYID" value="#.KEYID,#" TYPE="HIDDEN">
<INPUT NAME="DESC" value="#.DESC,#" TYPE="HIDDEN">
<INPUT NAME="MAXRECORDS" value="#MAXRECORDS,#" TYPE="HIDDEN">
<INPUT NAME="LASTRECORD" value="#LASTRECORD,#" TYPE="HIDDEN">
<INPUT NAME="FIRSTRECORD" value="#FIRSTRECORD,#" TYPE="HIDDEN">
</FORM>
```

```
<!-- DCL END SECTION --></TABLE>
</body>
</html>
```

Notice that each company value in the table is a HTML link. When the user selects a record by clicking one of the links, the web server starts the client program you specified in the HREF attribute of the HTML page. The client program also receives the command line parameters you specified on the HREF line.

These variables are archive-specific and must match the archive keys contained in the application index (APPIDX) for that archive. For example, if an archive used the keys ACCOUNT, NAME, and POLNO instead of Key1, Key2, and Key3, you must modify the HTML template to use those key names to work with that archive. Also, you may need to change the table headings, such as Customer Account, Customer Name, Location, and so on, to match those used in a particular archive.

The client program receives the request of type REQTYPE, MTC in this case, and runs the rules registered for this REQTYPE.

If the search returns more than 20 records, the system displays the first 20 along with a Next button which lets the user display the next 20 records, and so on. When there are less than 20 records left to display, the system displays those records without the Next button.

The RECORDS.HTM template contains a HTML form for doing this. Here is the example of this HTML form:

```
<!-- DCL BEGIN SECTION;NAME=MORERECORDS;IF MORERECORDS=YES; -->
<FORM METHOD=POST ACTION="#EXENAME,#" >
<INPUT VALUE="Next #MAXRECORDS,# matches" TYPE="submit">
<INPUT NAME="REQTYPE" value="SCH" TYPE="HIDDEN">
<INPUT NAME="USERID" value="#USERID,%s#" TYPE="HIDDEN">
<INPUT NAME="CONFIG" value="#CONFIG,%s#" TYPE="HIDDEN">
<INPUT NAME="FIELDS" value="KEY1,KEY2,KEYID" TYPE="HIDDEN">
<INPUT NAME="PARTIALMATCH" value="YES" TYPE="HIDDEN">
<INPUT NAME="RESTART" value="ARCKEY" TYPE="HIDDEN">
<INPUT NAME="ARCKEY" value="#RECORDS15.ARCKEY,#" TYPE="HIDDEN">
<INPUT NAME="KEY1" value="#.KEY1,#" TYPE="HIDDEN">
<INPUT NAME="KEY2" value="#.KEY2,#" TYPE="HIDDEN">
<INPUT NAME="KEYID" value="#.KEYID,#" TYPE="HIDDEN">
<INPUT NAME="DESC" value="#.DESC,#" TYPE="HIDDEN">
<INPUT NAME="MAXRECORDS" value="#MAXRECORDS,#" TYPE="HIDDEN">
<INPUT NAME="LASTRECORD" value="#LASTRECORD,#" TYPE="HIDDEN">
<INPUT NAME="FIRSTRECORD" value="#FIRSTRECORD,#" TYPE="HIDDEN">
</FORM>
<!-- DCL END SECTION --></TABLE>
```

This form is used only if the MORERECORDS variable was in the attachment with the value *YES*. This form is presented by the browser as one button. When the user selects the Next ## matches button, the web server executes the client program specified in the ACTION= part of this HTML form.

The client program gets the request type SCH. This form is similar to regular Search form, with the exception of RESTART value. The restart value is specifying for the Internet Document Server search rules where to start searching again. Generally, restart value works the same way as FIELDS. The value specifies comma-delimited archive table column names. For each of these names there should be a HTML variable with a corresponding value. In this case, restart is ARCKEY and there is ARCKEY= in the HTML form.

Note the RECORDS15.ARCKEY,# value for the restart position. It means get the value from the record 20. This number should be changed to the same number as the MAXRECORDS value.

The rules needed by the Internet Document Server (contained in the DOCSERV.INI file) to process the RCP request and post the results to the DOC Client are:

```
< ReqType:MTC >
    function = atcw32->ATCLogTransaction
    function = atcw32->ATCLoadAttachment
    function = dprw32->DPRSetConfig
    function = atcw32->ATCUnloadAttachment
    function = mtcw32->MTCLoadFormset
    function = dprw32->DPRRotateFormsetPages
    function = mtcw32->MTCPrintFormset
```

The processing of the MTC request one the Internet Document Server creates the following attachment variables:

- Results - Success or error code
- REMOTEPRINTFILE - full file name of the created PDF file

The client program changes this name to be relative to the web server HTML contents directory.

Setting Up the Printout Page

The printout page is the page that is returned by client program as a result of the processing of the PRT request. The client creates this page by processing the PRINTOUT.HTM HTML template. Again, the printout page could contain any information the user might want, including links to other pages, links to email, and so on.

The example template has a link to the Adobe internet site for downloading Acrobat Reader. The printout page does not display the Adobe Acrobat PDF file, it contains a link to the PDF file.

The following is the PRINTOUT.HTM template:

```
<html>
<head>
<base href="#BASELOCATION,#">
<meta http-equiv="keywords" content="Documerge Retrieve">
<title> RETRIEVE RECORD </title>
</head>
<body bgcolor = "#FFFFFF" link=#0000ff vlink=#2525b5
<alink=#ff0000 text="#000000" >
<body>
<IMG ALIGN=RIGHT TOP SRC="nlogo30s.gif" alt=fsilogo>
```

```
<hr>
    Selected Record:
<TABLE BORDER=2 CELLPADDING=1 COLOR="blue">
<TR><TD><TH>  Company </TH></TD>
    <TD>   Line of Business </TD>
    <TD>   PolicyNum </TD>
    <TD>   Date </TD>
    <TD>   Description </TD>
</TR>
<TR>
    <TD><TH> #KEY1,# </TH></TD>
    <TD> #KEY2,# </TD>
    <TD> #KEYID,# </TD>
    <TD> #CREATETIME,DATE=# </TD>
    <TD> #DESC,# </TD>
</TR>

</TABLE>
Click here to see the document
<hr>
Download Adobe&#174 Acrobat&#174 Reader Software (Required to view
PDF files)
</A>
<IMG ALIGN=LEFT TOP SRC="getacro.gif" alt="Download Adobe logo"> </A>
<hr>
<!--DCL BEGIN SECTION;NAME=ERRORFILE; -->
Click here to see the errors
<!-- DCL END SECTION -->
</body>
</html>
```

Again, variables on in the file are archive-specific and must match the archive keys contained in the application index (APPIDX) for the archive. For example, if a particular archive used the keys ACCOUNT, NAME, and POLNO instead of Key1, Key2, and Key3, the HTML template must be modified to use those key names to work with that archive. Also, the table headings (Customer Account, Customer Name, Customer Location, and so on) may need to be changed to match those used in a particular archive.

When the user click on the HTML link to the Adobe Acrobat PDF file, the web server sends the PDF file to the browser, and the browser automatically displays the file using the Adobe Acrobat Reader, provided the viewer has been properly installed.

LIMITATIONS

Here is a summary of the AFP, Xerox Metacode and PDF limitations you should keep in mind:

AFP LOADER LIMITATIONS

- The AFP loader works with AFP output produced by Image Sciences' products. It assumes records are delimited by a blocking scheme similar to files produced under MVS. At the beginning of the file is a four-byte block length. A four-byte record data length follows this.

  The record data length indicates the length of the next piece of data, in this case, an AFP command. Additional record data lengths and associated data follow this until the block length is exhausted. At this point, a new block length is expected and the process repeats itself. The AFP page segment and overlay files must use the same format as the AFP print stream being converted. The AFP loader should also work on AFP print streams without the logical block and record data lengths.

- All fonts used by an AFP print stream must be found in the font cross-reference (FXR) file.
- Large print-ready files (more than 100 pages) will process slowly.
- The AFP loader cannot display charts and inline graphics. Inline graphic support only applies to the Metacode loader.

METACODE LOADER LIMITATIONS

- The PrtType settings must match the settings used to produce the print-ready Metacode file.
- All fonts used by a Metacode print stream must be found in the font cross reference (FXR) file or in a system logo (LOG) file.
- Rotated text may not display properly. Short bind back pages will display upside-down. Landscape pages display sideways.
- Large print-ready files (more than 100 pages) will process slowly.

PDF LIMITATIONS

The system does not currently support the full set of Adobe Acrobat PDF capabilities. Here are a some of the limitations.

- If the PostScript Font Name/Setup Data setting in the FXR does not match a PDF base font, the system maps these PostScript font names to PDF base font names:

  Courier-Italic maps to Courier-Oblique

  Courier-BoldItalic maps to Courier-BoldOblique

  Univers-Medium maps to Helvetica

  Univers-Bold maps to Helvetica-Bold

  Univers-MediumItalic maps to Helvetica-Oblique

Univers-BoldItalic maps to Helvetica-BoldOblique

Finally, if the PostScript font name fails to map to a PDF base font name using the preceding rules, then fixed pitch fonts will map to Courier and proportional fonts will map to Helvetica. If a font has bold, italic, or bold and italic attributes, the Courier or Helvetica PDF base font with corresponding attributes will be used.

- Only the ANSI code page (also known as code page 1004) is supported for PDF files. Normally, this will only be an issue if your documents include international characters. If you have used AGFA fonts for printing, this should not be an issue.
- The system currently supports four standard page size in the PDF file:
  - Letter (8.5 x 11 inches)
  - Legal (8.5 x 14 inches)
  - A-4 (8.26x 11.69 inches)
  - Executive (7.25 x 10.5 inches)

  Portrait and landscape page orientations are available for these standard page size. The customized page size will be converted into Letter size with corresponding orientation.
- Page-at-a-Time downloading of PDF files is supported. To take advantage of this you must have an Acrobat 3.0 or higher viewer with an appropriate web browser and web server. The web browser would have to support a proposed HTTP extension for specifying byte ranges of a file to be downloaded, and the web server must support byte range downloading using a CGI script or other mechanism.
- Although Acrobat Reader 3.0 supports variable fields, radio buttons, push buttons, list boxes, and hypertext links, the system does not support creation of these objects within a PDF file.
- The Metacode loader can load fonts directly. Using this capability slows performance and increases the size of PDF files. Do not use this capability to load all fonts if you are making PDF files—doing so causes the PDF driver to crash.

Chapter 2

Printstream Bridge Rules

The Printstream Bridge includes rules you can use to convert print streams.These rules run on all supported platforms except where noted.

The rules are divided into these categories:

- TIFF Rules (TPD) on page 40
- Metacode Rules (MTC) on page 47

The rule names are case sensitive.

NOTE: For information on Internet Document Server rules, see the SDK Reference.

TIFF Rules (TPD)

Use these rules to convert TIFF files into PDF documents for the Internet Document Server. The rules are listed in alphabetical order, as shown below:

Originally, the TPD rules could only print single page TIFF files into a PDF file. The system embedded CCITT Group 4 single strip TIFF files into the PDF file for performance reasons and stored other types of compressed and uncompressed TIFF file data directly into the PDF file.

In Shared Objects version 11.2, the system was enhanced to let you process multi-page CCITT Group 4 single strip TIFF files and other types of multi-page TIFF files. This lets the system print single page, multi-page, and a combination of single and multi-page TIFF files into a PDF file, including color TIFF, dual resolution TIFF, and 32-bit TIFF files.

TPDCreateFormset

Use this rule to create a PDF file from a TIFF, BMP, or JPEG file. On RUNF, this rule creates a DSIValue named TPDFORMSET and locates the stem attachment variable named TIFFNAME. For each of the stem values called NAME, the rule creates a page in PDF format.

NOTE: This rule is only available on Windows 32-bit platforms.

Syntax

```
long _DSIAPI TPDCreateFormset (DSIHANDLE hdsi,
            char * pszParms,
            unsigned long ulMsg,
            unsigned long ulOptions )
```

Parameters

Parameter	Description
DSIHANDLE hdsi	the DSI instance handle
char * pszParms	a pointer to the rule parameter string
unsigned long ulMsg	DSI_MSG???? message, such as DSI_MSGRUNF
unsigned long ulOptions	options

This rule can submit a combination of TIF, BMP, and JPG bitmap files in one request by specifying their types. The input attachment variables NAME and TYPE are sent to IDS with a full file name and type for each bitmap. If the bitmap file name is sent to IDS without the bitmap type, this rule checks for the source type attachment variable SRCTYPE. If this variable does not exist, *TIF* is used as the default type.

Here is an example of the request type setup:

```
[ReqType:INI]
    function = tpdw32->TPDInitRule
[ReqType:TPD]
    function = atcw32->ATCLogTransaction
    function = atcw32->ATCLoadAttachment
    function = dprw32->DPRSetConfig
    function = atcw32->ATCUnloadAttachment
    function = tpdw32->TPDCreateFormset
    function = tpdw32->TPDPrintFormset
```

Here are the input attachment variables:

Variable	Description
CONFIG	This is the name of the configuration, such as *SAMPCO*.
TIFFNAME	This is the number of input bitmap files.
SRCTYPE	This is the source bitmap file type, such as *TIF*.

Variable	Description
TIFFNAME1.NAME	This is the name and path of the first bitmap file, such as *d:\docserv\mstrres\sampco\tif1.tif*.
TIFFNAME1.TYPE	The type of the first bitmap file.
TIFFNAME2.NAME	This is the name and path of the second bitmap file.
TIFFNAME2.TYPE	The type of the second bitmap file.
TIFFNAME3.NAME	This is the name and path of the third bitmap file.
TIFFNAME3.TYPE	The type of the third bitmap file.

NOTE: You can have as many TIFFNAME#.NAME/TIFFNAME#.TYPE variables as necessary.

The rule also tries to locate PDFNAME in the input attachment as the name of the output file. If one cannot be located, the rule generates a unique name and adds it to the output attachment as REMOTEFILENAME.

This rule uses the TIFFPATH option to locate TIFF, BMP, or JPEG files if the name of the TIFF, BMP, or JPEG file in the TIFFNAME variable does not already have a path.

```
< TIFF2PDF >
    TIFFPath =
```

On RUNR message, this rule locates the TPDFORMSET value, destroys the form set and deletes the value.

This rule depends on the TPDInitRule rule being registered on the INI request.

Returns

Success or failure

Errors

Error	Description
TPD0001	The system cannot locate DSILocateValue &NAME&. Make sure the TPDInit rule was executed.
TPD0002	The call to API &APINAME& failed.
TPD0003	The system cannot load file &TIFFNAME&.
TPD0004	The system cannot locate the stem variable &NAME&.
TPD0005	The stem variable &NAME& has an invalid value (0).

TPDCreateOutput

Use this rule to create a PDF output file. This rule uses the TPDFORMSET value created by the TPDLoadFormset rule. The rule tries to locate PDFNAME in the input attachment as the name of the output file. If it cannot locate PDFNAME, the rule generates a unique name and adds it to the output attachment as REMOTEFILENAME.

Syntax

```
long _DSIAPI TPDCreateOutput (DSIHANDLE hdsi,
            char * pszParms,
            unsigned long ulMsg,
            unsigned long ulOptions )
```

Parameters

Parameter	Description
DSIHANDLE hdsi	the DSI instance handle
char * pszParms	a pointer to the rule parameter string
unsigned long ulMsg	DSI_MSG???? message, such as DSI_MSGRUNF
unsigned long ulOptions	options

NOTE: This rule depends on the TPDInit rule being registered on the INI request.

Errors

Error	Description
TPD0001	Cannot DSILocateValue &NAME&. Make sure the TPDInit rule was executed.
TPD0002	The call to API &APINAME& failed.

See also

TPDInitRule on page 46

TPDLoadFormset on page 44

TPDLoadFormset

Use this rule to load bitmap files. On the RUNF message, the rule creates a DSIValue named TPDFORMSET to hold the form set handle created by the TPDStartFormset API, locates the stem attachment variable with the name TIFFNAME, and for each of the stem values NAME, calls TPDAddPage.

Syntax

```
long _DSIAPI TPDLoadFormset (DSIHANDLE hdsi,
            char * pszParms,
            unsigned long ulMsg,
            unsigned long ulOptions )
```

Parameters

Parameter	Description
DSIHANDLE hdsi	the DSI instance handle
char * pszParms	a pointer to the rule parameter string
unsigned long ulMsg	DSI_MSG???? message, such as DSI_MSGRUNF
unsigned long ulOptions	options

This rule uses this INI option:

```
< TIFF2PDF >
    TIFFPath =
```

to locate TIFF files if the name of the TIFF file in TIFFNAME variable does not have a path. The rule creates a DSI value called TPDFORMSETH that holds the FAP form set in memory.

On the RUNR message, the rule locates the TPDFORMSET value, destroys the form set by calling TPDStopFormset, and deletes the value. It also deletes the TPDFORMSETH value.

NOTE: This rule depends on the TPDInit rule being registered on the INI request.

Here is an example of the server INI configuration:

```
[ReqType:TPD]
    function = atcw32->ATCLogTransaction
    function = atcw32->ATCLoadAttachment
    function = dprw32->DPRSetConfig
    function = atcw32->ATCUnloadAttachment
    function = tpdw32->TPDLoadFormset
    function = dprw32->DPRAddLogo,TPDFORMSETH
    function = dprw32->DPRAddText,TPDFORMSETH
    function = tpdw32->TPDCreateOutput
```

Errors

Error	Description
TPD0001	Cannot DSILocateValue &NAME&. Make sure the TPDInit rule was executed.

Error	Description
TPD0002	The call to API &APINAME& failed.
TPD0003	The file named &TIFFNAME& cannot be loaded.
TPD0004	The stem variable &NAME& cannot be located.
TPD0005	The stem variable &NAME& has invalid value (0).

See also TPDInitRule on page 46

TPDInitRule

Use this rule to initialize the TIFF2PDF Bridge. On INIT message, this rule creates a DSIValue named TPDHANDLE which holds the handle to the TIFF2PDF Bridge.

On TERM message, this rule terminates the TIFF2PDF Bridge and deletes DSIValue TPDHANDLE.

NOTE: This rule is only available on Windows 32-bit platforms.

Syntax

```
long _DSIAPI TPDInitRule (DSIHANDLE hdsi,
                          char * pszParms,
                          unsigned long ulMsg,
                          unsigned long ulOptions )
```

Parameters

Parameter	Description
DSIHANDLE hdsi	the DSI instance handle
char * pszParms	a pointer to the rule parameter string
unsigned long ulMsg	DSI_MSG???? message, such as DSI_MSGRUNF
unsigned long ulOptions	options

You should use this rule as an INIT rule, only on the INI request type.

Returns

Success or failure

See also

TPDLoadFormset on page 44

METACODE RULES (MTC)

Use these rules to convert a Metacode print stream into documents for the Internet Document Server. The rules are listed in alphabetical order, as shown below:

MTCLoadFormset

Use this rule to load the Metacode or AFP print stream into a DAP form set. This rule creates a variable called MTCFORMSET with the value of the DAP form set handle. This rule expects the value METACODEFILE in the attachment with the name of the file to load. This rule destroys the DAP form set on the DSI_MSGRUNR message.

Syntax

```
long _DSIAPI MTCLoadFormset ( DSIHANDLE hInstance,
                              char * pszParms,
                              unsigned long  ulMsg,
                              unsigned long  ulOptions )
```

NOTE: The DPRPrint rule also works with the Documanage Bridge as well as the Documaker Bridge. If you include the MTCLoadFormset rule in the rule list, the DPRPrint rule will work with the form set loaded from that rule as well.

Parameters

Parameter	Description
DSIHANDLE hInstance	DSI instance handle
char * pszParms	Pointer to rule parameter string
unsigned long ulMsg	DSI_MSG???? message, such as DSI_MSGRUNF
unsigned long ulOptions	Options

This rule uses these options in the MasterResource control group in the DAP.INI file:

```
< MasterResource >
    DefLib =
    XrfFile =
    FormLib =
```

You can also use the following INI option to tell the system where to look for your Metacode files:

```
< Metacode2PDF >
    MetacodePath =
```

The MetacodePath option specifies where the MET files are for the Printstream Bridge.

Attachment variables

The required attachment variables are:

Variable	Description
USERID	The user ID
METACODEFILE or AFPFILE	The name of the Metacode file to load. If you omit the path, the MetacodePath option in the Metacode2PDF control group defines where the file is located. The name of the AFP file to load. If you omit the path, the AFPPath option in the AFP2PDF control group defines where the file is located.
PRTINPUTTYPE	The name of the printer control group (PrtType:XXX) to use for your INI settings. The default printer group is XER for the Metacode printer group and AFP for the AFP printer group.

This rule creates and destroys the MTCFORMSET DSI value.

Returns

Success or failure

Errors

Error	Description
MTC0001	cannot locate METACODEFILE in the attachment
MTC0010	DLL version mismatch, internal API failure
MTC0011	attachment value METACODEFILE is empty
MTC0010	print API failure, probably configured incorrectly in the INI file

For more information on error parameters see the HTML template ERRORS.HTM.

MTCPrintFormset

Use this rule to return a print output. This rule requires that the MTCFORMSET DSI variables created. Use the MTCLoadFormset on page 48 rule to create this variable.

Syntax

```
long _DSIAPI MTCPrintFormset ( DSIHANDLE hInstance,
                               char * pszParms,
                               unsigned long  ulMsg,
                               unsigned long  ulOptions )
```

Parameters

Parameter	Description
DSIHANDLE hInstance	DSI instance handle
char * pszParms	Pointer to rule parameter string
unsigned long ulMsg	DSI_MSG???? message, such as DSI_MSGRUNF
unsigned long ulOptions	Options

Attachment variables

There are no required attachment variables. If present, the system will use the following attachment variable:

Variable	Description
PrintPath	The full path for the output PDF file. If you omit this variable, the system uses the PrintPath option in the Attachments control group to determine the location of PDF file.

This rule generates a unique file name for the PDF file it creates and adds the name to the attachment as REMOTEPRINTFILE. The file name also includes path information.

This rule expects the MTCFORMSET variable be created with the DAP form set handle. It is similar to DPRPrint, but does not do any recipient filtering.

This rule can use the following control group and option in the DAP.INI file:

```
< Attachments >
    PrintPath =
```

Returns

Success or failure

Errors

Error	Description
MTC0017	Cannot locate DSI variable MTCFORMSET. Make sure you registered the MTCLoadFormset rule.
MTC0010	Cannot initialize the print subsystem. Check your INI file options

For the error parameters see the HTML template ERRORS.HTM.

APPENDIX B

Choosing a Paper Size

The system supports a wide variety of paper sizes including US and international sizes. The following tables show the paper sizes you can choose from:

- US Standard Sizes on page 54
- ISO Sizes on page 55
- Japanese Standard Sizes on page 58

You can also find the following related information in this topic:

- Printer Support for Paper Sizes on page 59
- Paper Sizes for AFP Printers on page 63

US Standard Sizes

These paper sizes are commonly used in the United States and Canada. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

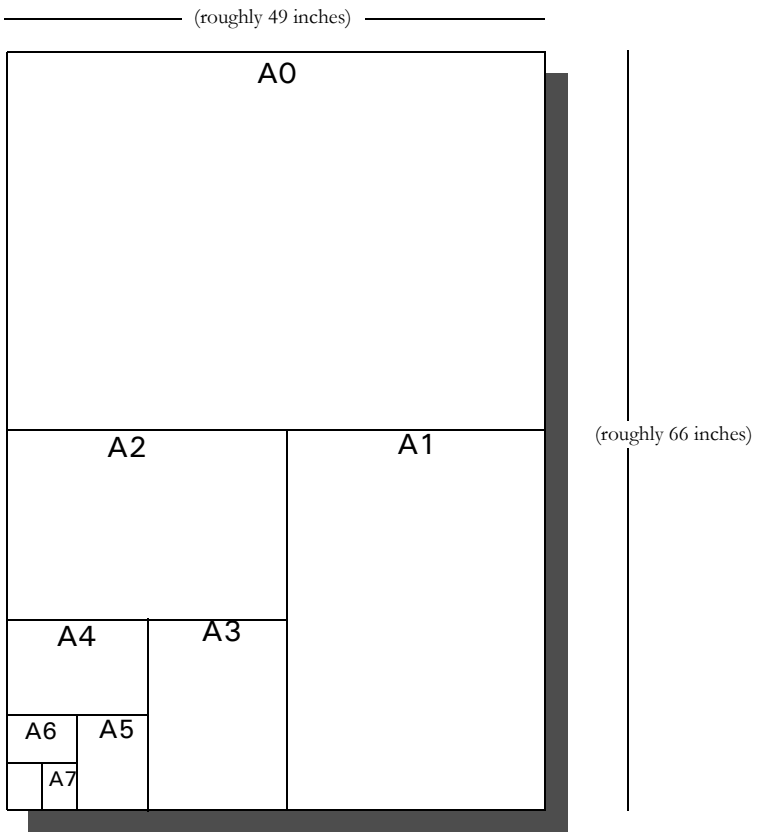
		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
US letter	0	20400 x 26400	216 × 279	8½ x 11
US legal	1	20400 x 33600	216 × 356	8½ x 14
US executive	3	17400 x 25200	190 × 254	7¼ 10½
US ledger	4	40800 x 26400	432 x 279	17 x 11
US tabloid	5	26400 x 40800	279 × 432	11 x 17
US statement	6	13200 x 20400	140 x 216	5½ x 8½
US folio	7	20400 x 31200	216 x 330	8½ x 13
US fanfold	8	35700 x 26400	378 x 279	14⅞ x 11
Custom	98	any x any	any x any	any x any

ISO Sizes

The International Organization for Standardization (ISO) paper sizes, which are based on the earlier Deutsche Industrie Norm (DIN) sizes, are used throughout the world except in Canada, the United States, and Japan. There are three main series of paper sizes: A, B, and C.

ISO A sizes

The A series of sizes are typically used for correspondence, books, brochures, and other printed materials. This diagram shows most of the various A sizes. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.



Width x Height

Name	Code	FAP units	Millimeters	Inches (approximate)
ISO A0	20	79464 x 112345	841 x 1189	33 1/8 x 46 1/4
ISO A1	21	56125 x 79464	594 x 841	23 3/8 x 33 1/8
ISO A2	22	39685 x 56125	420 x 594	16 1/2 x 23 3/8
ISO A3	23	28063 x 39685	297 x 420	11 3/4 x 16 1/2
ISO A4	2	19842 x 28063	210 x 297	8 1/4 x 11 3/4

		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
ISO A5	25	13984 x 19842	148 x 210	5 7/8 x 8 1/4
ISO A6	26	9921 x 13984	105 x 148	4 1/8 x 5 7/8
ISO A7	27	6992 x 9921	74 x 105	2 7/8 x 4 1/8
ISO A8	28	4913 x 6992	52 x 74	2 x 2 7/8
ISO A9	29	3496 x 4913	37 x 52	1 1/2 x 2
ISO A10	30	2457 x 3496	26 x 37	1 x 1 1/2
ISO 2A	32	112345 x 158927	1189 x 1682	46 3/4 x 66 1/4
ISO 4A	34	158927 x 224690	1682 x 2378	66 1/4 x 93 5/8

ISO B sizes

The B series of sizes are designed primarily for posters, wall charts, and similar items where the difference between each A size represents too large a jump. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
ISO B0	40	94487 x 133605	1000 x 1414	39 1/8 x 55 1/8
ISO B1	41	66802 x 94487	707 x 1000	27 7/8 x 39 1/8
ISO B2	42	47244 x 66802	500 x 707	19 5/8 x 27 7/8
ISO B3	43	33354 x 47244	353 x 500	13 7/8 x 19 5/8
ISO B4	44	23622 x 33354	250 x 353	9 7/8 x 13 7/8
ISO B5	45	16630 x 23622	176 x 250	7 x 9 7/8
ISO B6	46	11811 x 16630	125 x 176	5 x 7
ISO B7	47	8315 x 11811	88 x 125	3 1/2 x 5
ISO B8	48	5858 x 8315	62 x 88	2 1/2 x 3 1/2
ISO B9	49	4157 x 5858	44 x 62	1 3/4 x 2 1/2
ISO B10	50	2929 x 4157	31 x 44	1 1/4 x 1 3/4
ISO 2B	52	133605 x 188974	1414 x 2000	55 3/4 x 78 3/4
ISO 4B	54	188974 x 267209	2000 x 2828	78 3/4 x 111 1/4

ISO C sizes

The C series of sizes are designed for making envelopes and folders to take the A series of sizes. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

		Width x Height		
Name	**Code**	**FAP units**	**Millimeters**	**Inches (approximate)**
ISO C0	60	86645 x 122550	917 x 1297	36 7/8 x 51
ISO C1	61	61228 x 86645	648 x 917	25½ x 36
ISO C2	62	43275 x 61228	458 x 648	18 x 25½
ISO C3	63	30614 x 43275	324 x 458	12¾ x 18
ISO C4	64	21638 x 30614	229 x 324	9 x 12¾
ISO C5	65	15307 x 21638	162 x 229	6 3/8 x 9
ISO C6	66	10772 x 15307	114 x 162	4½ x 6 3/8
ISO C7	67	7653 x 10772	81 x 114	3¼ x 4½
ISO C8	68	5386 x 7653	57 x 81	2¼ x 3¼
ISO C9	69	3779 x 5386	40 x 57	1 5/8 x 2¼
ISO C10	70	2646 x 3779	28 x 40	1 1/8 x 1 5/8
ISO DL	71	10394 x 20787	110 × 220	4 1/3 x 8 2/3

The DL size is for a sheet 1/3 of the A4 size. This is the most common size of envelope.

Japanese Standard Sizes

Japan has its own standard paper sizes, called the Japan Industrial Standard (JIS). The JIS A series is identical in size to the ISO A series. The JIS B series, however, does not match the ISO B series. There is no equivalent to the ISO C series. This table shows the JIS paper sizes. The height and width are in FAP units (2400 per inch), millimeters, and inches. The inch dimensions are approximate.

		Width x Height		
Name	Code	FAP units	Millimeters	Inches (approximate)
JIS B0	80	97322 x 137573	1030 x 1456	40½ x 57¼
JIS B1	81	68787 x 97322	728 x 1030	28¾ x 40½
JIS B2	82	48661 x 68787	515 x 728	20¼ x 28¾
JIS B3	83	34393 x 48661	364 x 515	14¼ x 20¼
JIS B4	84	24283 x 34393	257 x 364	10⅛ x 14¼
JIS B5	85	17197 x 24283	182 x 257	7¼ x 10⅛
JIS B6	86	12094 x 17197	128 x 182	5 x 7¼
JIS B7	87	8598 x 12094	91 x 128	3½ x 5
JIS B8	88	6047 x 8598	64 x 91	2½ x 3½
JIS B	89	4252 x 6047	45 x 64	1¾ x 2½
JIS B10	90	3024 x 4252	32 x 45	1¼ x 1¾

Printer Support for Paper Sizes

This table outlines the various paper sizes supported by the different print drivers. The table includes information for the PDF, RTF, HTML, Metacode, PCL 5, PCL 6, GDI, PostScript, and AFP print drivers. The PDF, RTF, HTML, and Metacode print drivers support all paper sizes.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
US letter	X	X	X	X	X	X
US Legal	X	X	X	X	X	X
US executive	X	X	X	X	X	X
US ledger	X	X	X	X	X	X
US tabloid	X	Y	US letter	X	X	X
US statement	X	JIS B5	US executive	X	X	X
US folio	X	US legal	US legal	X	X	X
US fanfold	X	US ledger	US ledger	X	X	X
ISO 4A	X	Y	US letter	US letter	US letter	C
ISO 2A	X	Y	US letter	US letter	US letter	C
ISO A0	X	Y	US letter	US letter	X	C
ISO A1	X	Y	US letter	US letter	X	C
ISO A2	X	Y	US letter	US letter	X	C
ISO A3	X	X	X	X	X	X
ISO A4	X	X	X	X	X	X

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 63 for more information.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
ISO A5	X	X	X	X	X	X
ISO A6	X	X	X	X	X	X
ISO A7	X	ISO A6	ISO C5	ISO A6	X	C
ISO A8	X	ISO A6	ISO C5	ISO A6	X	C
ISO A9	X	ISO A6	ISO C5	ISO A6	X	C
ISO A10	X	ISO A6	ISO C5	ISO A6	X	C
ISO 4B	X	Y	US letter	US letter	US letter	C
ISO 2B	X	Y	US letter	US letter	US letter	C
ISO B0	X	Y	US letter	US letter	X	C
ISO B1	X	Y	US letter	US letter	X	C
ISO B2	X	Y	US letter	US letter	X	C
ISO B3	X	Y	US letter	US letter	X	C
ISO B4	X	JIS B4	US ledger	X	X	X
ISO B5	X	JIS B5	X	X	X	X
ISO B6	X	JIS B6	ISO C5	X	X	X
ISO B7	X	ISO A6	ISO C5	ISO A6	X	C
ISO B8	X	ISO A6	ISO C5	ISO A6	X	C
ISO B9	X	ISO A6	ISO C5	ISO A6	X	C

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 63 for more information.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
ISO B10	X	ISO A6	ISO C5	ISO A6	X	C
ISO C0	X	Y	US letter	US letter	X	C
ISO C1	X	Y	US letter	US letter	X	C
ISO C2	X	Y	US letter	US letter	X	C
ISO C3	X	Y	US letter	X	X	C
ISO C4	X	JIS B4	US ledger	X	X	C
ISO C5	X	X	X	X	X	C
ISO C6	X	JIS B6	ISO C5	X	X	C
ISO C7	X	ISO A6	ISO C5	ISO A6	X	C
ISO C8	X	ISO A6	ISO C5	ISO A6	US letter	C
ISO C9	X	ISO A6	ISO C5	ISO A6	US letter	C
ISO C10	X	ISO A6	ISO C5	ISO A6	US letter	C
ISO DL	X	X	X	X	X	X
JIS B0	X	Y	US letter	US letter	X	C
JIS B1	X	Y	US letter	US letter	X	C
JIS B2	X	Y	US letter	US letter	X	C
JIS B3	X	Y	US letter	US letter	X	C
JIS B4	X	X	X	US fanfold	X	X

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 63 for more information.

Paper size	PDF, RTF, HTML, and Metacode	PXL[1]	PCL[2]	GDI[2]	PST[3]	AFP[4]
JIS B5	X	X	X	X	X	X
JIS B6	X	X	X	X	X	X
JIS B7	X	ISO A6	ISO C5	ISO A6	X	C
JIS B8	X	ISO A6	ISO C5	ISO A6	X	C
JIS B9	X	ISO A6	ISO C5	ISO A6	X	C
JIS B10	X	ISO A6	ISO C5	ISO A6	X	C

Sizes marked with an *X* are fully supported by the corresponding driver.

Sizes marked with a *Y* are supported by sending the paper dimensions in millimeters to the printer.

Sizes that refer to another size substitute the referred size when *paper size matching* is turned on. If paper size matching is not turned on, the behavior depends upon the specific driver. To turn on paper size matching, use this INI option:

```
< PrtType:XXX >
    PaperSizeMatching = Yes
```

[1] When paper size matching is not turned on, the PCL 6 (PXL) driver sends the paper dimensions in millimeters to the printer.

[2] When paper size matching is not turned on, these drivers substitute US letter.

[3] This driver does not use paper size matching. US letter is substituted for the unsupported paper sizes

[4] Sizes marked with a C are supported, but are commented out of the AFP formdef source file called F1FMMST.DAT, See Paper Sizes for AFP Printers on page 63 for more information.

Paper Sizes for AFP Printers

The AFP formdef source file (F1FMMST.DAT) contains support for the following paper sizes, but since this file contains support for so many paper sizes, its size could affect printer performance. To limit the effect, some of the paper sizes are commented out, as shown in this table:

Size	Commented out?
Letter	No
Legal	No
Executive	No
Ledger	Yes
Tabloid	Yes
Statement	Yes
Folio	Yes
Fanfold	Yes
ISO A3	Yes
ISO A4	No
ISO A5	Yes
ISO A6	Yes
ISO B4	Yes
ISO B5	Yes
ISO B6	Yes
ISO DL	Yes
JIS B4	Yes
JIS B5	Yes
JIS B6	Yes

NOTE: The F1FMMST.DAT and F1FMMST.FDF files can be found in the FMRES master resource library (MRL).

The commented source line begins with an asterisk (*). To add support for another paper size, open the F1FMMST.DAT file and delete the asterisk at the beginning of each line that references a paper size you want to add.

Because the AFP formdef is composed on medium map names that specify page orientation, paper size, tray selection, and duplex settings, there are 31 groups of medium map settings. Each of these groups contains the 57 possible paper sizes. So, for each paper size you add, there are 31 sources lines you must *uncomment* to fully support a paper size for all orientations, trays, and duplex settings.

After you uncomment the lines that reference the paper size you want to add, run the AFPFMDEF utility to rebuild your AFP formdef file with the new information. For more information on this utility, see the Docutoolbox Reference.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

X