Appendix A. Extracting AFP Resource contents

This appendix describes how to write a simple JAVA program to extract the contents of AFP Resource objects.

A.1 When to use this program

AFP resources, as a rule, are considered platform independent. This is true for AIX, Windows NT, and OS/390. It is relatively easy to move resources, such as overlays, page segments, fonts, form definitions, and page definitions between these other platforms.

It is also fairly easy to move those resources to the AS/400 system. Prior to V5R1, this was a two step operation. First, transfer the resource AFP data stream into an AS/400 physical file. Then, run the appropriate create command, namely CRTOVL, CRTPAGSEG, CRTFNTRSC, CRTFORMDF, or CRTPAGDFN.

With V5R1, these steps are combined in the interface, referred to as AFP Manager, which is included in Client Access Express.

There are times when a company must move AFP resources to other platforms. They may have custom resources that they want to use on another platform but no longer have the source to recreate them. For example, one division of a company may have created an overlay using the AFP Driver based on a word processing document. Another division of the same company that runs its operations on AIX may decide that it wants to use the same overlay. If the original word processing document is not available, the second group of users would have to recreate the overlay from scratch.

The problem is that, as part of the iSeries architecture, all objects have an object context area that contains additional information to manage security, tracking changes, control how the object is to be used, and so on. For example, you can update the contents of a file or run a program, but you cannot “run” a file or update the contents of a program. In this light, there are no standard iSeries server commands shipped with OS/400 that allow you to access the contents of the AFP objects.

The AFP Utilities for iSeries licensed program includes the Convert Overlay to Physical File Member (CVTOVLPFM) and Convert Page Segment to Physical File Member (CVTPAGSPFM) commands. These are used to extract the AFPDS information from iSeries overlays and page segments and put it into physical file members that can then be transferred to the other platforms. However, if a customer does not have this package, or they need to exchange other types of resources, they do not have a simple system tool to use.

The solution to this problem is to use a Java program and the AS/400 Toolbox for Java to extract the AFP data stream from the resource.

A.2 Requirements

This document assumes that you are compiling and running the Java program from a Windows client. In the course of running the program, you sign on to the iSeries server and the contents of the requested objects are copied to the local directory on the client. A TCP/IP connection is required between the iSeries server and the client. The specific instructions assume that you are using a Microsoft Windows client, although the Java developers kit and iSeries Toolbox may be installed on a variety of platforms.

To create and run this program from a Windows environment, you must have:

- The Java software Developers Kit
- IBM Toolbox for Java

The Developer Kit for Java and Toolbox for Java are included with every OS/400 shipped. They are also preloaded on new iSeries systems but must be separately installed when upgrading to the current OS/400 operating system on existing RISC systems. For information on compiling and running Java programs directly on the iSeries see the iSeries Information Center.

This chapter describes a scenario that will work even if you do not have Java licensed programs installed on the iSeries.

A.2.1 Java Software Developers Kit

The following steps describe how to download the Java Software Developers Kit (SDK) from the Sun Microsystems Java Technology Site. Sites may change over time. Use your best judgement to navigate if the Web site has changed and the same path is not apparent. Versions and releases of the packages change frequently, so use the designation for the current release where appropriate:

1. Log on to: `http://www.javasoft.com`
2. Select **Products & APIs**.
3. Select **Java™ 2 Platform, Standard Edition (J2SETM)**.
4. Select **Java™ 2 SDK, Standard Edition, v x.x.x (SDK)**.
5. Select your platform (for example, **Microsoft Windows**).
6. Download **Java 2 SDK, v x.x.x Software for Windows 95 / 98 / 2000 / NT / ME 4.0 (Intel Platform)**. Select one large bundle (approximately 31 MB) or multiple disk size pieces (less than 1.44 MB each). If you choose the multiple disk option, you must also download the installation instructions. Click the **Continue** box.
7. Read the Terms and Conditions document and click the **ACCEPT** box at the bottom.
8. Select the appropriate FTP site for your geography.
9. Use the **Save As...** dialogue box to display the target directory.
10. Open the directory where you saved the files and run the installation program (**j2sdk-x_x_x_x-win.exe**). Or, follow the installation instructions for the multiple file download. A folder called `jdkx.x.x_x` is created on your workstation. It contains the java compilation and runtime code.

A.2.2 IBM Toolbox for Java or JTOpen

The IBM Toolbox for Java is a library of Java classes that give Java programs easy access to iSeries or AS/400 data and resources. JTOpen is the open source version of Toolbox for Java.

There are two ways to obtain the IBM Toolbox for Java:

- Client Access Express
- IBM Toolbox for Java Web site

A.2.2.1 Loading Java toolbox from Client Access Express

Java toolbox is shipped as part of Client Access Express. If you have not yet installed it, perform a Selective Setup. For information on installing Client Access components, refer to *Client Access Express for Windows - Setup*, available from the iSeries Information Center at:
`http://publib.boulder.ibm.com/pubs/html/as400/infocenter.html`

The installation process of Client Access Express Java Toolbox creates a directory (\Program Files\IBM\Client Access\jt400\lib). Within that directory, a file called JT400.zip or JT400.jar exists, depending on what release of Client Access you are using. If you have the JT400.zip file, do *not* unzip it.

A.2.2.2 Loading Java toolbox from the Web

If you are not running Client Access Express, follow these steps to download the AS/400 Toolbox for Java from the IBM Toolbox for Java and JTOpen Web site:
http://`www-1.ibm.com/servers/eserver/iseries/toolbox/downloads.htm`

1. From the Web site, select the link to IBM Toolbox for Java and JTOpen.
2. You are prompted to supply a user ID and password to register.
3. Enter the registration information and click **Accept License**.
4. Select **Single File Download: jtopen_x_xx.zip**. It is a very large file (over 13 MB) with over 1,300 files. You only need one file from within it.
5. Select a temporary directory to which to download it.
6. Create a directory called `JTOpen` on your hard drive.
7. Run an unzip program. Select only the /lib/jt400.jar file and direct it to the JTOpen directory.

A.2.3 Preparing your Windows environment

Once you have obtained the necessary software, modify your Windows environment to access the necessary resources automatically. This is done by updating the PATH and CLASSPATH. The method for doing this depends on the version of Microsoft Windows operating system you are running.

A.2.3.1 Setting PATH and CLASSPATH in Windows 95 or 98

For Windows 95 or Windows 98, the additions are made in the AUTOEXEC.BAT file.

These are examples of PATH and CLASSPATH statements that are added to the end of an AUTOEXEC.BAT file. In this case, the JTOpen version of the toolbox is downloaded from the Web, which explains the JTOpen entry:

```
rem path & classpath entries added for AS400 toolbox - extract resources
SET PATH=%PATH%;c:\jdk1.x.x_xx\bin
SET CLASSPATH=%CLASSPATH%;c:\JTOpen\lib\jt400.jar
```

A.2.3.2 Setting PATH and CLASSPATH in Windows NT or 2000

For Windows NT or 2000, update the PATH and CLASSPATH using a GUI interface.

1. Right-click **My Computer**.
2. Select **Properties.**
3. Select the **Environment** tag.
4. Enter `PATH` for the Variable, and `c:\jdk1.x.x_xx\bin` for the Value.
5. Repeat these steps for CLASSPATH and the appropriate entry for the jt400.* file.

A.2.4 Entering and compiling the program

The sample program is listed in A.2.6, “Sample Java program listing” on page 7. You may enter the program using any ASCII text editor. If you are reading this document in a softcopy format, you may be able to cut and paste the code. Windows Notepad suffices for a short program such as this one. There are other more sophisticated editors for Java programming available that provide built in formatting and color coding. VisualAge for Java is such an offering from IBM.

Give your Java source program a `.java` file extension. The name of the file must match the name given in the *class* statement, which is ExportAFP.java in this example. Be aware that Java language, and even the program name, is case sensitive. To compile the program, enter the following information at the C:\>:

```
javac ExportAFP.java
```

This creates the ExportAFP.class class module.

A.2.5 Running the sample program

The program expects the following parameters to be passed to it:

- Library name
- The names of any number of AFP resource objects

The AFP resource object names must be entered with the appropriate name extension as shown in Table 1.

Table 1. resource file name extensions

Resource object type	Resource file name extension
Overlay	.OVL
Page segment	.PAGSEG
Font resource	.FNTRSC
Page definition	.PAGDFN
Form definition	.FORMDF

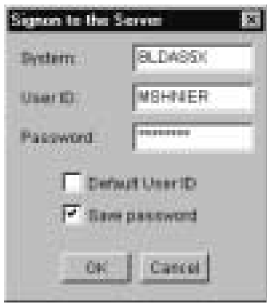
To extract the overlay called INVOICE and three page segments (SS2TOP, SSBOT1, and SSWM) from library MSHNIER, enter the following command:

```
java ExportAFP MSHNIER INVOIC.OVL SS2TOP.PAGSEG SSBOT1.PAGSEG SSWM.PAGSEG
```

Note: As mentioned earlier, the program name is case sensitive. However, the program was written to accept the parameters in upper or lower case. Therefore, it would be equally acceptable to enter the command as:

```
java ExportAFP mshnier invoic.ovl ss2top.pagseg ssbot1.pagseg sswm.pagseg
```

The iSeries server name, user ID, and password are not hard coded in the program. You are prompted to enter that information at run time as shown in Figure 1. You may enter the system name, or its IP address (such as 9.29.5.13).


Signon to the Server

System: BLDASSK
User ID: MSHNIER
Password: ********

[ ] Default User ID
[✓] Save password

Figure 1. Enter system name, user ID, and password

If you want, you can modify the program to include any, or all three, of the parameters. See the comment lines in the program listing that begin with “//” for syntax. If you choose to hard code your password in the program, you must make sure you protect your copy of the program source and compiled class object from unauthorized use.

The names given to the exported objects are generated by adding `.afp` to the end of the original object name. For example, the contents of INVOICE.OVL is exported to invoice.ovl.afp. The IBM AFP Viewer may be used to view overlay and page segment objects that use this naming convention.

The program writes a status message for each file it extracts. Figure 2 shows an example of the screen you see after running the program.

```
C:\AFPJava>java ExportAFP MSHNIER INVOIC.OVL SS2TOP.PAGSEG SSBOT1.PAGSEG SSWM.PAGSEG
Exporting /QSYS.LIB/MSHNIER.LIB/INVOIC.OVL to C:\AFPJava\invoic.ovl.afp
Exporting /QSYS.LIB/MSHNIER.LIB/SS2TOP.PAGSEG to C:\AFPJava\ss2top.pagseg.afp
Exporting /QSYS.LIB/MSHNIER.LIB/SSBOT1.PAGSEG to C:\AFPJava\ssbot1.pagseg.afp
Exporting /QSYS.LIB/MSHNIER.LIB/SSWM.PAGSEG to C:\AFPJava\sswm.pagseg.afp
```

Figure 2. Example of running Java ExportAFP program

A.2.6 Sample Java program listing

Figure 3 on page 8 contains the sample program described in this Appendix.

```
import com.ibm.as400.access.AS400;
import com.ibm.as400.access.AFPResource;
import com.ibm.as400.access.PrintObjectInputStream;

import java.io.File;
import java.io.FileOutputStream;

public class ExportAFP {
  public static void main(String args[]) {

    AS400 as400 = new AS400();

    // You may hardcode the system, username and/or password as follows.
    // AS400 as400 = new AS400("system", "username", "password");

    try {

      int resourceCount = args.length;

      String libraryPath = "/QSYS.LIB/"
                         + args[0].toUpperCase()
                         + ".LIB/";

      String resourcePath;
      AFPResource resourceObj;

      File outputFile;
      FileOutputStream outputStream;
      PrintObjectInputStream afpDataStream;
      byte[] buffer = new byte[4096];
      int bytesRead;

      for ( int i=1; i<resourceCount; i++ ) {
        resourcePath = "/QSYS.LIB/"
                     + args[0].toUpperCase()
                     + ".LIB/"
                     + args[i].toUpperCase();
        resourceObj = new AFPResource(as400, resourcePath);
        afpDataStream = resourceObj.getInputStream();
        outputFile = new File(args[i].toLowerCase() + ".afp");
        outputStream = new FileOutputStream(outputFile);

        System.out.println("Exporting "
                         + resourcePath
                         + " to "
                         + outputFile.getAbsolutePath());
        while ( (bytesRead = afpDataStream.read(buffer)) != -1 )
          outputStream.write(buffer, 0, bytesRead);

        outputStream.close();
        afpDataStream.close();
      }

      as400.disconnectAllServices();
      System.exit(0);
    } catch (Exception fmv) {
      System.out.println(fmv);
      as400.disconnectAllServices();
      System.exit(-1);
    }
  }
}
```

Figure 3. Sample Java program for extracting AFP resources