Client for Java Administrator's Guide

For other guides in this document set, go to the Document Center

MetaFrame® Presentation Server Client For Java, Version 8.x

Citrix® MetaFrame® Presentation Server 3.0
Citrix® MetaFrame® Access Suite

Last updated: 21 November 2003 (SC)

Contents

Chapter 4 Configuring the Client for Java

Chapter 5 Improving the Performance of the Client for Java

Chapter 6 Limitations of the Client for Java

Before You Begin

This guide is for system administrators responsible for installing, configuring, and maintaining the MetaFrame Presentation Server Client for Java. This guide assumes knowledge of:

- The server to which the clients connect
- The operating system on the client device
- Java-enabled Web browsers, HTML, and JavaScript
- Installation, operation, and maintenance of network and printing systems
- Web server configuration

How to Use this Guide

To get the most out of this guide, review the table of contents to familiarize yourself with the topics discussed:

Chapter	Contents
Chapter 2, “Introduction to the Client for Java”	Gives a list of features and system requirements.
Chapter 3, “Deploying the Client for Java”	Describes how to deploy the client.
Chapter 4, “Configuring the Client for Java”	Describes how to start and configure connections to servers.
Chapter 5, “Improving the Performance of the Client for Java”	Provides advice about how to improve performance after initial configuration.
Chapter 6, “Limitations of the Client for Java”	Describes known issues with the client and how to work around them.
Appendix A, “Client for Java Parameters and File Descriptions”	Provides a complete listing of all the applet parameters plus details of other files referred to in the guide.

Accessing Documentation

This administrator's guide is part of the MetaFrame Presentation Server documentation set. The documentation set includes online guides that correspond to different features of MetaFrame Presentation Server. Online documentation is provided as Adobe Portable Document Format (PDF) files.

Use the *Document Center* to access the complete set of online guides. The Document Center provides a single point of access to the documentation that enables you to go straight to the section of documentation that you need. The Document Center includes:

- A list of common tasks and a link to each item of documentation.
- A search function that covers all the PDF guides. This is useful when you need to consult a number of different guides.
- Cross-references between documents. You can move between documents as often as you need using the links to other guides and the links to the Document Center.

Important To view, search, and print the PDF documentation, you need to have the Adobe Acrobat Reader 5.0.5 with Search or a later version with Search. You can download Adobe Acrobat Reader for free from Adobe Systems' Web site at http://www.adobe.com/.

If you prefer to access the guides without using the Document Center, you can navigate to the component PDF files using Windows Explorer. If you prefer to use printed documentation, you can also print each guide from Acrobat Reader.

More information about Citrix documentation, and details about how to obtain further information and support, is included in *Getting Started with MetaFrame Presentation Server*.

Introduction to the Client for Java

Overview

The Client for Java is a Java applet that provides access to applications running on a server farm from any computer device with a standard Web browser. The applet is a download-and-run, zero-install client, optimized for use in Web environments where it is not possible or desirable to install software on the client device.

New Features in this Release

The following list outlines all the new features in this release.

For information on preexisting features, refer to the Client Feature Matrix on the Client Download page of the Citrix Web site.

Local Root Certificates

The comprehensive set of root certificates stored in the Java runtime environment is automatically used instead of the seven root certificates built into the client. This feature is available only when using J2SE (Java 2 Platform, Standard Edition) software.

Improved Client Drive Mapping

The user's home directory is automatically mapped to drive H at the start of a session. This feature is not available when using the Microsoft JVM.

A new version of the **Settings** dialog is available for J2SE environments. It provides an improved **Client Drive Mapping** tab that makes it easier for the user to configure drive mapping.

Printer Auto Detection

When using a J2SE 1.4.x environment that supports the Java Print Service API, the client automatically detects printers available to the client device. This means that users do not have to configure printers manually. For PostScript-capable printers, a generic Postscript driver is automatically configured on the server, and the resulting PostScript output is sent directly to the printer. For non-PostScript printers, the Universal Print Driver (see "Universal Print Driver Support" on page 10) is automatically configured.

Note Mac OS X provides a J2SE 1.4.x environment but does not provide the Java Print Service API, so printers are not auto-detected.

Universal Print Driver Support

The Universal Print Driver (UPD) is a standard Windows print driver that encapsulates print jobs in PCL4 (Printer Control Language 4) format. A client-based interpreter renders the print job using the client device's local print driver and printing services. The UPD generates smaller print jobs, which can significantly improve performance when printing over WAN or dial-up connections. Using UPD also increases security on the server because the number of drivers used is restricted to those known to UPD.

The Client for Java supports only UPD1 (UPD2 supports printing in color and at higher resolutions).

Logon Look and Feel

The appearance of the **Progress** dialog has been improved. This dialog now remains visible until the server starts to display the progress of the user's logon to the session. This enables the user to check that the logon process is continuing satisfactorily.

Terminal Services Client Access License Improvements

For Windows 2000 Server, Microsoft supports TS CAL (Terminal Services Client Access License) equivalency. This means that connecting to a Windows 2000 Server from a Windows 2000 (or later) client platform should not consume a CAL. This is now supported when the Client for Java is run on a Windows 2000 (or later) client system connecting to MetaFrame Presentation Server.

Note that Windows XP Home edition does consume a CAL.

General Performance Improvements

High Throughput Networking

This provides support for larger network packet sizes and also allows data transfer over virtual channels (for example, client drive mapping and printer mapping) to be temporarily interrupted. For example, if the user moves the mouse or types at the keyboard while a large file is being transferred over client drive mapping, the drive mapping transfer is temporarily interrupted to give priority to the mouse and keyboard handling.

Better ICA Performance

New compression technology results in less data being sent over the network. This results in, for example, faster video rendering, file transfer, and printing, providing a better overall user experience.

SpeedScreen Browser Acceleration

SpeedScreen browser acceleration provides major performance improvements for users connecting to Internet Explorer published on a server. Users can interact with the browser while graphically rich pages or large images are being downloaded. Scrolling performance in Internet Explorer is also greatly improved on graphically rich pages. This feature is available only when using Internet Explorer 5.5 or later on the server, and is supported only in J2SE 1.4.x environments on the client.

ThinImage

The server uses lossy compression for graphics. This reduces bandwidth consumption and provides faster rendering for the entire desktop session.This feature is supported only in J2SE 1.4.x environments.

Dynamic Session Reconfiguration

For seamless windows (see “Seamless Support” on page 12 for further details), the client detects local desktop size change and requests the server to update the underlying session size when the local desktop size changes. The client cannot detect changes to the local color depth.

Introduction to the Client for Java

The Client for Java is a signed Java applet that allows users to connect to MetaFrame Presentation Server computers.

The client has the following advantages:

- **No client installation**. You do not need to install any software on any client device. Users require only a Java-compatible Web browser. Setup is transparent and automatic.
- **Minimal download size**. At the most basic level of functionality, the applet is around 350 KB in size, providing a faster download than any other client.
- **Platform-independent**. The Client for Java can run on any client device running a Web browser either with the Microsoft JVM or J2SE environment 1.3 or greater.

The applet resides on a Web server and is deployed using an HTML page with an <applet> tag. Users run the client by opening the HTML page using a browser that has Java support. When the page opens, the Java applet is automatically downloaded to the client device. The applet then runs and connects to the server or published application specified in the <applet> tag.

Unlike the ActiveX, Netscape Plug-in, or Win32 Web Clients, which are downloaded once and then saved for future use by client systems, the applet is not stored permanently by the client system. However, most browsers cache Java applets so that they do not need to be downloaded each time they are used. J2SE Plug-in environments provide a separate cache for Java applets, which you configure in the Plug-in control panel.

Seamless Support

Seamless support is provided as an option on the client. It has three main aspects:

- Seamless windows
- Session sharing
- Connection Center, a tool that enables users to manipulate both seamless and non-seamless ICA connections

To provide seamless support on the client, Citrix recommends deploying the client through the Web Interface, because this provides the most effective interface for the features provided. You can deploy the client using the sample HTML pages provided with the client package, but this requires more work on your part.

Seamless Windows

Seamless windows means that each remote application appears in a separate resizable window on the client desktop. Users can resize the application window, minimize it, and copy and paste text between published applications and applications running locally on the client device. Copy/paste also works for non-text objects when used between applications sharing an ICA session.

Note Seamless windows are not supported on Mac OS X. If the Java Client is configured for seamless mode and run on these platforms, a non-seamless session is launched.

Session Sharing

Session sharing allows seamless application launches to share a single connection rather than creating a new connection for each application. This reduces the system overhead and therefore improves response times for users who have several applications open at the same time. Applications launched in existing sessions also launch more quickly, because a new connection and associated resources do not need to be created.

Connection Center

Connection Center enables users to:

- Disconnect a session
- Switch between full screen and seamless mode
- View properties such as the ICA encryption setting and the user name
- Log off a server session
- Close a published application
- Configure client settings

Client for Java Requirements

To run the Client for Java, the client system must have the following:

- A Web browser with either the Microsoft JVM, or Java 2, Standard Edition Version 1.3 or greater, configured to accept signed Java applets. For more information about signed applets, see “Using Signed Java Applets” on page 24.
- Network access to the Web server that stores the client files.

Java Environments

A large number of Java-enabled environments are available, and their functionality varies from platform to platform. To validate proper functionality of the Client for Java, Citrix selects a representative group of platforms for testing.

For English and other European languages, the client has been tested with:

- Internet Explorer 5.x/6.x on Windows 98, Windows Me, Windows NT 4.0 Workstation, Windows 2000 Professional, and Windows XP Professional and Home editions, with the Microsoft VM and Sun JRE1.4.x
- Safari 1.x with Apple JVM 1.4.x on Mac OS X 10.x or later
- Netscape 7.x on Solaris SPARC 9
- Netscape 7.x (glibc) and Mozilla 1.x on Linux x86 RedHat 9

For Japanese, the client is supported only with Internet Explorer 5.x/6.x on Windows NT 4.0, Windows 2000, and Windows XP. The limited platform support for Japanese is due to technical problems on some of the systems: these have been reported to the appropriate vendors.

On some platforms you need to install additional software to use audio mapping. See "Client Audio Mapping" on page 47 for more details.

For details of any known limitations of particular platforms or browsers, see Chapter 6, and consult the Readme file for any late-breaking issues.

Deploying the Client for Java

This chapter describes a basic deployment of the applet, followed by details of how to customize the applet and how to specify parameters. A typical example of how you might deploy the client is also provided.

Before You Begin

To deploy the client, you need:

- A copy of the client package. You can download the package from the Citrix Web site or copy it in uncompressed form from the Components CD. Citrix recommends that you obtain the latest version of the client from the Web site.

  On the Web site, the client package is available in two formats:

  - .zip, primarily for Windows systems
  - .tar.gz, primarily for UNIX systems

  Both have identical contents.
- A means of uncompressing and unpacking the .zip or .tar.gz package, if you download this from the Web site. If you are copying files from the Components CD you do not need to uncompress them.
- Administrator access to a Web server.

Unpacking the Client for Java

This section describes the contents of the client package.

To unpack the client

1. Copy the client package to a suitable location on the Web server. For Microsoft IIS servers, copy the package to a folder in the Web root directory (typically C:\inetpub\wwwroot). For UNIX systems, consult the Web server documentation.
2. If you downloaded the compressed package from the Web site, extract the program files from the .zip or .tar.gz package to the same folder, using a suitable decompression utility.

A number of files are created on the Web server. Some of these files are the Java archives that make up the applet. There are two types of archive package, as follows:

Archive type	Description
*N.jar files	Netscape Object signed Java archives. Compatible with: Netscape 6.x/7.x, Mozilla 1.x, and other browsers using a J2SE environment Internet Explorer on Windows platforms with Java Plug-in[1] 1.3.1 or later
*M.cab files	Authenticode signed cabinet files. Compatible with: Internet Explorer 4.x or later on Windows systems with the Microsoft VM
	[1]Download the Java Plug-in from http://www.java.com.

For each type of signed archive package there are a number of different components. For Microsoft Authenticode signed Java archives, the components are named as follows:

Archive File	Approximate Size[1]	Description
JICAEngM.cab	548KB	Complete archive. This archive contains the contents of all of the other archives apart from cryptojM.cab and sslM.cab, which must be included if required.
JICA-coreM.cab	354KB	Core archive. This archive provides only a basic connection. You add functionality by using it in conjunction with the other component archives described below.

cryptojM.cab	115KB	Encryption component. Required for all encryption.
sslM.cab	118KB	SSL component. Adds SSL and TLS encryption support.
JICA-configM.cab	58KB	User configuration component. Adds support for the Status bar, buttons, and **ICA Settings** dialog box.
JICA-audioM.cab	11KB	Audio component. Adds client audio mapping.
JICA-cdmM.cab	23KB	CDM component. Adds client drive mapping.
JICA-clipboardM.cab	11KB	Clipboard component. Adds client clipboard mapping.
JICA-printerM.cab	21KB	Printer component. Adds client printer mapping.
JICA-sicaM.cab	15KB	ICA encryption component. Adds ICA encryption support.
JICA-zlcM.cab	70KB	SpeedScreen latency reduction component. Adds support for local text echo and mouse feedback.
JICA-seamlessM.cab	60KB	Seamless and Connection Center components. Adds support for seamless windows and Connection Center.
		[1] Sizes of other types of archive package vary from those listed here.

If you are not using Microsoft Authenticode signed Java archives, choose the appropriate archive for the Java Runtime Environment you are using. For example, to deploy the core archive for Netscape, choose the Netscape Object signed archive, JICA-coreN.jar.

Getting Started with the Sample HTML Files

This section describes how to use the sample HTML pages supplied in the package to create an HTML page that specifies the correct archives for the user's browser.

First the sample HTML pages are described, then instructions for customizing them are provided. If you intend to use seamless windows and Connection Center, read the instructions on how to edit seamless1.html, but also keep the desktop.html instructions close for reference. If you do not want to use this functionality, read the instructions for editing desktop.html. For information on the benefits of seamless support and Connection Center, see "Seamless Support" on page 12.

To access the Connection Center, session sharing, and seamless windows functionality, Citrix strongly recommends that you use the Web Interface, which automates the steps that you otherwise need to implement yourself.

Seven sample HTML files are supplied in the client package:

- **index.html**. This page contains links to and descriptions of the six launching pages:
  - **desktop.html, application.html, and autoproxy.html.** If you do not want to implement seamless windows and Connection Center, use these pages.
  - **seamless1.html, seamless2.html, and seamless3.html.** To implement seamless windows and Connection Center, use these pages.
- **desktop.html**. This page launches a desktop session to a server. To make a connection with this page, you need to specify an address for the server.
- **application.html**. This page launches a connection, with 128-bit ICA encryption enabled, to a published application. You need to specify the name of the published application and the name of a server to use for server location.
- **autoproxy.html**. This page launches a connection to a published application through a proxy server, using proxy auto detection. You need to specify the name of the published application and the name of a server to use for server location. For further details of server location, see "Configuring Network Protocol and Server Location" on page 29.
- **seamless1.html, seamless2.html, and seamless3.html.** These pages start remote applications using Connection Center and seamless windows. The only difference between the three files is that they each start a different application. The applications are launched using an existing ICA session when possible. When session sharing is not possible, a new ICA session is created. You need to specify the name of a published application and the name of a server to use for server location.

Editing desktop.html

Desktop.html contains the following <applet> tag:

```
<applet name="javaclient"
    codebase="../"
    code="com.citrix.JICA"
    archive="JICA-coreN.jar,JICA-configN.jar"
    width="640"
    height="480">
    <param name="cabinets" value="JICA-coreM.cab,JICA-configM.cab">
    <param name="Address" value="plateau">
    <param name="End" value="end.html">
</applet>
```

The <applet> tag is used to configure the client. Some parameters are specified inside the <applet> tag as follows:

- Applet name. This is an optional, unique name for the applet. Use this name to refer to the applet when writing scripts. In desktop.html, the applet name **javaclient** is used by a script that displays a warning message if the user tries to close the Web browser window when an ICA session is running. It is also needed for PAC (proxy auto configuration) file support.
- Codebase. The path from the HTML page to the client archives. Change this path if it is not correct for your deployment.
- Code. The name of the class file that is executed. For the client without the Connection Center, this is always com.citrix.JICA.
- Archive. Specify Netscape Object signed archives here. Specify Authenticode (Internet Explorer) signed archives in a <param> tag with name = “cabinets”. In desktop.html, a Netscape Object signed core archive is specified here and Authenticode signed archives are specified in a <param> tag so that both forms of the client are supported. If you specify multiple archives in the same place, separate them with a comma.

  Note If you want users to be able to, for example, map drives and printers, you must specify the necessary archives here.

- Width. The width of the applet window, in pixels.
- Height. The height of the applet window, in pixels.

All other parameters are specified using <param> tags, located between the <applet> and </applet> tags. Use the <param> tags in the form:

```
<param name="parametername" value="valuename">
```

where *parametername* is the name of the parameter you are specifying and *valuename* is the value you are allocating to it.

To customize and use desktop.html

1. Open desktop.html in a plain text editor and find the <applet> tag section:

```
<applet name="javaclient"
 codebase="../"
 code="com.citrix.JICA"
 archive="JICA-coreN.jar,JICA-configN.jar"
 width="640"
 height="480">
 <param name="cabinets" value="JICA-coreM.cab,JICA-configM.cab">
 <param name="Address" value="plateau">
 <param name="End" value="end.html">
</applet>
```

   This is the section that launches the client.

2. Change the Address value to the address of a server on your local network.
3. Change the relative path specified for the codebase if it is not correct for your deployment.
4. Publish the sample HTML pages using your Web server. See the Web server documentation for more information about how to do this.
5. On the client device, open a Web browser and open the URL for the sample HTML pages. The index.html page opens.
6. Click the **Minimal Desktop** link. The applet appears.
7. To connect to the server, click **Connect** or **Click to connect**. To configure the client using the **ICA Settings** dialog box, click **Settings**.

You can edit application.html and autoproxy.html in the same way. The additional parameters used in these examples are described in Chapter 4.

To change the warning message displayed when a user tries to close an active ICA session

If you try to close the Web browser window when using an ICA session created with one of the example Web pages, a warning message appears. The message is defined in this section of the HTML page:

```
function onBeforeUnload() {
    var connected = document.javaclientname.isConnected();
    if (connected) {
        alerted = true;
        return "Closing this window will disconnect your ICA session";
    }
}
```

where *javaclientname* is the name of the applet.

To change the message displayed, edit the text in the HTML page.

Editing seamless1.html

Seamless1.html contains the following applet tag:

```
<applet name="javaclient"
    code="com.citrix.ConnectionCenter"
    codebase="../"
    archive="JICA-coreN.jar,JICA-configN.jar,JICA-seamlessN.jar"
    width="600"
    height="400">
    <param name="cabinets" value=
        "JICA-coreM.cab,JICA-configM.cab,JICA-seamlessM.cab">
    <param name="Address" value="Notepad">
    <param name="InitialProgram" value="#Notepad">
    <param name="HTTPBrowserAddress" value="plateau">
    <param name="TWIMode" value="on">
</applet>
```

The applet tag is similar to that used in desktop.html (see “Editing desktop.html” on page 18), but note the following differences:

- Code. The name of the class file that is executed. For the client with the Connection Center, this is always com.citrix.ConnectionCenter.
- Archive and Cabinets. The JICA-seamless archive is required for Connection Center and seamless windows.
- Width and Height. These are set to 600 and 400 pixels respectively, which are appropriate dimensions for the Connection Center user interface.
- TWIMode. This parameter enables seamless windows. Seamless windows are required for session sharing. If you use Connection Center without seamless windows, there is no session sharing, in other words each application is launched in a separate ICA connection and you gain no reduction in system resource overhead.

You can customize the sample HTML files and specify additional HTML files to launch your own published applications. If you have many applications, it may be easier to provide users with HTML links that all reference a server-side script that generates the <applet> tag as needed, based on the selected link.

To customize and use seamless1.html

1. Open seamless1.html in a plain text editor and find the <applet> tag section:

```
<applet name="javaclient"
 code="com.citrix.ConnectionCenter"
 codebase="../"
 archive="JICA-coreN.jar,JICA-configN.jar,JICA-seamlessN.jar"
 width="600"
 height="400">
 <param name="cabinets" value="JICA-coreM.cab,
        JICA-configM.cab,JICA-seamlessM.cab">
 <param name="Address" value="Notepad">
 <param name="InitialProgram" value="#Notepad">
 <param name="HTTPBrowserAddress" value="plateau">
 <param name="TWIMode" value="on">
</applet>
```

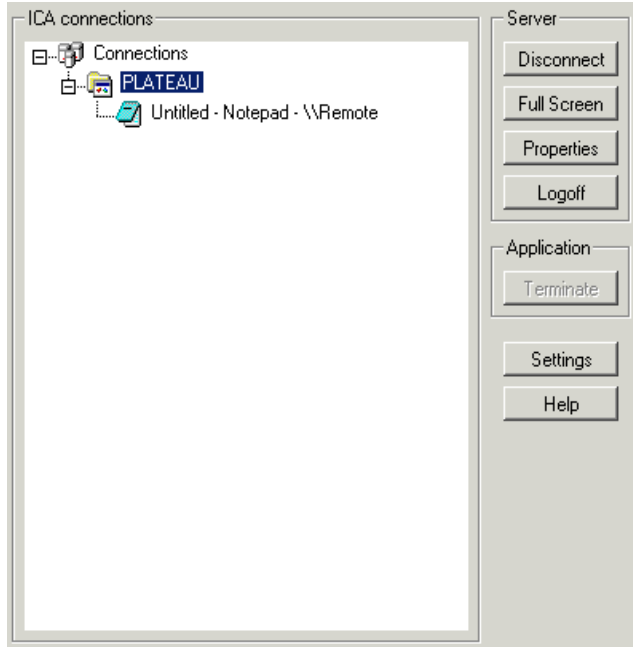
   This is the section that launches the client.

2. Change the value of the Address parameter to the name of the published application to which you want to connect.
3. Change the value of the InitialProgram parameter to the application name specified in Address, preceded by a # symbol. For example, if the published application is called **Word**, use the following parameters:

```
<param name="Address" value="Word">
<param name="InitialProgram" value="#Word">
```

4. Change the relative path specified for the codebase if it is not correct for your deployment.
5. Change the value of the HTTPBrowserAddress parameter to the address of the server used for HTTP browsing.
6. Publish the sample HTML pages using your Web server. See your Web server documentation for more information about how to do this.
7. On the client device, open a Web browser and open the URL for the sample HTML pages. The index.html page opens.

8. Click the **Launch seamless application 1** link. The first time in each Web browser session that you select a seamless link, the Connection Center applet starts up, launches an ICA connection to the specified application, and displays it in a separate window. The illustration below shows the Connection Center window:



Screenshot of the Connection Center window

9. If you select another seamless link while this session is open, the Connection Center applet starts up again. However, this new applet is displayed as a progress indicator rather than a duplicate Connection Center, and it closes itself when the new application is opened. In this way the launching and management of all applications is centralized by the initial Connection Center instance, which performs session sharing when possible.

Double-clicking a window node on the Connection Center tree brings that window to the front. If the window is minimized, double-clicking the node has no effect.

Right-clicking a window node displays the standard **System** menu.

Closing the Connection Center window disconnects all connected sessions, after prompting the user for confirmation.

Using Signed Java Applets

Due to security restrictions imposed by Java, many Java environments do not permit users to connect to other computers on the network when using Java applets.

When a Java applet attempts to make a connection to the server specified in the HTML page, the Java security manager detects the attempt to connect to another computer and cancels the operation. The result of this security restriction is that, under normal conditions, a client system can connect to a server only if the server is also the same device as the Web server that contains the applet class files.

To overcome this restriction, the client uses signed archives. The signature confirms that the files being downloaded came from Citrix and have not been altered since the signature was applied. You must ensure that users' Web browsers are configured to accept signed Java applets.

When attempting a connection to the server, the user is prompted with the Citrix signed certificate. When the user accepts the signature, the connection is permitted.

Deployment Example

In the following example, you want to make the desktop for a server called buster available to users. Buster is a server that runs Microsoft IIS and MetaFrame Presentation Server. You want the users to be able to use the drives on their client devices during ICA sessions, and to be able to print to local or network printers.

1. You go to the Citrix Web site and download JICAComponents.zip to C:\inetpub\wwwroot on buster.
2. You extract the files to C:\inetpub\wwwroot.
3. You open desktop.html in the examples folder using Notepad, then make and save the following changes:
   - To specify the correct server, you change the value of the Address parameter as follows:

     ```
     <param name="Address" value="buster">
     ```

   - To enable Internet Explorer users to map client drives and printers, you add JICA-cdmM.cab and JICA-printerM.cab to the Cabinets parameter, so that it reads as follows:

     ```
     <param name="cabinets" value="JICA-coreM.cab,JICA-configM.cab,
         JICA-cdmM.cab,JICA-printerM.cab">
     ```

- To enable other users to map client drives and printers, you add JICA-cdmN.jar and JICA-printerN.jar to the Archive attribute, so that it reads as follows:

```
archive="JICA-coreN.jar,JICA-configN.jar,JICA-cdmN.jar,
    JICA-printerN.jar">
```

4. You check that the users' Web browsers are configured to accept signed Java applets.
5. You publish desktop.html using the IIS Manager tool, and tell the users the URL of the page (http://buster/desktop.html).

Configuring the Client for Java

The client can be configured using:

- An HTML page, as described in this chapter.
- The **ICA Settings** dialog box.

  If a feature can be configured by the user through the **ICA Settings** dialog box, this is made clear in the relevant section of this chapter. To display the **ICA Settings** dialog box, click the **Settings** button. In seamless mode, the **Settings** button is on the Connection Center. If the client is not in seamless mode, the **Settings** button is on the status bar. To get help on the tasks you can carry out using the dialog box, click **Help**.

  You can prevent users configuring their own settings by removing the **Settings** button or the Status bar, as described in "Status Bar and Settings Button" on page 34.
- The Web Interface.

  The Web Interface automatically generates the necessary Web pages to launch the client. See the *Web Interface Administrator's Guide* for details of how to use the Web Interface to configure the client.

Note Drive mapping can be configured only through the **ICA Settings** dialog box.You cannot configure client drive mapping on an HTML page or through the Web Interface because this is a violation of the client's security.

Topics covered in this chapter include:

- Initial configuration
  - Specifying the language for the user interface
  - Configuring network protocol and server location
  - Changing the client name
  - Passing parameters to published applications

- User interface configuration
  - Setting the number of colors used in the ICA session window
  - Showing or hiding the status bar and Settings button
  - Controlling client behavior at the start and end of sessions
  - Specifying what type of keyboard to use
  - Configuring hotkeys
- Client device mapping
  - Client drive mapping
  - Client printer mapping
  - Client audio mapping
- Integrating the client with security solutions
  - Proxy servers
  - The Secure Gateway for MetaFrame Presentation Server and SSL Relay
  - Firewalls
  - ICA encryption
- Advanced configuration options

Initial Configuration

Starting the Client for Java in a Specific Language

The client allows you to specify which language to use to display the user interface. By default, a session uses the language specified for the client device to display the user interface. If you specify a language code that is not recognized or not supported, English is used.

To set the client language

Specify the following parameter in the HTML page:

```
<param name="Language" value="yourlanguage">
```

where *yourlanguage* is the two letter abbreviation for the language you want to use.

The standard two letter abbreviations are as follows:

English	en
French	fr
German	de
Spanish	es
Japanese	ja

For example, to use Japanese as the language on a non-Japanese device when connecting to the server named CitrixServer, create an applet tag like the following:

```
<applet code="com.citrix.JICA"
    archive="JICAEngN.jar"
    width="1024" height="768">
    <param name="cabinets" value="JICAEngM.cab">
    <param name="Address" value="CitrixServer">
    <param name="Language" value="ja">
</applet>
```

If you use languages other than English, ensure that your Web server sends HTML files with the correct Content-Type and Charset to avoid possible corruption of the applet parameter strings. Configuration details depend on the server software in use.

When troubleshooting suspected problems with parameter string encoding, it can be useful to copy the strings outside the applet tag and check that they display correctly in the Web browser.

Configuring Network Protocol and Server Location

This section explains how to:

- Set the network protocol used by the client
- Control the way in which the client locates servers
- Configure a list of servers for business recovery

Setting the Network Protocol

The network protocol setting allows you to control the way the client searches for servers and how it communicates with them.

The protocols are:

- TCP/IP + HTTP - The client uses the HTTP protocol to search for servers. The client communicates with the server using ICA protocol over TCP/IP. This is the default protocol.

- SSL/TLS + HTTPS - The client uses the HTTPS protocol to search for a list of servers. The client communicates with the server using the SSL or TLS protocols. These protocols are described in detail in "Integrating the Client with the Secure Gateway for MetaFrame Presentation Server or SSL Relay" on page 51.

To change the network protocol to SSL/TLS+HTTPS

Add the following parameter to the HTML page:

```
<param name= "SSLEnable" value="on">
```

Server Location

Server location (also called server browsing) is the mechanism by which a client discovers an appropriate server to host a given application. Depending on the server configuration, this can involve taking load balancing into account so that the user's application is run on the least loaded server.

The default browser server address is **ica**. You must set specific server addresses for MetaFrame Presentation Server computers unless your networking environment is configured with a DNS record for **ica**. The client uses the HTTP or HTTPS protocol respectively to contact the servers.

Example

In the following example, the HTTPBrowserAddress parameter is specified to be the server **Wizard**. This browser server is responsible for locating an appropriate server to run the published application **Notepad**.

```
<html>
    <body>
        <applet code="com.citrix.JICA"
            archive="JICAEngN.jar"
            width="640" height="480">
            <param name="cabinets" value="JICAEngM.cab">
            <param name="HTTPBrowserAddress" value="Wizard">
            <param name="Address" value="Notepad">
            <param name="InitialProgram" value="#Notepad">
        </applet>
    </body>
</html>
```

Business Recovery

Business recovery provides consistent connections to published applications in the event of a browser server disruption. You can define up to three groups of servers to which you want to connect: a primary and two backups. Each group can contain from one to five servers.

To specify a business recovery server group

Use the following parameters to specify server groups:

- **HTTPBrowserAddress, HTTPBrowserAddress2 to HTTPBrowserAddress5**: Specify the primary group of servers
- **HTTPBrowserAddress6 to HTTPBrowserAddress10**: Specify the first backup group of servers
- **HTTPBrowserAddress11 to HTTPBrowserAddress15**: Specify the second backup group of servers

Fill in any unused server addresses with five dashes (-----). These dashes are required to fill in any gaps in the list but are not required at the end of the list.

Example

In the following example, the primary group of servers contains **Arthur**, **Morgana**, and **Merlin**. The first backup group contains the servers **Excalibur** and **Stone**. There is no secondary backup group.

```
<html>
    <body>
        <applet code="com.citrix.JICA"
            archive="JICAEngN.jar"
            width="640" height="480">
            <param name="cabinets" value="JICAEngM.cab">
            <param name="HTTPBrowserAddress" value="Arthur">
            <param name="HTTPBrowserAddress2" value="Morgana">
            <param name="HTTPBrowserAddress3" value="Merlin">
            <param name="HTTPBrowserAddress4" value="-----">
            <param name="HTTPBrowserAddress5" value="-----">
            <param name="HTTPBrowserAddress6" value="Excalibur">
            <param name="HTTPBrowserAddress7" value="Stone">
        </applet>
    </body>
</html>
```

Changing the Client Name

The *client name* is used to identify the client to the server, and is also used to name printers. See “Client Printer Mapping” on page 43 for more information about printers. You may need to change the client name if you are trying to access resources shared with another client with the same name.

By default, the client uses the value for the client device’s host name (if it exists and is not set to **localhost**) as the client name reported to the server. If the client cannot use the client device’s host name, it uses **AnonJava** as the client name. The client name sent to the server is always truncated to 20 characters.

You can change the client name on the HTML page or by using the **ICA Settings** dialog box.

To change the client name

To change the client name, you need to add two parameters on the HTML page:

```
<param name="client.wfclient.UseHostname" value="off">
<param name="client.wfclient.Clientname" value="yourclient">
```

where *yourclient* is the client name you want to use.

Passing Parameters to Applications

When connecting to a published application, you specify the application name using the InitialProgram parameter on the HTML page. For example, to connect to an application called **Notepad**, specify the following parameters:

```
<param name="Address" value="Notepad">
<param name="InitialProgram" value="#Notepad">
```

You can also specify a command-line parameter that the server will pass to a published application when it runs that application. For example, if you provide a file name parameter to Notepad, Notepad will start up with that file loaded.

The parameters that an application will honor are built into the application and do not have to be file names.

To pass a parameter to an application

Specify the following parameters on the HTML page:

```
<param name="Address" value="application">
<param name="InitialProgram" value="#application">
<param name="Param" value="parameter">
```

where *application* is the name of the published application and *parameter* is the parameter. You can specify a maximum value length of 256 characters for these parameters. For example, to open a file called M:\new.txt in Notepad, specify the following parameters:

```
<param name="Address" value="Notepad">
<param name="InitialProgram" value="#Notepad">
<param name="Param" value="M:\new.txt">
```

Note For parameter passing to work, you must configure the published application to receive parameters by appending %* to the published application's command line. For example:

```
notepad %*
```

Full details of how to publish applications and set up commands are in the documentation included in the MetaFrame Presentation Server package.

File name parameters are interpreted by the remote application relative to the server's file system. If you want to pass a client-side file to a remote published application, you must use client drive mapping (see "Client Drive Mapping" on page 42).

User Interface Configuration

Window Properties

You set the dimensions of the applet panel in which ICA sessions run using the Width and Height attributes of the <applet> tag, as described in "Editing desktop.html" on page 18. You can also specify the size of the remote session by using the DesiredHRes and DesiredVRes parameters. If you do not specify these last two parameters, the remote session fits into the applet area available when any border and status bar are added.

The number of colors used in the session window is defined with a parameter on the HTML page.

To set the number of colors used for the ICA session window

Specify the following parameter in the HTML page

```
<param name="DesiredColor" value="2|4|8">
```

where 2 specifies 256 colors, 4 specifies thousands of colors, and 8 specifies millions of colors. The client cannot be configured to use only 16 colors but can display applications published in 16 color mode; these are run in 256 colors.

Status Bar and Settings Button

The status bar displays status information about the client. When the user clicks the **Settings** button, the **ICA Settings** dialog box appears.

You can display or hide the status bar and **Settings** button using parameters on the HTML page. Both are displayed by default. However, if you do not want users to make configuration changes, you may decide to hide the **Settings** button. If you decide that maximum screen real estate is a priority, you may decide to hide the status bar.

When the client is in seamless mode, the status bar is not visible and the user accesses the **Settings** dialog box from the Connection Center.

To hide the status bar

Specify the following parameter on the HTML page:

```
<param name="ShowStatusBar" value="no">
```

To hide the Settings button

Specify the following parameter on the HTML page:

```
<param name="ShowSettingsButton" value="no">
```

Note To display the **Settings** button and status bar when using the component archives, you must include the JICA-config archive. This functionality is included in the complete archives.

Auto-Reconnect and Session Termination

Note The information in this section does not apply if you are using Connection Center. Sessions always start automatically and, if the network connection is lost, an attempt is always made to reconnect.

You can control how the client behaves when starting or ending a session by specifying parameters on the HTML page.

You can configure the client to connect when the user clicks the startup splash screen, or to connect automatically, as the HTML page displays. You can also change what happens when a session ends, either as a result of the user terminating it or because there was a problem.

To change the client startup

Specify the following parameter on the HTML page:

```
<param name="Start" value="Manual|Auto">
```

If you set this parameter to **Manual** (the default), the user must click to connect to a server. If you set it to **Auto**, the message **Connecting to server** appears as the HTML page is displayed and the user is automatically connected to the server.

To change what happens when a session ends

Specify the following parameter on the HTML page:

```
<param name="End" value="Manual|Auto|Terminate|URL">
```

where:

- **Manual** - displays the startup splash screen when the session ends, and the message **Click to reconnect**. To reconnect, the user can click anywhere on the splash screen.
- **Auto** - displays the **Reconnecting** dialog box when the session ends for any reason. The number in the dialog box counts down to 0 and the client reconnects.
- **Terminate** - displays either **Connection Terminated** or **Connection Error** when the session ends, depending on whether the user chose to end the session or whether there was a problem that caused the session to end.
- ***URL*** - displays the splash screen and redirects in two seconds to the specified URL. You can specify the URL of any Web page.

The HTML examples supplied with the client include the End parameter. The value specified is a URL to a page called end.html. The applet tag section of each of the examples has this parameter:

```
<param name="End" value="end.html">
```

When you end the session, the client redirects to end.html, which contains a script to close the browser window. You can edit end.html to display anything you want.

To change the time-out period for automatic reconnection

The default time-out period is five seconds. To change the period, specify the following parameter in the HTML page:

```
<param name="ReconnectDelay" value="delay">
```

where *delay* is the delay in seconds. Specifying this parameter does not affect the delay before connection to an HTML page if you specified a URL for the End parameter.

Note If the Start and End parameters are both set to **Auto**, the startup splash screen is displayed and you must click on it to connect.

Keyboard and Mouse

The client lets you specify what type of keyboard to use in ICA sessions. By default, if you do not specify a keyboard preference, the session uses the default keyboard for the user's MetaFrame Presentation Server computer.

You can specify the keyboard type on the HTML page or the user can do it using the **ICA Settings** dialog box.

Note Different Java environments handle keyboard events differently. If you experience problems with unexpected keyboard behavior, consider upgrading or changing your Java Runtime Environment and/or Web browser.

The client supports the use of any keyboard supported by the server to which the user is connecting.

To specify a keyboard other than the server's default

Specify the following parameter on the HTML Page:

```
<param name="user.wfclient.keyboardlayout" value="layout">
```

where *layout* is a value from the server's list of supported keyboards. A list of the supported keyboards is provided in "Supported Keyboard Layouts" on page 85.

For example, to specify a Danish keyboard, create an HTML page like the following:

```
<html>
    <body>
        <applet code="com.citrix.JICA"
            archive="JICAEngN.jar"
            width="640" height="480">
            <param name="cabinets" value="JICAEngM.cab">
            <param name="Address" value="CitrixServer">
            <param name="user.wfclient.keyboardlayout" value="Danish">
        </applet>
    </body>
</html>
```

The client supports the following keyboard types to distinguish between subtypes of the Japanese keyboard layout:

```
"(Default)"
"IBM PC/XT or compatible keyboard"
"101 Keyboard (Japanese)"
"106 Keyboard (Japanese)"
"NEC PC-9800 on PC98-NX (Japanese)"
"NEC PC-9800 on PC98-NX 2 (Japanese)"
"NEC PC-9800 Windows 95 and 98 (Japanese)"
"NEC PC-9800 Windows NT (Japanese)"
"Japanese Keyboard for 106 (Japanese)"
"DEC LK411-JJ Keyboard (Japanese)"
"DEC LK411-AJ Keyboard (Japanese)"
```

Note If you are using a 109 key Japanese keyboard, specify the keyboard type as **106 Keyboard (Japanese)**.

To specify a Japanese keyboard type

Specify the following parameter on the HTML page:

```
<param name="user.wfclient.keyboardtype"
value="101 Keyboard (Japanese) | 106 Keyboard (Japanese)">
```

For example, to specify a Japanese 106 key keyboard, create an HTML page like the following:

```
<html>
    <body>
    <applet code="com.citrix.JICA"
    archive="JICAEngN.jar"
    width="640" height="480">
    <param name="cabinets" value="JICAEngM.cab">
    <param name="Address" value="CitrixServer">
    <param name="user.wfclient.keyboardlayout" value="Japanese">
    <param name="user.wfclient.keyboardtype"
                value="106 Keyboard (Japanese)">
    </applet>
    </body>
</html>
```

Japanese IME Support

The client provides a choice of options for using a Japanese Input Method Editor (IME). Users can configure these options using the **ICA Settings** dialog box. Alternatively, you can set the keyboard layout parameter in the applet tag: this overrides the users' settings.

You can choose between using a client-side IME or a server-side IME.

The benefit of using a client-side IME is that users can choose their preferred IME that they have installed on the client device, and also that they do not have to deal with one IME for local applications and another potentially different IME with a different dictionary for server-side applications. When using a client-side IME, the user does not compose the text "in place" (that is, at the insertion point where the user types), but in a separate composition window.

The benefit of using a server-side IME is that the user composes the text "in place."

To use a client-side IME for connections to servers running Windows Server 2003

Specify the following parameter on the HTML Page:

```
<param name="user.wfclient.keyboardlayout"
            value="Japanese (client IME only)">
```

To use a client-side IME for connections to servers running Windows 2000

Note This section does not apply to servers running Windows 2000 with Service Pack 4. If you have installed Service Pack 4, you can enter Japanese characters into the logon dialog using a client-side IME.

When connecting to MetaFrame Presentation Server on Windows 2000 with Service Pack 3 or earlier, users cannot use a client-side IME to enter Japanese characters into the session logon dialog. If they want to use a client-side IME but they also need to enter Japanese characters in the logon dialog, specify the following parameter on the HTML page:

```
<param name="user.wfclient.keyboardlayout"
            value="Japanese (client and server IME)">
```

This allows users to use the server-side IME to enter credentials in the logon dialog. After they log on to the session, they should turn off the server-side IME and use the client-side IME.

When the server-side IME is being used to enter logon credentials with this keyboard layout, if the server-side IME is in Kana mode, Shift+0 does not generate the "wo" character. There are three workarounds for this problem:

- Use the “Japanese (server IME only)” option as described in the following section
- Use Roman-input mode instead of Kana input mode
- Enter the “wo” character using the drawing interface provided by the IME pad

When using the Web Interface, select between “Japanese (client IME only)” and “Japanese (server IME only).” Do not select the keyboard layout “Japanese (client and server IME),” because users provide credentials using the Web Interface and do not need to manually log on to the session.

To use a server-side IME

Specify the following parameter on the HTML page:

```
<param name="user.wfclient.keyboardlayout"
            value="Japanese (server IME only)">
```

Alternatively, select “(Server Default)” for the keyboard layout and connect to a server with a Japanese keyboard layout with IME configured as the default keyboard layout.

Using Applications that Require a 3-Button Mouse

In the Client for Java, the middle button of a 3-button mouse can be emulated by clicking both buttons of a 2-button mouse at the same time.

Hotkeys

Hotkeys are used to control the behavior of the client and as substitutes for the standard Windows hotkeys. For example, if you want to display the Windows Security Desktop on a Windows PC, you press CTRL+ALT+DEL. If you are running the client on a Windows PC and are working in a MetaFrame Presentation Server session, this key combination opens the Security Desktop on the local device. Hotkey functionality allows you to map common key combinations like CTRL+ALT+DEL to a key combination such as CTRL+F1 that is ignored by your local operating system. When you press this new combination, the client sends CTRL+ALT+DEL to the server, displaying the Windows Security Desktop in your session.

You can specify hotkeys on the HTML page or the user can do it by using the **ICA Settings** dialog box.

Client hotkeys use a pair of keys. The first is a modifier key and the second is a character. The following hotkeys are available:

Hotkey (Default Mapping)	Description
Hotkey2 (Shift+F3)	Close Remote Application. The Close Remote Application hotkey disconnects applications opened in an ICA session. If no programs are open, the session is disconnected after the user is prompted for confirmation.
Hotkey3 (Shift+F2)	When the client is running in seamless mode, this hotkey toggles between seamless mode and "windowed" mode. When the client is not in seamless mode, this hotkey is used when a reconnected session is larger than the applet panel; it toggles between embedding the session inside the applet panel with scroll bars and displaying it in a separate window. When the client is running in full screen mode, this hotkey toggles the title bar on and off.
Hotkey4 (CTRL+F1)	Substitute for the standard Windows hotkey CTRL+ALT+DEL. The CTRL+ALT+DEL hotkey displays the Windows Security Desktop in the ICA session.
Hotkey5 (CTRL+F2)	Substitute for the standard Windows hotkey CTRL+ESC. On MetaFrame Presentation Server computers, the remote Windows **Start** menu appears.
Hotkey6 (ALT+F2)	Substitute for the standard Windows hotkey ALT+ESC. This hotkey is used to bring the focus to maximized and minimized windows of programs that are open in an ICA session, in the order that they were opened.
Hotkey7 (ALT+PLUS)	Substitute for the standard Windows hotkey ALT+TAB. Use this hotkey to cycle through applications that are open in the ICA session. A popup box appears and displays the programs as you cycle through them. The chosen application receives keyboard and mouse focus.
Hotkey8 (ALT+MINUS)	Substitute for the standard Windows hotkey ALT+Shift+TAB. Like the ALT+TAB hotkey, this key sequence cycles through applications that are open in the ICA session but in the opposite direction. The chosen application receives keyboard and mouse focus.
Hotkey9 (CTRL+F3)	Substitute for the standard Windows hotkey CTRL+Shift+ESC. Use this hotkey to display the Task Manager.
Hotkey10 (CTRL+F4)	Toggle SpeedScreen latency reduction. Toggles mouse click feedback and local text echo on and off.
Hotkey 11 (ALT+*)	DIsplay the **ICA Settings** dialog box.

To change the hotkey sequence from the default

For each hotkey, specify two parameters on the HTML page: one for the shift state and a second for the character state, as follows:

```
<param name="user.wfclient.hotkey*shift" value="shiftstate">
<param name="user.wfclient.hotkey*char" value="character">
```

where star (*) is the hotkey number; *shiftstate* is ctrl, shift, alt, or (none); and *character* can be any of the following:

F1 through F12, tab, star, plus, minus, escape, (none)

Specifying **(none)** for the character disables the hotkey.

The following example describes how to map the Close Remote Application hotkey to the key sequence CTRL+F1 and the ALT+TAB hotkey key sequence to Shift+TAB.

```
<html>
    <body>
        <applet code=com.citrix.JICA
            archive="JICAEngN.jar"
            width=640 height=480>
            <param name="cabinets" value="JICAEngM.cab">
            <param name="Address" value="CitrixServer">
            <param name="user.wfclient.hotkey2shift" value="ctrl">
            <param name="user.wfclient.hotkey2char" value="f1">
            <param name="user.wfclient.hotkey7shift" value="shift">
            <param name="user.wfclient.hotkey7char" value="tab">
        </applet>
    </body>
</html>
```

Client Device Mapping

The client supports *client device mapping* for connections to servers. Client device mapping allows a remote application running on the server to access printers and disk drives attached to the local client machine. The applications and system resources appear to the user at the client machine as if they are running locally. Ensure that client device mapping is supported on your server before using these features.

This section includes more information about:

- Client drive mapping
- Client printer mapping
- Client audio mapping

Client Drive Mapping

Client drive mapping can make any specified directory on the client machine, including CD-ROMs, available to the user during ICA sessions. When a server is configured to allow client drive mapping, users can access their locally stored files, work with them during their ICA sessions, and then save them again either on a local drive or on a drive on the server.

The user's home directory is automatically mapped to drive H. Note that this feature is not available when using the Microsoft JVM.

When drive mapping is configured, the client attempts to use it for all connections. If the server does not support drive mapping, or if the Java environment is configured not to allow access to local drives, drive mapping is not available.

Drive mapping can be configured only by the user through the **ICA Settings** dialog box.You cannot configure client drive mapping on an HTML page or through the Web Interface because this is a violation of the client's security.

To make client drive mapping available to users when using the core archives, you must include the JICA-cdm archive. If you do not, the **Drive Mapping** tab does not appear in the **ICA Settings** dialog box.

Configure client drive mapping by making the **ICA Settings** dialog box available to users the first time they log on and instructing them to configure drive mapping. The **ICA Settings** dialog box is described in the Help. See "Copying Client Drive Mapping Configuration" on page 59 for information about how to copy client drive mapping configuration to multiple users.

Using Mapped Drives

Once configured, mapped drives are transparent and appear the same as other network drives on the server. However, due to the way Java accesses file systems, mapped drives cannot:

- **Lock files**. Applications sometimes lock the files that are in use to prevent other applications from accessing them at the same time and possibly corrupting them. However, the client cannot lock local files. To prevent file corruption, warn users not to access the same file with two or more applications at the same time.
- **Use file attributes**. Attributes cannot be set on files on a mapped drive.
- **Set date and time**. The date and time stamps are not set on files created or edited on mapped drives.
- **Report drive capacity and usage**. Users must use the operating system of their local machine to determine the capacity of mapped drives.

Client Printer Mapping

There are two ways to map printers, depending on your environment:

- In a J2SE 1.4.x environment, printers are auto-detected (see "Auto-Detected Printers" on page 43). No manual configuration is necessary.
- If you are not in a J2SE 1.4.x environment, you must configure printers manually using any of the following:
  - The **ICA Settings** dialog box (described in the Help). This is the easiest way to do it.
  - The HTML page (described in "Manually Configured Printers" on page 44).
  - The Printer Management node on the Presentation Server Console. This tool is described in Chapter 11 of the *MetaFrame Presentation Server Administrator's Guide*.

To make client printer mapping available to users, you must specify the JICA-printer archive in the <applet > tag or use the full archive. If you do not, the Printer Mapping functionality is not available, so the relevant tab does not appear in the **ICA Settings** dialog box. Users with auto-detected printers still need access to the **Printer Mapping** tab in case they need to modify their printer settings.

Note Before users can print to a client printer from MetaFrame Presentation Server for UNIX, printing must be enabled by the MetaFrame administrator. For further information, see the *MetaFrame Presentation Server for UNIX Administrator's Guide,* and the manual pages.

Auto-Detected Printers

In a J2SE 1.4.x environment that supports the Java Print Service API, the client automatically detects all printers available to the client device, including USB printers, and makes them available to the session. This means that no mapping or manual configuration is required.

For PostScript-capable printers, a generic Postscript driver is configured on the server, and the resulting PostScript output is sent directly to the printer. For non-PostScript printers, the Universal Print Driver (UPD) is configured.

To modify printer settings, users select the **Printer Mapping** tab on the **ICA Settings** dialog box. If a print job requires color or advanced printing options such as duplex printing, users should configure an appropriate native driver. If they configure both a native driver and a UPD driver, the server uses the native driver if it is available, otherwise it uses the UPD driver.

Users cannot delete auto-detected printers unless the Java environment detects that they are no longer available.

Note Mac OS X provides a J2SE 1.4.x environment but does not provide the Java Print Service API, so printers are not auto-detected.

Manually Configured Printers

When you configure printers manually, as described below, users find those printers mapped to their sessions and ready for use when they log on. When they log off, their printer mappings are deleted from the server. The printers are automatically mapped again the next time they log on.

Note You cannot configure USB printers manually.

To configure printers manually

Specify the printer name, the port name, and the driver, using the following parameters in the HTML page:

```
<param name="user.localclientprinters" value="printername">
<param name="user.printername.port" value="portname">
<param name="user.printername.driver" value="drivername">
```

where:

- ***printername*** **-** Can be any name by which you want to identify the printer.
- ***portname*** **-** Specifies a file name, port name, or printer IP address (or network name and print queue).

  Note When mapping printers attached to a Macintosh you can specify only a file name, not a port name or printer IP address.

  When printing to a file, the output file is composed of printer machine code. This file can be sent to a printer using a platform-specific utility. For example, use a command prompt on Windows platforms to send the file to a printer by copying the file to a printer port.

  When printing to a port, specify the port. A typical port on Windows is LPT1. On Linux or UNIX systems, the port is similar to /dev/lp0. Check your operating system documentation for more information.

  When printing to a network printer, specify the printer's IP address or network name and print queue (*ipaddress*:*printqueuename* or *networkname*:*printqueuename*).

- ***drivername*** - Specify the printer driver. This name is case-sensitive and must exactly match the driver name on the server. To check the driver list on servers running MetaFrame Presentation Server:

  1. Double-click **My Computer.**
  2. Double-click **Printers**, then double-click **Add Printer**.
  3. In the Add Printer wizard, be sure My Computer is selected. Click **Next**.
  4. Select a port, such as LPT1. Click **Next**.

     The list under Printers contains the printer driver names.

You can also make a printer the default printer for a session by specifying the following parameter:

```
<param name="user.printername.comment" value="WFCDefault">
```

where *printername* is the name of the printer. If the server is set to connect only to the user's default printer, this sets the manually configured printer to be the default printer.

Note The server must have the correct printer driver installed, as specified either in the **ICA Settings** dialog box or on the HTML page using the Driver parameter. If the correct driver is not installed, the printer is not configured. In this case, you must install the correct printer driver on the server.

You can change the list of drivers that appears in the **ICA Settings** dialog box by editing the ICAPrinterDrivers.txt file. See "Changing the List of Printer Drivers" on page 59.

Examples: Manually Configuring Individual Printers

The following example demonstrates configuring a printer by specifying parameters on the HTML page. This HTML page is suitable for use only with a Windows client system. In this example, the printer's name is LocalPrinter1. It is connected to the client's LPT1 port and has a driver named HP LaserJet .

```
<html>
    <body>
    <applet code="com.citrix.JICA"
    archive="JICAEngN.jar"
    width="640" height="480">
    <param name="cabinets" value="JICAEngM.cab">
    <param name="Address" value="CitrixServer">
    <param name="user.localclientprinters" value="LocalPrinter1">
    <param name="user.LocalPrinter1.port" value="lpt1:">
    <param name="user.LocalPrinter1.driver" value="HP LaserJet">
    </applet>
    </body>
```

```
</html>
```

In the following example, the printer's name is NetPrinter1 and has a driver named HP LaserJet. The printer is a network printer that exists on a network print server with an IP address of 192.168.1.24 and a print queue named FLOOR2_LJ.

The port name of a network printer can be either the printer's IP address and print queue, or its network name and print queue.

```
<html>
    <body>
    <applet code="com.citrix.JICA"
    archive="JICAEngN.jar"
    width="640" height="480">
    <param name="Address" value="CitrixServer">
    <param name="user.localclientprinters" value="NetPrinter1">
    <param name="user.NetPrinter1.port" value="192.168.1.24:FLOOR2_LJ">
    <param name="user.NetPrinter1.driver" value="HP LaserJet">
    </applet>
    </body>
</html>
```

Example: Manually Configuring Multiple Printers

The following example shows how you would configure both printers from the previous two examples. Note how the two printer names, LocalPrinter1 and NetPrinter1, are both specified in the user.localclientprinters parameter.

```
<html>
    <body>
    <applet code="com.citrix.JICA"
        archive="JICAEngN.jar"
        width="640" height="480">
        <param name="Address" value="CitrixServer">
        <param name="user.localclientprinters"
            value="LocalPrinter1,NetPrinter1">
        <param name="user.LocalPrinter1.port" value="lpt1:">
        <param name="user.LocalPrinter1.driver" value="HP LaserJet">
        <param name="user.NetPrinter1.port"
            value="192.168.1.24:FLOOR2_LJ">
        <param name="user.NetPrinter1.driver"
            value="HP LaserJet 400 Series PS">
    </applet>
    </body>
</html>
```

Client Audio Mapping

Client audio mapping enables applications running on the server to play sounds through a sound device installed on the client computer. To make client audio mapping available to users, you must enable it using a parameter on the HTML page.

Starting the client while an audio application is running on your desktop can disable audio mapping. Do not run audio applications while starting the client.

Note Java Plug-in 1.4.1 or later is required for audio to work on Linux platforms. The client explicitly turns off audio when running on Linux platforms with earlier Java environments because these environments can crash or hang on Linux when audio is enabled.

To enable client audio mapping

Add the following parameter to the HTML page:

```
<param name="ClientAudio" value="on">
```

You control the amount of bandwidth used by client audio mapping by configuring the ICA Settings on the server, as described in Chapter 7 of the *MetaFrame Presentation Server Administrator's Guide*.

Note If the server is set to use **Low** quality audio, client audio mapping for the client is disabled.

Integrating the Client with Security Solutions

You can integrate the client with a range of security technologies, including proxy servers, firewalls, and SSL/TLS based systems. This section describes:

- Making a connection through a SOCKS proxy server or Secure proxy server (also known as Security proxy server, HTTPS proxy server, or SSL tunneling proxy server)
- Integrating the client with the Secure Gateway for MetaFrame or SSL Relay solutions with Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols
- Connecting to a server across a firewall using alternate addressing
- Using ICA encryption to help protect data on local networks

For full details of the security technologies listed above and advice on security planning, see the *MetaFrame Presentation Server Administrator's Guide*.

Connecting Through a Proxy Server

Proxy servers are used to limit access into and out of your network, and to handle connections between clients and servers. The Client for Java supports both SOCKS and Secure proxy protocols.

Configure the client to work with a proxy server by specifying parameters on the HTML page, through the **ICA Settings** dialog box, or through the Web Interface. This section describes how to specify a proxy server on the HTML page. Performing this task using the **ICA Settings** dialog box is described in the Help. Configuring the client through the Web Interface is described in the *Web Interface Administrator's Guide*.

Some proxy servers require authentication when a connection is requested. Specify a proxy logon name and password on the HTML page when configuring the client. If authentication is required and you do not specify the details on the HTML page, users are prompted to enter the logon name and password when they open an ICA connection.

Note The client supports only Basic proxy authentication when connecting to a Secure proxy, and user name/password authentication when connecting to a SOCKS proxy. Proxy authentication therefore does not work with proxy servers configured to use other authentication schemes such as NTLM, Kerberos, and Digest.

You can configure the client to use a proxy server in any of the following three ways:

- Enabling proxy auto detection
- Using a PAC (proxy auto configuration) file to automate proxy configuration
- Manually specifying proxy server details

Note Proxy detection and PAC files are not supported on Mac OS X. To connect through a proxy server, you must configure the proxy details manually, as described in "Manually Specifying Proxy Server Details" on page 50.

Enabling Proxy Auto Detection

If you are deploying the client in an organization with many proxy servers, consider using proxy auto detection. Proxy auto detection obtains proxy details from the local Web browser settings. It is also useful if you cannot determine which proxy server will be used when you configure the client. Proxy auto detection can be used with:

- Internet Explorer 4.0 or later for Windows using Microsoft's VM or the Sun Plug-in
- Netscape 6.x or later for Windows, UNIX, and Linux
- Other Web browsers that use the Sun Plug-in; for example, Mozilla

---

Note If you are using Internet Explorer with Microsoft's VM and Internet Explorer is set to automatically detect proxy settings, but http://wpad/wpad.dat (the file that contains the proxy details) cannot be contacted, the client tries a direct connection. To disable this fallback behavior, set the ProxyAutoDetectFallback parameter to No.

To automatically detect the proxy server settings used by the Web browser (proxy auto detection)

Specify the following parameter in the HTML file:

```
<param name="ProxyType" value="auto">
```

Using a PAC File to Automate Proxy Configuration

You can also automate proxy configuration by using a PAC file. A PAC file is a JavaScript file that is served from a local Web server and used to automatically configure the proxy settings of Web browsers.

Note that you cannot specify this method of proxy configuration through the **ICA Settings** dialog box.

To obtain the proxy server settings from a PAC file

Specify the following parameters in the HTML file:

```
<param name="ProxyType" value="script">
<param name="ProxyAutoConfigURL"
    value="http://webserver.mycompany.com/myproxies.pac">
```

where http://*webserver.mycompany.com/myproxies.pac* is the URL for the PAC file.

Manually Specifying Proxy Server Details

If you are manually specifying the proxy server, you need to know its address. You also need to know its port number if it is not set to 1080 for a SOCKS proxy server or 8080 for a Secure proxy server.

To specify the proxy server details manually

Note If you are configuring the proxy manually, confirm these details with your security administrator. ICA connections cannot be made if these details are incorrect.

Specify the following details using parameters in the HTML file:

- The address of the proxy server.
- The port number of the proxy server (if not 1080 for SOCKS or 8080 for Secure proxy).
- The protocol of the proxy server: SOCKS proxy or Secure proxy.
- In the case of a SOCKS proxy, the protocol version number. Alternatively, you can omit the version number; the client will then try SOCKS Version 5 and fall back to SOCKS Version 4 if necessary.
- If the proxy requires authentication and you are supplying it through the HTML page, the logon name and password.

Parameter	Description
ProxyType=none | auto | socks | socksv4 | socksv5 | secure | script	none - no proxy auto - use the Web browser's settings socks - use SOCKS - automatically detect the version socksv4 - use SOCKS Version 4 socksv5 - use SOCKS Version 5 secure - Use Secure proxy script - use a PAC file (specified by ProxyAutoConfigURL)
ProxyHost=*address:port*	Address and port (if required) of the proxy server
ProxyUsername	Proxy Username
ProxyPassword	Proxy Password

Parameter	Description
ProxyExcludeList=*address1*; *address2*; etc...	A semicolon-separated list of addresses of servers that the client must connect to directly—not through the proxy server.
ProxyAutoConfigURL	The URL for the PAC file. Use with the parameter ProxyType.
ProxyAutoDetectFallback	Specifies whether or not to make a direct connection to the server if Internet Explorer is set to automatically detect proxy settings, but the file that contains the settings (http://wpad/wpad.dat) is not accessible. Values are **yes** or **no**. The default value is **yes**.

For example, to connect to a server named Norbert using a SOCKS proxy server, Version 5, at the IP address 10.45.1.3 and port 1080, use an HTML page like the following:

```
<html>
    <body>
        <applet code="com.citrix.JICA"
            archive="JICAEngN.jar"
            width="640" height="480">
            <param name="cabinets" value="JICAEngM.cab">
            <param name="Address" value="Norbert">
            <param name="ProxyType" value="socksv5">
            <param name="ProxyHost" value="10.45.1.3:1080">
            <param name="ProxyUsername" value="dentres">
            <param name="ProxyPassword" value="sangle">
        </applet>
    </body>
</html>
```

Integrating the Client with the Secure Gateway for MetaFrame Presentation Server or SSL Relay

You can integrate the client with the Secure Gateway or with an SSL relay service. The client supports both SSL and TLS protocols:

- SSL provides strong encryption to increase the privacy of your ICA connections and certificate-based server authentication to ensure that the server you are connecting to is a genuine server.

- TLS (Transport Layer Security) is the latest, standardized version of the SSL protocol. The Internet Engineering Taskforce (IETF) renamed it TLS when they took over responsibility for the development of SSL as an open standard. TLS secures data communications by providing server authentication, encryption of the data stream, and message integrity checks. Because there are only minor technical differences between SSL Version 3.0 and TLS Version 1.0, the certificates you use for SSL in your MetaFrame installation also work with TLS. Some organizations, including US government organizations, require the use of TLS to secure data communications.

For more information about Secure Gateway, see the *Secure Gateway Administrator's Guide*.

Configuring and Enabling the Client for SSL and TLS

SSL and TLS are configured in the same way, use the same certificates, and are enabled with the same parameter. You configure SSL and TLS using parameters on the HTML page or with the Web Interface.

When SSL and TLS are enabled, each time you initiate a connection the client tries to use TLS first, then tries SSL. If it cannot connect with SSL, the connection fails and an error message appears.

If you are using a J2SE environment, the comprehensive set of root certificates stored in the environment is automatically used.

If you are not using a J2SE environment, the seven certificates built in to the client are used. These are listed below: check that they meet the security requirements for your organization.

If your organization has a certification authority and you need to specify your own certificates, see "Configuring the Client for Use with Your Security Solution" on page 53.

Certificate	Issuing Authority
Class4PCA_G2_v2.crt	VeriSign Trust Network
Class3PCA_G2_v2.crt	VeriSign Trust Network
BTCTRoot.crt	Baltimore Cyber Trust Root
GTECTGlobalRoot.crt	GTE Cyber Trust Global Root
GTECTRoot.crt	GTE Cyber Trust Root
Pcs3ss_v4.crt	Class 3 Public Primary Certification Authority
SecureServer.crt	Secure Server Certification Authority

To enable SSL and TLS

1. If you have not already done so, include the appropriate cryptoj and ssl archives in the archive parameter in the applet tag on the HTML page. These archives are not included in the complete JICAEng archive and you must include them before SSL or TLS encryption can be used.
2. Configure the Web server so that the HTML page specifying the applet can be delivered to the Web browser only through an SSL/TLS (https://) connection.

   CAUTION Security is seriously compromised if this step is omitted.

3. Change the network protocol to SSL/TLS+HTTPS and enable SSL by adding the following parameters to the HTML page:

   ```
   <param name="SSLEnable" value="on">
   <param name="HTTPBrowserAddress" value="myserver.mycompany.com">
   ```

   where *myserver.mycompany.com* is the fully qualified domain name of a MetaFrame Presentation Server computer.

To force TLS connections

To force clients to connect only with TLS, you must specify TLS on the Secure Gateway server or SSL relay service. See the *Secure Gateway Administrator's Guide* or SSL relay service documentation for more information.

Specify the following parameter in the HTML file:

```
<param name="SecureChannelProtocol" value="TLS"
```

Configuring the Client for Use with Your Security Solution

When you enable the SSL/TLS+HTTPS protocol, the default settings apply. The client attempts to connect using the TLS protocol. If a TLS connection cannot be established, the client tries SSL. If an SSL connection cannot be established, the connection fails and an error message appears.

You must perform further configuration if:

- You are using a Secure Gateway server that is configured to run in relay mode. If you are not sure what this means, see the *Secure Gateway Administrator's Guide* or contact your security administrator for more information.
- You are not using the default Cipher Suite. You may need to use the Com or Gov cipher suites to comply with your organization's security regulations. If you are not sure, contact your security administrator.

- Your organization has its own certification authority. You may need to specify alternative root certificates to comply with your organization's security regulations. If you are not sure, contact your security administrator. It is important to verify the authenticity of a root certificate before installing it.Your organization should have a procedure in place for users to check the root certificate as they install it.

This section describes how to configure these solutions.

To specify a Secure Gateway server that is configured to run in Relay mode

Add the following parameter to the HTML page:

```
<param name="SSLProxyHost" value="address:port">
```

where *address* is the fully qualified domain name of the Secure Gateway server and the same domain name specified in the server certificate. If the server port is not 443, specify the port. You can also specify a Secure Gateway server address in the **ICA Settings** dialog box.

To use a different Cipher Suite

Add the following parameter to the HTML page:

```
<param name="SSLCiphers" value="All|Com|Gov"
```

where:

Option	Description
All	SSL_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_SHA SSL_RSA_WITH_3DES_EDE_CBC_SHA
Com	SSL_RSA_WITH_RC4_128_MD5 SSL_RSA_WITH_RC4_128_SHA
Gov	SSL_RSA_WITH_3DES_EDE_CBC_SHA

To use your own certificates in a J2SE environment

In a J2SE environment you use the Java keytool utility to import SSL certificates for each client device. For information on how to use this utility, refer to the Sun documentation. Here is a sample command line showing how to use keytool to import a root certificate:

```
keytool -import -trustcacerts -alias zoo1 -file c:\bin\zoo.cer -
keystore c:\Program Files\Java\j2rel.4.2_01\lib\security\cacerts
```

The keystore parameter is mandatory.

Certificates generated by Microsoft Certificate Server cannot be imported into the keystore with the 1.4.2 environment. If you need to use this type of certificate, import them using the procedure for non-J2SE environments described below.

To use your own root certificates in a non-J2SE runtime environment

To use your own root certificates in a non-J2SE environment, you need to create an archive that contains your certificates, add this archive to the list of client archives, and then use parameters to specify the number of certificates and their names:

1. Download a development kit. Before you can create the archive, you must obtain an appropriate kit.
   - To create .jar files for Netscape and other Java environments, download a copy of the Sun JDK from http://www.java.sun.com/
   - To create .cab files for Internet Explorer, download the Cabinet Software Development Kit from http://www.microsoft.com
2. Create the archive as a .jar or .cab file.
   - To make a .jar file containing the certificates A.crt, B.crt, and C.crt, install the Sun JDK tools (and put them on your path), then type the following at the command line:

     ```
     jar -cf MyCertificates.jar A.crt B.crt C.crt
     ```

     This command generates an archive called MyCertificates.jar. Specify the archive in the archive parameter in the HTML page. Use commas to separate multiple archives in an applet tag.
   - To make a .cab file containing the certificates A.crt, B.crt, and C.crt, install the Cabinet Software Development Kit (and ensure the CABARC tool is in your path), then type the following at the command line:

     ```
     CABARC n MyCertificates.cab A.crt B.crt C.crt
     ```

     This command generates an archive called MyCertificates.cab. Specify the archive in the cabinets parameter of your HTML page. Use commas to separate multiple archives in a cabinets parameter.

3. Specify which SSL certificates the client should use. Use the following parameters:

   SSLNoCACerts - The number of specified certificates in the archive. The default value of zero indicates that only the set of certificates built in to the client will be used.

   SSLCACert0, SSLCACert1, ... SSLCACert<n> - The names of SSL root certificates to use for validating the SSL certificate obtained from the SSL relay. The number of root certificates you specify must match the number specified in the SSLNoCACerts parameter. With these parameters, you can also specify the client's built-in certificates, in addition to your own custom certificates (the former reside in the ssl archive, the latter reside in the archive you created following the steps above).

Here is an example of an <applet> tag that enables SSL with the client and uses the custom root certificates A.crt and B.crt, in addition to the client's built-in root certificates, Verisign Class 4, and Baltimore.

```
<applet code="com.citrix.JICA"
    width="640" height="480">
    <param name="cabinets"
    value="JICA-coreM.cab,sslM.cab,cryptojM.cab,MyCertificates.cab">
    <param name="Address" value="myserver.mycompany.com">
    <param name="SSLEnable" value="on">
    <param name="SSLNoCACerts" value="4">
    <param name="SSLCACert0" value="Class4PCA_G2_v2.crt">
    <param name="SSLCACert1" value="BTCTRoot.crt">
    <param name="SSLCACert2" value="A.crt">
    <param name="SSLCACert3" value="B.crt">
</applet>
```

Connecting to a Server Across a Firewall

Network firewalls can allow or block packets based on the destination address and port. If you are using the client through a network firewall that employs IP address translation, you must specify the following parameters:

- **UseAlternateAddress**: Use an alternate address across a firewall (specified by the parameter HTTPBrowserAddress). The values are **0** (default; actual address is used) and **1** (alternate address is used). If this parameter is set to **1**, the parameter HTTPBrowserAddress must also be specified. Setting this parameter to **0** is the same as not using the parameter.

  You can also enable alternate addressing using the **ICA Settings** dialog box.

- **HTTPBrowserAddress**: The external Internet address of a server.

Note All servers in the farm must be configured with their alternate (external) address. See the *MetaFrame Presentation Server Administrator's Guide* for more information.

For example, to connect to a server across a firewall in applet mode and use an alternate address for the server **Fountain**, make an HTML page like the following:

```
<html>
    <body>
    <applet code="com.citrix.JICA"
        archive="JICAEngN.jar"
        width="640" height="480">
        <param name="cabinets" value="JICAEngM.cab">
        <param name="Address" value="Fountain">
        <param name="HTTPBrowserAddress" value="177.17.1.7">
        <param name="user.wfclient.UseAlternateAddress" value="1">
    </applet>
    </body>
</html>
```

Using ICA Encryption

The default level for ICA encryption is **Basic**. To enable ICA encryption levels higher than **Basic**:

1. Ensure the server is configured to allow the selected encryption level or greater. To enable encryption levels higher than **Basic**, the server must support RC5 encryption.
2. If you are using the core archive, include the cryptoj and JICA-sica archives in the archive attributes or cabinets parameter on the HTML page. If you are using a complete archive, include only cryptoj.
3. Set the appropriate encryption level, as described below.

To specify ICA encryption

Use the following parameter on an HTML page:

```
<param name="EncryptionLevel" value="0|1|2|5">
```

The number you enter corresponds to the encryption level required as follows:

Level	Description
0	No encryption (needs server setup; default is Basic).
1	Basic encryption. This is the default.
2	RC5 128-bit encryption during authentication. After the logon process is successfully completed, the encryption level changes to Basic.
5	RC5 128-bit. This is intended for users who are dealing with sensitive data and need a high level of privacy and integrity.

To create a connection to a server called CitrixServer using 128-bit ICA encryption using Netscape 6, create an HTML page like the following:

```
<html>
    <body>
    <applet code="com.citrix.JICA"
    archive="JICA-coreN.jar,cryptojN.jar,JICA-sicaN.jar"
    width="640" height="480">
    <param name="cabinets"
        value="JICA-coreM.cab,cryptojM.cab,JICA-sicaM.cab">
    <param name="Address" value="CitrixServer">
    <param name="EncryptionLevel" value="5">
    </applet>
    </body>
</html>
```

Advanced Configuration

Precedence of Configuration Locations

This section describes the different places parameters can be specified and the precedence of each place.

The client configuration is stored in three places:

- The HTML page, stored on a Web server
- The ICA file, served from a server running the Web Interface or from a standard Web server
- The appsrv.ini file, stored on the user's local device

Parameters specified on the HTML page take precedence over those specified in the ICA file. Similarly, parameters specified in the ICA file take precedence over those in the appsrv.ini file. Note that some parameters can be specified only in the .ini file.

The location of the appsrv.ini configuration file depends on the client operating system and Web browser. The locations for some common Web browser and operating system combinations are as follows:

Sun Plugin 1.3.x and 1.4.x (Win32):

```
%USERPROFILE%\Citrix
```

Microsoft VM (Win32):

```
%SYSTEMROOT%\Java\Citrix
```

Netscape 6.x and later, Mozilla 1.x (Win32):

```
%USERPROFILE%\Citrix
```

Netscape 6.x and later, Mozilla 1.x (UNIX and Linux):

```
$HOME/.citrix
```

Safari 1.x (Mac OS X):

```
<User's home directory>/.citrix
```

Copying Client Drive Mapping Configuration

If you are deploying the client to multiple users, you typically use the Web Interface.

Alternatively, you can configure common settings through the HTML page or the ICA file. However, client drive mapping settings can be configured only through the **ICA Settings** dialog box and are stored only in the appsrv.ini file. They cannot be configured through the ICA file or the HTML page. You can deploy client drive mapping settings by configuring client drive mapping on one device and copying the appsrv.ini file to the correct location on the users' devices.

Changing the List of Printer Drivers

When you configure printers using the **ICA Settings** dialog box, you set the driver by choosing one from a list.

You can change the list of drivers to match the printer drivers that are available on the server by editing the ICAPrinterDrivers.txt file. This plain text file is included in the client package and is located in the same directory as the client archives. The contents of the default file are listed in "ICAPrinterDrivers.txt File" on page 84.

Use a text editor such as Notepad to open the file. Add or remove driver names by deleting or adding names to the file, one driver name per line. You can add the driver names in any order.

Improving the Performance of the Client for Java

This chapter describes how you can improve the performance of the client using:

- Session sharing
- Data compression
- Data caching
- Queuing mouse movements and keystrokes
- SpeedScreen latency reduction

Advice is also provided on how to improve performance over a low-bandwidth connection.

Session Sharing

If you are not using seamless windows, consider moving to this mode of working to gain the performance benefits of session sharing. These include a reduction in system overhead for both client and server and quicker application launching.

Data Compression

Data compression reduces the amount of data that needs to be transferred over the network. However, additional processor resources are required to compress and decompress the data. If your connection is bandwidth-limited, enabling data compression improves performance.

You can specify two levels of data compression, standard and maximum. Maximum data compression uses more processor power and memory and may reduce performance on slow devices. Data compression is enabled by default.

To enable maximum data compression

Specify the following parameters on the HTML page:

```
<param name="Compress" value="on">
<param name="MaximumCompression" value="on">
```

To disable data compression

Specify the following parameter on the HTML page:

```
<param name="Compress" value="off">
```

Bitmap Caching

Bitmap caching stores commonly used images on a local disk. If the connection is bandwidth-limited, using bitmap caching increases performance. If the client is on a high-speed LAN, you do not need bitmap caching.

You can configure bitmap caching using parameters on the HTML page or the user can do it through the **ICA Settings** dialog box.

To enable bitmap caching for a connection

Specify the following parameter on the HTML page:

```
<param name="PersistentCacheEnabled" value="on">
```

The bitmap cache directory is stored in the directory specified by the standard Java system property user.home. The location of the user.home directory depends on the Java environment or Web browser you are using. If necessary, use the **ICA Settings** dialog box to change the bitmap cache directory.

Queuing Mouse Movements

When queuing is enabled, the client sends mouse movement updates less frequently to the server. This prevents the connection from becoming overburdened with excessive mouse events, which would degrade performance. Disabling queuing makes the ICA session more responsive to mouse movements. By default, mouse movement updates are sent every 150 milliseconds.

To change the update period for mouse movement queuing

Specify the following parameter on the HTML page:

```
<param name="MouseTimer" value="period">
```

where *period* is the update period you are using, in milliseconds. To disable mouse movement queuing, set the period to zero.

Using SpeedScreen Latency Reduction

SpeedScreen latency reduction improves performance over high latency connections by providing instant feedback to the user in response to typed data or mouse clicks.

Note SpeedScreen latency reduction works only when it is enabled on the server to which you are connecting. See the MetaFrame Presentation Server documentation for more details.

SpeedScreen latency reduction is not available on the Japanese client.

To enable mouse click feedback

Add the following parameter to the HTML page:

```
<param name="ZLMouseMode" value="0|1|2">
```

where **0** is off, **1** is on, and **2** is automatic. When mouse click feedback is set to automatic, the client enables it when required.

To enable local text echo

Add the following parameter to the HTML page:

```
<param name="ZLKeyboardMode" value="0|1|2">
```

where 0 is off, 1 is on, and 2 is automatic. When local text echo is set to automatic, the client enables it when required.

Improving Performance over a Low-Bandwidth Connection

If you are using ICA over a low-bandwidth connection, such as a modem or cellular telephone, you can make a number of changes to your client configuration and the way you use the client that will improve performance:

- **Reduce the size of the applet.** If you do not require the entire functionality of the client, specify only the archives necessary to provide the functions you require. Chapter 3 of this guide explains how to specify component archives. Reducing the size of the applet can greatly reduce the download time.

- **Change the client configuration.** On devices with limited processing power, or where limited bandwidth is available, there is a trade-off between performance and functionality. The client provides both user and administrator with the ability to choose an acceptable mixture of rich functionality and interactive performance. Making one or more of the following changes can reduce the bandwidth that the connection requires and improve performance:
  - **Enable maximum data compression.** Compression reduces the size of the data that is transferred over the ICA connection (see "Data Compression" on page 61).
  - **Enable the bitmap cache.** Bitmap caching stores commonly used images locally on the client so that they do not have to be transferred over the ICA connection every time they are needed (see "Bitmap Caching" on page 62).
  - **Queue mouse movements.** When queuing is enabled, the client sends mouse updates to the server less frequently. Increasing the queuing period may improve performance on a low-bandwidth connection (see "Queuing Mouse Movements" on page 62).
  - **Enable SpeedScreen latency reduction.** SpeedScreen latency reduction improves performance over high latency connections by providing instant feedback to the user in response to typed data or mouse clicks (see "Using SpeedScreen Latency Reduction" on page 63).
  - **Reduce the window size.** Change the window size to the minimum size users can comfortably use (see "Editing desktop.html" on page 18).
  - **Reduce the number of colors.** Reduce the number of colors to 256 (see "Window Properties" on page 33).
  - **Disable client audio mapping.** If your users do not need sound, disable client audio mapping (see "Client Audio Mapping" on page 47) or remove the JICA-audio archive.
  - **Disable clipboard mapping**. If users do not need to copy and paste text, do not include the JICA-clipboard archive.
  - **Disable client drive mapping**. If users do not need to map drives, do not include the JICA-cdm archive.
  - **Disable printing**. If users do not need to map printers, do not include the JICA-printer archive.
- **Change the way you use the client.** ICA technology is highly optimized and typically does not have high CPU and bandwidth requirements. However, users on a low-bandwidth connection should consider the following to preserve performance:

- **Avoid accessing large files using client drive mapping.** When you access a large file with client drive mapping, the file is transferred over the ICA connection. On slow connections, this may take a long time.
- **Avoid printing large documents on local client printers.** When you print a document on a local client printer, the print file is transferred over the ICA connection. On slow connections, this may take a long time.
- **Avoid playing multimedia content.** Playing multimedia content uses a lot of bandwidth and can cause reduced performance.

- **Use the latest client and server software.** Citrix is continually enhancing and improving ICA performance with each release, and many performance features require the latest client and server software in order to function. This is particularly relevant for Version 8.x of the client: in order to gain the benefits of high throughput networking the server needs to be running MetaFrame Presentation Server 3.0 for Windows, or MetaFrame Presentation Server 1.2 for UNIX.

Limitations of the Client for Java

This chapter describes some known limitations of the client that you or your users may encounter and suggests workarounds where applicable.

The sections are in alphabetical order of operating system, followed by a Miscellaneous section and a section that deals with Japanese-specific issues. The final section tells you how to report any further issues you may find with the client.

Linux and Solaris

Loss of Keyboard Focus

On Linux and Solaris platforms, when using a J2SE 1.4 Runtime Environment, the applet in embedded or non-seamless mode can lose keyboard focus after opening a client dialog box. The keyboard focus is successfully passed to the dialog box when the dialog box is opened, but it is lost when the dialog box is closed.

For Solaris, the workaround is to give keyboard focus to another application's window, then give it back to the browser window containing the embedded client applet. For Linux, click in the applet to restore keyboard focus.

Clipboard Support on X11

- Clipboard support on the Client for Java works only with applications that use the X11 CLIPBOARD selection. Specifically, Motif and Gnome applications are fine, but KDE applications do not work. Xterm can be configured with X11 resources to use the CLIPBOARD selection, for example:

```
XTerm*VT100.Translations: #override \n\
~Ctrl ~Meta <Btn2Up>:\
insert-selection(PRIMARY,CLIPBOARD,CUT_BUFFER0)\n\
<BtnUp>: select-end(PRIMARY,CLIPBOARD,CUT_BUFFER0)\n
```

GNU Emacs can also be configured to use CLIPBOARD with the following Emacs lisp:

```
(setq x-select-enable-clipboard t)
```

- The xclipboard utility (present on most X11 systems) may be of use in transferring data between the PRIMARY and CLIPBOARD selections.

Mac OS X

- To right-click when connected to a MetaFrame Presentation Server computer with a Macintosh, hold down the command key and click the mouse button.
- Characters generated using the Option keys on Macintosh may not be supported by the current Windows font in your ICA session. If the character produced is not the expected character, choose a Windows font in the ICA session that supports the character. After producing the desired character, you can return to the usual font.
- The client cannot load when the Java archive files are hosted on a Web server that requires cookies or authentication before serving the files. Ensure that the Web server does not require authentication to serve these files.

  This issue also affects the HTML help and .ica files.

  In particular, asp or jsp applications that manage session state using cookies are affected.
- When using client drive mapping, non-ASCII characters in client-side file names are displayed on the server as question marks (?). To avoid this problem, restrict file names to ASCII characters.

Client Printer Mapping

When using the client with Safari, the list of printer drivers in the **ICA Settings** dialog box does not appear.

To solve this problem, you need to add the ICAPrinterDrivers.txt file to an archive and add this archive to the list of client archives specified on the HTML page. The ICAPrinterDrivers.txt file is a plain text file, included in the client package and located in the same directory as the client archives.

To add the ICAPrinterDrivers.txt file to an archive

1. Before you can create the archive, you must obtain an appropriate development kit. To create .jar files, download a copy of the Sun JDK from http://java.sun.com/.

2. To make a .jar file containing ICAPrinterDrivers.txt, install the Sun JDK tools (and put them in your path), then type the following at the command line:

   ```
   jar -cf PrinterDrivers.jar ICAPrinterDrivers.txt
   ```

   This command generates an archive called PrinterDrivers.jar.

Specify the archive on the HTML page. Note that multiple archives in an applet tag must be separated by commas.

Windows

You may encounter display problems when using J2SE 1.3.x or later, depending on your computer's graphics card. The solution is to set the Java system property (either -Dsun.java2d.noddraw=true or -Dsun.java2d.ddoffscreen=false). On Windows systems, you can configure this in the Java Plug-in control panel applet.

Internet Explorer

- The browser does not correctly route some ALT key events. For example, pressing ALT+F when an application running in the ICA session has focus sends the event to both the application and to the browser.
- When Web browser scroll bars are in use, the session is sometimes repainted incorrectly.
- To display dead key characters, you must press the chosen dead key and then press the space bar twice.
- When using Microsoft's Virtual Machine for Java, status messages are displayed either in the Web browser's status bar or (if present) in the client status bar. However, status messages displayed in the Web browser's status bar are not permanently visible; instead, they flash on and off as the mouse pointer enters and exits the applet. This means that indications of SSL/TLS encryption are not always visible, and on some devices these messages may be impossible to see. To avoid this, either ensure that the client status bar is used, or use the Sun Java Plug-in.

Internet Explorer Security Settings on Windows Server 2003

For Windows Server 2003 the default security settings when you log on with administrator privileges are:

- Internet = High
- Local Intranet = Medium-Low

- Trusted Sites = Medium

This means that if you log on as an administrator and try to use the client over the Internet, by default it will not work because you cannot enable the usage of Java environments for the High security level through Internet Explorer's Internet Options.

There are two ways you can avoid this problem:

- Follow the instructions given in Microsoft Knowledge Base article 182569.
- Accept the Internet site that provides the Java applet as a trusted site. The default security level for a trusted site is Medium, therefore the Java environment is enabled and you can use the client.

If you log on to Windows Server 2003 as an ordinary user as opposed to an administrator, the default security level for the Internet is Medium; therefore the problem does not occur.

Miscellaneous

Client Drive Mapping

A remote application generates an error message because it cannot lock a file.

Due to Java limitations, Java programs cannot lock files. This means that files cannot be locked on mapped local drives.

To avoid generating error messages, disable the Safelock option on the HTML page:

```
<param name="Safelock" value="off">
```

When Safelock is disabled, if a remote application attempts to lock a file on a mapped drive, the client will report success. However, this does not mean that the file is locked; you must take precautions to ensure that another application does not write to or delete the same file, because this could result in data corruption.

Keyboard

- When connecting to a server, the client cannot detect the state of the Scroll Lock, Num Lock, or Caps Lock keys when the client receives focus.
- In rare circumstances, typing at the keyboard can produce no effect. If this happens, explicitly setting the Keyboard Layout should resolve the problem. See "Keyboard and Mouse" on page 36.

Mouse Pointer Support

- The Microsoft JVM supports only 14 different mouse pointers, so in these environments the client attempts to map remote pointer shapes to one of these 14.
- Arbitrary mouse pointers are not supported on Mac OS X.
- In shadowed sessions, the mouse pointer does not move when the shadowing user moves his or her mouse pointer.

Mouse Wheel Support

Mouse wheels are supported only in J2SE 1.4.x environments.

NTLM Authentication

When using J2SE 1.4.2, if the codebase of the applet points to a Web site that is protected by NTLM authentication, the client takes many hours to start. This issue has been reported to Sun. The only workaround is to make the client available on a site protected by both Digest and NTLM; Digest is then used instead of NTLM.

Seamless Windows

- On some platforms, the “outline” drag-box that is displayed when you move or resize seamless windows can cause unattractive repaint effects. If you experience this, you can either specify a solid drag-box type or eliminate the drag-box entirely by setting the parameter TWIDragBoxType to **solid** or **none** respectively.
- When using seamless mode to run applications that provide shadows around menus (for example Microsoft Office XP), extraneous windows are visible around the edges of the menus in certain Java environments. To avoid this, upgrade to a J2SE1.4.x environment or use the Microsoft JVM.
- Dual-headed displays are supported only when using a J2SE runtime environment. On earlier systems you can get seamless windows only on the primary display.

Security

SSL/TLS is not supported when using key lengths greater than 2048 bits.

Server Names with non-ASCII Characters

For servers to be contactable by client operating systems, it is generally necessary for the server's host name and domain name to be compliant with the Domain Naming System (DNS). That is, the names must contain only upper- and lower-case ASCII a to z, digits 0 to 9, and dash (-). This may not be necessary when client operating systems and the DNS server support multinational characters.

Published application names can comprise any characters because these are always resolved by the ICA browser. Note that any ICA browser addresses have the same DNS restrictions as the server names described above.

On some systems, you may not be able to connect to a server if its name does not conform to this requirement (that is, a server with a Japanese name or a name containing other non-ASCII characters).

To avoid this, either:

- Conform to the DNS requirement and use only ASCII characters in server names.

—Or—

- Specify ICA browser addresses as IP addresses and also do not enable DNS resolution for the server. If you do both of these things, the servers can have names that do not comply with the DNS requirement. However, note that SSL will not work in this scenario because DNS names are required for certificate validation. In addition, the Web Interface will not work because DNS is used to resolve the hostname part of the Web Interface URL.

Universal Print Driver and PCL4

- Horizontal lines in text and images sometimes appear jagged in landscape mode
- Print output near the page edge is sometimes clipped when you are using small page margins

To avoid these problems, use the appropriate native printer driver instead of the UPD. Configure the driver using the **Settings** dialog and ensure it is available on the server.

Japanese-Specific Issues

Using the Client with the Microsoft JVM

Microsoft Internet Explorer using the Microsoft JVM does not support Extended Unicode (EUC-JP) encoding. If .ica files are encoded as EUC-JP (for example, when generated by the Web Interface running on UNIX Web servers), text such as published application names is not interpreted correctly and the connection fails.

Using a Local IME to Input Japanese Characters

If you use the Microsoft IME Standard 2002 on your local device to input Japanese characters during a session, the characters you enter may not be displayed correctly in the temporary input window if using the Microsoft JVM. This problem can be avoided by changing the browser's JVM to use a J2SE environment.

Entering Japanese Characters in Shadowing Sessions

If you are shadowing another session, make sure you use the server IME. By default, the server side IME is used, but if you have been using the client IME, you need to change your keyboard layout setting. To change the setting, select the **General** tab on the **ICA Settings** dialog box. Choose **(Server Default)** or **Japanese (server IME only)** from the Keyboard Layout options. Alternatively you can choose **Japanese (client and server IME)** and use the server IME during shadowing.

Entering the Long Vowel Sound Symbol (—) in Kana Input Mode

When using the server IME with a J2SE environment to enter Japanese characters in Kana input mode, press the Shift key with the long vowel symbol key (—) to enter the long vowel symbol. This is not necessary when using the client IME.

Entering Japanese Characters in Applications Using the Client IME

If you use the client IME, displayed characters may be corrupted when you are using certain applications. If this happens, use the server IME instead.

The IME toolbar Is Inaccessible After Minimizing

When you are using the server-side IME in a connection to Windows 2000 Server, if the IME toolbar is minimized it becomes inaccessible. This occurs with all seamless sessions and non-seamless sessions to published applications, but is not a problem for non-seamless desktop sessions. In the latter case, you can retrieve the IME toolbar from the Windows System Tray. In the other cases, the System Tray is not available.

The workaround is to connect to a non-seamless desktop session and restore the IME toolbar from the System Tray. This is not an issue when connecting to Windows 2003 Server, as MetaFrame Presentation Server prevents the toolbar from being minimized.

Entering Characters After Reconnecting Using the Client IME

If you reconnect to a session from a different browser or with a different window size setting from the original session, you may see a dialog box warning you that the video mode for the existing session cannot change. You cannot input characters using the client IME until you close the dialog box by clicking **OK**.

Sometimes this dialog box is displayed behind the application window. If you cannot enter characters using the client IME after reconnecting, even if you do not see a dialog box, try minimizing the application window in the browser window to see whether the warning dialog box appears. Close the dialog box by clicking **OK**, then restore the application window.

Hotkey Support for Japanese Keyboards

On certain hardware platforms, some Java environments may have problems with the special Japanese modifier keys, such as Hankaku, Zenkaku, and Hiragana, which are used to control the IME on the server.

If you experience this problem, first ensure that you have the latest version of a J2SE environment for your platform. If the problem persists, you can interact with the IME by clicking the IME buttons using your mouse.

Alternatively you can define hotkeys to simulate the effect of the IME keys. For example, you can define F1 as the Katakana key. This technique for mapping hotkeys is similar to that used for defining special key combinations for the client, such as using CTRL+F1 to send the key combination CTRL+ALT+DEL.

You define Japanese hotkeys in the same way as English hotkeys: either in the <applet> tag or by using the **ICA Settings** dialog box, as described in "Hotkeys" on page 39.

The full set of possible Japanese hotkeys is:

HotkeyMuHenkanChar
HotkeyMuHenkanShift

HotkeyPrevKouhoChar
HotkeyPrevKouhoShift

HotkeyKatakanaChar
HotkeyKatakanaShift

HotkeyHankakuChar
HotkeyHankakuShift

HotkeyKanjiBangoChar
HotkeyKanjiBangoShift

HotkeyNextKouhoChar
HotkeyNextKouhoShift

HotkeyAllKouhoChar
HotkeyAllKouhoShift

HotkeyHiraganaChar
HotkeyHiraganaShift

HotkeyRomajiChar
HotkeyRomajiShift

HotkeyEisuChar
HotkeyEisuShift

Reporting Issues

If you have any issues with the client that are not described in this chapter, please take the following steps:

1. Make sure you are running the latest version of the client.
2. Read the information in the Readme file.
3. Check the relevant section of this guide.
4. If the problem still persists, report the problem to your Citrix support representative and provide all the following information:
   - The client operating system platform.
   - The Web browser and version number and, if you are using a Java environment Plug-in, the Plug-in version number.

- The complete applet tag from the Web browser's **View Source** (or equivalent) menu. This shows the resulting HTML after any client- or server-side scripting.
- The contents of any ICA file being used. To get these, add the ICAFileDebug parameter with value = True to the <applet> tag.
- Any error messages displayed in the Java console.
- If applicable, any Java stack trace that is displayed after clicking **Details** in an error dialog box.
- The MetaFrame Presentation Server platform and MetaFrame Presentation Server version.
- The Web Interface version number, if applicable.
- Any relevant security information; for example, whether you are using ICA encryption or SSL.
- An explanation of how to reproduce the problem.

Parameters and File Descriptions for the Client for Java

Parameters

The following is a complete list of the parameters that can be specified to provide additional features and customization. The Address parameter is the only required parameter; all other parameters in these lists are optional.

Most of the parameters are specified on the HTML page. However, the client drive mapping parameters listed in this section are specified in the client-side appsrv.ini file.

General Configuration

Parameter	Description
Address	The address of the server or the name of the published application. If a published application is entered as an address value, the InitialProgram parameter must also be specified.
Cabinets	Specifies Authenticode signed .cab file archives, used with the Microsoft VM on Internet Explorer for Windows.
Clientname	The client name. Use in the form client.wfclient.Clientname. Use with the parameter client.wfclient.UseHostname=off.
CREnabled	Specifies whether or not content redirection is enabled. Values are **yes** or **no**. The default value is **yes**.
Domain	The name of the domain for the user name.

Parameter	Description
HTTPBrowserAddress HTTPBrowserAddress2 to 15	The address of a browser server. This parameter is used when TCP/IP+HTTP or SSL/TLS+HTTPS browsing has been specified. This parameter is also used to designate groups of primary and backup servers. Note that when the XML Service on the server is not configured to use the default port 80, you must append :<*port number*> to this parameter, substituting <*port number*> with the port number your server's XML Service is configured to use. For HTTPBrowserAddress the default is **ica**. There is no default for HTTPBrowserAddress 2 to 15.
Icafile	An ICA file for the client to use. The value entered must be a valid URL. There is no default.
ICAPortNumber	The default ICA port number is 1494. A different port number can be specified using this parameter or by appending the port number to the address value; for example, **CitrixServer:1495**.
InitialProgram	The name of the initial program to run after connecting to the server. If you are connecting to a published application, add a # symbol before the program name.
Language	Causes the client's user interface components to appear in a language other than the language of the client device.
Param	Passes a parameter such as a file name to a published application.
Password	The password of the user. The Password parameter cannot be used to specify an encrypted password. To specify an encrypted password, use an ICA file or .ini file that contains an encrypted password.
SpeedScreenBA	Specifies whether or not SpeedScreen browser acceleration is enabled. Values are **yes** or **no**. The default value is **yes**.
TWIDisableSessionSharing	Specifies whether or not session sharing is disabled. Used in conjunction with Connection Center. The default value is **no**.
TWIMode	Enables seamless windows. If you are using seamless windows, you must set this parameter to **on**; otherwise, set it to **off.**

Parameter	Description
UseHostname	Specifies using the hostname as the client name. Use in the form client.wfclient.UseHostname.
Username	The user name to use during logon.
WorkDirectory	The path of the working directory where the initial program is run after the user connects to the server.

User Interface Parameters

Parameter	Description
Border	Turns the border around the ICA session in the browser window on or off. Values for this parameter are **on** or **off**. The default value is **off**.
BorderWidth	The border width in pixels. The default value is 6.
DesiredColor	The color depth of the ICA session windows. The values are 2 (256 colors), 4 (thousands of colors), and 8 (millions of colors). The default value is 256 colors. 16 color mode is not supported but the client can connect to applications published in 16 color mode, in which case 256 colors are used.
DesiredHRes	The height of the ICA session window, if you want the session size to be different from the applet size. If this parameter is not specified, the Height parameter is used and the session height is therefore the same as the applet height. There is no default.
DesiredVRes	The width of the ICA session window, if you want the session size to be different from the applet size. If this parameter is not specified, the Width parameter is used and the session width is therefore the same as the applet width. There is no default.
End	Controls the client's behavior when you terminate a session. The values are **manual** (default), **auto**, **terminate**, and ***URL***.
Height	The height of the ICA session window. This parameter is specified as an attribute in the <applet> tag.
Hotkey*n*Shift Hotkey*n*Char	Sets hotkeys that can be used to control various client functions. Use in the form user.wfclient.Hotkey*n*Shift or user.wfclient.Hotkey*n*Char where *n* is the number of the hotkey. *n* can be 2, 3, 4, 5, 6, 7, 8, 9, 10, or 11. For further information on how to set hotkeys, see "Hotkeys" on page 39.
KeyboardLayout and KeyboardType	The type of keyboard. Use in the form user.wfclient.KeyboardType and user.wfclient.KeyboardLayout.

ShowSettingsButton	Specifies whether or not to show the **Settings** button. You must include the complete JICAEng archive or the JICA-config archive to display the **Settings** button. Values for this parameter are **yes** or **no**. The default value is **yes**.
ShowStatusBar	Specifies whether or not to show the status bar. You must include the complete JICAEng archive or the JICA-config archive to display the status bar. Values for this parameter are **yes** or **no**. The default value is **yes**.
Start	Controls the client's behavior when you start a session. The values are **manual** (default) and **auto**.
Width	The width of the ICA session window. This parameter is specified as an attribute in the <applet> tag.

Client Audio Mapping Parameters

Parameter	Description
ClientAudio	Enables client audio. The values are **on** and **off**. The default is **off**.

Client Printer Mapping Parameters

Parameter	Description
Comment	Sets a default printer. Must be specified as user.*printername*.Comment, where *printername* is the name allocated with the parameter LocalClientPrinters.
Driver	The printer driver. Use in the form user.*printername*.Driver, where *printername* is the name allocated with the parameter LocalClientPrinters.
LocalClientPrinters	Used for passing information about client printers to the server. Must be specified as <param name="user.localclientprinters" value="*printername*">. To specify more than one printer, separate the printer names with commas.
Port	The printer port. Use in the form user.*printername*.Port, where *printername* is the name allocated with the parameter LocalClientPrinters.

Client Drive Mapping Parameters

Although the user specifies most client drive mapping parameters using the **ICA Settings** dialog box, there are three parameters that you can specify only in the client-side appsrv.ini file:

Parameter	Description
DriveRemovable<x>	{yes | no} Specifies whether or not drive x is removable, that is, whether it is a floppy drive or a CD-ROM drive. The default is **no**.
DriveMappingHomeDrive	{single letter} Specifies the drive letter to use for the home drive. The default is **H**.
DriveMappingAutoDetectHome	{yes | no} Specifies whether or not to auto-detect the user's home drive. The default is **yes**.

Performance Tuning Parameters

Parameter	Description
Compress	Sets data compression. The values are **on** to enable data compression and **off** to disable it. By default, compression is enabled.
MaximumCompression	Sets high level data compression. The values are **on** to enable greater data compression and **off** to select normal data compression. Requires the Compress parameter to be enabled.
MouseTimer	The time (in milliseconds) between the mouse movement updates that are sent to the server. Set to 0 to disable queuing. The default is 150.
PersistentCacheEnabled	Enables or disables bitmap caching. Must be specified as "user.wfclient.PersistentCacheEnabled". Values are **on** or **off**. The default value is **off**.
PersistentCacheMinBitmap	The size of the smallest bitmap to cache, in KB. The default is 8.
PersistentCacheSize	The size of the bitmap cache in MB. The default is 10.
ZLKeyboardMode	SpeedScreen latency reduction mode. The values are 0 (off), 1 (on), or 2 (auto). The default is 0.
ZLMouseMode	SpeedScreen latency reduction mode. The values are 0 (off), 1 (on), or 2 (auto). The default is 2.

Security Integration Parameters

Parameter	Description
EncryptionLevel	The level of ICA encryption to use for an ICA connection. Values are as follows: 0 = No encryption, 1 = basic encryption, 2 = RC5 128-bit encryption during authentication only, 5 = RC5 128-bit The default is 1.
ProxyAutoConfigURL[1]	The location of a Proxy Auto Configuration (PAC) file for automatic proxy configuration.
ProxyAutoDetectFallback	Specifies whether or not to make a direct connection to the server if Internet Explorer is set to automatically detect proxy settings and the file that contains the settings (http://wpad/wpad.dat) is not accessible. Values are **yes** or **no**. The default value is **yes**.

ProxyExcludeList[1]	A semicolon- or comma-separated list of addresses of servers that the client must connect to directly—not through the proxy server.
ProxyHost[1]	The location and port of the proxy server.
ProxyType	The type of proxy server. Values are: none = no proxy auto = use the Web browser's settings socks = use SOCKS and automatically detect the version socksv4 = use SOCKS Version 4 socksv5 = use SOCKS Version 5 secure = use Secure proxy script = use a PAC file (specified by ProxyAutoConfigURL)
ProxyUsername[1] ProxyPassword[1]	Proxy server logon credentials.
SecureChannelProtocol	Specifies the SSL/TLS protocol version. Values are **SSL**, **TLS**, or **detect**. If you specify **detect**, the client connects using the protocol requested by the server. Default = **detect**.
SSLCACert0... SSLCACert*	The name of an additional certificate.
SSLCiphers	An alternative cipher suite. Values are **Gov**, **Com**, or **All**. Default = **All**.
SSLEnable	Enables SSL and TLS encryption protocols. Used with the BrowserProtocol parameter. Values are **on** or **off**. Default = **off**.
SSLNoCACerts	The number of additional certificates. Default = 0.
SSLProxyHost	A Secure Gateway (relay mode) address.
UseAlternateAddress	Specifies whether or not to use an alternate server address across a firewall. Used in the form user.wfclient.UseAlternateAddress. Values are 0 (actual address is used) or 1 (alternate address is used). Default = 0.

[1] These parameters are meaningful only if you specify ProxyType.

ICAPrinterDrivers.txt File

The contents of the default ICAPrinterDrivers.txt file are as follows:

```
HP LaserJet
HP DeskJet
HP OfficeJet
HP LaserJet Series II
HP LaserJet III
HP LaserJet 4
HP LaserJet 5
HP LaserJet 4000 Series PCL
HP LaserJet 4000 Series PS
HP LaserJet 4050 Series PCL
HP LaserJet 4050 Series PS
HP LaserJet 5000 Series PCL
HP LaserJet 5000 Series PS
HP LaserJet 8000 Series PCL
HP LaserJet 8000 Series PS
HP LaserJet 8100 Series PCL
HP LaserJet 8100 Series PS
Canon Bubble-Jet BJC-70
Canon Bubble-Jet BJ-200ex
Canon Bubble-Jet BJC-600
Canon LBP-4
Canon LBP-8II
Canon LBP-8III
Epson Stylus Pro ESC/P 2
Epson Stylus COLOR ESC/P 2
Epson Stylus Photo ESC/P 2
Epson EPL-3000
Epson EPL-4000
Epson EPL-5000
Epson EPL-6000
Epson EPL-7000
Epson EPL-8000
Epson EPL-9000
Lexmark Optra
```

Supported Keyboard Layouts

The keyboard layouts supported are:

```
"(Server Default)"
"Albanian"
"Belarusian"
"Belgian Dutch"
"Belgian French"
"Brazilian (ABNT)"
"British"
"Bulgarian (Latin)"
"Bulgarian"
"Canadian English (Multilingual)"
"Canadian French (Multilingual)"
"Canadian French"
"Croatian"
"Czech (QWERTY)"
"Czech"
"Danish"
"Dutch"
"Estonian"
"Finnish"
"French"
"German (IBM)"
"German"
"Greek (220) Latin"
"Greek (220)"
"Greek (319) Latin"
"Greek (319)"
"Greek Latin"
"Greek"
"Hungarian 101-Key"
"Hungarian"
"Icelandic"
"Irish"
"Italian (142)"
"Italian"
"Japanese"
"Japanese (client and server IME)"
"Japanese (client IME only)"
"Japanese (server IME only)"
"Korean"
"Latin American"
"Latvian (QWERTY)"
"Latvian"
"Lithuanian"
"Norwegian"
"Polish (214)"
"Polish (Programmers)"
"Portuguese"
"Romanian"
"Russian (Typewriter)"
"Russian"
```

```
"Serbian (Cyrillic)"
"Serbian (Latin)"
"Slovak (QWERTY)"
"Slovak"
"Slovenian"
"Spanish Variation"
"Spanish”
"Swedish"
"Swiss French"
"Swiss German"
"Taiwan"
"Turkish (F)"
"Turkish (Q)"
"Ukrainian"
"United Kingdom"
"US"
"US-Dvorak for Right hand"
"US-Dvorak for left hand"
"US-Dvorak"
"US-International"
```

Index

J

K

L

M

P

Q

R

S

T

U

W