

Oracle Forms 12c

Client Deployment Configuration Options

ORACLE WHITE PAPER | MAY 2016





Table of Contents

Introduction	2
Oracle Forms 12c Client Deployment Configuration Options	3
Java Applet Embedded in HTML	3
JNLP Embedded in HTML	4
Java Web Start	4
Forms Standalone Launcher	5
Conclusion	8



Introduction

In the early 1990's, the Internet (World Wide Web) began to evolve from a system designed and used by government organizations to one used globally by the public. This transition began a wave of related technologies being born. The Internet allowed people to have immediate access to information once thought to be out of reach. Web technology, specifically Web browsers, gave developers an opportunity to deliver their applications to end-users without the need to install software on the end-user's machine.

Until the 1990's, Oracle Forms was delivered as a two-tier technology. With its client and server concept, Oracle Forms required that the technology's runtime software be installed and configured on the end-user's machine. This meant that each end-user machine would require an administrator to install runtime software one machine at a time. Often this was a difficult task given that an organization could have thousands of end-users, who could be located anywhere in the world. As we moved toward and into the twenty first century, Oracle realized that Oracle Forms customers would begin expecting to have access to their applications in the same way as so many other applications, through a web browser. Oracle Forms 6i was the first version to fully support application deployment via a web browser. Because of the need for tight integration on the client tier, it was necessary to use Java Applet technology to host the applications. So, although applications were launched from a browser, it was actually the Java Runtime Environment (JRE) on the client that ran or hosted the applications. After more than fifteen years of web browsers supporting integration with the Java Plugin, many browser vendors are moving to a plugin-free model. As a result, Oracle Forms needs new ways of being deployed if it is to continue using Java technology on the client tier.

Beginning with Oracle Forms 12c, product users can now choose from several client configuration options. With several options available, administrators can choose which option best suits their needs. This document aims to describe each of the configuration options and how to use them. It will also explain the advantages and limitations of each.

Oracle Forms 12c Client Deployment Configuration Options

Beginning with Oracle Forms 12c, there are now four supported client deployment configuration options. The available options are as follows.

- » Java Applet embedded in HTML
- » JNLP embedded in HTML
- » Java Web Start (JWS)
- » Forms Standalone Launcher (aka Standalone or FSAL)

Although each option will have minimal to no impact on the appearance and behavior of any application, each will be contained slightly differently. Enabling and configuring any of these can be accomplished in the Forms Web Configuration page (i.e. formsweb.cfg) in Fusion Middleware Control.

Any examples provided in this paper assume that you have properly configured Secure Socket Layer (SSL) in the server environment, as this is the most secure way to run any web deployed application. All options can also be used with non-SSL, but this is not recommended. Refer to the Oracle HTTP Server and WebLogic Server Administration Guides for details on properly configuring SSL in your environment.

Java Applet Embedded in HTML

This option was the first offered when browsers began supporting the Java Plug-in. Its configuration is the default for any Oracle Forms installation, 12.2.1 and older. This option can be used to give the appearance that the Forms application (applet) is embedded in a web page. This is often desirable when the HTML content surrounding the Forms application contains related or integrated information. This can also be helpful when using the Forms JavaScript integration feature. Single sign-on and single sign-off are also fully supported in this configuration.

Disadvantages of using this option include the requirement of a Java Plug-in and the need for a certified browser that supports the Java Plug-in. The loss of usable space taken up by the browser's window, toolbar, and menu will also need to be tolerated.

Using the Applet embedded in HTML option is simple because this is the default configuration. Simply enter a URL in the browser that appears something like one of the following.

- » `https://example.com/forms/frmservlet`
- » `https://example.com/forms/frmservlet?config=default`

These parameters/values are required for this configuration and are found in the Forms Web Configuration page of Fusion Middleware Control.

Non-WebUtil Enabled Forms	WebUtil Enabled Forms
<ul style="list-style-type: none">» <code>baseHTML=base.htm</code>» <code>baseHTMLjpi=basejpi.htm</code>	<ul style="list-style-type: none">» <code>baseHTML=webutilbase.htm</code>» <code>baseHTMLjpi=webutiljpi.htm</code>

JNLP Embedded in HTML

Embedded JNLP is very similar to embedded Applet, however the application is treated more like a Java Web Start application although embedded within a web page. Like an embedded Applet, embedded JNLP fully supports JavaScript integration, single sign-on, single sign-off, and the ability to visually embed the form in a web page. Embedded JNLP has the added advantage of base-64 encoding the JNLP content. This content includes most of the parameter/value pairs configured for the application. Because the base-64 encoded text is not human readable, curious end-users will be discouraged from attempting to alter any of the parameters. It should be understood that the base-64 encoding is not a security mechanism. The base-64 encoding used in this configuration is required by Java and helps to improve the performance of delivery from server to client. This configuration requires the Java Plug-in and a certified browser that supports the Java Plug-in.

To use this configuration you can either use the provided example configuration named “jnlp” or create your own.

- » <https://example.com/forms/frmservlet?config=jnlp>

These parameters/values are required for this configuration and are found in the Forms Web Configuration page of Fusion Middleware Control.

Non-WebUtil Enabled Forms	WebUtil Enabled Forms
<ul style="list-style-type: none">» <code>basejnlp=base.jnlp</code>» <code>baseHTMLjpi=basejpi_jnlp.htm</code>	<ul style="list-style-type: none">» <code>basejnlp=webutil.jnlp</code>» <code>baseHTMLjpi=basejpi_jnlp.htm</code>

When using Java Web Start or Embedded JNLP, if the application uses custom jar files (e.g. `jacob.jar`, `icons.jar`, `example.jar`, etc), these should be added to `extensions.jnlp`. The file is located in `Oracle_Home\forms\java`. Open this file in a text editor and make the appropriate entries. An example is included in the file. Each entry should be added on its own line.

Java Web Start

Java Web Start is considered a semi-browserless configuration. The use of Java Web Start will give an Oracle Forms application the appearance of being a natively installed application rather than a web app because when running, the application is not contained by the boundaries of the browser. This is often desirable with Point of Sale applications where the only application used on the device is the POS application or in cases where the application is designed to use the full screen. Different from the functionality provided by the use of `separateFrame=true` (available in the previous two configurations only), the use of Web Start allow you to close the browser window used to call the application once it has started.

Oracle Forms’ use of Java Web Start allows applications to be called from a browser using a hyperlink or by directly entering a URL. Alternatively, the application can be run from a JNLP file stored on the end-user machine. This method eliminates the need for a browser, except when the application is single sign-on protected. Java Web Start can also be used to call an Oracle Forms application from the command line. Although there are several variations of how Java Web Start can be used, if the application requires the use of single sign-on, it must be called from a browser. Attempts to call an SSO protected application from a static JNLP file or the command line will fail.

Because this is a mostly browser-less configuration, features like single sign-off and JavaScript integration are not supported when using the Java Web Start configuration.

This configuration requires the Java Plug-in be installed if calling from a browser. If not calling from a browser, either the Java Plug-in or Java Development Kit (JDK) installation is required. A browser is optional and would only be required if single sign-on is used.

To use this configuration you can either use the provided example configuration named “webstart” or create your own. The application can be called from a browser or you can use the command line or custom script.

- » `jnlps://example.com/forms/frmservlet?config=webstart`
- » `https://example.com/forms/frmservlet?config=webstart`
- » `javaws "https://example.com/forms/frmservlet?config=webstart"`

The “jnlp” and “jnlps” protocols (used in above example) are supported with Java 8u92+ on Microsoft Windows.

The following are required parameters/values for this configuration and are found in the Forms Web Configuration page of Fusion Middleware Control.

Non-WebUtil Enabled Forms	WebUtil Enabled Forms
<ul style="list-style-type: none">» <code>basejnlp=base.jnlp</code>» <code>webstart=enabled</code>	<ul style="list-style-type: none">» <code>basejnlp=webutil.jnlp</code>» <code>webstart=enabled</code>


When using Java Web Start or Embedded JNLP, if the application uses custom jar files (e.g. `jacob.jar`, `icons.jar`, `example.jar`, etc), these should be added to `extensions.jnlp`. The file is located in `Oracle_Home\forms\java`. Open this file in a text editor and make the appropriate entries. An example is included in the file. Each entry should be added on its own line.

Forms Standalone Launcher

The Forms Standalone Launcher is a fully browser-less configuration. Further, it does not require that the Java Plug-in be installed. It does require Java on the client, but it can be any one of the following; JRE, JDK, or Server JRE.

With this configuration, applications are launched from the command line or custom script files by calling the platform specific “java” executable. This configuration fully isolates the application from the Java Plug-in or Java Web Start and the browser. Using the Forms Standalone Launcher can provide the appearance of a fully native application. Single sign-on, single sign-off, and JavaScript integration are not supported with this configuration.

This configuration requires that a small jar file (`frmsal.jar`) be stored on the end-user machine. The file can be transferred to the end-user machine using any desirable method (e.g. web download, email, ftp, etc). This file is the Forms Standalone Launcher. It is version specific, so it cannot be used with other Forms versions or patch levels.



The file is staged on the server in the `Oracle_Home\forms\java` directory. However, a helpful usage guide that includes a download link can be easily accessed when the server is running. Administrators can easily disable, remove, or edit this page if desired. Navigate to the following to display the page.

» `https://example.com/forms/html/fsal.htm`

To use this configuration you can either use the provided example configuration named “standaloneapp” or create your own. The application can be called from the command line or custom script. You cannot use a browser with this configuration. The command line entry would look something like the following.

» `java -jar frmsal.jar -url "https://example.com/forms/frmservlet?config=standaloneapp"`

» `java -jar frmsal.jar -url "https://example.com/forms/frmservlet?config=standaloneapp" -t 30000`

Refer to the Usage Guide mentioned above for more details on command line syntax.

The following are key parameters/values used by this configuration and are found in the Forms Web Configuration page of Fusion Middleware Control.

Non-WebUtil Enabled Forms	WebUtil Enabled Forms
» <code>baseSAAfile=basesaa.txt</code>	» <code>baseSAAfile=webutilsaa.txt</code>
» <code>fsalcheck=true</code> (<i>optional, but recommended</i>)	» <code>fsalcheck=true</code> (<i>optional, but recommended</i>)

In this release, the Forms Standalone Launcher does not have the ability to determine which Java version is being used. It will be up to the Administrator and/or user to ensure that an appropriate Java version is used. Refer to Figure 1 below for a simple Microsoft Windows script (batch) example.

```

1  @ REM Set desired Java version's home here.
2  @ SET JAVA_HOME="C:\Program Files (x86)\Java\jre1.8.0_92"
3  @ REM Assume file is in user's home directory
4  @ SET FSAL_HOME=%USERPROFILE%
5  @
6  @ :BEGIN
7  @ ECHO.
8  @ IF NOT EXIST %JAVA_HOME% GOTO NOJAVA
9  @ IF NOT EXIST %FSAL_HOME% GOTO NOFSAL
10 @   SET PATH=%JAVA_HOME%\bin;%PATH%
11 @   CLS
12 @ REM The following command line is broken into two lines using a caret.
13 @ REM Remove the caret if using a single line for this command.
14 @   start /min "Oracle Forms 12c Console - Do not close." java -jar %FSAL_HOME%\frmsal.jar ^
15 @     -url "https://example.com:9001/forms/frmservlet?config=standaloneapp"
16 @   GOTO :DONE
17 @
18 @:NOJAVA
19 @ ECHO The required Java version was not found.
20 @ SET /p JAVA_HOME=Enter alternative Java home directory:
21 @ ECHO.
22 @ IF NOT EXIST %JAVA_HOME% GOTO :EOF
23 @   GOTO :BEGIN
24 @
25 @:NOFSAL
26 @ ECHO Oracle Forms Standalone Launcher not found.
27 @ SET /p FSAL_HOME=Enter alternative FSAL home directory:
28 @ ECHO.
29 @ IF NOT EXIST %FSAL_HOME% GOTO :EOF
30 @   GOTO :BEGIN
31 @
32 @:DONE
33 @ REM Close shell - do not do this while troubleshooting.
34 @ EXIT

```

Figure 1 - Microsoft Windows script example.



Conclusion

Oracle Forms 12c offers a variety of possible client configurations options. Choosing which is best will depend on the application's needs and desired appearance at runtime. Because users can choose which option suits them best, no longer will they be constrained to a single possibility. If choosing the Java Web Start or Forms Standalone Launcher options, administrators no longer have to be concerned about compatibility between browser and Java versions. This should make application deployment and manageability easier.







Oracle Corporation, World Headquarters

500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries

Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

-  blogs.oracle.com/oracle
-  facebook.com/oracle
-  twitter.com/oracle
-  oracle.com

Integrated Cloud Applications & Platform Services

Copyright © 2016, Oracle and/or its affiliates. All rights reserved. This document is provided for information purposes only, and the contents hereof are subject to change without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document, and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group. 0116

Oracle Forms 12c Client Deployment Configuration Options
May 2016
Author: Michael Ferrante
Contributing Authors: Oracle Forms Development