Oracle® Fusion Middleware

Working with Oracle Forms

14.1.2.0.0
F46700-02
May 2025

Oracle Fusion Middleware Working with Oracle Forms, 14.1.2.0.0

F46700-02

Primary Author: Oracle Corporation

Contents

Preface

Audience

This manual is intended for software developers and application administrators who are interested in deploying Oracle Forms applications to the Web with Oracle Fusion Middleware.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customer access to and use of Oracle support services will be pursuant to the terms and conditions specified in their Oracle order for the applicable services.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Documents

You can refer the Oracle Fusion Middleware Library for additional information.

- For 14*c* Oracle Forms information, see Oracle Forms Documentation Library.
- Oracle Forms Developer Online Help, available from the Help menu in the Form Builder.
- For Oracle Forms white papers and other resources, see Oracle.com.
- For upgrade information, see Fusion Middleware Upgrade Documentation.
- For release-related information, see Fusion Middleware Release Notes.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Part I

Getting Started

This part introduces you to the Oracle Forms Services. It provides an overview of the development and deployment environment for Oracle Forms, and provides references where you can find more information on associated components in Oracle Fusion Middleware. It also provides pointers to features and improvements in Forms Services.

Specifically, this part contains the following chapters:

Oracle Forms

Oracle Forms is a component of Oracle Fusion Middleware used to develop and deploy Forms applications. The Forms applications provide a user interface to access the Oracle Database in an efficient and tightly-coupled way.

Oracle Forms includes:

- Oracle Forms Developer: Used to develop and compile Forms applications
- Oracle Forms Services: A server component used to deploy the applications

Oracle Forms Developer

Oracle Forms Developer is used to develop a form that can access an Oracle database and present the data.

Wizards and utilities are provided to speed up application development. The source form (*.fmb) is created and compiled into an "executable" (*.fmx). The Forms application is run (interpreted) by the Forms Runtime process.

Refer to the following documentation for Oracle Forms Developer:

- Oracle Form Builder Online Help, which is accessible from Oracle Form Builder, provides information on how to use Oracle Forms Developer to develop and compile Forms applications.
- Obsolete features of Oracle Forms Developer and instructions for upgrading your Forms applications, as described in Preparing to Upgrade.

Oracle Forms Services

Oracle Forms Services is a comprehensive application framework optimized to deploy Forms applications in a multitiered environment. It takes advantage of the ease and accessibility of the Web and elevates it from a static information-publishing mechanism to an environment capable of supporting complex applications.

The Forms applications that you design and develop in Oracle Forms Developer are deployed on Oracle Fusion Middleware. These applications run on the middle tier (see Oracle Forms Services Architecture). The user interface is presented on the client tier as a Java applet.

This guide describes the configuration files, and environment variables that you can use to customize deployment of Forms applications. It also provides information on performance, logging and monitoring your deployment. You can use Oracle Fusion Middleware Control to manage the configuration files, and environment variables, and monitor the deployment.

How Oracle Forms Services Launches a Forms Application

When a user first starts an Oracle Forms application, the Forms servlet reads one of the Forms base template files. Any variables (%*variablename*%) in the template file are replaced with the appropriate parameter values specified in the Forms Web Configuration (`formsweb.cfg`) file, and from query parameters in the URL request (if any).

You can easily modify the Web configuration in Oracle Fusion Middleware Control according to your requirements. Oracle Forms Services Architecture describes the processes that are involved in deploying and running a typical Forms application.

Oracle WebLogic Server

Oracle WebLogic Server, a component of Fusion Middleware Infrastructure, is an application server for building and deploying enterprise Java EE applications with support for new features for lowering cost of operations, improving performance and supporting the Oracle applications portfolio.

Regardless of whether you want to create a staging, production, or testing environment, you begin by creating a WebLogic domain. A WebLogic domain includes managed servers, of which one is configured as an Administration Server. The Administration Server maintains configuration data for a domain. For information about Oracle WebLogic Server, see Introduction in *Understanding Oracle WebLogic Server*.

During configuration, a managed server for Oracle Forms is created (WLS_FORMS), as described in About WebLogic Managed Servers and HTTP Server.

Oracle Fusion Middleware

Oracle Fusion Middleware includes Web servers, application servers, content management systems, and developer tools that provide complete support for development, deployment, and management of software applications.

Among the components are Oracle Forms Services, Oracle WebLogic Server, and Oracle Fusion Middleware Control, which together provide the technology to fully realize the benefits of three-tier computing.

You can manage and monitor Oracle Forms using Oracle Fusion Middleware Control.

For a complete overview, list of components, and conceptual information about Oracle Fusion Middleware, see:

- About Key Oracle Fusion Middleware Concepts in *Understanding Oracle Fusion Middleware*.
- Getting Started Managing Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

About Installing or Upgrading Oracle Forms

Before you install or upgrade Oracle Forms or applications, review these topics:

To prepare for installing Oracle Forms, see Preparing to Install.

To plan an upgrade of Oracle Forms, see Planning an Upgrade of Oracle Forms.

To upgrade applications developed in versions 10.1.2 or older, see Preparing to Upgrade. This topic includes the list of changed or obsolete features in Oracle Forms.

Oracle Forms Services Architecture

Oracle Forms Services is a middle-tier application framework for deploying complex, transactional Forms applications to a network such as an intranet or the Internet.

The Forms Services architecture is made up of three-tiers. Developers build Forms applications with Forms Developer and deploy them with Forms Services. Developers can also take applications that were previously deployed in client/server and move them to a three-tier architecture. Some minor changes in application code may be required when moving to a three-tier architecture.

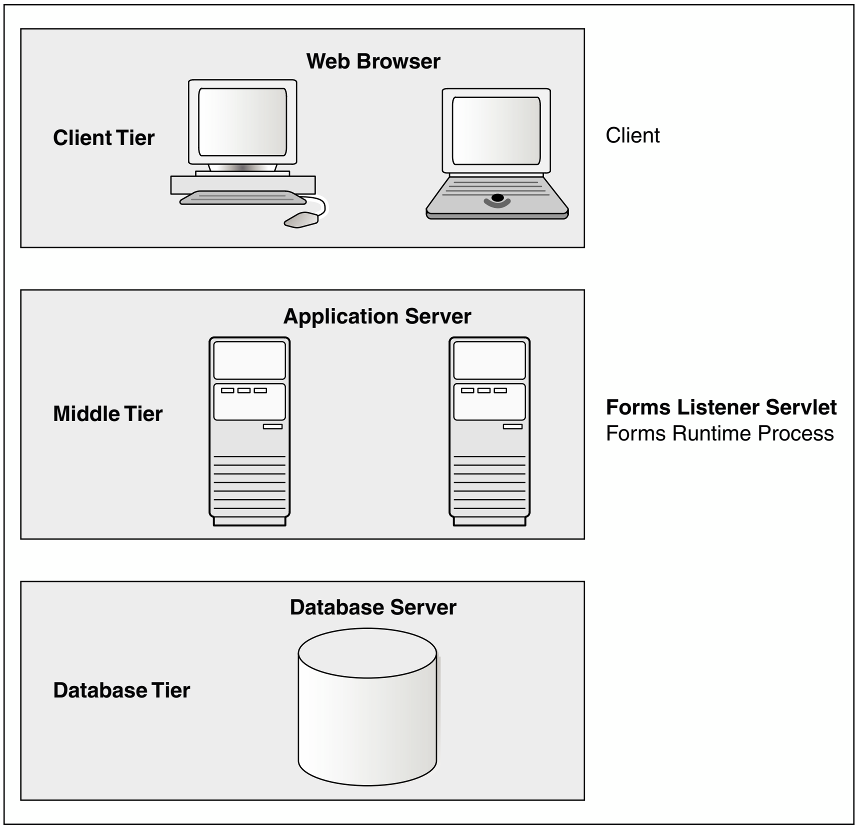
The three tiers of the architecture are as follows:

- **Client Tier:** The client tier (top of Figure 1) may contain one of the following two configurations:
  - The Forms Standalone Launcher (FSAL) and a Java Runtime Environment or Java Development Kit (JDK).
  - The Java Plug-In/Java Runtime (JPI/JRE), used to launch the application with Java Web Start (JWS).

    > **Note:**
    >
    > Some browsers, such as Microsoft Edge with IE mode, may still include the required components for this configuration but this configuration is no longer recommended for use.

- **Middle Tier**: The middle tier (middle of Figure 1) is the application server, where application logic, server software, and compiled Forms application modules are stored. The **Forms Listener servlet** and the **Forms Runtime process** also reside on the middle tier.
- **Database Tier**: The database tier (bottom of Figure 1) is the database server, where database server software is stored.

Figure 1   Three-tier configuration for running a form



Forms Listener Servlet

The Forms Listener servlet is a broker between the Java client and the Forms Runtime process. The servlet takes connection requests from Java client processes and initiates a Forms Runtime process on their behalf.

If your application uses Forms REST integration, the calls to REST are executed by the Forms Listener servlet.

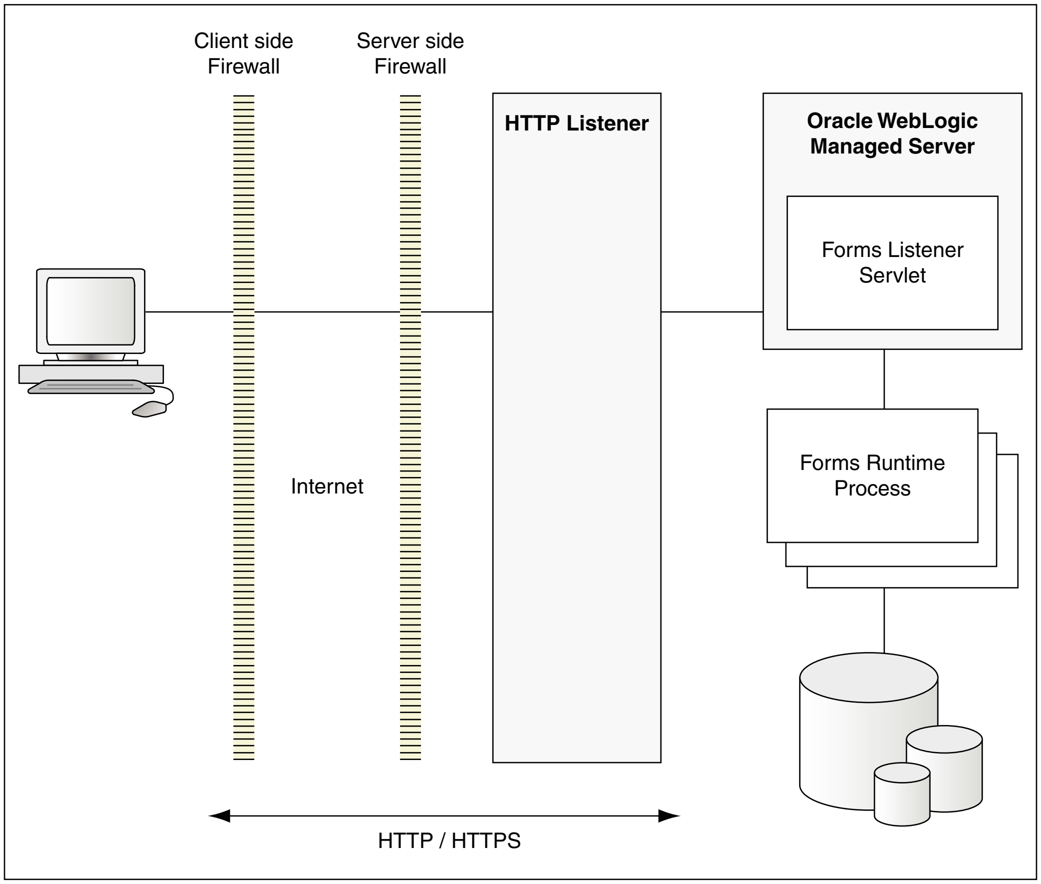
The Figure 2 shows how the client sends HTTP requests and receives HTTP responses from Forms Services. Oracle Forms Services uses the Forms Listener servlet to start, stop, and communicate with the Forms Runtime process. In this image, the client is to the left. In the center of the image, the HTTP Listener acts as the network endpoint for the client, keeping the other server computers and ports from being exposed at the firewall.

The Forms Runtime process, in the right side of the image, executes the code contained in a particular Forms application. The Forms Listener servlet manages the creation of a Forms Runtime process for each client and manages the network communications between the client and its associated Forms Runtime process.

Note:

The Forms Listener servlet is configured for you during the Oracle Fusion Middleware installation process.

Figure 2   Architecture using the Forms Listener Servlet



Forms Runtime Process

The Forms Runtime process plays two roles: as a server, when communicating with the client tier, and as a client when communicating with the database server. When it communicates with the client tier, it manages requests from client tiers and sends metadata to the client to describe the user interface. When it communicates with the database server, it queries the database server for requested data.
For each Oracle Forms session, there is one Oracle Forms Runtime process on the application server. This process is where Oracle Forms actually runs, and manages application logic and processing. It also manages the database connection; queries and updates data; runs any PL/SQL in the form; executes triggers; and so on. It uses the same forms, menus, and library files that were used for running in client/server mode.

The Forms Runtime process also uses a Java Virtual Machine (JVM), as needed, to run custom Java imported into the application, calls to Oracle Reports, or calls to Oracle Business Intelligence Publisher (also referred to as the Analytics Publisher). In order to limit the amount of memory consumed by the Forms Runtime process, the Forms JVM Controller can be configured and enabled. With the JVM Controller enabled, memory needed by this JVM can be shared across multiple JVMs, thereby reducing the total memory consumption and freeing more resources on the server. To manage JVM usage and pooling, see Configuring and Managing Java Virtual Machines.

Part II
Deployment

This part describes how to deploy Forms applications. This chapter also includes information about configuring and using Oracle Forms Application Deployment Services (FADS).

This part contains the following chapters:

Oracle Forms Services in Action

This topic describes the steps to run Forms Services in Oracle Fusion Middleware and how the configuration files are used.

Be aware that if you run an out-of-the-box Forms URL with no arguments the user will be shown the default test-form, which displays the Forms version number information. If it is desired that this information not be displayed, the administrator can simply modify the [default] config section in the Forms Web Configuration so that a different form is specified (or no form is specified at all, in which case users will get an error message when they try that URL rather than seeing the form that includes the version number). For example, assume Oracle HTTP Server (OHS) is running on port 7777 on a computer called *example.com*. Also assume no modifications have been made to the standard configuration created during the Oracle Fusion Middleware installation process.

When a user runs an Oracle Forms Services application, the following sequence of events occur:

1. The user calls Forms using a URL:
   `http://example.com:7777/forms/frmservlet?config=myapp&form=hrapp`

   In this example, the top level form module to be run is called *hrapp* using the configuration section called *myapp*.

2. Oracle HTTP Server listener receives the request. It finds `/forms` path in the URL and forwards the request to the correct Oracle WebLogic Managed Server based on the WebLogic handler mappings. The mapping is defined in `forms.conf`.

   > **Note:**
   >
   > Using Oracle HTTP Server in front of WebLogic Server is optional. Choosing to do so will require that `forms.conf` be configured post installation. The included example within the file can be used as an example of appropriate settings. Once settings have been saved, the file should be moved to the OHS configuration file directory that contains other `.conf` files, see Enabling Oracle HTTP Server with Oracle Forms Services.

3. Oracle WebLogic Managed Server maps the request to the Oracle Forms Services application that has a context root named `/forms`. It maps the request to the Forms servlet using the frmservlet mapping specified in the `application.xml` file.

4. The Forms servlet running on the Oracle WebLogic Managed Server processes the request. The Forms servlet:

   - Opens the servlet configuration file (`formsweb.cfg` by default), which is located in `$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config`.

   - Determines which configuration section to use in the `formsweb.cfg` file. In this example, the URL contains the query parameter `config=myapp`, therefore, the [`myapp`] section is used.

   - Determines which template file (for example, baseHTML, basejnlp, or basesaa.txt) file to use, based on:

     a. which client configuration (for example, browser, web start, or FSAL) made the request;

     b. what platform the user is running; and

     c. the settings of various parameters in the `formsweb.cfg` file.

   - Reads the template file, and returns the contents as an HTML, JNLP, or text (based on the client configuration choice) after performing variable substitutions as follows: Whenever a variable (like `%myParam%`) is encountered, the Forms servlet looks for a matching URL query parameter (for example, `&myParam=xxx`), or, failing that, looks for a matching parameter in the `formsweb.cfg` file. If a matching parameter is found, the variable (`%myParam%`) is replaced with the parameter value.

     In this example, the template file contains the text `%form%`. This is replaced with the value "hrapp".

5. Depending on which template file the Forms servlet selected, the page returned to the client contains an applet, object embed, or jnlp tag to start the Forms applet. The Forms client runs in the JVM environment provided by Oracle Java plug-in, Web Start, or standalone Java executable, depending on the request type.

6. To start the Forms applet, its Java code must first be loaded. The location of the applet is specified by the applet codebase and archive parameters.
   The virtual path definition in the `weblogic.xml` file for `/forms/java` allows the applet code to be loaded from the Web server.

   > **Note:**
   >
   > The Forms applet code is only loaded over the network the first time the user runs an Oracle Forms Services application or if a newer version of Oracle Forms Services is installed on the Web server. Otherwise, it is loaded from the Java cache on the local disk.

7. Once the Oracle Forms Services applet is running, it starts a Forms session by contacting the Forms Listener servlet at URL `http://example.com:7777/forms/lservlet`.

8. The Oracle HTTP Server listener receives the request. It forwards the request to Oracle WebLogic Managed Server, since the path `/forms/lservlet` matches a servlet mapping in the `web.xml` file (the one for the Forms Listener servlet).

9. The Forms Listener servlet (lservlet) starts a Forms run-time process (`frmweb.exe` or `frmweb`) for the Forms session.
10. Communication continues between the Forms applet and the Forms run-time process, through the Listener Servlet, until the Forms session ends.
11. The attribute value in a URL (such as the name of the form to run) is passed to the Forms run-time process. Part of the serverArgs value in the template file is `%form%`, which is replaced by "hrapp". Therefore, the run-time process runs the form in the file "hrapp.fmx". This file must be present in any of the directories named in the `FORMS_PATH` environment setting, which is defined in the environment file (`default.env` by default).
12. The Forms sessions end when either of the following occurs:
    - The top-level form is exited (for example, by the PL/SQL trigger code which calls the "exit_form" built-in function). The user is prompted to save changes if there are unsaved changes. `exit_form(no_validate)` exits the form without prompting.
    - If the user quits the session without first saving pending changes, all changes will be lost.

Application Deployment

Once you have created your application in Forms Developer, you are ready for application Web deployment. Oracle Forms Services accesses an application in Oracle Fusion Middleware through a specified URL.

The URL then accesses the HTTP Listener, which communicates with the Listener Servlet. The Listener Servlet starts a Forms run-time process (`frmweb.exe` on Windows or `frmweb` on UNIX and Linux) for each Forms Services session.

For information about how Forms Services run, see Oracle Forms Services in Action.

The following section are included:

- Deploying Your Application
- Specifying Parameters
- Creating Configuration Sections in Fusion Middleware Control
- Specifying Special Characters in Values of Runform Parameters

Deploying Your Application

To deploy a basic form with the default parameters set up by Oracle Fusion Middleware Config Wizard:

1. Create your application in Oracle Forms Developer and save or copy the related source files (`.fmb`, `.mmb`, `.pll`, `.olb`) to the desired location on the application server where they will be hosted.

   The source files are design time files that can only be opened in Forms Developer. The executable files (`.fmx`, `.mmx`, `.plx`) are the runtime files created when you compile the source files (using the Forms Compiler) and are used for Web deployment.

   See the Help menu in the Form Builder for more information about Forms file types and how to use the Forms Compiler (`frmcmp`).

2. Using the Forms Compiler, generate executables from the source files. The compiler and its location can be found here:

   - On Unix platforms: `FORMS_INSTANCE/bin/frmcmp.sh`
   - On Microsoft Windows: `ORACLE_HOME\bin\frmcmp.exe`

   If no arguments are passed into the compiler at startup, it will attempt to launch its graphical user interface. Details about using the Forms Compiler can be found in the Forms Developer (Form Builder) Help. Alternatively, use the `-help` option to expose optional arguments. For example: `frmcmp -help`

   Usage example: `frmcmp.sh module=myForm.fmb module_type=form compile_all=yes userid=user1/user1@orcl`

3. Modify the `formsweb.cfg` file so that Oracle Forms Services can access your application module. You edit this file in the **Web Configuration** page of Fusion Middleware Control, see Configuring Forms Services.

   Table 1 shows the configuration of an application called "`my_application`" with a form module called "`form=hrapp.fmx`":

   Table 1 Example of Configuration Section Parameter Values

Configuration Section Name	Forms Module Name Value
my_application	hrapp.fmx

   When configured, the Oracle Forms Services module hrapp.fmx is accessible on the Web by entering "`...?config=my_application`" in the browser URL (the name of the **Web Configuration** section in `formsweb.cfg`).

   > **Note:**
   >
   > The name of the configuration section must not include spaces, must contain only alphanumeric characters, and must be unique.

4. Make sure the .fmx file location is specified in the `FORMS_PATH` environment variable.

   For example, in Windows, if your .fmx file is located in `d:\my_files\applications`, in the `FORMS_PATH`, include `d:\my_files\applications`. On Windows, use semicolons to separate directory locations if specifying multiple locations. On UNIX/Linux, use colons for separators. Specify this information in the **Environment Configuration** page for the environment file.

5. To modify an environment file, select the file in the **Environment Configuration** page of Fusion Middleware Control and add or edit environment variables as needed by your application. For example, you can add the environment variable shown in the following table.

   Table 2 Example of Environment Variable Values

Environment Variable Name	Environment Variable Value
NLS_LANG	`NLS_LANG=GERMAN_GERMANY.WE8ISO8859P15`

If you specified these environment variables in an environment file, specify this environment file in the respective configuration section of the formsweb.cfg in the **Web Configuration** page.

6. Enter the name of your application in the URL as shown:

   ```
   http://example.com:9001/forms/frmservlet?
   ```

   where "example" is the hostname of your computer and "9001" is the port used by your WebLogic Manager Server

   Once you have created a configuration section, add "config=" and the name of the configuration section. In this example, the URL to access hrapp.fmx is:

   ```
   http://example.com:9001/forms/frmservlet?config=my_application
   ```

Specifying Parameters

There are two ways to predefine parameter values for your Oracle Forms Services applications. You can define parameters by:

- Editing your application settings in the default section of the **Web Configuration** page of Fusion Middleware Control. The default configuration section displays the default values that are used by Oracle Forms Services.
- Managing (adding, editing, copying, deleting) other system and user parameter values in the named application configuration section (see Creating Configuration Sections in Fusion Middleware Control). For example, in the configuration section you create for myApp, you can add or change these parameters and their values, as shown in the following table.

Parameters specified in the named configuration section of a **Web Configuration** override the settings in the default section.

> **Note:**
>
> System Parameters cannot be overridden in the URL, while user parameters can.

Creating Configuration Sections in Fusion Middleware Control

Within the configuration sections you created in step 2 of Deploying Your Application, you can specify parameters for your Oracle Forms Services applications. You can specify any application and system parameters that are available in the default section for **Web Configuration** page.

For example, you can set the look and feel of the application to the Oracle look and feel by setting the lookAndFeel parameter to the value of oracle and clicking **Apply**.

You can also override the default parameter values in the named configuration section. For example, to predefine the connect information of an application to <username>/<password>@orcl, the parameter value for userid must be set in the named configuration section by changing the parameter value of userid to <username>/<password>@orcl.

For other parameters that you can edit, see Forms Configuration Parameters.

Editing the URL to Access Oracle Forms Services Applications

You can directly type unrestricted parameters in the URL that accesses your Oracle Forms Services application. Using the previous example, instead of specifying the `form` parameter in your configuration file, you could also type it into the URL as follows:

```
http://example.com:9001/forms/frmservlet?config=my_application&form=hrapp
```

You can use the ampersand (&) to call a combination of a form and named configuration parameters. In the above example, you are calling the form "hrapp" with the parameter settings you specified in "my_application".

> **Note:**
>
> Parameters specified in the URL override the parameters set in the configuration section, see Managing URL Security for Applications.

Specifying Special Characters in Values of Runform Parameters

Certain considerations apply if values passed to runform parameters contain special characters. This section describes these considerations, and compares the default behavior in this release with the behavior in prior releases.

Runform parameters are those that are specified in the serverArgs applet parameter of the template HTML file. The value specified for the serverArgs parameter in the template HTML file, after variable substitution, is sometimes referred to as the command-line parameters string. It consists of a series of blank-separated `name=value` pairs. The name must consist solely of alphanumeric or underscore characters. The value portion of a name=value pair can be an arbitrary string.

Default Behavior in the Current Release

The value of a runform parameter can be specified in one of three places:

1. In the value of the `serverArgs` parameter in the template HTML file (for example, `base.htm`).
2. In the value of a variable specified in the configuration file (for example, `formsweb.cfg`), which is substituted (directly or recursively) for a variable reference in (1). Such values are typically maintained using Fusion Middleware Control; see Configuring Forms Services.
3. As an attribute value in a URL, which is substituted directly for a variable reference in (1) or (2).

For case (3), URL syntax rules (as enforced by the browser and the application server) require that certain characters be entered as URL escape sequences ('%' followed by 2 hexadecimal digits representing the ASCII value of the character, for a total of three characters).

This requirement includes the `%` character itself (which must be entered as `%25`). In addition, Oracle Forms Services currently requires that the quote character ("") be entered as `%22`, even if the browser and the application server allow a quote to be entered without escaping.

URL syntax rules also allow a space to be entered as a `+` (as an alternative to the URL escape sequence `%20`). However in the value of the `otherparams` configuration parameter, a `+` is treated specially; it separates name=value pairs as opposed to indicating a space embedded in the value of a runform parameter.

For example, if a runform application has user parameters `param1` and `param2`, and you want to assign them the values 'a b' and 'c d', you do so by incorporating the following into a URL:

```
&otherparams=param1=a%20b+param2=c%20d
```

When specifying runform parameters in the template HTML files or in the configuration files (cases (1) and (2)), Forms requires URL escape sequences in some circumstances, allows them in others, and forbids them in still others.

Outside of the values of runform parameters, URL escape sequences must not be used. For example, the `=` in a name=value pair must always be specified simply as `=`, and the space that separates two adjacent name=value pairs must always be specified simply as " " (a single space character).

Within the value of a runform parameter, space (' ') must be specified as a URL escape sequence (`%20`). The HTML delimiter character (specified in the configuration file) must also be specified as a URL escape sequence. And when the runform parameter is specified in the template HTML file (case (1)), quote ("") must also be specified as a URL escape sequence (%22).

Any other 7-bit ASCII character may also be specified as a URL escape sequence, although this is not required (except possibly for `%`, as noted below). Certain additional restrictions apply to the `%` character. These include:

- If the HTML delimiter is `%` (the default), then an occurrence of `%` within the value of a runform parameter must be escaped (specified as `%25`). (This actually follows from the requirement stated above, that the HTML delimiter character be escaped). Furthermore, variable names must never begin with two hexadecimal digits that represent a 7-bit ASCII value (that is, two hexadecimal digits, the first of which is in the range 0-7).
- If the HTML delimiter is not `%`, then an occurrence of `%` must be escaped if it is immediately followed by an octal digit and then a hexadecimal digit. It is recommended that other occurrences of '%' also be escaped; but this is not a requirement.

> **Note:**
>
> You might choose to ignore this recommendation if you have existing template HTML files or configuration files created in prior releases, which use an HTML delimiter other than '%', and which contain '%' in runform parameter values.

Behavior in Previous Releases

Release 9.0.4 and later behave the same as the current release except that a quote must be escaped (%22) within the value of a runform parameter in a configuration file, and in the template HTML file.

Releases before 9.0.4 did not allow URL escape sequences in runform parameter values specified in the template HTML file or the configuration file (cases (1) and (2) above). In all three cases, it was difficult or impossible to specify certain special characters, notably space, quote, and apostrophe. Also, certain transformations were applied to the parameter value before passing it to runform. Most notably, if a value began and ended with an apostrophe, these were typically stripped off. However, these transformations were not well-defined, and they differed between the Web and client/server environments.

Obtaining the Behavior of Prior Releases in the Current Release

If your applications are dependent on the behavior of prior releases, you can obtain that behavior in the current release, by simply setting the value of the `escapeparams` variable to `False` in the configuration file (this can be accomplished using Fusion Middleware Control).

If you want to obtain the old behavior only for selected applications, you can specify different values for the `escapeparams` variable in different configuration sections. Applications that require the old behavior can specify a configuration section in which the `escapeparams` variable is set to `False`; applications that require (or tolerate) the behavior in the current release can specify a configuration section in which the `escapeparams` variable is set to `True`.

Considerations for Template Files

If you are creating your own template files, consider the following:

- It is recommended that a reference to the `escapeparams` variable (the string `%escapeparams%`, if '`%`' is the HTML delimiter character) appear at the beginning of the value of the `serverArgs` applet parameter, followed by a space. See the shipped `base.htm` file for an example.
- References to the `escapeparams` variable must appear nowhere else in the template file. If you choose to enclose the value of the `serverArgs` applet parameter in apostrophes instead of quotes, then within the value of a runform parameter in your template file, apostrophes must be escaped (%27). Quotes do not require escape sequences.
- It is permissible to omit the reference to the `escapeparams` variable from the beginning of the value of the `serverArgs` applet parameter. This results in the behavior of prior releases, regardless of the value specified in the configuration file for the `escapeparams` variable.

Considerations for Static HTML Pages

If you are invoking the runform engine using static HTML, and you want to obtain the behavior in the current release, then you must take certain steps.

> **Note:**
>
> The use of static HTML or JNLP is not recommended. It is recommended that all calls to Forms applications be routed through the Forms Servlet (`frmservlet`). Not doing so may result in unpredictable behavior.

The basic rule is that your static HTML must look like the HTML generated by the Forms servlet. Specifically, the value of the `serverArgs` applet parameter must begin with the string `escapeparams=true` (case-insensitive).

Also, in the value portion of each name=value pair, in the value of the `serverArgs` applet parameter, certain characters must be specified by a URL escape sequence, as listed in the following table.

Table 3   URL Escape Sequences for Static HTML pages

Characters that must be escaped	URL Escape Sequence
newline ' \n '	%0a
space ' '	%20
quote ' " '	%22
percent ' % '	%25
apostrophe ' ' '	%27
left parenthesis ' ('	%28
right parenthesis ') '	%29

It is also permissible to escape other 7-bit ASCII characters in the value portion of a `name=value` pair.

Here's an example of what the `serverArgs` applet parameter might look like in static HTML. This is for a form named "my form" (quotes not included), which is being passed the value "foo'bar" (quotes again not included) to the user-defined parameter named `myparam`.

```
<PARAM NAME="serverArgs" VALUE="escapeparams=true module=my%20form
myparam=foo%27bar">
```

Oracle Forms Application Deployment Services

The Forms Application Deployment Services (FADS) simplifies the process of packaging applications, deploying, configuring, and storing archived copies of the applications.

The Forms Application Deployment Services allow administrators or developers to package applications, deploy, configure, and store archived copies of the applications with the click of a button. To deliver and deploy a Forms application, you need to create an application package that contains Forms executable files, such as fmx, mmx, and plx files. In addition to generating executable files, applications may also require additional custom files, such as jars and html and possibly unique configuration settings.

Using the FADS web interface, you can check on the status of your deployments, deploy updated versions of your applications, delete no longer needed applications, and much more. A command line interface is also available, which may be helpful for Forms build integration and scripting automated deployment jobs where using a web interface may not be appropriate.

> **Note:**
>
> The FADS web interface supports 100MB maximum archive (.far) file size limit. Any attempt to deploy files larger than 100MB may result in timeout errors and failed deployment. For large files, use the provided command line utility instead of the Web interface.

The followings sections are included:

- Accessing FADS
- Using the Forms Applications Packager
- Application Package Deployment

Accessing FADS

To work with FADS, you can use the web interface or the command line interface.

Accessing the Web Interface

FADS is deployed to the WebLogic Server Administration Server. Therefore, to access FADS the WebLogic Server Administration Server must be running and accessible. To ensure the highest degree of security, we recommend that you enable SSL on the WebLogic Server Administration Servers. See Using Oracle Forms Services with the HTTP Listener and Oracle WebLogic Server.
To access the FADS web interface, use a URL similar to the following:

```
http://example.com:7001/fadsui
```

To access FADS, use the same credentials used to access Fusion Middleware Control (Enterprise Manager).

> **Note:**
>
> Some of the ADF errors and warnings related to FADS that appear in Admin Server logs can be ignored; unless FADS is not functioning properly.

Accessing the Command Line Interface

FADSCLI is the command line interface that allows users to connect and interact with the Forms Application Deployment Services of the same version as the FADSCLI utility.

By default, the FADS Command Line Interface connects to the server in non-SSL mode.

To use SSL mode, you can connect to the WebLogic Administration Server SSL port by passing the additional argument `ssl=true` to the various FADSCLI options. When running in SSL mode, the certificate has to be imported on the client side JVM, where you are running FADSCLI. Refer to the Java documentation on importing certificate to the Java keystore.

FADSCLI includes the FADSCLI scripts and its libraries. It is located in `FMW ORACLE_HOME`, in the following directory `ORACLE_HOME/forms/fads/fads-client`.

For information about the various FADSCLI options, see FADSCLI Options.

Using the Forms Applications Packager

You need to create an application package before you can deploy it using FADS. Use the Forms Application Packager (FAP) command line utility to package Forms Applications Archive files, referred to as FAR files.

The Forms Application Packager searches for the Forms artifacts under the artifacts directory and generates the FAR files. These FAR files can then be deployed to FADS running on the latest Oracle Fusion Middleware 14*c* release domain.

The following topics are included:

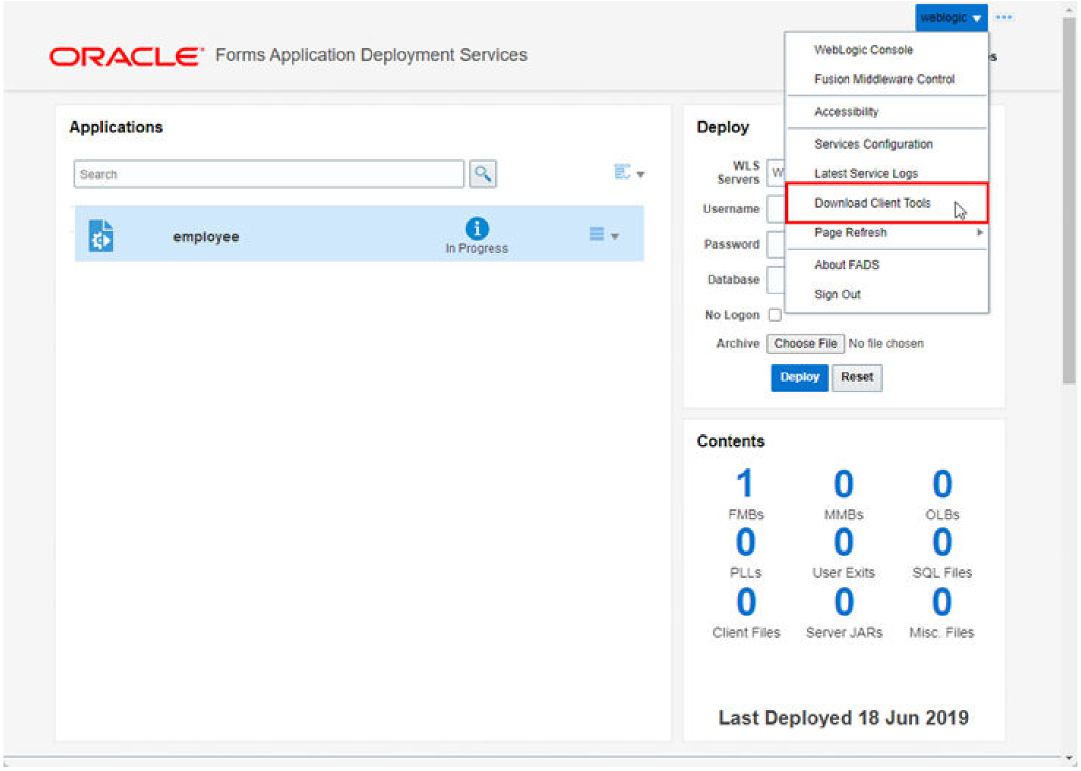
- Obtaining the Forms Applications Packager
- Creating an Application Package
- Understanding FAR File Contents

Obtaining the Forms Applications Packager

The Forms Application Packager searches for the Forms artifacts under the artifacts directory and generates the FAR files. These FAR files can then be deployed to FADS running on the latest Oracle Fusion Middleware 14*c* release domain. Before you can use the Forms Application Packager, download the client tools set from the web interface. The FAP utility is included in this set.
The Forms Application Packager includes the scripts (`fa_packager.sh` and `fa_packager.cmd`), and its dependent libraries. The Forms Application Packager utility is available:

- In the `$FMW_HOME/forms/fads/fads-client` directory, in current Oracle Fusion Middleware 14*c* release, and Oracle Forms Standalone Builder 14*c* installation.
- For download from the Forms Application Deployment Services web interface. Navigate to the ***<username>*** (for example, weblogic) drop down menu on the upper right side and select the **Download Client Tools** option.

Figure 3   Download Client Tools Option



The Client Tools download includes both the Forms Application Packager utility and the FADS Command-line Interface (FADSCLI).

Creating an Application Package

Use the various Forms Applications Packager arguments to create the application package.

Here is the syntax to use the Forms Applications Packager.

```
fa_packager.sh appName=<app_name> appVersion=<app_version>
artifactDirs=<artifact_dirs>
                       outputDir=<output_dir> preserveDir=<preserve_dir>
startForm=<startForm> ignoreExt=<ignore extn list>
```

The Forms Applications Packager arguments should always be passed in the same order as listed in the syntax. Refer Table 4 for more information on the arguments.

Table 4 Forms Application Packager Arguments

Argument	Description	Mandatory/Optional	Notes
`appName`	Forms Application name	Mandatory	The first argument always needs to be the Forms Application Name. Example: sales
`appVersion`	Forms Application version	Mandatory	The second argument always needs to be the Forms Application Name. This is the Forms Application version and it has not nothing to do with the Forms Product version. Example: 1.0
`artifactDirs`	Forms Applications Artifacts directory It is the directory where the Forms Application artifacts (Forms, Menus, PLSQL libraries, Object libraries) reside. Optionally, Forms applications related configuration can be included into the FAR file by creating the configuration contents in the files. as shown in Table 5.	Mandatory	The third argument always needs to be the artifacts directory. Avoid creating one top-level directory for all the Forms applications. Example: if you have Forms Applications Sales, Finance and Human Resources. Create a separate top-level directory for each one of these applications and pass that directory path as the artifacts directory. When specifying optional configuration files, place the files under the top level directory of the artifacts directory. The Forms Application Packager will pick up the configuration from these files and include it in the FAR file.

Table 4 (Cont.) Forms Application Packager Arguments

Argument	Description	Mandatory/Optional	Notes
`outputDir`	It is the directory where the FAR files are generated.	Optional **Note:** When preserve_dir argument is used then output_d	When used, it should always be the forth argument passed to the FA packager. If you provide the output directory, then fa_packager will generate the far files under this directory. If you don’t provide this argument then fa_packager will generate the far files under the current directory.

Table 4 (Cont.) Forms Application Packager Arguments

Argument	Description	Mandatory/Optional	Notes
		ir becomes mandatory.	
`preserveDir`	It is a Boolean argument (true/false) with a default value of false. It indicates if the directory structure from the artifact directory needs to be preserved in the FAR files.	Optional	When used, it should always be the fifth argument passed to the FA packager.

Table 5 Forms Applications Related Configuration Files

File	Notes	Destination Forms Configuration Files
`app.cfg`	The contents of this file will be added to formsweb.cfg during deployment by FADS. • You can define multiple application sections in this file. • You should not have an application section named “default”, nor should you name any Forms Application as default. • When the user does not provide app.cfg, an empty application section based on the application name is created in formsweb.cfg. • If userId parameter is added to this file, it will be ignored.	`formsweb.cfg`
`app.env`	The contents of this file will be added to a new env file during deployment by FADS. When the user does not provide app.env, a Forms environment configuration file is still created by FADS with all the required environment variables.	`<appname>.env`

Table 5   (Cont.) Forms Applications Related Configuration Files

File	Notes	Destination Forms Configuration Files
`app_jvmcontrollers.cfg`	The contents of this file will be added to jvmcontrollers.cfg during deployment by FADS.	`jvmcontrollers.cfg`
`app_registry.dat`	The contents of this file will be added to Registry.dat	`Registry.dat`

Understanding FAR File Contents

This topic describes the contents of the FAR files.

The contents of the FAR file which are generated by the Forms Application Packager, are shown in Table 6:

Table 6   FAR File Contents

Name	Description
Forms_application.xml	It is the FADS deployment descriptor, which contains all the metadata relevant to the packaged Forms Application.
fmb directory	Includes all the Forms (fmb) files from the Forms Application artifacts directory.
mmb directory	Includes all the Menus (mmb) files from the Forms Application artifacts directory.
pll directory	Includes all the PLSQL libraries (pll) files from the Forms Application artifacts directory.
olb directory	Includes all the Object libraries (olb) files from the Forms Application artifacts directory.
sql directory	Includes all the SQL files files from the Forms Application artifacts directory.
user_exits directory	Includes all the User Exit libraries (.so and .dll for Windows platform) files from the Forms Application artifacts directory.
client_jars directory	Includes the following artifacts from the Forms Application artifacts directory: • JAR files (*.jar) • HTML files (*.htm, *.html) • Java Script files (*.js) • Image and Icon files (*.jpeg, *.jpg, *.gif, *.png) • JNLP files (*.jnlp)
java_importer directory	If the Forms application has any Java Importer related jar files those should be placed under a directory named java_importer and placed under the top level directory under the Forms Application artifacts directory.
misc_files directory	Includes all Microsoft Word (*.doc, *.docx) files, Portable Document Format (*.pdf) files and Text (*.txt) files from the Forms Application artifacts directory.

Table 6 (Cont.) FAR File Contents

Name	Description
webutil 32-bit files directory	Includes all the 32-bit webutil files. The top-level `webutil` directory includes the `win32` subdirectory that contains all the 32-bit webutill files. Here is an example. `webutil/win32/*.dll`
webutil 64-bit files directory	Includes all the 64-bit webutil files. The top-level `webutil` directory includes the `win64` subdirectory that contains all the 64-bit webutill files. Here is an example. `webutil/win64/*.dll`

> **Note:**
>
> If the preserve_dir argument is passed, none of the directories, shown in Table 6, will be created. The sub-directories similar to the sub-directories under the artifacts directory will be created in the FAR file.

Application Package Deployment

After you create the application package, you can deploy it using FADS.

The following topics are included:

- Using the Web Interface
- Using the Command Line Interface

Using the Web Interface

The FADS web interface helps you to quickly deploy applications.

The following topics are included:

- Understanding the UI
- Deploying the Application Package

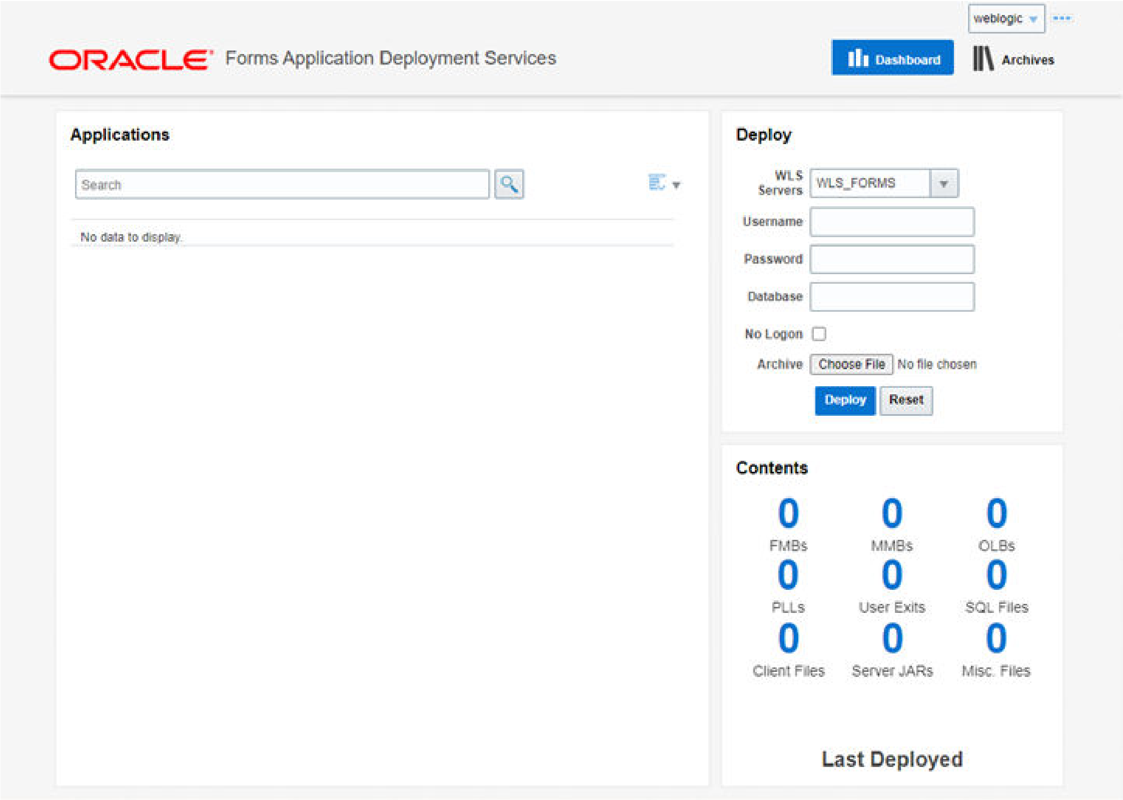
Understanding the UI

The FADS web interface includes the Dashboard and Archives tabs. You can also use the drop-down menu to access the configuration options.

Dashboard Tab

The Dashboard tab of the FADS web interface is displayed when you initially log on. As the name suggests, this tab allows you to deploy applications and view the deployment progress and status.

Figure 4   The Dashboard Tab



WLS Servers: WLS_FORMS

No Logon □

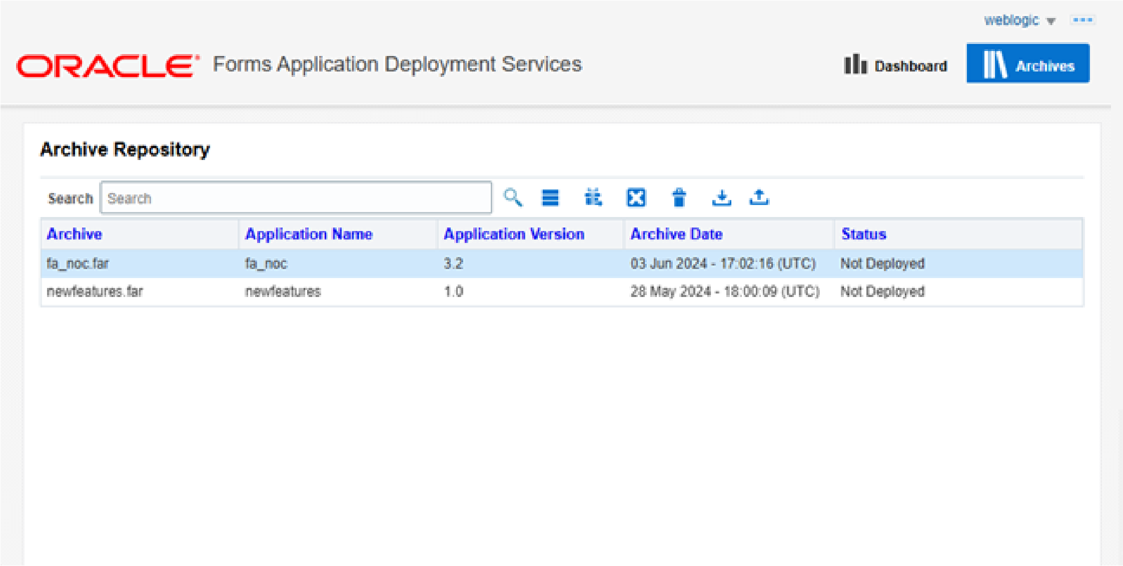
Archive: Choose File No file chosen

0 FMBs	0 MMBs	0 OLBs
0 PLLs	0 User Exits	0 SQL Files
0 Client Files	0 Server JARs	0 Misc. Files

Archives Tab

The Archives tab lists all applications deployed earlier. You can review the archive name, application name and version, and date. Use the Search field to locate a specific application in the list. For a selected application in the list, you can view details, deploy, stop deployment, delete, or download.

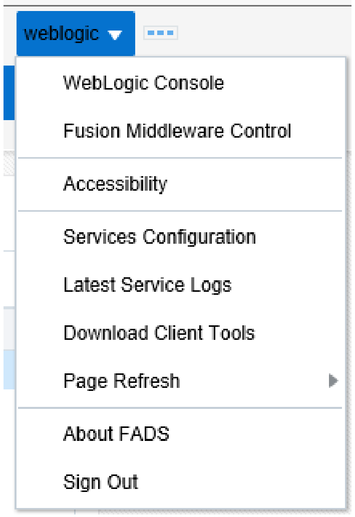
Figure 5   The Archives Tab



Archive	Application Name	Application Version	Archive Date	Status
fa_noc.far	fa_noc	3.2	03 Jun 2024 - 17:02:16 (UTC)	Not Deployed
newfeatures.far	newfeatures	1.0	28 May 2024 - 18:00:09 (UTC)	Not Deployed

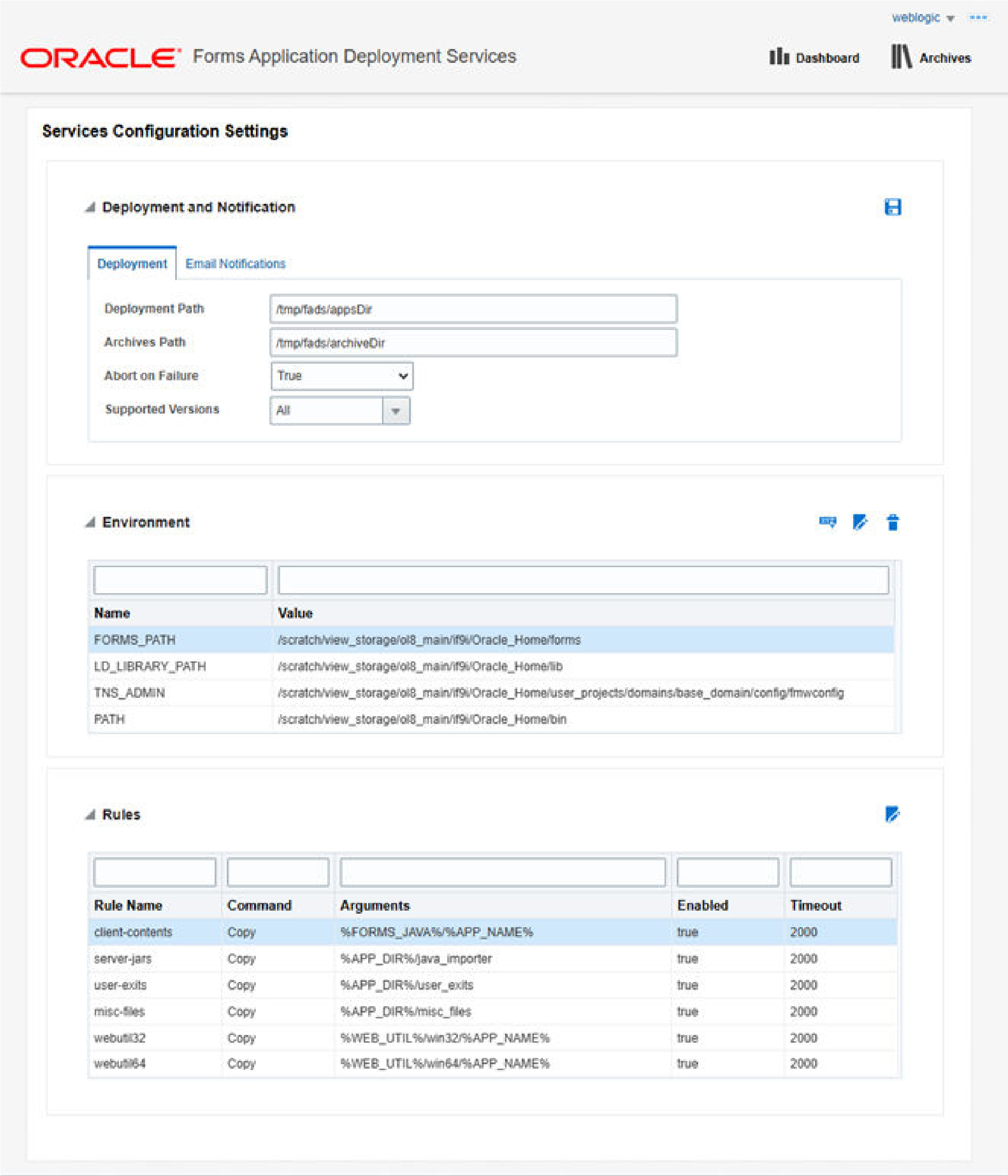
<username> Drop-Down Menu

The <username> drop-down menu (for example, weblogic) includes multiple options. Using these options you can view service configuration, download tools, view logs, and specify page refresh interval.



Click the Services Configuration option to view the Services Configuration page that lists FADS-related configuration settings. The page includes the Environment, Settings, and Rules sections that are prepopulated.

Figure 6   The Services Configuration Page



Services Configuration Settings

Deployment and Notification

Deployment | Email Notifications

Deployment Path	/tmp/fads/appsDir
Archives Path	/tmp/fads/archiveDir
Abort on Failure	True
Supported Versions	All

Environment

Name	Value
FORMS_PATH	/scratch/view_storage/ol8_main/if9i/Oracle_Home/forms
LD_LIBRARY_PATH	/scratch/view_storage/ol8_main/if9i/Oracle_Home/lib
TNS_ADMIN	/scratch/view_storage/ol8_main/if9i/Oracle_Home/user_projects/domains/base_domain/config/fmwconfig
PATH	/scratch/view_storage/ol8_main/if9i/Oracle_Home/bin

Rules

Rule Name	Command	Arguments	Enabled	Timeout
client-contents	Copy	%FORMS_JAVA%/%APP_NAME%	true	2000
server-jars	Copy	%APP_DIR%/java_importer	true	2000
user-exits	Copy	%APP_DIR%/user_exits	true	2000
misc-files	Copy	%APP_DIR%/misc_files	true	2000
webutil32	Copy	%WEB_UTIL%/win32/%APP_NAME%	true	2000
webutil64	Copy	%WEB_UTIL%/win64/%APP_NAME%	true	2000

Deploying the Application Package

To deploy the application package you created with Forms Applications Packager, use the Dashboard tab in the FADS web interface.

1. Access the FADS web interface.
2. Navigate to the Deploy section on the Dashboard tab.
3. Select the server from the WLS server list.
4. Specify the database user credentials needed to generate this Forms application.

5. Enter the database name needed to generate this Forms application.
6. Browse and select the application package (`.far`) file and click **Deploy**.

   The Contents section is updated to show the files contained in the selected application package. The Applications section displays the deployment status.
7. Review the diagnostic logs for FADS services for information on any failures that might occur during deployment of FAR files or errors that occur with FADS deployment services.

   Access the FADS diagnostics logs by clicking on the drop-down arrow next to the current user's name on the upper right of the page, and then click on Latest Service Logs.

Using the Command Line Interface

Use the FADS command line interface to deploy multiple applications and to schedule deployment of applications.

The FADS command line interface is efficient when you need to deploy:

- Multiple applications simultaneously
- Updates to multiple existing applications simultaneously
- Schedule nightly or regular updates for selected applications

The FADS CLI provides multiple options to help you deploy and manage applications.

Table 7 List of FADSCLI Options

Option	Description
`listApps`	Displays all the Forms applications deployed to the FMW domain through the Forms Application Deployment Services.
`listArchives`	Displays all the Forms Application Archive (far) files that reside in the Archives Repository in Forms Application Deployment Services.
`deployApp`	Deploys a given Forms Application Archive (far) file to the Forms Application Deployment Services running on the FMW domain.
`deployApps`	It is batch mode of deployment, where it deploys all the Forms Application Archive (far) files, under a given archives directory to the Forms Application Deployment Services running on the FMW domain.
`deployArchive`	Deploys a Forms Application Archive file residing in the Archives Repository to to the Forms Application Deployment Services running on the FMW domain.
`undeployApp`	Undeploys / deletes a Forms application that is currently deployed to the Forms Application Deployment Services.
`describeApp`	Describes the contents of a Forms Application that is currently deployed to the Forms Application Deployment Services.
`deleteArchive`	Deploys a Forms Application Archive file residing in the Archives Repository
`downloadLog`	Downloads the deployment logs of a a Forms Application that is deployed to the Forms Application Deployment Services.
`downloadArchive`	Deletes a Forms Application Archive file residing in the Archives Repository.

For more information on these options, see FADSCLI Options.

FADSCLI Options

This topic describes the options available for FADSCLI.

Help

The following script describes `Help` for FADSCLI.

```
sh-4.1$ ./fadscli.sh
fadscli Help :
------------------------------------------------------------------------------
------------------------------------------------
         options          |
description

------------------------------------------------------------------------------
------------------------------------------------
         listApps         |   displays all the applications deployed to the
FMW domain
         listArchives     |   displays all the far files that reside in the
archives repository
         deployApp        |   deploys the far file to the FMW
domain
         deployApps       |   deploys all the far files in the local archives
directory (archivedir) to the FMW domain
         deployArchive    |   deploys the far file (archivename) from the
archive repository to the FMW domain
         undeployApp      |   undeploys/deletes the Forms application from
the FMW domain
         describeApp      |   describes the contents of an application that
is deployed on the FMW domain
         deleteArchive    |   deletes the archive from the archive
repository
         downloadLog      |   gets the deployment logs of an application that
is deployed to the FMW domain
         downloadArchive  |   downloads an archive from the archive
repository


Usage :
     fadscli.sh option hostname=hostname portno=portno username=username
password=password

-----------------------------o-p-t-i-o-n-s-----a-r-g-u-m-e-n-t-
s--------------------------------------------------------------

     fadscli.sh listApps hostname=<hostname> portno=<portno>
username=<username> password=<password>
     fadscli.sh listArchives hostname=<hostname> portno=<portno>
username=<username> password=<password>
     fadscli.sh deployApp hostname=<hostname> portno=<portno>
username=<username> password=<password> farfile=<path to the far file>
dbuser=<dbuser> dbpassword=<dbpassword> dbalias=<dbalias>
managedserver=<managedserver1,managedserver2>
```

```
     fadscli.sh deployApps hostname=<hostname> portno=<portno>
username=<username> password=<password> archivedir=<path to the directory
containing far files> dbuser=<dbuser> dbpassword=<dbpassword>
dbalias=<dbalias> managedserver=<managedserver1,managedserver2>
     fadscli.sh deployArchive hostname=<hostname> portno=<portno>
username=<username> password=<password> archivename=<archivename>
appname=<appname> appversion=<appversion> dbuser=<dbuser>
dbpassword=<dbpassword> dbalias=<dbalias>
managedserver=<managedserver1,managedserver2>
     fadscli.sh undeployApp hostname=<hostname> portno=<portno>
username=<username> password=<password> appname=<appname>
appversion=<appversion>
     fadscli.sh describeApp hostname=<hostname> portno=<portno>
username=<username> password=<password> appname=<appname>
appversion=<appversion>
     fadscli.sh deleteArchive hostname=<hostname> portno=<portno>
username=<username> password=<password> archivename=<archivename>
appname=<appname> appversion=<appversion>
     fadscli.sh downloadLog hostname=<hostname> portno=<portno>
username=<username> password=<password> appname=<appname>
appversion=<appversion>
     fadscli.sh downloadArchive hostname=<hostname> portno=<portno>
username=<username> password=<password> archivename=<archivename>
appname=<appname> appversion=<appversion>

 optional argument: pass argument ssl=true when running HTTPS/SSL
------------------------------------------------------------------------------
------------------------------------------------
sh-4.1$
```

listApps

```
sh-4.1$ ./fadscli.sh help listApps

listApps : displays all the applications deployed to the FMW domain

usage:

     fadscli.sh listApps hostname=<hostname> portno=<portno>
username=<username> password=<password>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
```

listArchives

```
sh-4.1$ ./fadscli.sh help listArchives

listArchives : displays all the far files that reside in the archives
repository

usage:

     fadscli.sh listArchives hostname=<hostname> portno=<portno>
username=<username> password=<password>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
```

deployApp

```
sh-4.1$ ./fadscli.sh help deployApp

deployApp : deploys the far file to the FMW domain

usage:

     fadscli.sh deployApp hostname=<hostname> portno=<portno>
username=<username> password=<password> farfile=<path to the far file>
dbuser=<dbuser> dbpassword=<dbpassword> dbalias=<dbalias>
managedserver=<managedserver1,managedserver2>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
farfile       : path of the far file that is to be deployed
dbuser        : Forms application database username
dbpassword    : Forms application database password
dbalias       : Forms application database alias
managedserver : Forms Managed Servers where the Forms application should be
deployed
```

deployApps

```
sh-4.1$ ./fadscli.sh help deployApps

deployApps : deploys all the far files in the local archives directory
(archivedir) to the FMW domain

usage:

     fadscli.sh deployApps hostname=<hostname> portno=<portno>
username=<username> password=<password> archivedir=<path to the directory
containing far files> dbuser=<dbuser> dbpassword=<dbpassword>
dbalias=<dbalias> managedserver=<managedserver1,managedserver2>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
archivedir    : local directory path containing the far files
dbuser        : Forms application database username
dbpassword    : Forms application database password
dbalias       : Forms application database alias
managedserver : Forms Managed Servers where the Forms application should be
deployed
```

deployArchive

```
sh-4.1$ ./fadscli.sh help deployArchive

deployArchive : deploys the far file (archivename) from the archive
repository to the FMW domain

usage:

     fadscli.sh deployArchive hostname=<hostname> portno=<portno>
username=<username> password=<password> archivename=<archivename>
appname=<appname> appversion=<appversion> dbuser=<dbuser>
dbpassword=<dbpassword> dbalias=<dbalias>
managedserver=<managedserver1,managedserver2>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
archivename   : Name of the archive in archive repository
appname       : Forms application name
```

```
appversion    : Forms application version
dbuser        : Forms application database username
dbpassword    : Forms application database password
dbalias       : Forms application database alias
managedserver : Forms Managed Servers where the Forms application should be
deployed
```

undeployApp

```
sh-4.1$ ./fadscli.sh help undeployApp

undeployApp : undeploys/deletes the Forms application from the FMW domain

usage:

     fadscli.sh undeployApp hostname=<hostname> portno=<portno>
username=<username> password=<password> appname=<appname>
appversion=<appversion>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
appname       : Forms application name
appversion    : Forms application version
```

describeApp

```
sh-4.1$ ./fadscli.sh help describeApp

describeApp : describes the contents of an application that is deployed on
the FMW domain

usage:

     fadscli.sh describeApp hostname=<hostname> portno=<portno>
username=<username> password=<password> appname=<appname>
appversion=<appversion>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
appname       : Forms application name
appversion    : Forms application version
```

deleteArchive

```
sh-4.1$ ./fadscli.sh help deleteArchive

deleteArchive : deletes the archive from the archive repository

usage:

     fadscli.sh deleteArchive hostname=<hostname> portno=<portno>
username=<username> password=<password> archivename=<archivename>
appname=<appname> appversion=<appversion>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
archivename   : Name of the archive in archive repository
appname       : Forms application name
appversion    : Forms application version
```

downloadLog

```
sh-4.1$ ./fadscli.sh help downloadLog

downloadLog : gets the deployment logs of an application that is deployed to
the FMW domain

usage:

     fadscli.sh downloadLog hostname=<hostname> portno=<portno>
username=<username> password=<password> appname=<appname>
appversion=<appversion>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
appname       : Forms application name
appversion    : Forms application version
```

downloadArchive

```
sh-4.1$ ./fadscli.sh help downloadArchive

downloadArchive : downloads an archive from the archive repository
```

```
usage:

     fadscli.sh downloadArchive hostname=<hostname> portno=<portno>
username=<username> password=<password> archivename=<archivename>
appname=<appname> appversion=<appversion>


arguments:
--------------------------------------------------------------
hostname      : Weblogic Admin Server hostname
portno        : Weblogic Admin Server port (ssl or non-ssl)
username      : Weblogic Admin username
password      : Weblogic Admin password
archivename   : Name of the archive in archive repository
appname       : Forms application name
appversion    : Forms application version
```

Part III

Client Configurations

This part describes the client configuration options available with Oracle Forms. Select an option based on your requirements.

The following client configurations are available:

- Oracle Forms Standalone Launcher
- Java Web Start
- Java Applet Embedded in HTML
- JNLP Embedded in HTML

Example configurations for each of the above options is provided in the Forms web configuration, `formsweb.cfg` file.

Also, review Browser Considerations for information on browsers supported for the client configuration options.

Oracle Forms Standalone Launcher

The Forms Standalone Launcher (FSAL) offers an alternative way for end-users to run Forms applications rather than using a browser.

- About Forms Standalone Launcher
- Installing the Forms Standalone Launcher
- Using FSAL to Run Applications
- File Caching
- Security Considerations
- FSAL Reference

About Forms Standalone Launcher

The Forms Standalone Launcher (FSAL) provides a browser-less, client/server-like experience for users to access and interact with applications. The application runs in its own parent window, unlike when running a form embedded in an html page in a browser.

As a result, there is no risk of accidentally navigating away from the running form by pressing the Back or Forward button, or clicking a bookmark, and so on. With FSAL, the application is hosted on a centralized application server such as WebLogic Server. This means the Forms application modules are securely stored on the remote middle-tier server.

Although the user will not have direct access to the Forms modules, such as FMX, MMX, PLX, and so on. which are on the remote server, they will be able to run these applications using a typical URL previously used when running in a browser.

The use of FSAL supports all the same functionality found when running a form in a browser, except event driven single sign-off. Single sign-on is supported. JavaScript integration support for applications launched using FSAL can be added/enabled with a provided add-on (WJSI) and a third-party library (Eclipse/Jetty). FSAL can be launched using custom protocol handlers from web pages.

FSAL does not rely on Java Deployment technologies—such as Plug-in, Web Start, and so on —and is not dependent on a browser. It does require the Java Runtime on the end-user machine.

Requirements

FSAL requires Oracle Java version Java 17 or newer installed on the user’s machine. These versions are only available in 64-bit Java Development Kit (JDK) distributions.

Java 17 and newer distributions are available for most common end-user platforms and can be used to run FSAL. JDK distributions are generally intended for Java developers and includes far more than a typical user should need. The JDK includes a complete JRE as well as tools for developing, debugging, monitoring, and maintaining Java applications.

You can use any distribution of a supported version as long as it supports running a Java application and was certified with the Oracle Forms version being used. Refer to Oracle Fusion Middleware Supported System Configurations for this Forms version in order to ensure you are using a certified Java version.

The distribution you choose may depend on the application’s needs. It is recommended that the contents of the Oracle Java installation chosen be carefully reviewed and your application be thoroughly tested before moving to production.

Custom JRE Support

Forms also supports using a custom JRE distribution that you create using Oracle Java version 17 or newer. These versions of Java include these utilities for creating a custom distribution:

- The `jdeps` utility: Used to determine which Java modules are needed
- The `jlink` utility: Used to create the custom JRE
- The `jpackage` utility: Used to package your custom JRE into an installable distribution if desired

This guide doesn't cover how to use these utilities to create a custom JRE distribution. Refer instead to your Java documentation for the steps.

Installing the Forms Standalone Launcher

Download the FSAL java JAR file (`frmsal.jar`) and save it to your local drive.

You can download the `frmsal.jar` file from the **Usage/Syntax** web page available with the installation. This utility can be stored anywhere as long as the user has access to that directory and utility. Storing it in the user's home directory is recommended, but not required.

This file is version-specific—you cannot use `frmsal.jar` from one installation against another. For example, you cannot use `frmsal.jar` downloaded from a Microsoft Windows server to run an application against a UNIX/Linux server or vice versa. The launcher is not client- platform specific. It can be used on any platform that supports running a Forms certified Oracle Java version.

> **WARNING:**
>
> Do not change the name of the `frmsal.jar` file. Some functionality in FSAL expects the default filename. Changing the filename may result if some functionality not working correctly.

To download the `frmsal.jar` file:

1. Access the **Usage/Syntax** web page using a URL like this:

   ```
   https://<your server>:<your port>/forms/html/fsal.htm
   ```

2. Click the available download link and save the file to your local drive.

Using FSAL to Run Applications

To run an application using FSAL, you simply have to open a command prompt and provide the required command line arguments, Forms application URL, and other details.

If your needs are more advanced—for example, your app needs to communicate through a proxy server or you want to use custom protocol handlers—refer to the other topics in this section.

Although the Forms Standalone Launcher and the applications it runs are not related to a web browser or its applications, it is possible to communicate with a web page from an FSAL application. For details, refer to Oracle Forms and JavaScript Integration .

- Running an Application with FSAL
- Enabling JavaFX for Java 17+
- Running An Application Through a Proxy Server
- Launching FSAL with a Custom Protocol Handler

Running an Application with FSAL

Launch an application in FSAL using a command prompt.

Knowing the complete Forms application URL is necessary in order to service the request. Currently, the use of URL redirects or rewrites is not supported, but may be technically possible depending on the server configuration. FSAL expects to receive a fully qualified URL that points to the Forms environment.

A desktop shortcut, script, or batch file can be used in place of a hyperlink in order to make starting the application simple and less error prone.

> **Note:**
>
> This utility is not recommended for non-interactive purposes such as unattended load testing. Launching too many simultaneous or sequential sessions on a user's machine may result in unpredictable and undesirable behavior.

To start an application with FSAL:

1. Open a shell such as DOS on Microsoft Windows, then enter this command to verify the desired and certified Oracle Java version is found.

   ```
   java -version
   ```

   The result should indicate the desired Oracle Java version. If not, the system's `PATH` may not be properly set. Make the needed correction then continue.

2. Change to the directory where the `frmsal.jar` is stored, then enter the following command using the required server name and port:

   > **Note:**
   >
   > If the server is not running `SSL` enabled and instead is using `HTTP`, the protocol (`http://`) can be omitted from the URL entry, as it is assumed.

   ```
   java -jar frmsal.jar -url "https://<server>:<port>/forms/frmservlet?
   config=standaloneapp"
   ```

   In this example, the application associated with the configuration section titled `standaloneapp` will run. You can use any configuration section as long as it contains the entries found in the `standaloneapp` example provided in the Forms Web Configuration.

3. If required, create a desktop shortcut, batch script file, or custom executable to make launching the application easier and more seamless.

Oracle Forms generated output, typically seen in the Java Console, appears in the shell used to start the application. If the `javaw` or similar command is used rather than `java`, a console may not be shown. If using the `java` command, closing the shell that started the application will terminate the application.

This behavior can be altered to accommodate the application needs using various shell commands and associated switches. Refer to the operating system documentation for information on using the command shell on the user's platform.

Refer to the **FSAL Usage** page provided in the installation for a complete list of command line arguments. To access this page, enter this URL in a browser:

```
http://<server>:<port>/forms/html/fsal.htm
```

Enabling JavaFX for Java 17+

If you are using Oracle Java 17 or newer "Long Term Support" (LTS) releases to run FSAL, you'll need to enable JavaFX if your applications use the Forms audio feature. JavaFX is necessary for the audio feature to function properly.

If the applications do not use the Forms audio feature, you can skip these steps.

To enable JavaFX, you'll need the JavaFX SDK. You can download this SDK from Gluon's JavaFX page at https://gluonhq.com/products/javafx. Make sure to download the version of the JavaFX SDK that most closely matches the Java version used to run FSAL.

> **Note:**
>
> JavaFX is now an Open-Source project hosted by Gluon. JavaFX versions newer than 8 are supported and licensed by Gluon and/or the open source community. Be sure to review the Terms of Use and support options before downloading and using. Refer to https://openjfx.io and https://github.com/openjdk/jfx.
>
> Gluon is not affiliated with Oracle and JavaFX is not supported by Oracle.

To enable JavaFX on the user's machine:

1. Extract the JavaFX SDK into a directory where the user has runtime (`read`, `execute`) permissions.
2. Alter the typical FSAL startup command to include JFX.

   For example:

   ```
   java --module-path C:/javafx-sdk-17.0.4/lib --add-
   modules=javafx.media,javafx.swing -jar frmsal.jar -url "https://
   <server>:<port>/forms/frmservlet?config=standaloneapp"
   ```

Running An Application Through a Proxy Server

If an application needs to communicate through a proxy server, you can include proxy settings using arguments with the java command when you launch the application in FSAL.

In many cases, users will access a Forms application while within a corporate network. In some cases, this means that the user's machine requires the appropriate proxy configuration in order to access both internal and external content. In the case of using FSAL, the browser and system level settings may not be visible to the shell that launches the application. Therefore, you may need to include such settings at the time FSAL is run.

You can do this by using arguments that either reference the system proxy settings or include the specific proxy settings.

To use the system proxy settings, enter this command to run the application:

```
java -Djava.net.useSystemProxies=true -jar frmsal.jar -url "https://
<server>:<port>/forms/frmservlet?config=standaloneapp"
```

The argument, `-Djava.net.useSystemProxies=true`, causes Java to attempt the call using the proxy settings provided at the system level.

> **Note:**
>
> This method may not be supported when the system is configured to use Automatic Configuration Scripts (such as `wpad.dat`). Refer to the official Java documentation for details.

You can also specifically include proxy settings. Here are two examples depending on the protocol (HTTPS or HTTP):

```
java -Dhttps.proxyHost=<proxyserver> -Dhttps.proxyPort=<proxyserver port
number> -Dhttps.nonProxyHosts="localhost|example.com" -jar frmsal.jar -url
"https://<server>:<port>/forms/frmservlet?config=standaloneapp"
```

```
java -Dhttp.proxyHost=<proxyserver> -Dhttp.proxyPort=<proxyserver port
number> -Dhttp.nonProxyHosts="localhost|example.com" -jar frmsal.jar -url
"http://<server>:<port>/forms/frmservlet?config=standaloneapp"
```

See Java Networking and Proxies.

Launching FSAL with a Custom Protocol Handler

You can use a hyperlink with a custom FSAL protocol to launch a Forms application. FSAL recognizes two special protocols—fsal for non-SSL requests and fsals for SSL requests.

To use a custom protocol, register the custom protocol on the user's machine and include a properly formatted hyperlink on the desired web page.

Using a Custom Protocol in a Hyperlink

Here is an HTML example that uses a hyperlink with a custom protocol that can be used to launch FSAL with a named Forms configuration, standaloneapp:

```
<a href="fsal://<SERVER>:<PORT>/forms/frmservlet?config=standaloneapp">Run
FSAL</a>
```

Registering a Custom Protocol

The steps needed to register this new protocol on the user's machine depend on the operating system. This section provides the steps to register a new protocol on a Microsoft Windows machine. For registering custom protocol handlers on other operating systems, refer to the Operating System documentation or contact the vendor for assistance.

> **WARNING:**
>
> Modifying the Windows Registry can cause irreversible damage if performed incorrectly. Be sure to create a Registry backup and System Restore Point before attempting. Do not attempt this if you are unfamiliar with editing the Windows Registry.

The following commands can either be run individually on a Windows DOS command prompt or executed from a script. Regardless of the approach, the user must have Administrator privileges since this will write to the Windows Registry.

The examples in this section assume:

- The latest Forms frmsal.jar is stored in the user's home directory.

- The latest Oracle Java 8 (32-bit) JRE is installed. Any certified Oracle Java version can be used, however be sure to adjust the path in the examples below to match the location of the java executable.

In the following commands, be sure to carefully review the last entry in each section. Be sure the path to `javaw.exe` and the path to `frmsal.jar` on the user's machine represents your situation. Note that the Java path used is a special path created when installing the JRE only. If using the JDK, this path will need to be updated.

Commands for non-SSL Requests

```
reg add HKEY_CLASSES_ROOT\fsal /t REG_SZ /d "Oracle Forms Standalone
Launcher" /f
reg add HKEY_CLASSES_ROOT\fsal /v "URL Protocol" /t REG_SZ /d "" /f
reg add HKEY_CLASSES_ROOT\fsal /v "UseOriginalUrlEncoding" /t REG_DWORD /d
"00000001"
reg add HKEY_CLASSES_ROOT\fsal\shell /f
reg add HKEY_CLASSES_ROOT\fsal\shell\open /f
reg add HKEY_CLASSES_ROOT\fsal\shell\open\command /t REG_SZ /d "\"C:\Program
Files (x86)\Common Files\Oracle\Java\javapath\javaw.exe\" -jar %USERPROFILE%
\frmsal.jar -url \"%1\"" /f
```

Commands for SSL Requests

```
reg add HKEY_CLASSES_ROOT\fsals /t REG_SZ /d "Oracle Forms Standalone
Launcher with SSL/TLS" /f
reg add HKEY_CLASSES_ROOT\fsals /v "URL Protocol" /t REG_SZ /d "" /f
reg add HKEY_CLASSES_ROOT\fsals /v "UseOriginalUrlEncoding" /t REG_DWORD /d
"00000001"
reg add HKEY_CLASSES_ROOT\fsals\shell /f
reg add HKEY_CLASSES_ROOT\fsals\shell\open /f
reg add HKEY_CLASSES_ROOT\fsals\shell\open\command /t REG_SZ /d "\"C:\Program
Files (x86)\Common Files\Oracle\Java\javapath\javaw.exe\" -jar %USERPROFILE%
\frmsal.jar -url \"%1\"" /f
```

Once the above have been successfully executed, the protocols `fsal://` or `fsals://` can be used in a browser in place of `http://` or `https://`.

File Caching

Like the Java Plugin or a browser, FSAL caches files that may be reused the next time an application is launched to improve startup performance.

Caching Messages

When starting FSAL from a Unix/Linux shell or Windows DOS shell, the directory where cached files are stored is displayed in the shell output during the loading process. Also displayed is whether the files are being downloaded from the server or reused from the existing cache.

If downloading new files (not cached), you will see a message like this:

```
Inspecting archive files in cache directory C:\Users\<user
name>\AppData\Local\Temp\frmsal\<server name>\14.1.2.0
Downloading archive file frmall.jar to cache subdirectory 8ymuqdqvdfe13a0d5vvema0dh
```

If previously cached files are used, you will see a message like this:

```
Inspecting archive files in cache directory C:\Users\<user
name>\AppData\Local\Temp\frmsal\<server name>\14.1.2.0
Using cached archive file frmall.jar from cache subdirectory 8ymuqdqvdfe13a0d5vvema0dh
```

Changing `tmpdir`

To control the location of the cache files, change the location of `tmpdir` when starting the application. Consider the following example for Microsoft Windows users. Note that it uses the value of the Windows system variable `%USERNAME%` as the first level directory.

```
java -Djava.io.tmpdir=C:/%USERNAME%/fsal -jar frmsal.jar -url "https://
<server>:<port>/forms/frmservlet?config=standaloneapp"
```

> **Note:**
>
> When the user is on a shared Unix/Linux platform this may cause problems because the downloaded cached files will be owned by the first user who ran the application. As a result, subsequent users may not have sufficient permissions to overwrite the old cached files with new ones. By creating unique locations for each user this problem can be avoided.
>
> ```
> java -Djava.io.tmpdir=/u01/$user/fsal -jar frmsal.jar -url "https://
> <server>:<port>/forms/frmservlet?config=standaloneapp"
> ```

Disabling File Caching

For some applications, you may not want to use cached files, but instead always download from the server. This may also be true when troubleshooting technical issues.

To disable file caching:

1. Open Fusion Middleware Control (FMC), then edit the FSAL template file (`basesaa.txt` or `webutilsaa.txt` based on your application) to include this line:

   ```
   ignoreSaaCache=%ignoreSaaCache%
   ```

2. In the Forms Web Configuration, add a new parameter `ignoreSaaCache` to the `[standaloneapp]` configuration or whichever configuration is being used to run the application.
3. Set its value to `TRUE`.

Cached files will now be ignored and downloading files from the server will occur each time the application is started.

Security Considerations

Review the topics in this section when planning and configuring FSAL, the user's machine, and your network.

Ensuring the highest degree of security for an application, the data exchanged within it, and the network on which it is hosted should be considered the most important aspects of any application deployment. A weakened security layer can be responsible for sensitive data breaches and malicious system attacks. It is your responsibility to ensure that proper security efforts are being used to protect the applications, the data, and their hosting systems.

> **Note:**
>
> This section offers several ways you can improve the security related to using the FSAL, but these are only examples. It is your responsibility to understand the concepts and research any that you do not fully understand. Improperly implementing any security configuration may put your system at risk.
>
> After configuring your system, carefully review and test your changes before assuming they are correct.
>
> The suggestions provided here may not be unique to FSAL or even Oracle Forms in general. Many are standard suggestions that relate to the technologies in use. Therefore, the availability of additional information is widespread. Do not use this section as the only source of information for securing your application, its data, and its environment.

- Secure Socket Layer
- Configuring SSL/TLS with FSAL
- Troubleshooting SSL/TLS
- Signed Code

Be sure to also review the *Resources* section included at the end of this section.

Secure Socket Layer

Using Secure Socket Layer (SSL) and Transport Layer Security (TLS) to run any application communicating on the network is very important to securing data. The use of `SSL`/`TLS` for running any and all applications should be considered a requirement and not optional.

Secure Socket Layer (SSL) and Transport Layer Security (TLS) are cryptographic protocols used to provide encrypted communication between a source and destination of network traffic. These protocols work by creating a trusted connection, which is most often established by a public and private key exchange.

To run an application using `SSL`/`TLS` with FSAL requires that the `SSL`/`TLS` certificate public key be included in the user’s Java TrustStore. This may require you to import the certificate(s) if that has not already been included in the TrustStore or provided by a known Certificate Authority such as DigiCert, Entrust, or Comodo.

> **Note:**
>
> The terms “KeyStore” and “TrustStore” are used interchangeably in this section.

This key must be imported into the Java TrustStore that is being used by FSAL to run the form. While often only the “root” certificate is needed, in some cases more than one certificate may need to be imported, as a chain of certificates may exist (for example root, intermediate, user).

If you try to run an application using `SSL/TLS` and a needed certificate is not found or was improperly imported, you may see a Java error like one of these

```
java.security.cert.CertificateException: No subject alternative names present …

java.security.cert.CertificateException: No name matching <server>:<port> found …

javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: …
```

Following one of the above or similar errors, you will see an FSAL error:

```
FRM-92490: Unable to fetch applet parameters from server.
```

This is because a secure connect could not be established and the parameters needed to start and run the application were not permitted to be downloaded.

Configuring SSL/TLS with FSAL

Import and store SSL/TLS certificates using either the FSAL Certificate Importer or the Java provided tools. The FSAL Certificate Importer is the default method.

Using The FSAL Certificate Importer

On receiving initial feedback from the server, FSAL checks the TrustStore to determine if the delivered certificate is known. If known and deemed trusted, the application runs without interruption.

If the certificate is unknown, the user sees a prompt indicating this and allowing the user to import a new certificate. This prompt includes a **More Information** link that provides details about the delivered certificate.

If the user clicks **OK**, the new certificate is inserted into a newly created Java TrustStore used by Forms/FSAL. This FSAL created TrustStore is used by FSAL for all subsequent requests unless disabled by the user by setting the runtime switch `cert_truststore` to "`java`".

This new Java TrustStore is created in the user’s home directory, specifically in:

```
C:\Users\<USER NAME>\Oracle\forms
```

The TrustStore file name is `formstruststore`. Changing the name of the generated TrustStore is not supported. If the file is not found, a new empty file is created. Because this is a Java generated and standard TrustStore file, the tools found in the Oracle Java installation (`keytool.exe`) can be used to administer this file if necessary.

When running an application, you can use the `autoImportCert` switch to import unknown certificates automatically. If set to true, the user will not see the prompt before the certificate is imported. This option is not recommended and is provided mostly for testing purposes. Here is an example of how to use it:

```
java -jar frmsal.jar -url "https://<server>:<port>/forms/frmservlet?
config=standaloneapp" -autoImportCert true
```

For additional information, refer to the FSAL command line arguments listed on the Usage web page. The list of command line arguments is also included here: FSAL Command Line Arguments.

Configuring the KeyStore Manually

Refer to the command line options provided near the end of this document for details on reverting to the previous behavior.

If using the FSAL SSL/TLS Certificate Importer is not desirable, follow these steps prior to using FSAL. It may also be necessary to complete these steps if a compatibility issue is identified with the certificate(s) in use and the FSAL Certificate Importer. Some non-standard or self-generated certificates may not be compatible with the FSAL Certificate Importer.

1. Obtain an SSL/TLS certificate from a known Certificate Authority and configure Oracle HTTP Server (or WebLogic Server) according to the instructions provided by the documentation for those components.
2. Obtain the public key portion of the certificate.
   The public key portion of the certificate is be needed on the user's machine. If you do not have the public key, there are several ways to obtain it. Obtaining the key chain can be done from any machine that has access to the server. Here is an example of how to obtain the key(s) using the `openssl` command. Make sure to replace <*server*>:<*port*> with the server name and SSL/TLS port number.

   ```
   openssl s_client -showcerts -connect <server>:<port> > output.txt
   ```

   > **Note:**
   >
   > This is pre-installed on most Unix/Linux platforms, but can be obtained for Microsoft Windows.

   The result of running this command will be saved in a file named "output.txt". The file will contain one or more certificates (depending on the certificate type and how it was created). The certificate is the contents between and including the BEGIN and END header/footer.
3. Copy each certificate to its own text file.
   Be sure to include the BEGIN and END text exactly as shown below and the contents between them. Do not include extra lines above or below the **-----BEGIN CERTIFICATE-----** or **-----END CERTIFICATE-----** when saving the file, but do include these header/footer entries.

   Here is an example:

   ```
   -----BEGIN CERTIFICATE-----
   MIIFKDCCBBCgAwIBAgIBPTANBgkqhkiG9w0BAQsFADCBrjEpMCcGA1UEAxMgTmlj
   Y2subWFuc0BvcmFjbGUuY29tMRAwDgYDVQQLEwdTdXBwb3J0MQ8wDQYDVQQKEwZP
   cmFjbGUxGTAXBgNVBAcTEENvbG9yYWRvIFNwcmluZ3MxETAPBgNVBAgTCENvbG9y
   . . .
   -----END CERTIFICATE-----
   ```

> **Tip:**
>
> Alternatively, run the following to directly generate the needed certificate file:
>
> ```
> openssl s_client -showcerts -servername <server name> -connect
> <server>:<port> | openssl x509 -outform PEM > cert.cer
> ```

4. Import the public key(s) into the KeyStore on the user's machine (for example, `cacerts`). If it is a certificate chain, be sure to import all keys in the chain (signer, intermediate, root, and so on). This means it may be necessary to run the keytool utility more than once. To import the certificate public key(s) use the Java keytool utility, which is included in most Oracle Java distributions.

   Here is an example of how import each certificate using the keytool utility. Refer to the Oracle Java documentation for details on how to use the keytool utility.

   ```
   keytool -importcert -alias <server_name> -keystore <JAVA_HOME>/jre/lib/
   security/cacerts -file cert.cer
   ```

   > **Note:**
   >
   > The alias must be unique. If there are more than one certificate in the chain each must have a unique alias. However, the root should be imported first and should use the server's name as its alias. Attempting to use an alias that was previously used will result in an error. It is recommended that the server's name be used in the alias. For example, the root certificate should be imported first and should be named `myserver`, the second `myserver2`, and the third `myserver3`.

5. Run FSAL with SSL, including the switch, `-cert_truststore`, set to java in order to bypass the automatic certificate importer:

   ```
   java -jar frmsal.jar -url "https://<server>:<port>/forms/frmservlet?
   config=standaloneapp" -cert_truststore java
   ```

Troubleshooting SSL/TLS

If you encounter problems with your certificates, try these steps where appropriate.

- If certificates were manually inserted into the Java Keystore, verify the correct KeyStore was updated.

  By default, a KeyStore is found in `JRE_HOME/lib/security` or in the user's home directory. The default file name (when not using the FSAL Certificate Importer) is `cacerts`. Verify that the file modified date/time is the same as when the import command was executed.

  When using the FSAL Certificate Importer the TrustStore name is `formstruststore`.

- Verify that all certificates in the chain have been imported.

In many cases, only the root certificate is needed. If it is believed that there is a certificate chain, all the certificates can be listed for review using this Java command:

```
keytool -list -keystore "C:\java\jdk\jre\lib\security\cacerts"
```

- If the KeyStore or TrustStore updated is not in the `JRE_HOME` or user home directory, copy it to the `JRE_HOME`. Be sure to create backup files of any files before overwriting.
- Enable Java SSL/TLS debugging.

  To enable debugging mode, run FSAL and the desired form as follows:

  ```
  java -Djavax.net.debug=all -jar frmsal.jar -url https://<server>:<port>/
  forms/frmservlet?config=standaloneapp
  ```

  The output can also be redirected to a text file as follows (assumes Microsoft Windows):

  ```
  java -Djavax.net.debug=all -jar frmsal.jar -url "https://<server>:<port>/
  forms/frmservlet?config=standaloneapp" > C:/existing_directory/output.txt
  ```

- If the KeyStore or TrustStore that contains the needed certificate(s) is not the default (in the default location), the `keystore` or `truststore` switch can be used to explicitly reference the desired file.

  ```
  java -Djavax.net.ssl.keystore=C:/somewhere/mycacerts -jar frmsal.jar -url
  "https://server:port/forms/frmservlet?config=standaloneapp"
  ```

  or

  ```
  java -Djavax.net.ssl.truststore=C:/somewhere/mycacerts -jar frmsal.jar -
  url "https://server:port/forms/frmservlet?config=standaloneapp"
  ```

Signed Code

For Oracle Forms, a digital signature certificate is added to the Oracle-provided Forms Java JAR files. This digital signature helps to ensure that the application owner is known and trusted by the user.

Forms, Java Plug-in, and Java Web Start

To run a Java application from a browser with the Java Plug-in or Java Web Start, the user would likely click a hyperlink or bookmark or manually type a URL into a browser. This can potentially put the user's system at risk. To protect against a link opening a malicious Java application on the user’s machine, the Java Plug-in and Java Web Start have special security features, including a requirement to only permit signed (and trusted) applications to run. Unsigned applications are not permitted and are blocked.

All Java JAR files provided by Oracle Forms that run on the user’s machine are signed in order to comply with Java’s security requirements. This allows Forms applications to run safely with Java Web Start. It is further required that custom JAR files (those not provided by Oracle) also be properly signed with a trusted certificate. Signing JARs not provided by Oracle is the customer’s responsibility.

Note:

> Using self-generated certificates is not recommended and should be avoided. Although using self-generated certificates may seem to function, support for this functionality may be removed from Java in the future.

Refer to the `jarsigner` utility and its documentation for details on how to add a certificate to your own JAR files.

FSAL and Signed JARs

JAR signing is recommended for all deployments, including when using FSAL. Since the inherent risk associated with using a browser is mostly avoided when using FSAL, this client configuration is quickly becoming the preferred approach. That said, you should still carefully consider all security aspects when planning application deployment.

Although running native Java applications like FSAL do not have the same level of built-in security checks found in the Java Plug-in and Java Web Start, limited levels of signed code verification can be enabled.

Note:

> The following has been deprecated in Java versions newer than version 11. Warnings may appear if enabling this functionality using Java 17 to Java 23. As of Oracle Java 24 or newer, this functionality will be desupported and the ability to execute the functionality will be removed. These instructions are provided for legacy purposes only.

To enable signed code awareness for FSAL and specifically your application:

1. Obtain the public keys associated with all JAR files used in the application.
   Oracle provided JARs are signed when delivered and this certificate is included in the KeyStore by default. If the signer's public key for the JAR files used is not available, use the following Java command for each of the JARs to obtain their certificates:

   ```
   keytool -printcert -rfc -jarfile <yourJarFile>.jar
   ```

   You should see output like this in response:

   ```
   -----BEGIN CERTIFICATE-----
   MIIG7zCCBNegAwIBAgIQCl39nB4pDWNpkID7lx+kLjANBgkqhkiG9w0BAQsFADBp
   MQswCQYDVQQGEwJVUzEXMBUGA1UEChMORGlnaUNlcnQsIEluYy4xQTA/BgNVBAMT
   ...
   2ROzDhwQmGHIbDXUo8jXX7NGHdpyZovPm//IO/zp9d6Anka/B3viZlii4eAz6Xbv
   yIafqfh5fS9ZLb4kSHKgf87f4Zlm8YElQF+ZJ+rGVT5xArbEjxcoLyzpmmLWCPeI
   yTifK4P1GgeEhdWol7/R25bAKyPr9OrySnQx9vnUbmKa120i13/mhJm2gXp7qG+P
   uxx1l3SL/yU8gbmOdRno6CcANWLUvQ5FjMVppV8FVPf+7gyhsKrAYP2SMNy2F0qi
   1OIF
   -----END CERTIFICATE-----
   ```

> **Note:**
>
> The first certificate returned should be the "signer" certificate and the one needed for the steps that follow.

2. Copy the contents of the first certificate into a text file (`foo.pem` in this example). The copied contents should begin with "`-----BEGIN CERTIFICATE-----`" and end with "`-----END CERTIFICATE-----`".

3. On the user's machine, import each public key into the Java KeyStore that will be used at runtime:
   `keytool -importcert -alias `*`foo`*` -file C:\foo.pem`

   The above alias can be any alphanumeric string, however be sure to remember it as it will be needed in a coming step. Also, because the `-keystore` switch was not included, a KeyStore in the user's home directory is used (or created if one does not exist). The file will be named ".keystore". To use a different KeyStore and/or KeyStore name, add the `-keystore` switch as desired.

4. Repeat the above step for each certificate.

5. Create a `.java.policy` file in the user's home directory if one does not exist or if you want a fully customized file. Otherwise, use the default file.
   `<JAVA_HOME>``\jre\lib\security\java.policy`
   Add the following to it:

   ```
   keystore ".keystore";
   grant signedBy "foo" {
      permission java.security.AllPermission;
   };
   ```

   The "foo" reference is a pointer to the alias used in step 2. Add additional `grant signedBy` sections for each key that was imported in step 2. The value of keystore is a pointer to the KeyStore file you expect to use.

6. Run your Forms application using FSAL with this slightly modified command entry:
   ```
   java -Djava.security.manager -jar frmsal.jar -url "https://
   yourserver:port/forms/frmservlet?config=standaloneapp"
   ```

When running the application, JAR certificates are matched with those in the KeyStore. If they do not match, the app will not run. In the above example, it is assumed that the default `java.policy` is used or a `.java.policy` was created in the user's home directory.

```
java -Djava.security.manager -Djava.security.policy=${user.home}/
foo.policy -jar frmsal.jar -url "https://<server>:<port>/forms/
frmservlet?config=standaloneapp"
```

You can also access a custom policy file from another directory on the user's machine or a remote location (URL).

```
java -Djava.security.manager -Djava.security.policy=https://
<server>:<port>/foo.policy -jar frmsal.jar -url "https://
<server>:<port>/forms/frmservlet?config=standaloneapp"
```

> **Note:**
>
> To ensure users are taking advantage of this validation, a startup script or similar should be used. Using a script will help prevent the possibility of typographical errors or the possibility of not using the security manager.

Single Sign-on

Single Sign-on is an authentication concept that offers many valuable advantages to web applications. The idea of integrating Forms with Single Sign-on (SSO) is often believed to have a significant benefit over using the Forms database login functionality because database credentials for running Forms applications can be hidden from the user.

To use SSO with FSAL you must perform the same configuration steps needed for using SSO with Forms in any other case. For information about how SSO is configured with Forms and how it works, see Using Forms Services with Oracle Access Manager.

To enable SSO for a specific application, set `ssoMode=true` in the desired application configuration of the Forms Web Configuration settings associated with the application to be protected.

FSAL Reference

Refer to this section for information on FSAL parameters and command line arguments. You'll also find links to related documentation.

Topics:

- FSAL Web Configuration Parameters
- FSAL Command Line Arguments

FSAL Web Configuration Parameters

Refer to this topic for details about FSAL servlet and applet parameters.

Table 8 FSAL Web Configuration Parameters

Servlet/Applet Parameter	Description
`ignoreSaaCache`	Specifies whether to ignore the existing cache and download fresh files from the server. Valid values: `TRUE`, `FALSE` Default: `FALSE`

Table 8 (Cont.) FSAL Web Configuration Parameters

Servlet/Applet Parameter	Description
`ignoreMissingSaaArchives`	Specifies whether to ignore configured but missing JAR files at startup. When set to `TRUE`, any occurrence of a missing archive results in an error but the session tries to continue. This is the default behavior. When set to `FALSE`, FSAL tries to continue, if possible, but fails if not. **Note:** This should generally only be used for testing and debugging. Valid values: `TRUE`, `FALSE` Default: `FALSE`
`fsalEnableAutoUpdate`	Specifies whether FSAL should automatically update itself if it does not match the server version. Auto update is only supported for patch updates. Valid values: `TRUE`, `FALSE` Default: `TRUE`
`fsalJavaVersion`	Specifies the required Java version to run the application. See `fsalJavaVersion` in Web Configuration Parameters for possible value combinations.
`fsalUpdateDialogText`	Specifies the text to display in the AutoUpdate dialog when it is presented. Valid value: A string with maximum length less than 128 characters
`ssoSaaBrowserLaunchTimeout`	Specifies, in seconds, how long the Forms servlet will wait for the initial request from the browser that was launched by FSAL for SSO authentication. If the interval expires, a fatal error `FRM-93249` is reported. Valid values: Integers in the range 1-300. Default: `15`
`ssoSaaBrowserPageTimeout`	Specifies, in seconds, how long the Forms servlet will wait for the user to enter data into a browser page during SSO authentication for an FSAL application. If the interval expires, a fatal error `FRM-93382` or `FRM-93383` is reported. Valid values: Integer >= 15, or 0 (wait indefinitely). Default: `0`

Table 8   (Cont.) FSAL Web Configuration Parameters

Servlet/Applet Parameter	Description
`ssoSaaWaitInterval`	Specified the interval, in seconds, after which FSAL reissues requests to the Forms servlet while SSO authentication is proceeding in the launched browser. Larger values reduce network traffic but increase the chances of a timing out, producing a fatal error `FRM-93248`). Valid values: Integer >= 5, or 0 (do not reissue requests). Default: `25`
`ssoSuccessLogonURL`	The URL to redirect to if SSO authentication completes successfully for an FSAL application.

FSAL Command Line Arguments

Refer to this topic for details about FSAL command line arguments.

Table 9   FSAL Command Line Arguments

Argument	Description
`-url`	Specifies the fully qualified URL needed to run the Forms application. The URL should include the necessary configuration reference that includes the needed FSAL settings (for example, `config=standaloneapp`). The URL must be surrounded by quotation marks. **Note:** When not using SSL/TLS, the protocol `http://` can be omitted from the URL entry.
`-t`	Specifies, in milliseconds, the amount of time the launcher should wait for the server to provide its initial response before timing out. Valid values: Positive integers (>= 1) Default: `60000` (ms)
`-showConfig`	Specifies if the Forms web configuration parameters should be display on command line when the application is loaded. Valid values: `TRUE`, `FALSE` Default: `FALSE`

Table 9 (Cont.) FSAL Command Line Arguments

Argument	Description
`-showDetails`	Specifies whether to display additional details on application loading and startup. Valid values: `0`, `1`, `2`, `99` Value descriptions: • `0`: No additional information should be displayed. • `1`: Show location from where resources are loaded. • `2`: Show SSL/TLS related information when certificate information is not in the FSAL TrustStore. • `99`: Shows all the details. **Note:** Other values are reserved for future use. Default: `0`
`-changeFSALStorePass`	Specifies whether to change the password of FSAL custom TrustStore. A default password is set at creation time. That password is: `changeit` In most cases, there is little reason to set a password for the truststore on a user's machine unless it contains private key/certificate information. Valid values: `TRUE`, `FALSE` Default: `FALSE`
`-autoImportCert`	Specifies whether the SSL/TLS certificates should be imported without any user interaction. This argument is ignored if `cert_truststore` is set. Valid values: TRUE, FALSE Default: FALSE

Table 9 (Cont.) FSAL Command Line Arguments

Argument	Description
`-cert_truststore`	Specifies whether to use the Forms TrustStore or a custom KeyStore for storing SSL/TLS certificates. A value of "forms" or unset will result in using a Forms generated Java TrustStore. A value of "java" will cause FSAL to look in the Java default KeyStore for storing and verifying SSL/TLS certificates. **Note:** When using the Java default store, you must manually import needed certificates into it before running any SSL/TLS application. Valid values: `forms`, `java` Default: `forms`
`-bypassHostnameVerification`	Specifies whether hostname verification failures are ignored when validating SSL/TLS certificates. Setting this to `TRUE` allows the application to run even though the connection cannot be verified. This is inherently insecure and is strongly discouraged. This option is ignored if the `cert_truststore` option is set to "java". Valid values: `TRUE`, `FALSE` Default: `FALSE`
`-clearcache`	Specifies whether to clear the FSAL cache. The `–url` argument is ignored if used with `–clearcache`. Valid values: `TRUE`, `FALSE` Default: `FALSE`

Java Web Start

Java Web Start is considered a semi-browserless configuration.

The use of Java Web Start (JWS) gives an Oracle Forms application the appearance of a natively-installed application rather than a web app. This is because when running, the application is not contained within 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. Because this is typically a browser-less configuration,single sign-off events are not supported when using this configuration.

JavaScript integration support for JWS does not exist by default. However, JS integration can be enabled with a provided add-on and a third party library (Eclipse/Jetty). For more information, refer to Forms and JavaScript Integration for Java Web Start and Forms Standalone Launcher.

Unlike the functionality provided by the use of `separateFrame=true` (available in the JNLP Embedded in HTML and Java Applet Embedded in HTML configurations only), the use of Web Start allows you to close the browser window used to call the application after it has started or not use a browser at all. 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 fail.

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

To use this configuration, 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"
```

> **Note:**
>
> The `jnlp` and `jnlps` protocols used in the examples are supported with Java 8u92+ on Microsoft Windows. Also, SSO is not supported when launching with the `JNLP://` or `JNLPS://` protocols.

To implement this configuration, configure these parameters in the Forms Web Configuration page of Fusion Middleware Control.

Table 10   Parameters for Java Web Start

Type	Parameters
Non-WebUtil Enabled Forms	• `basejnlp=base.jnlp` • `webstart=enabled`
WebUtil Enabled Forms	• `basejnlp=webutil.jnlp` • `webstart=enabled`

Note:

When using Java Web Start, if the application uses custom jar files, for example `jacob.jar`, `icons.jar`, or `example.jar` add these to the `extensions.jnlp` file. The file is located in the `ORACLE_HOME\forms\java` directory. Open this file in a text editor and add the needed entries based on the example included in the file. Each entry should be added on its own line.

Java Applet Embedded in HTML

This is the default configuration and gives the appearance that the Forms application (applet) is embedded in a web page.

Note:

Although Forms is shipped with the components needed to allow Forms applications to run as an embedded applet in some browsers (such as MS Edge with IE mode), this configuration is no longer recommended for use.

This helps 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 supported in this configuration.

To use this configuration, you require a Java Plug-in and a certified browser that supports the Java Plug-in.

Enter a URL in the browser, such as `https://example.com/forms/frmservlet` or `https://example.com/forms/frmservlet?config=default`.

To implement this configuration, configure these parameters in the Forms Web Configuration page of Fusion Middleware Control.

Table 11 Parameters for Java Applet Embedded in HTML

Type	Parameters
Non-WebUtil Enabled Forms	• `baseHTML=base.htm` • `baseHTMLjpi=basejpi.htm`
WebUtil Enabled Forms	• `baseHTML=webutilbase.htm` • `baseHTMLjpi=webutiljpi.htm`

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.

> **Note:**
>
> Although Forms is shipped with the components needed to allow Forms applications to run as an embedded applet in some browsers (such as MS Edge with IE mode), this configuration is no longer recommended for use.

Similar to an embedded Applet, embedded JNLP 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 (client side HTML source). This content includes most of the parameter and value pairs configured for the application. Because the base-64 encoded text is not human readable, it deters end-users from reading the parameters. Because much of the content is base-64 encoded, start up performance is slightly improved.

> **Note:**
>
> Base-64 encoding is not a security mechanism.; it is used by Java and helps to improve delivery performance from server to client.

For this configuration, you require a Java Plug-in and a certified browser that supports the Java Plug-in.

To use this configuration, either use the provided example configuration (named `jnlp`) or create your own.

```
https://example.com/forms/frmservlet?config=jnlp
```

To implement this configuration, configure these parameters in the Forms Web Configuration page of Fusion Middleware Control.

Table 12 Parameters for JNLP Embedded in HTML

Type	Parameters
Non-WebUtil Enabled Forms	• `basejnlp=base.jnlp` • `baseHTMLjpi=basejpi_jnlp.htm`
WebUtil Enabled Forms	• `basejnlp=webutil.jnlp` • `baseHTMLjpi=basejpi_jnlp.htm`

> **Note:**
>
> When using Embedded JNLP, if the application uses custom jar files, for example `jacob.jar`, `icons.jar`, or `example.jar` add these to the `extensions.jnlp` file. The file is stored in the `ORACLE_HOME\forms\java` directory. Open this file in a text editor and add the needed entries based on the example included in the file. Each entry should be added on its own line.

Browser Considerations

The following links provide information about client browser, Java version and the latest supported platforms.

- For product certification requirement documents, see the Oracle Fusion Middleware Supported System Configurations page.
- For system requirement information, see Oracle Fusion Middleware System Requirements and Specifications document.
- For Oracle's general browser policy, see the Oracle Software Web Browser Support Policy page.

> **Note:**
>
> Internet Explorer and Microsoft Edge with IE-Mode are the only Forms-certified browsers that support the use of the Java Plugin. This plugin is needed to run Forms applications that are embedded in a web page. Because Internet Explorer is no longer supported by Microsoft, it is not recommended for use any longer.
>
> All Forms-certified browsers can be used for launching applications configured to use Java Web Start. This is the recommended configuration if using a browser is desired. Alternatively, the Forms Standalone Launcher can be used.

Part IV
Administration

This part describes how to configure and manage Forms services. To configure deployment of Forms applications and perform most management tasks for a Forms instance, use Fusion Middleware Control. Configuring and managing Forms Services also includes managing and configuring environment variables; URL security for applications; Fonts, Icons, Images used by Forms Services; language detection; key mappings and others.
This part contains the following chapters:

- Fusion Middleware Control and Oracle Forms
- Configuring Forms Services
- Managing Environment Variables
- Managing User Sessions
- Creating Your Own Applet Definition Template Files
- Deploying Fonts, Icons, and Images
- Enabling Language Detection
- Enabling Key Mappings
- Oracle Forms Configuration Helper Script

> **Note:**
>
> The Oracle Forms Services installation includes numerous tools intended to be used for administering and controlling the environment. Except where instructed otherwise, these tools should be used when configuration changes are necessary. Similar is true when starting or stopping any of the servers associated with the WLS Domain. Attempting to make configuration changes directly (manually) within any of the configuration files or attempting to control the servers using operating system commands is not recommended. Doing so can result in configuration inconsistencies or corruption that may cause unpredictable behavior or an inability to restart or stop servers.

Fusion Middleware Control and Oracle Forms

Fusion Middleware Control is a Web-based tool that is launched from a web browser.

The default URL for Fusion Middleware Control is `http://<example.com>:7001/em`

Use the Web-based Oracle Fusion Middleware Control to:

- Monitor metrics for a Forms Services instance, as described in Monitoring Forms Services Instances.
- Manage user sessions, as described in Managing User Sessions

- Configure parameters for a Forms Services instance, as described in Configure Parameters with Fusion Middleware Control.
- Configure Forms Trace and monitor trace metrics, as described in Enable and Configure Forms Trace and Taking Advantage of Oracle Diagnostics and Logging Tools.
- Configure multiple environment files, as described in Managing Environment Variables.
- Configure and use JVM pooling, as described in Managing JVM Pooling from Fusion Middleware Control.

Accessing Forms Services with Fusion Middleware Control

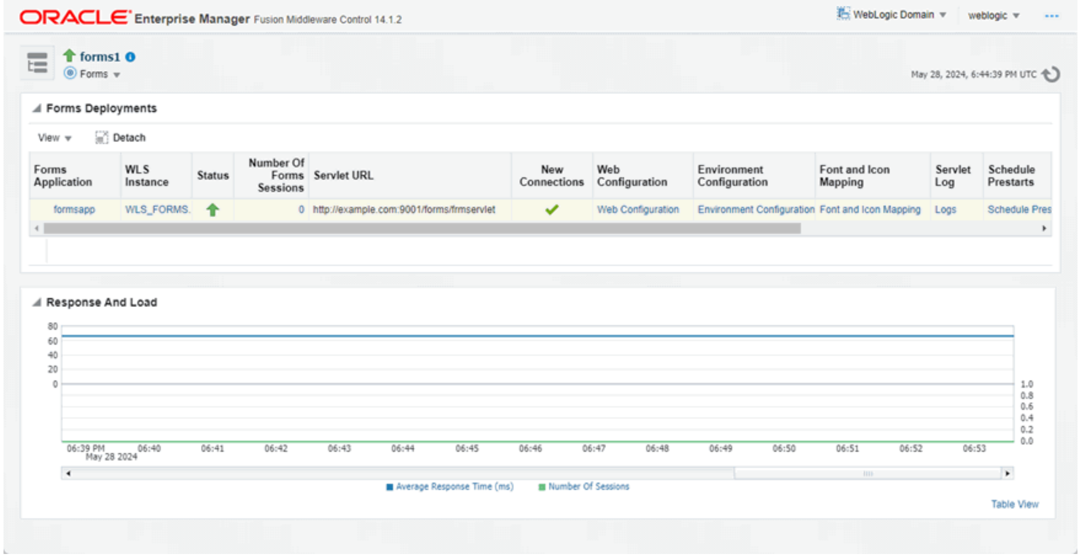
To perform most management tasks for a Forms instance using Fusion Middleware Control, start by navigating to the Forms home page in Fusion Middleware Control.

For introductory information about using Fusion Middleware Control, see Overview of Oracle Fusion Middleware Administration Tools in the *Administering Oracle Fusion Middleware* guide.

To navigate to the **Forms Home** page in Fusion Middleware Control:

1. Navigate to the home page for the Fusion Middleware Control that contains the Forms instance you want to manage.
2. Expand the navigation side panel and expand the Forms node. Click the link for the Forms instance you want to access.

Figure 7   Forms Home page



Forms Application	WLS Instance	Status	Number Of Forms Sessions	Servlet URL	New Connections	Web Configuration	Environment Configuration	Font and Icon Mapping	Servlet Log	Schedule Prestarts
formsapp	WLS_FORMS.	↑	0	http://example.com:9001/forms/frmservlet	✓	Web Configuration	Environment Configuration	Font and Icon Mapping	Logs	Schedule Pres

3. The Forms Home page provides information on the Forms applications that are deployed on the Forms instance.

The information displayed on the Forms Home page, is describes in Table 13.

Table 13   Forms Deployment Fields

Field	Description
Forms Application	Lists the names of the Forms applications that are deployed on the Oracle WebLogic Server instance. Click the name to view the Forms application home page.
WLS Instance	Name of Oracle WebLogic Server instance where the application is deployed.
Status	Indicates the status of the forms application. A green up arrow indicates the application is running. A red down arrow indicates the application is not started.
Number of Forms Sessions	Displays the number of active forms sessions.
Servlet URL	Displays the URL for the Forms servlet.
New Connections	Indicates whether new connections are enabled or not.
Web Configuration	Link to the Web Configuration page.
Environment Configuration	Link to the Environment Configuration page.
Fonts and Icon Mapping	Link to the Fonts and Icon Mapping page.
Servlet Logs	Link to the Servlet Logs.
Prestart Scheduling	Link to the Prestart scheduling page.
Advanced	Link to the Advanced Configuration page.

To access the Forms Menu in Fusion Middleware Control:

1. Navigate to the Forms home page in Fusion Middleware Control.
2. Click **Forms** on the top left. This displays the Forms Menu. Table 14 lists the Menu Selections that are available in the Forms Menu.

Table 14   Forms Menu Options

Select	To Display
Home	Forms Home page. This page displays a list of the Forms deployments and their details. This page also displays the Response and Load statistics and a set of useful links in the Resource Center.
Monitoring - Performance Summary	Performance Summary page. This page displays a set of default performance charts that show the values of specific performance metrics, see Monitoring Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.
Monitoring - Servlet Log	Log Messages page. Oracle Fusion Middleware components generate log files containing messages that record all types of events.
JVM Controllers	JVM Controllers page. This page manages the JVM controller for the Forms instance.
Schedule Prestart	Prestart scheduling page. This page manages Forms prestart scheduling.
User Sessions	User Sessions page. This page monitors and traces User Sessions within a Forms instance.
Web Configuration	Web Configuration page. This page configures deployment of Forms applications and manage configuration sections and parameters in `formsweb.cfg`.

Table 14 (Cont.) Forms Menu Options

Select	To Display
Trace Configuration	Trace Configuration page. This page manages the settings used for tracing of user sessions.
JVM Configuration	JVM Configuration page. This page modifies the JVM controllers that can be subsequently spawned for the Forms instance.
Environment Configuration	Environment Configuration page. This page manages environment variables that define environment settings for Forms run time.
Advanced Configuration	Advanced Configuration page. This page allows you to edit a Forms configuration or template file and apply the settings.
Fonts and Icons Mapping	Fonts and Icons Mapping page. This page helps to change, add, or delete parameters in the Registry.dat file.
Security	Displays information about the following: • **Forms OPSS Resource Administration**: This page administers the Oracle Platform Security Services (OPSS) resources. • **Forms LDAP Association**: This page administers the Forms LDAP resources. • **Forms Runtime LDAP Association**: This page helps to associate and disassociate a forms deployment with an Oracle Internet Directory host to enable Single Sign-On functionality. • **Resource Migration**: This page helps in migrating resources from LDAP to OPSS.
General Information	Displays information about the Target Name, Version, Oracle Home, and Host.

> **Note:**
>
> For the pages that include a **Help** icon, click the **Help** icon to access the page-level help. The page-level help describes each element in the page.

Configuring Forms Services

Use the **Web Configuration** page in Fusion Middleware Control to configure deployment of Forms applications by modifying `formsweb.cfg`.

When modifying any configuration files managed by Fusion Middleware Control (FMC), ensure you edit the files from within FMC. Do **not** modify the files using any external text editor. Doing so may result in the changes not being saved or lost.

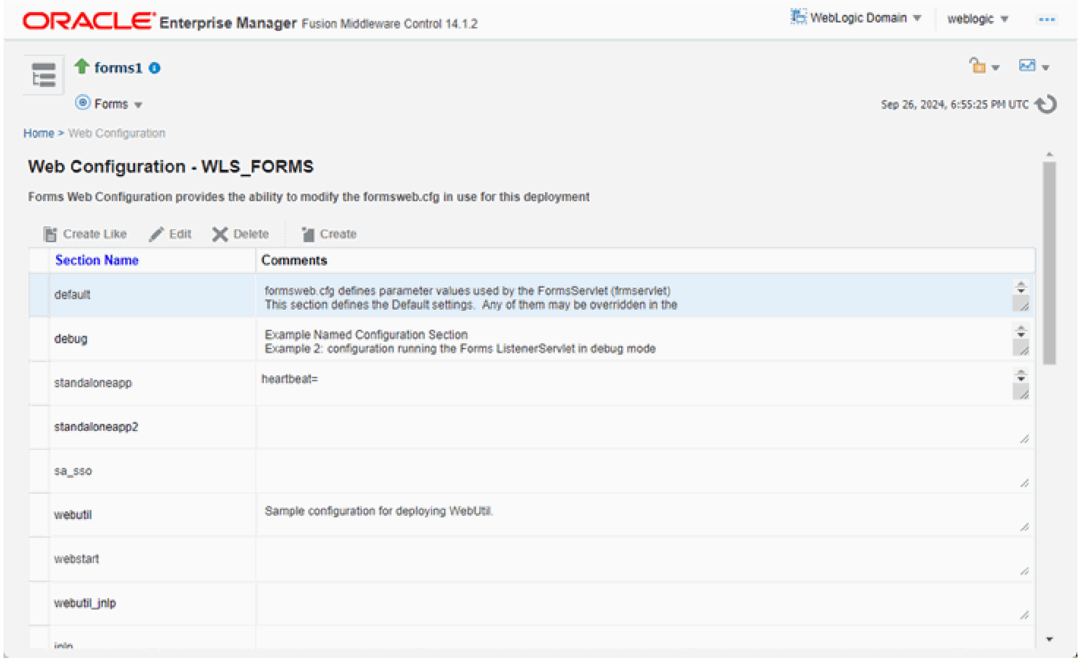
To access Web Configuration page:

1. Access Fusion Middleware Control.
2. From the Fusion Middleware Control main page, click the Target Navigation link to expose the navigation side panel.
3. In the tree, expand the **Forms** node and click the instance you want to configure.

4. From the Forms page and in the Forms Deployments table, click **Web Configuration**.

   The **Web Configuration** page (Figure 8) is displayed.

Figure 8   Web Configuration Page



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain ▾
weblogic ▾
forms1
Forms ▾
Sep 26, 2024, 6:55:25 PM UTC
Home > Web Configuration

Web Configuration - WLS_FORMS

Forms Web Configuration provides the ability to modify the formsweb.cfg in use for this deployment

Section Name	Comments
default	formsweb.cfg defines parameter values used by the FormsServlet (frmservlet) This section defines the Default settings. Any of them may be overridden in the
debug	Example Named Configuration Section Example 2: configuration running the Forms ListenerServlet in debug mode
standaloneapp	heartbeat=
standaloneapp2	
sa_sso	
webutil	Sample configuration for deploying WebUtil.
webstart	
webutil_jnlp	
jnlp	

5. See Table 15 and Table 16 for the tasks that you can do.

> **Note:**
>
> As with most Web applications, it is easy to lose unsaved changes by switching pages. Be sure to save any changes you make through Fusion Middleware Control to Forms configuration or environment files before proceeding to other pages.
>
> The length of time it takes for changes to be saved is affected by the number of lines you have changed. For example, an additional fifty lines of comments takes longer to save than just the deletion of a single entry.

Common Tasks in Web Configuration Page

Read about the tasks available to edit and modify the configuration file and parameters in the Web Configuration page.

The following table provides information about the common tasks that you can do to edit configuration with the sections of a configuration file and their parameters.

Table 15   Common Tasks for Working with Configuration Sections

Task	Description	Comment
Create Like	Creates a copy of a configuration section.	Use to create a configuration section based on the parameters of an existing configuration section.
Edit	Opens the Edit Description dialog.	Allows editing of the text description of a configuration section.
Delete	Opens the Confirmation dialog when deleting a configuration section.	Irrevocably deletes a configuration section and its contents when you click Delete in the Confirmation dialog.
Create	Opens the Create Section dialog.	Creates a configuration section. You must supply a required name and an optional description for it.

The following table provides information about the tasks that you can do to modify the parameters within a named configuration section

Table 16   Common Tasks for Working with Parameters

Task	Description	Comment
Show	Drop down list for selecting named groups of parameters in a configuration section.	Use for viewing and editing groups of parameters, as described in see Forms Configuration Parameters. The groups of parameters include: • basic • sso • trace • plugin • HTML • applet • advanced • all
Revert	Enables you to revert all changes made to parameters in a configuration section since the last apply.	Does not allow you to revert individual changes in a configuration section.
Apply	Applies and activates all changes made to parameters in a configuration section.	Once applied, you cannot revert changes to individual parameters.
Hide Inherited	Enables you to hide or display parameters that are inherited from a parent configuration section.	Use this to view parameters that have been explicitly added to a configuration section or to view all parameters (including those that are inherited from the default section).
Add	Displays the Add Parameter dialog.	Add a parameter to a configuration section based on a mandatory name and an optional value and description.
Delete	Deletes a parameter.	There is no Confirmation dialog. Once applied, you cannot revert changes to individual parameters.

Table 16 (Cont.) Common Tasks for Working with Parameters

Task	Description	Comment
Override	Allows overriding and editing of a parameter which is inherited from the default section.	Click Apply to save and activate your changes.

Configure Parameters with Fusion Middleware Control

For a description and the location of the Forms servlet configuration file (`formsweb.cfg`), see Configuration Files.

There are several configuration parameters that specify files. Typically, the following values and their parameters should appear in the default configuration section, as shown in Table 17.

> **Note:**
>
> The parameter `basejnlp` is not included in the default configuration section, but appears in the examples sections provided in the installation.

Table 17 Default Configuration Parameters that Specify Files

Parameter	Value	Default Location (When path not specified)
`baseHTML`	`base.htm`	`FORMS_INSTANCE/server`
`baseHTMLjpi`	`basejpi.htm`	`FORMS_INSTANCE/server`
`basejnlp`	`NULL`	`FORMS_INSTANCE/server`
`envFile`	`default.env`	`DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config`
`baseSAAfile`	`basesaa.txt`	`FORMS_INSTANCE/server`

All of these parameters specify file names. The default path are specified in the table above.

Managing Configuration Sections

You can manage configuration sections by creating, editing, duplicating, and deleting a named configuration sections.

The following sections are included:

- Creating a Configuration Section
- Editing a Named Configuration Description
- Duplicating a Named Configuration
- Deleting a Named Configuration

Creating a Configuration Section

You can create a configuration section in formsweb.cfg from the **Web Configuration** page of Fusion Middleware Control. These configurations can be requested in the end-user's query string of the URL that runs a form.

To create a configuration section:

1. Start the Fusion Middleware Control.
2. From the Fusion Middleware Control main page, click the link to the Forms Services instance that you want to configure.
3. From the Forms menu list, select the **Web Configuration**.
4. From the Change Center menu, select **Lock & Edit** to start editing the configuration.

> **Note:**
>
> Lock and Edit makes sure that you are not overwriting others changes. Only one person can make changes per transactional unit.
>
> In case you do not want to save the changes, you can click **Undo All Changes** in the Change Center menu.

5. Click **Create** at the top of the **Web Configuration** region.

   The **Create Section** dialog appears.
6. Enter a name and description for the configuration section and click **Create**.

> **Note:**
>
> The name must not contain any special characters such as #, *.

   The configuration section is added.

7. To activate the changes, click **Activate Changes**.

For example, to create a configuration to run Forms in a separate browser window with the Oracle look and feel, create a section called sepwin and add the following parameters from Table 18:

Table 18 Sample Parameters to Add to a Configuration Section

Parameter	Value
form	`<module>`
separateFrame	`True`
lookandfeel	`Oracle`

Your users would type the following URL to launch a form that uses the "sepwin" (or the name you applied) configuration:

```
http://server:port/forms/frmservlet?config=sepwin
```

Editing a Named Configuration Description

You can edit the description (comments) for a named configuration from the **Web Configuration** page.

> **Note:**
>
> You can make a backup of the configuration section you are about to edit by duplicating it first, as described in Duplicating a Named Configuration.

To edit a named configuration description:

1. In the **Web Configuration** region, select the row containing the configuration section you want to edit.
2. Click **Edit**.
3. The **Edit Description** dialog appears.
4. Enter the text for the comment.
5. Click **Save**.

   The **Edit Description** dialog box is dismissed, and your changes are saved.

Duplicating a Named Configuration

You can make a copy of a named configuration for backup purposes, or create configuration sections from existing configurations or other duplicates.

To duplicate a named configuration:

1. In the **Web Configuration** region, select **Create Like**.
2. In the Create Like dialog, from the **Section to Duplicate** menu list, select the name of an existing configuration section you want to duplicate.
3. In the **New Section Name** field, enter a name for the configuration section. The name for the configuration section must be unique.
4. Click **Create**.

   A section with the same parameters, parameter values and comments of the section you are duplicating is created.

Deleting a Named Configuration

When you delete a named configuration, you delete *all* the information within it. If you only want to delete specific parameters, see Managing Parameters.

To delete a named configuration:

1. From the **Web Configuration** region, select the row of the configuration section you want to delete.
2. Click **Delete**.

The **Confirmation** dialog appears.

3. Click **Delete**.

   The configuration section is deleted.

   Oracle Enterprise Manager returns to the **Web Configuration** page and displays the remaining configurations.

> **Note:**
>
> You cannot delete the Default configuration section.

Managing Parameters

Use Fusion Middleware Control to manage parameters within a named configuration. You can add, edit, or delete parameters from the Section pane of Fusion Middleware Control.

To edit a new or overridden parameter in a configuration section:

1. From the **Web Configuration** region, select the row of the configuration section that contains the parameter(s) you want to edit.
2. In the Section region, select the parameter group from the **Show** menu list. The parameters of the group are displayed.
3. Select the row of the parameter you want to edit. Enter the Value and Comments.

   > **Note:**
   >
   > You can edit new or overridden parameters. Inherited parameters must first be overridden so they can be edited. In Figure 9, `form`, `height`, and `width` are examples of overridden parameters.

4. Click **Apply** to save the changes or **Revert** to discard them.

To add a parameter to a configuration:

1. In Fusion Middleware Control, from the **Web Configuration** region, select the configuration section row to which you want to add a parameter.
2. Click Add to add a parameter.

   The Add dialog box is displayed.
3. Enter the Name, Value and Comments for the parameter.
4. Click **Create** to add the parameter.
5. Click **Apply** to save the changes or **Revert** to discard them.

To delete a parameter in a configuration:

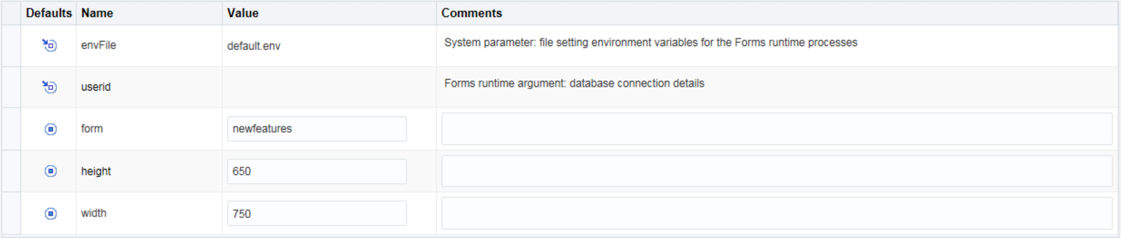
1. In Fusion Middleware Control, from the **Web Configuration** region, select the configuration section row that contains the parameter you want to delete.
2. In the Sections region, from the Show menu list, select the parameter group that contains the parameter you want to delete.

3. Select the row that contains the parameter you want to delete.
4. Click **Delete**.
5. Click **Apply** to save the changes or **Revert** to discard them.

> ### Note:
>
> You can only delete user-defined parameters. Inherited parameters (such as `envFile` in Figure 9) cannot be deleted.

> ### Note:
>
> When you delete an overridden parameter, the parameter is not deleted but instead regains its inherited status.

Figure 9 Parameter States



	Defaults	Name	Value	Comments
	↘◎	envFile	default.env	System parameter: file setting environment variables for the Forms runtime processes
	↘◎	userid		Forms runtime argument: database connection details
	◉	form	newfeatures	
	◉	height	650	
	◉	width	750	

This image shows a screenshot that displays the different icons for the various parameter states in Fusion Middleware Control.

Forms Configuration Parameters

This section provides information about Forms configuration parameters.

These parameters can be specified in the Forms Web Configuration (`formsweb.cfg`), as described in preceding sections. Many of these parameters can also be specified in the URL. A value in the URL overrides a value from `formsweb.cfg`.

Note that a few configuration parameters are applet parameters and are defined in the applet definition template files provided by Oracle. The value for these parameters is based on configuration parameters that have the same names as the applet parameters. For descriptions of these configuration parameters and applet parameters, refer to Web Configuration Parameters. Some parameters in the Web Configuration are servlet parameters and should not be added to the template files.

Managing Environment Variables

Use the **Environment Configuration** page of Fusion Middleware Control to manage environment variables. From this page, you can add, edit, or delete environment variables as necessary.

The environment variables such as `PATH`, `ORACLE_HOME`, and `FORMS_PATH` for the Forms run-time executable (`frmweb.exe` on Windows and `frmweb` on UNIX) are defined in `default.env`. The Forms listener servlet calls the executable and initializes it with the variable values provided in the environment file, which is found in the `$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config` directory by default.

Any environment variable that is not defined in `default.env` is inherited from the Oracle WebLogic Managed Server. The environment file must be named in the `envFile` parameter in the Default section of the **Web Configuration** page.

A few things to keep in mind when customizing environment variables are:

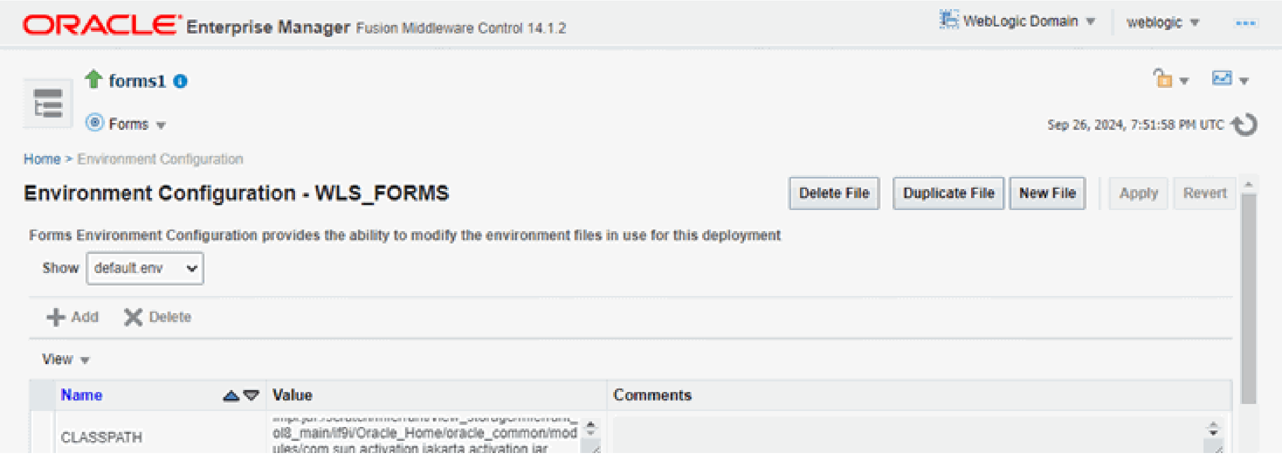
- Environment variables may also be specified in the Windows registry. Values in the environment file override settings in the registry. If a variable is not set in the environment file, the registry value is used.
- The server does not require restarting for configuration changes to take effect.
- Existing Forms processes are not affected by environment variables that were defined after they were started.
- Environment variables not set in the environment file or Windows registry are inherited from the environment of the parent process, which is the Oracle WebLogic Managed Server.

Managing Environment Configuration Files

To access the Environment Configuration page:

1. Start Fusion Middleware Control.
2. From the Fusion Middleware Control main page, click the link to the Oracle Forms Services instance that you want to configure.
3. From the Forms menu list, select **Environment Configuration**.

Figure 10   Environment Configuration page


ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Sep 26, 2024, 7:51:58 PM UTC
Home > Environment Configuration
Environment Configuration - WLS_FORMS
Delete File
Duplicate File
New File
Apply
Revert
Show
default.env
Add
Delete
View
Name
Value
Comments
CLASSPATH

Forms Environment Configuration provides the ability to modify the environment files in use for this deployment

Name	Value	Comments
CLASSPATH	ol8_main/lf9i/Oracle_Home/oracle_common/modules/com.sun.activation.jakarta.activation.jar	

To duplicate an environment configuration file:

1. From the **Environment Configuration** page, click Duplicate File.
   The Duplicate File dialog is displayed.
2. Select the file which you want to duplicate and enter a unique name for the file.
3. Click Duplicate to create the file.

To delete an environment configuration file:

1. In the **Environment Configuration** page, from the **Show** menu list, select the environment configuration file you want to delete.
2. Click **Delete File**.
   The Confirmation dialog is displayed.
3. Click **Yes** to confirm the deletion.

> **Note:**
>
> You cannot delete `default.env`. You can delete only user-defined environment configuration files.

To view an environment configuration file:

1. In the **Environment Configuration** page, from the Show menu list, select the environment configuration file that you want to view.
2. The parameters and their values are displayed.

Configuring Environment Variables

To edit an environment variable:

1. In the **Environment Configuration** page, select the row of the parameter that contains the environment variable you want to edit.
2. Enter the Value and Comments.
3. Click **Apply** to save the changes or **Revert** to discard them.

To add an environment variable:

1. From the **Show** menu list, select the environment configuration file to which you want to add the variable.
2. Click **Add** to add a parameter.

   The Add dialog box is displayed.
3. Enter the Name, Value and Comments.
4. Click **Create**.
5. Click **Apply** to save the changes or **Revert** to discard them.

To delete an environment variable:

1. From the **Show** menu list, select the environment configuration file where you want to delete an environment variable.
2. Select the rows of the parameters you want to delete. You can delete more than one parameter at a time.
3. Click **Delete**.
4. Click **Apply** to save the changes or **Revert** to discard them.

Forms Environment Variables

Forms runtime configuration environment variables include the following:

- Environment variables that specify the location of resources, such as file names, URLs, hosts, and ports.
- Environment variables used to limit the consumption of resources, such as memory and disk space.
- Environment variables used for performance tuning.
- Environment variables used for tracing.
- Environment variables that restrict the end user's access to databases, without affecting application semantics, specifically, `FORMS_ALLOW_DB_CONNECT_STRING`, `FORMS_DBALIAS_SPECIAL_CHARS`, and `FORMS_RESTRICT_ENTER_QUERY`. These environment variables are also documented in the Form Builder online help.

For descriptions of runtime configuration environment variables, refer to Environment Variables.

Environment variables that affect runtime application semantics are documented in the Form Builder online help. In the online help, navigate to List of Application Semantics Environment Variables under Environment Variables to review details. These include environment variables that specify locale-specific behavior, such as language and format masks.

Environment variables that affect the Form Builder, Compiler, or API are documented in the Form Builder online help. In the online help, navigate to List of Builder Environment Variables under Environment Variables to review details.

Managing User Sessions

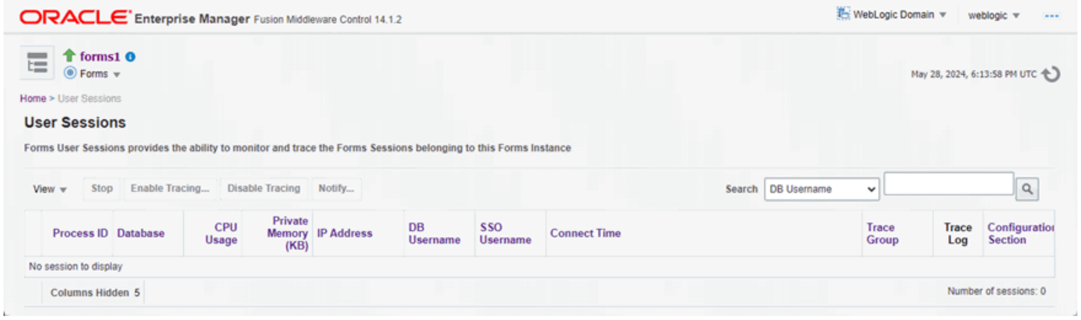
Administrators can manage user sessions, and related features such as monitoring, debugging and tracing using Fusion Middleware Control.

A user session starts when the frmweb process starts. Use the Forms User Sessions pages to monitor and trace the Forms sessions within a Forms Instance. The Forms User Sessions page is accessed from the Forms menu list by selecting **User Sessions**.

To view Forms user sessions:

1. Start Fusion Middleware Control.
2. From the Forms menu list, select **User Sessions**. The **User Sessions** page will be displayed.

Figure 11 User Sessions page



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
May 28, 2024, 6:13:58 PM UTC
Home > User Sessions

User Sessions

Forms User Sessions provides the ability to monitor and trace the Forms Sessions belonging to this Forms Instance

Process ID	Database	CPU Usage	Private Memory (KB)	IP Address	DB Username	SSO Username	Connect Time	Trace Group	Trace Log	Configuratio Section
No session to display										

Columns Hidden 5

Number of sessions: 0

Table 19 User Sessions Page Fields

Field	Description
Process ID	The process ID of the user session.
Database	The database name used by the Forms application for the user session. Click the Database name to view the Database Sessions page.
CPU Usage	The percentage of CPU used by the run-time process.
Private Memory (KB)	The memory used by the run-time process. On Linux platforms, private memory is not the actual private memory but indicates the Resident Set Size (RSS).
IP Address	The IP address of the client computer used to connect to Forms Services.
DB Username	Database user name.
SSO Username	Single Sign-On user name.
Connect Time	The time when the user connected to Forms Services. If the client connection time and client IP are empty, the session is a prestarted session, which is not yet connected to any client.

Table 19   (Cont.) User Sessions Page Fields

Field	Description
Trace Group	The trace group used for tracing the user session. When tracing is enabled, this column shows the trace group name or the events being traced. The events are displayed if the events of the trace group that was enabled for the session have been later modified in the trace configuration. Notice that the Trace group name that is displayed may not be indicate the accurate events being traced if built-ins are used to control the tracing.
Trace Log	Displays the trace log if one exists for the user session.
Configuration Section	Indicates the configuration section used by the Forms application.
Form Name	Indicates the module name of the form application.
CPU Time	Indicates total CPU time used by forms sessions since Connect time.

To enable new Forms user sessions:

By default, new Forms user sessions are permitted to be started. You can disable or prevent new user sessions from being started by using Fusion Middleware Control to set the `allowNewConnections` parameter to false.

1. Start Fusion Middleware Control.
2. From the Forms menu, select **Web Configuration**.
3. Select the default configuration section. `allowNewConnections` cannot be overridden in named sections.
4. In the Sections region, find and edit the value for the `allowNewConnections` parameter. A value of `true` (default) enables new user sessions, whereas `false` disables them.
5. Click **Apply** to save the changes.

When new user sessions are disabled, attempted connections are directed to a URL identified by the `formsweb.cfg` parameter `connectionDisallowedURL` (in the default section). You must specify a complete and valid URL as the value.

If `connectionDisallowedURL` is not specified, then the following message is displayed in the browser:

```
The Forms Servlet is not allowing new connections.
```

When you disable new user sessions, existing forms sessions are unaffected and the Oracle WebLogic Managed Server instance remains up.

To terminate a Forms user session:

1. Select the link to the Forms Services instance that has the user session to be terminated.
2. From the Forms menu, select **User Sessions**.
3. Click the row of the user session to be deleted.
4. Click **Stop**.
5. The Confirmation dialog is displayed.
6. Click **Yes**.

The user session is deleted and the Runform instance is terminated.

To search for a Forms user sessions:

1. From the Forms menu, select **User Sessions**.
2. Select the column name in which you want to search.
3. Enter the search string.
4. Click the blue arrow to search. The search results are displayed.

To sort the list of Forms user sessions:

1. From the Forms menu, select **User Sessions**.
2. Move the mouse over the column.
3. Click the up or down arrow to sort in ascending or descending order. The page is refreshed showing the sorted user sessions. You can sort in order of all columns except Trace Logs.

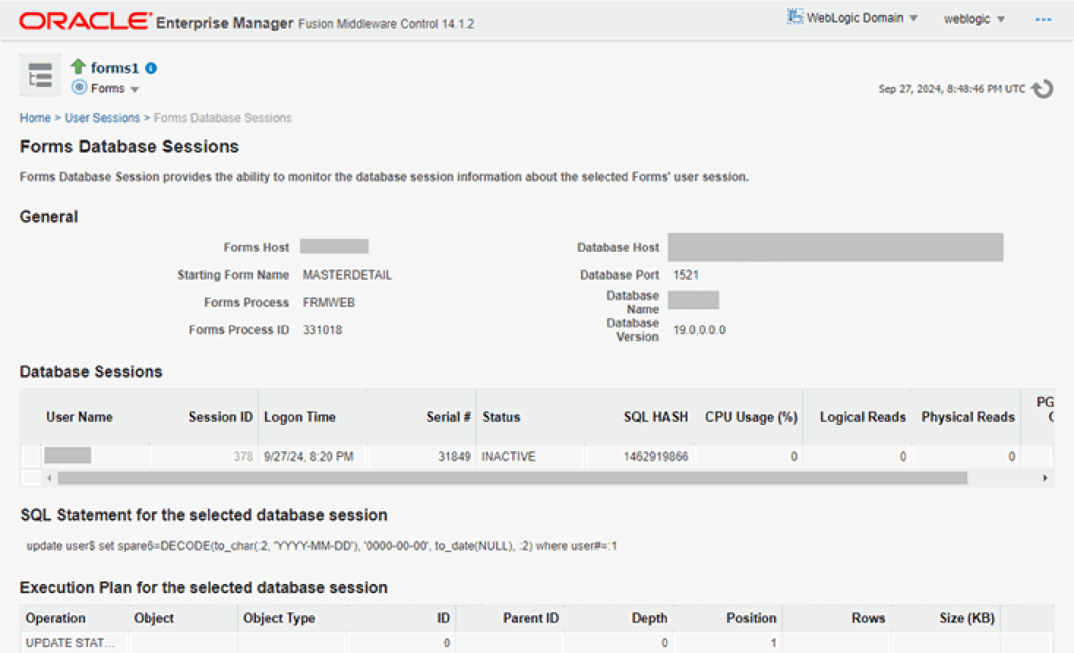
To customize your view of Forms user sessions:

1. From the User Sessions page, click View.
2. From the View menu, you can select:
   - **Show All** to view all columns.
   - The specific columns you want displayed.
   - **Reorder Columns** to organize the order of display of the columns.
   - **Show More Columns** to hide or display specific columns.

To view database sessions for a Forms user session:

1. From the Forms menu, select **User Sessions**.
2. Click the Database name in the Database column.
   Log in to view the Database Sessions page. You need Database Administrator privileges to log in to Database Sessions page.

Figure 12   Database Sessions Page



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Sep 27, 2024, 8:48:46 PM UTC

Home > User Sessions > Forms Database Sessions

Forms Database Sessions

Forms Database Session provides the ability to monitor the database session information about the selected Forms' user session.

General

Forms Host [redacted]
Starting Form Name MASTERDETAIL
Forms Process FRMWEB
Forms Process ID 331018

Database Host [redacted]
Database Port 1521
Database Name [redacted]
Database Version 19.0.0.0.0

Database Sessions

User Name	Session ID	Logon Time	Serial #	Status	SQL HASH	CPU Usage (%)	Logical Reads	Physical Reads	PG(
[redacted]	378	9/27/24, 8:20 PM	31849	INACTIVE	1462919866	0	0	0	

SQL Statement for the selected database session

```
update user$ set spare6=DECODE(to_char(:2, 'YYYY-MM-DD'), '0000-00-00', to_date(NULL), :2) where user#=:1
```

Execution Plan for the selected database session

Operation	Object	Object Type	ID	Parent ID	Depth	Position	Rows	Size (KB)
UPDATE STAT...			0		0	1		

3. The following three tables describes the information displayed in the Database Sessions page.

Table 20   Database Sessions Page

Field	Description
Username	Database username used for connection to the database.
Session ID	Database session identifier.
Logon Time	Date and time when user logged on to the session.
Serial #	Session serial number. Used to uniquely identify a session's objects. Guarantees that session-level commands are applied to the correct session objects if the session ends and another session begins with the same session ID.
Status	Indicates whether the session is active or not.
SQL HASH	Used to identify the SQL statement executed
CPU Usage (%)	CPU Usage (in percentage) on the Database system for the given session.
Logical Reads	Number of Logical Reads for the given session.
Physical Reads	Number of Physical Reads for the given session.
PGA (Program Global Area) Memory	Size of PGA (Program Global Area) Memory after an interval.

Table 21   Details of Selected Database Session

Field	Description
SQL Statement for the selected Database Session	Displays the most recent SQL statement.

Table 22   Execution Plan for the Selected Database Session

Field	Description
Operation	Name of the internal operation performed in the execution step (for example, TABLE ACCESS).
Object	Name of the table or index.
Object Type	Type of the object.
ID	A number assigned to each step in the execution plan.
Parent ID	ID of the next execution step that operates on the output of the current step.
Depth	Depth (or level) of the operation in the tree. It is not necessary to issue a CONNECT BY statement to get the level information, which is generally used to indent the rows from the PLAN_TABLE table. The root operation (statement) is level 0.
Position	Order of processing for all operations that have the same PARENT_ID.
Rows	Estimate, by the cost-based optimizer, of the number of rows produced by the operation.
Size (KB)	Estimate, by the cost-based optimizer, of the number of bytes produced by the operation.
Cost	Cost of the operation as estimated by the optimizer's cost-based approach. For statements that use the rule-based approach, this column is null.
Time (sec)	Elapsed time (in seconds) of the operation as estimated by the optimizer's cost-based approach. For statements that use the rule-based approach, this column is null.
CPU Cost	CPU cost of the operation as estimated by the optimizer's cost-based approach. For statements that use the rule-based approach, this column is null.
I/O Cost	I/O cost of the operation as estimated by the optimizer's cost-based approach. For statements that use the rule-based approach, this column is null.

To send notification messages to Forms user sessions:

1. Start Fusion Middleware Control.
2. From the Forms menu select **User Sessions**.
3. Select the user sessions required to receive the notifications.

   You can select multiple user sessions by using **CTRL** key and selecting the additional sessions.
4. Click the **Notify** button above the table.
5. In the dialog box, select the **Notification Value** to be sent to the user.

   > **Note:**
   >
   > Refer to the Form Builder Help for information about how to receive Administrator Notifications in your application.
6. Click OK to send the notification.

Creating Your Own Applet Definition Template Files

Consider creating your own applet definition template files (by modifying the templates provided by Oracle). By doing it, you can hard-code standard Forms parameters and parameter values into the template.

Your template can include standard text, a browser window title, or images (such as a company logo) that would appear on the first Web page users see when they run Web-enabled forms. Adding standard parameters, values, and additional text or images reduces the amount of work required to customize the template for a specific application. To add text, images, or a window title, you must include the appropriate tags in the template HTML file.

See Specifying Special Characters in Values of Runform Parameters for information about coding the `serverArgs` applet parameter.

Any user-added customized configuration files (such as user client registry files or user key binding files or multiple environment files) must be copied to the same directory as the corresponding default configuration file.

For example, if the user has created a French environment configuration file `default_fr.env`, then it must be placed in the `$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config` directory.

Variable References in Template HTML Files

When a variable reference occurs within a string delimited by quotes or apostrophes (for example, the value of an applet parameter), then when the value of the variable is substituted for the variable reference, HTML escape sequences replaces the HTML metacharacters ('&', '<', '>', quote, and apostrophe).

This sequence is *not* done for variable references outside delimited strings. Therefore, such variables should be specified in the `restrictedURLparams` system default configuration parameter, for security reasons.

Deploying Fonts, Icons, and Images

You can specify the default location and search paths for fonts, icons, and images in Registry.dat. Custom color schemes settings and gradient canvas color settings are also set in Registry.dat.

The following sections are included:

- Managing Registry.dat with Fusion Middleware Control
- Creating Custom Runtime Color Scheme
- Customizing Gradient Canvas Colors
- Managing Application Fonts
- Deploying Application Icons, Images or Audio Files
- Splash screen and Background Images

- Custom JAR Files Containing Icons and Images, and Audio Files
- Customizing Smart Bar Size
- Registry Settings

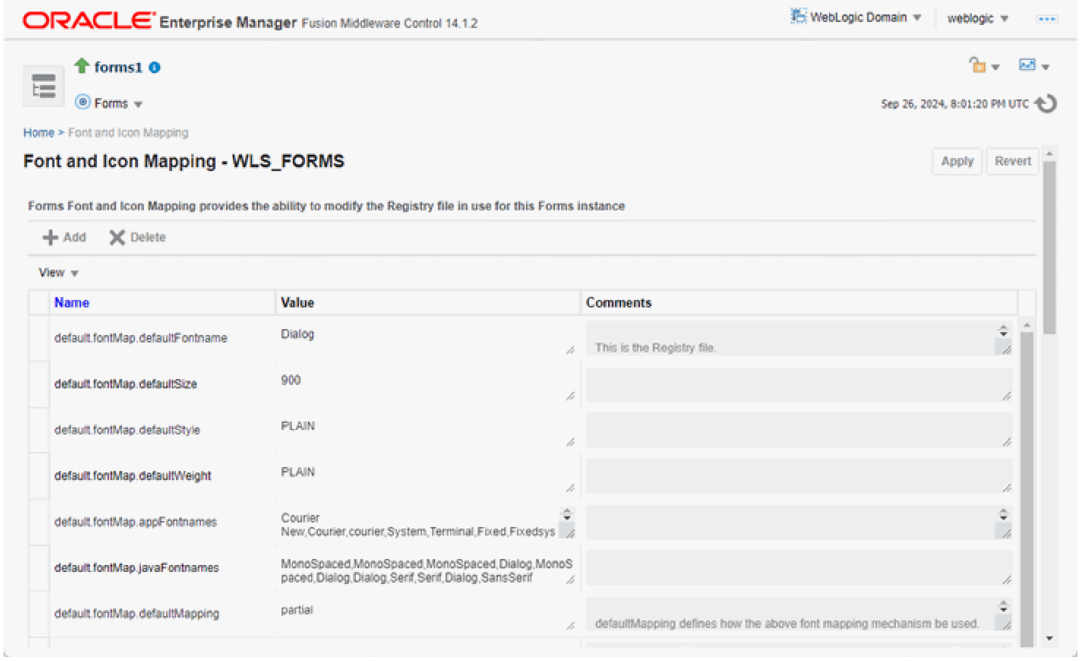
Managing Registry.dat with Fusion Middleware Control

Use Fusion Middleware Control to change, add, or delete parameters from `Registry.dat`.

To access the Fonts and Icon Mapping page:

1. Start Fusion Middleware Control.
2. From the Forms menu list, select **Font and Icon Mapping**. The **Font and Icon Mapping** page is displayed.

Figure 13   Font and Icon Mapping Page



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Sep 26, 2024, 8:01:20 PM UTC
Home > Font and Icon Mapping
Apply
Revert
Add
Delete
View

Font and Icon Mapping - WLS_FORMS

Forms Font and Icon Mapping provides the ability to modify the Registry file in use for this Forms instance

Name	Value	Comments
default.fontMap.defaultFontname	Dialog	This is the Registry file.
default.fontMap.defaultSize	900	
default.fontMap.defaultStyle	PLAIN	
default.fontMap.defaultWeight	PLAIN	
default.fontMap.appFontnames	Courier New,Courier,courier,System,Terminal,Fixed,Fixedsys	
default.fontMap.javaFontnames	MonoSpaced,MonoSpaced,MonoSpaced,Dialog,MonoSpaced,Dialog,Dialog,Serif,Serif,Dialog,SansSerif	
default.fontMap.defaultMapping	partial	defaultMapping defines how the above font mapping mechanism be used.

To edit a Registry.dat parameter value:

1. Start Fusion Middleware Control.
2. From the Forms menu list, select **Font and Icon Mapping**.
3. Select the row containing the parameter to modify and change the value(s) for it in the **Value** text field.
4. Click **Apply** to save the changes.

To add a Registry.dat parameter and its value:

1. From the Forms menu list, select **Font and Icon Mapping**.
2. Click **Add**.

   The Add dialog appears.
3. Enter the name, value, and comments for this parameter.

4. Click **Create**.
5. Click **Apply** to save or **Revert** to discard the changes.

To delete a Registry.dat parameter and its value:

1. From the Forms menu list, select **Font and Icon Mapping**.
2. Select the row containing the parameter to delete and click **Delete**.
3. The parameter is deleted.
4. Click **Apply** to save or **Revert** to discard the changes.

Creating Custom Runtime Color Scheme

Oracle Forms delivers nine predefined color schemes that are set using the applet parameter `colorscheme`. Beginning with Oracle Forms 12*c*, customized color schemes can be created in lieu of using what is provided. To enable a custom colorscheme, set `customcolorscheme=<COLOR SCHEME NAME>` in formsweb.cfg. The customizations are configured in Registry.dat. An example named sample is included in Registry.dat. To use this example, set customcolorscheme=sample in formsweb.cfg.

> **Note:**
>
> All virtual colors in the custom color scheme must have valid values representing the desired color. NULL values are considered invalid. If any entry is invalid the application will use the color scheme set in the `colorscheme` parameter for all colors. The only exception is `desktop`. The `desktop` value is permitted to be blank, but all others must contain valid values.

Color values can either be hexadecimal (e.g. 0xFFFFFF) or RGB sets (e.g. 255,255,255). For example:

```
# Sample custom color scheme, where scheme name is "sample".
colorScheme.sample.description=Sample custom color scheme
colorScheme.sample.lightest=0xFFFF33
colorScheme.sample.lighter=0xFFCC33
colorScheme.sample.light=0xCC3333
colorScheme.sample.dark=0x993333
colorScheme.sample.darker=0x660033
colorScheme.sample.darkest=0x003333
colorScheme.sample.selection=0x4169E1
colorScheme.sample.pinstripe1=0xEE82EE
colorScheme.sample.pinstripe2=0xF5DEB3
colorScheme.sample.desktop=0x5111F6
colorScheme.sample.smartBar=0xFFFFFF
```

Customizing Gradient Canvas Colors

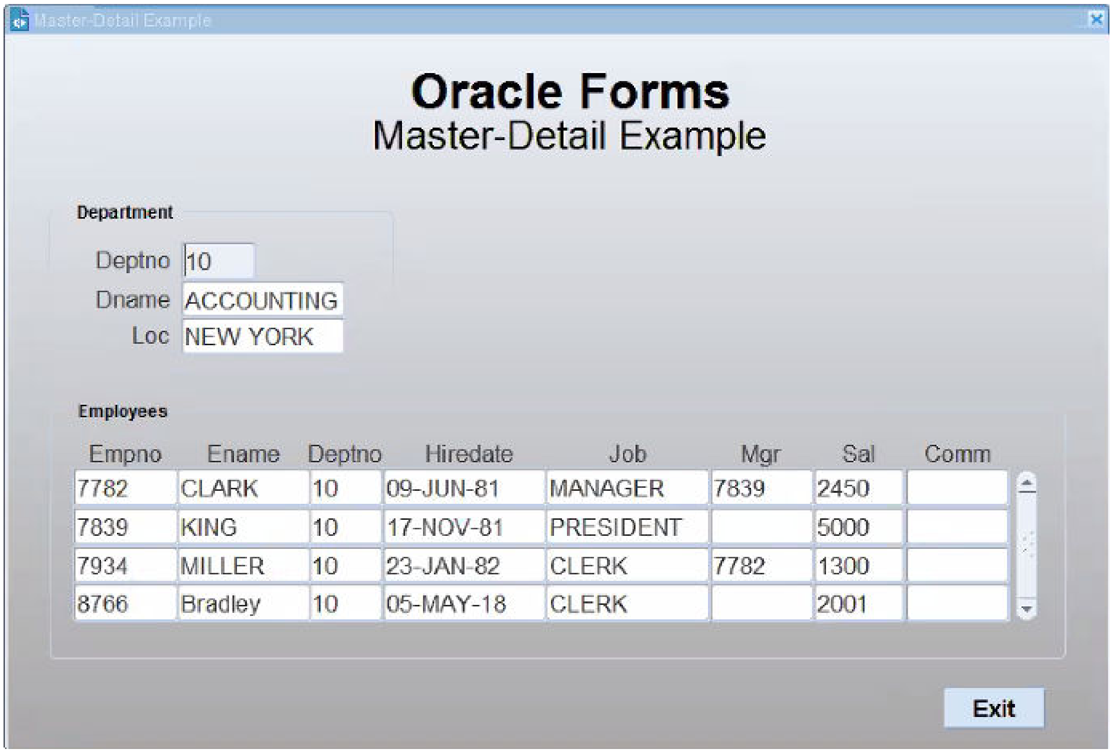
Gradient canvas colors are set at design-time, but are not visible until runtime.

The starting color is based on the background color set for the canvas. If no color is set, the color implemented by the runtime colorscheme will be used. At design-time, the developer can choose the starting position from where the gradient flows (None, Left, Top, Right, or Bottom).

The difference in color from its starting point to its end point is set in the runtime configuration Registry.dat using these settings:

```
default.gradient.redDelta
default.gradient.greenDelta
default.gradient.blueDelta
```

These settings only apply when the canvas background color has a specified value. Each specifies an integer value representing how much to add (or subtract) from the start value to reach the end color. The color range assumes a value between 0 and 255. Thus, if the start color had an RGB value of `150,150,150` and each of the Delta properties was set to 50, then the end color would be `200,200,200`. Similarly, the properties can have negative values. As an example, if all were set to -50, the end color would be `100,100,100`. Setting the red, green, blue Delta values to 0 or unset would result in no change to the color. However, if left unset and the canvas background color is unset (<Unspecified>) the colorscheme colors are used and the gradient will go from `OLAFDark` to `OLAFVeryLight`.



Department

Deptno	10
Dname	ACCOUNTING
Loc	NEW YORK

Employees

Empno	Ename	Deptno	Hiredate	Job	Mgr	Sal	Comm
7782	CLARK	10	09-JUN-81	MANAGER	7839	2450	
7839	KING	10	17-NOV-81	PRESIDENT		5000	
7934	MILLER	10	23-JAN-82	CLERK	7782	1300	
8766	Bradley	10	05-MAY-18	CLERK		2001	

Using this feature in cases where a hidden stacked canvas that is larger than the main canvas and is shown later in the session's life is not recommended. This is because the gradient fill pattern is drawn only at initial startup. Altering the view area at runtime may result in an incomplete gradient flow. It should be noted that using a gradient in both the content and stacked canvas will result in color transitioning not being in sync with each other. This is because the coloring on one canvas is unrelated to the coloring on another. Additional usage and limitation information can be found in the Form Builder Help.

Managing Application Fonts

You can use Fusion Middleware Control to change the default font and font settings in the `Registry.dat` file.

All font names are Java Font names. Each of these parameters represents the default property to use when none is specified.

To change the font settings for a deployed application:

1. Start Fusion Middleware Control.
2. From the Forms menu list, select **Font and Icon Mapping**.
3. Change any of the settings described in the following table to reflect your desired font settings:

Table 23 Default Font Values

Property	Default Value	Description
`default.fontMap.appFontnames`	`Courier New,Courier,courier,System,Terminal,Fixedsys,Times,Times New Roman,MS Sans Serif,Arial`	Default Font Face mapping. Represents a comma delimited list of application font names. The number of entries in the `appFontname` list should match the number in the `javaFontname` list. The elements of the list are comma separated and *all* characters are taken literally; leading and trailing spaces are stripped from Face names. Notice that this file uses the Java 1.1 font names to handle the NLS Plane.
`default.fontMap.defaultFontname`	`Dialog`	Represents the default Java `fontName`

Table 23   (Cont.) Default Font Values

Property	Default Value	Description
`default.fontMap.defaultMapping`	`FULL`	Values are FULL, PARTIAL, and NONE. • If set to FULL, the Java font mapping mechanism is used. Application specified fonts are mapped to the Java fonts. If a specified Java font is not found in the `appFontnames` (`default.fontMap.appFontnames`) list, then the default Java font (defined in `default.fontMap.defaultFontname`) is used. • If set to PARTIAL, the Java font mapping mechanism is used. If an application specified font is not found in the `appFontnames` list, Forms attempts to use the application specified font. • If set to NONE, the Java font mapping mechanism is not used. The application specified font name is passed directly to Java and Forms will attempt to use it.
`default.fontMap.defaultSize`	`900`	Represents the default `fontSize`. Note that the size is multiplied by 100; for example, a 10pt font has a size of 1000.
`default.fontMap.defaultStyle`	`PLAIN`	Represents the default `fontStyle`, PLAIN or *`ITALIC`*.
`default.fontMap.defaultWeight`	`PLAIN`	Represents the default `fontWeight`, PLAIN or **`BOLD`**.
`default.fontMap.javaFontnames`	`MonoSpaced, MonoSpaced,MonoSpaced, Dialog,MonoSpaced,Dialog, Dialog,Serif,Serif, Dialog,Sans Serif`	Represents a comma delimited list of Java font names.

For example, to change your default font to Times New Roman, replace **Dialog** with **Times New Roman**.

You can change the default font face mappings:

```
default.fontMap.appFontnames=Courier New,Courier,
courier,System,Terminal,Fixed,Fixedsys,Times,Times New Roman,
MS Sans Serif,Arial
default.fontMap.javaFontnames=MonoSpaced,MonoSpaced,MonoSpaced,Dialog,
MonoSpaced,Dialog,Dialog,Serif,Serif,Dialog,SansSerif
```

> **Note:**
>
> Some fonts may not be supported in Java. For this reason you can specify (map) Java-supported fonts that appear when a non-supported font is encountered. In the previous sample, each font in `default.fontMap.appFontnames` corresponds to a font in `default.fontMap.javaFontnames`.

4. Click **Apply** to save the changes.

Using User Installed Fonts

Applications that have explicitly declared font faces can take advantage of fonts installed on the user's machine that have that font installed locally.

For example, if the application has Forms objects—like text item, button, and so on—and those items have been defined by the developer to use a font named "Times New Roman" for their labels, those items will appear at runtime using that named font if they are detected on the user's machine.

To enable this functionality, set the `Registry.dat` parameter `default.fontMap.defaultMapping` to one of these three possible values:

- `full`: (default) The Java font mapping mechanism is used. If a specified font is not found in the `appFontnames` list, the default font is used (as defined by the `defaultFontname` parameter in `Registry.dat`).
- `partial`: The Java font mapping mechanism is used. If a specified font is not found in the `appFontnames` list, the requested font is passed directly to Java and Java attempts to use the specified font (if found on the user's machine).
- `none`: The Java font mapping mechanism is not used. The requested font is passed directly to Java and Java attempts to use the specified system font (if found on the user's machine).

Deploying Application Icons, Images or Audio Files

When deploying an Oracle Forms application, the icon and image files used must be in a Web-enabled format, such as JPG or GIF (GIF is the default format). The same is true of audio files. For supported Audio files formats, please refer to the Java JFX documentation at `https://docs.oracle.com/javafx/2/api/javafx/scene/media/package-summary.html#SupportedMediaTypes`.

By default, the icons are found relative to the `DocumentBase` directory. That is, `DocumentBase` looks for images in the directory relative to the base directory of the application start HTML file. As the start HTML file is dynamically rendered by the Forms servlet, the Forms `webapp's` directory becomes the document base. The Forms `webapp's` directory is located at `$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_14.1.2/<random string>/war`.

For example, if an application defines the icon location for a button with `myapp/<iconname>`, then the icon is looked up in the directory `forms/myapp`.

To change the default location, set the `imageBase` parameter to `codebase` in the Web Configuration page of Fusion Middleware Control. Alternatively, you can change the `default.icons.iconpath` value of the `Registry.dat` file in the `$DOMAIN_HOME/config/`

`fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config/oracle/forms/registry` directory.

Setting the imageBase parameter to `codebase` enables Oracle Forms to search the `forms/java` directory for the icon files. Use this setting if your images are stored in a JAR file. Changing the image location in the `Registry.dat` configuration file is useful to store images in a central location independent of any application and independent of the Oracle Forms installation.

For audio files the concept is similar to the above, however `mediaBase` cannot be overridden by entries in `Registry.dat`. For audio files stored in JAR files, set `mediaBase=codeBase` in `formsweb.cfg`.

Storing Icons, Images, or Audio Files in a Java Archive File

If an application uses custom icons, images, or audio files, it is recommended you store them in a Java archive file and set the `imageBase` value to `codebase` (and mediaBase=codeBase if including audio files). The files can be packaged into a Java archive using the `jar` command of any Java Development Kit (Java JDK), so long as its version is the same or older than the version expected to be used on the end-user machine. Also, JAR files must be signed with a trusted certificate. For more information on how to create a JAR file, see Creating a JAR File for Icon, Images, and Audio Files.

In order for Oracle Forms to access the icon files stored in this archive, the archive must be stored in the `forms/java` directory. Also, the name of the archive file must be part of the `ARCHIVE` parameter used in the custom application section of the Web Configuration (`formsweb.cfg`) file.

If the application is running with Java Web Start (JWS), you must reference custom JAR files in `extensions.jnlp`, which is found in the `forms/java` directory. JWS will not read the value of the `ARCHIVE` parameter. This file must be manually edited (in a text editor). Be sure to stop the Forms managed server before editing. When the initial application starts, the JAR file is downloaded and stored in cache on the client until the archive file is changed.

For information on how to create JAR files, see Custom JAR Files Containing Icons and Images, and Audio Files.

> **Note:**
>
> Oracle Forms default icons (for example, icons present in the default smart icon bar) do not require deployment, as they are part of the `frmall.jar` file.

Adding, Modifying, and Deleting Icon Mappings

Use Fusion Middleware Control to add icon changes to the Registry.dat file used by your application.

To add icon mappings:

1. Start Fusion Middleware Control.
2. From the Forms menu, select **Font and Icon Mapping**.
3. Click **Add**.
   The Add dialog appears.
4. Enter the name, value, and an optional comment.

5. Click **Create** to create the mapping.
   The mapping is added to the list.
6. Click **Apply** to save the changes.

To modify icon mappings:

1. From the Font and Icon Mapping region, select the mapping you want to modify.
2. Change the name and value of the mapping. For example,
   - Modify the `iconpath` parameter specifying your icon location:
     `default.icons.iconpath=/mydir`

     (for an absolute path)

     or

     `default.icons.iconpath=mydir`

     (for a relative path, starting from the `DocumentBase` Directory)
   - Modify the `iconextension` parameter:
     `default.icons.iconextension=gif`

     or

     `default.icons.iconextension=jpg`
3. Click **Apply** to save and activate the changes.

To delete an icon mapping:

1. From the Font and Icon Mapping region, select the mapping you want to delete.
2. Click **Delete**.
3. The selected icon mapping is deleted.
4. Click **Apply** to save or **Revert** to discard the changes.

To reference the application file:

In a specific named configuration section in the Web Configuration (`formsweb.cfg`), modify the value of the `serverApp` parameter and set the value to the location and name of your application file.

For example:

```
[my_app]
```

```
ServerApp=http://example.com/appfile/myapp
```

(for an absolute path)

or

```
[my_app]
```

```
ServerApp=appfile/myapp
```

(for a relative path, relative to the `CodeBase` directory)

Table 24 describes the correct locations where to place your application icons:

Table 24 Icon Location Guide

Icon Location	When	How
DocumentBase	Default. Applications with few or no custom icons.	Store icons in forms `webapp's` directory or in a directory relative to it. The forms `webapp's` directory is located at `$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_14.1.2/<random string>/war`.
Java Archives	Applications that use many custom icons.	Set `ImageBase` to `codebase`, create Java archive file for icons, and add archive file to the archive parameter in `formsweb.cfg`.
Registry.dat	Applications with custom icons that are stored in a different location as the Oracle Forms install (can be another server). Useful to make other changes to the Registry.dat file such as font mapping.	Copy Registry.dat and change ServerApp parameter in `formsweb.cfg`.

Splash screen and Background Images

When you deploy your applications, you have the ability to specify a splash screen image (displayed during the connection) and a background image file.

Those images are defined in the HTML file or you can use the **Web Configuration** page in Fusion Middleware Control:

```
<PARAM NAME="splashScreen" VALUE="splash.gif">
```

```
<PARAM NAME="background" VALUE="back.gif">
```

The default location for the splash screen and background image files is in the DocumentBase directory containing the baseHTML file.

> **Note:**
>
> Image formats for splash screens and icons are the standard formats that are supported by `java.awt.Image`. For information about `java.awt.Image`, see Java Advanced Imaging (JAI) API in `http://www.oracle.com/technetwork/java/index.html`.

Custom JAR Files Containing Icons and Images, and Audio Files

Each time you use an icon or an image (for a splash screen or background), an HTTP request is sent to the Web server. To reduce the HTTP round-trips between the client and the server, you have the ability to store these files in a Java archive (JAR) file. Using this technique, only one HTTP round-trip is necessary to download the JAR file.

Creating a JAR File for Icon, Images, and Audio Files

The Java JDK comes with an executable called *jar*. This utility enables you to store files inside a Java archive, as described in https://docs.oracle.com/javase/8/docs/technotes/tools/windows/jar.html.

For example:

```
jar -cvf myico.jar Splash.gif Back.gif icon1.gif
```

This command stores three files (`Splash.gif`, `Back.gif`, `icon1.gif`) in a single JAR file called `myico.jar`.

> **Note:**
>
> Java JAR files must be signed with a trusted certificate to be accessible at runtime without displaying additional security warnings. See https://docs.oracle.com/javase/tutorial/deployment/jar/signindex.html.

Using Files Within the JAR File

The default search path for the icons and images is relative to the `documentBase`. However, when you want to use a JAR file to store those files, the search path must be relative to the `codebase` directory, the directory which contains the Java applet.

To use a JAR file to store files, you must specify that the search path is relative to `codebase` using the `imageBase` parameter in the Web Configuration or HTML file.

This parameter accepts two different values:

- **documentBase** The search path is relative to the `documentBase` directory. If no value is specified for `imageBase`, then the value of `documentBase` is used.
- **codeBase** The search path is relative to the `codeBase` directory, which gives the ability to use JAR files.

In this example, we use a JAR file containing the icons and we specify that the search should be relative to `codeBase`. If the parameter `imageBase` is not set, the search is relative to `documentBase` and the icons are not retrieved from the JAR file.

For example (`formsweb.cfg`):

```
archive=frmall.jar, icons.jar
```

```
imageBase=codeBase
```

For audio files, set `mediaBase=codeBase`.

Customizing Smart Bar Size

Oracle Forms application Smart Bar size and its icons can now be customized to fit different icon sizes. This helps the tool bar icons to expand or re-size proportionally when Oracle Forms application is running on a monitor with a higher resolution or DPI.

In earlier releases, unlike other objects in a Form, you could not customize, set, or scale the default size of the Smart Bar and its icons with `clientDPI`.

The current default height of the Smart Bar can fit icons of 16x16 pixel size. To customize or set the height of the Smart Bar to allow larger icons, a new applet parameter `smartBarHeight` has been added. Use the `smartBarIconScaling` parameter can be used in conjunction with the `smartBarHeight`parameter to specify if icons (when smaller) should be scaled to fit. For more information on these parameters, refer to Web Configuration Parameters.

A value of DYNAMIC mean that the Smart Bar inherits whatever value is provided in `clientDPI` settings. As a result the Smart Bar and the default icons size, like other objects in a Form, scale proportionately with `clientDPI` settings. If `clientDPI` settings is specified but the parameter value of `smartBarHeight` is not set, Smart Bar will not scale proportionately with `clientDPI`, and the default small size of Smart Bar and icons will be displayed.

> **Note:**
>
> The icons shipped with Forms for use in default Menu or Smart Bar will not be modified. If you want to set medium or large parameter value for `smartBarHeight`, you have to create your own customized menu bar and icons, that is icons with large pixel size, rather than using the default menu bar and Smart Bar icons shipped with Forms.

Customizing UI Borders

You can customize the borders of some UI items such edit items, display items, poplists, combo-boxes, tlists, and images using border settings in Registry.dat.

These settings let you control which borders are displayed as well as default, rollover, and highlight color for these borders.

- Setting UI Borders
- Setting Border Colors
- Setting Rollover Border Colors
- Setting Border Highlight Colors

Setting UI Borders

Draw one or more borders for items such as edit items, display items, poplists, combo-boxes, tlists, and image when using the Oracle look-and-feel. After you set which borders you want to display, specify which fields use these values through a builder property.

Set borders in `Registry.dat` using these variables: `default.border.right`, `default.border.left`, `default.border.top`, and `default.border.bottom`. These variables take Boolean values. The default is TRUE. See Registry Settings for more details.

For example, to set the borders so that the item looks like an underlined field, set `default.border.bottom` to TRUE and set the other three to FALSE. You'll then have to set the builder property for the items which should have this appearance.

Once you've set the registry variables for an item, set a builder property to specify whether a field uses these values. If the `Specified Border Sides` property is set to FALSE (the default), item bevels appear normally. For example, PLAIN bevels appear as a rectangle around the field, raised or lowered bevels have three dimensional appearance, and so on.

If `Specified Border Sides` is set to TRUE, then the item will appear to have a PLAIN bevel on those sides that are displayed.

This property can also be set and retrieved through plsql using `SET_ITEM_PROPERTY` and `GET_ITEM_PROPERTY`. The property to be used in plsql is BORDER_SIDES.

The XML converter has also been updated so that if you run the converter with `COMPAT_VERSION` set to 12.2.1.4.0, the XML file that gets generated won't include this property. This means that the XML can be read by an older version of the XML converter to create an fmb file that can be properly read by an older builder.

Testing UI Borders

When testing UI border appearance, keep the following in mind:

- Verify that each of the values takes effect as expected and that various combinations also work as expected. For example, setting left and bottom borders should show lines on the left and bottom of the field but not the top or right.
- A missing or invalid value for any of the registry entries should be ignored and treated as TRUE.
- This should impact items with any bevel style but only those items that have the new property turned on (either from the builder or through plsql) should get partial borders, otherwise they should continue to get the full border drawn with the specified bevel.
- Dynamically turning this property on or off should have the expected behavior.
- Dynamically changing the bevel to or from PLAIN should have the expected effect for items that don't have this property set. For items that do have it set, the appearance of the bevel shouldn't change.
- This should work for edititems, display items, poplists, combo boxes, tlists and images.
- This should work for single-line and multi-line (with or without a scroll bar) text items.
- Because of the way borders are created as class variables rather than instance variables, if you change some of these values in `Registry.dat`, you'll need to close the browser or run in a new JVM on the client before you will see the changes take effect.
- If this is used in conjunction with the new feature that allows edge-coloring to be specified, these partial borders should correctly pick up the edge coloring for fields that have that functionality specified (and use the default coloring for other fields).

Setting Border Colors

Specify a color for text-item, display-item, poplist, combobox, tlist, and image borders when using the Oracle look-and-feel.

Set the border color in `Registry.dat` using `default.border.color`. This variable takes comma-separated, RGB values such as "0,255,0". If the variable is not set or if the setting is invalid, the border color defaults to "red". See Registry Settings for more details.

If this variable is set, borders appear in the specified color rather than the default color of the current color scheme. Borders also appear to be PLAIN as opposed to whatever the specified bevel might have been.

> **Note:**
>
> This setting can be used in conjunction with border display variables. If border color and border display variables are set, only the displayed borders appear in the specified color. See Setting UI Borders.

Once you've set the registry variable for an item, set a builder property to specify whether a field uses this value. The `Show Border Color` builder property determines whether the border color is a applied to the item's borders. This property takes a Boolean value. If the property is set to TRUE, then the border appears in the specified color.

This property affects text-item, display-item, poplist, combo box, tlist, and image item types including both single and multi-line fields.

This property can also be set through `SET_ITEM_PROPERTY` and retrieved through `GET_ITEM_PROPERTY` using the pslql constant `BORDER_COLOR`. ER 33471569 now allows various OLAF color names to be used as well as RGB values.

The XML converter has also been updated so that if you run the converter with `COMPAT_VERSION` set to 12.2.1.4.0, the XML file that gets generated won't include this property. This means that the XML can be read by an older version of the XML converter to create an fmb file that can be properly read by an older builder.

Testing UI Borders

When testing border color appearance, keep the following in mind:

- Confirm it works properly for edit items (both single and multi-line, including with or without a scroll bar) as well as display items, poplists, combo boxes, tlists and images.
- Make sure it doesn't cause problems for any other item types
- Make sure that this works regardless of what the bevel-style is set to, the bevel/border will appear PLAIN (but colored) if this is enabled.
- Confirm that if partial borders are turned on, that it combines the coloring and partial borders correctly. This includes dynamically setting this value as well as dynamically setting the partial border support on/off for the item when this property is on.
- Test that for display-items, this correctly works if the value is changed from a single-line value to a multi-line value (or vice-versa). Specifically, does the highlighted border persist when that change happens?
- Need to confirm that the interaction between this property and the highlight border mentioned below works as expected. (For example, if I turn on highlight on an item it should use that color, and when I turn off highlight it should go back to the specified color for this (assuming border coloring is enabled).
- Verify that the right thing happens when the window is made inactive

Setting Rollover Border Colors

Specify a color to display text-item, display-item, poplist, combobox, tlist, and image borders when the user mouses over the item.

Set the border color in `Registry.dat` using `default.border.rolloverColor`. This variable takes comma-separated, RGB values such as "0,255,0". If the variable is not set or if the setting is invalid, the border color defaults to "red". See Registry Settings for more details.

When this variable is set, the rollover border color is displayed on mouse over regardless of the BEVEL setting of the item. Only those borders that are set for display show the rollover border color.

> **Note:**
>
> This setting can be used in conjunction with border display variables. If rollover color and border display variables are set, only the displayed borders appear in the specified color on mouse over. See Setting UI Borders.

Once you've set the registry variable for an item, set a builder property to specify whether a field uses this value. The `Rollover Border Color` builder property determines whether the rollover color is a applied to the item's borders on mouse over. This property takes a Boolean value. If the property is set to TRUE, then the border appears in the specified color when the mouse is over the item.

This property affects text-item, display-item, poplist, combo box, tlist, and image item types including both single and multi-line fields.

The rollover border color only appears for items which are enabled. Disabled items don't show the rollover color. This also means that, if a modal window is shown, the rollover border coloring won't work for items not in that window, since they are effectively disabled while the modal window is visible.

This property can also be set through `SET_ITEM_PROPERTY` and retrieved through `GET_ITEM_PROPERTY` using the pslql constant `BORDER_ROLLOVER`. ER 33471569 now allows various OLAF color names to be used as well as RGB values.

The XML converter has also been updated so that if you run the converter with `COMPAT_VERSION` set to 12.2.1.4.0, the XML file that gets generated won't include this property. This means that the XML can be read by an older version of the XML converter to create an fmb file that can be properly read by an older builder.

> **Note:**
>
> Note that the interaction between this mechanism and the other border color related features mentioned in this document can be a little tricky (specifically if someone changes some of the properties/turns on the highlighting while the mouse is over the item). Thus, if a change is made that chat would change the appearance of the border (highlighting is turned on, the bevel is changed, etc.) that change will appear, and nothing will happen when the mouse leaves the item. The next time the mouse enters the item again, the rollover affect will happen again (assuming it is still enabled for the item).

When testing this feature, among the aspects that should be tested we need to be sure to include:

- Confirm it works properly for edit items (both single and multi-line) and display-items, poplists, comboboxes, tlists and images. Note that when the mouse is over the scroll bar of a multi-line text-field, it will be treated as not in the item. Unfortunately, there is nothing we can do about that, so it is a minor limitation we will have to live with.
- Make sure it doesn't cause problems for any other item types
- Make sure that this correctly works if partial borders are enabled.
- Need to test that for display-items, this correctly works if the value is changed from a single-line value to a multi-line value (or vice-versa). Specifically, does the highlighted border persist when that change happens?
- Need to confirm that the interaction between this property and the other border properties works as expected, particularly when one of the properties is changed while the mouse is over the item (and thus has temporarily changed the border appearance).

Setting Border Highlight Colors

Specify a highlight color to display text-item, display-item, poplist, combobox, tlist, and image borders.

Set the highlight border color in `Registry.dat` using `default.border.highlightColor`. This variable takes comma-separated, RGB values such as "0,255,0". If the variable is not set or if the setting is invalid, the border color defaults to "red". See Registry Settings for more details.

To highlight a border, set `SET_ITEM_PROPERTY(BORDER_BEVEL)` and `SET_ITEM_INSTANCE_PROPERTY(BORDER_BEVEL)` to `HIGHLIGHTED`. Setting the bevel to `HIGHLIGHTED` instructs the client to draw the border using the `default.border.highlightColor` color value.

You can use this feature highlight items that have failed validation. Note that just like the normal border coloring, this coloring is maintained even if the window is made inactive, for example, if an alert or other window is raised. In the case of the window being inactive, the appearance will change slightly, but it will still be distinguishable as a highlighted field.

> **Note:**
>
> OLAF color names, such as "OLAFLight" and "OLAFDark", can be used as well as RGB values.

When testing this feature, among the aspects that should be tested we need to be sure to include:

- Confirm it works properly for edit items (single and multi-line) as well as display-items, poplists, comboboxes, tlists and images.
- Make sure it doesn't cause problems for any other item types
- Make sure it works properly both for set_item_property and set_item_instance_property
- Make sure that changing from one bevel to another (without ever using the HIGHLIGHTED value) still works correctly (this is because of the implementation, we need to make sure that setting the bevel to null (which we do when 'clearing' the color that we might have previously set) doesn't cause any problems.

- Changing the Registry value and running a form again should pick up the new color appropriately without needing to restart the browser (we create a different borderpainter for each color that we try to use, as opposed to just having one 'highlighted' borderpainter.
- Need to test that for display-items, this correctly works if the value is changed from a single-line value to a multi-line value (or vice-versa). Specifically, does the highlighted border persist when that change happens?
- If an item (or instance) has been set to be HIGHLIGHTED, GET_ITEM_PROPERTY and GET_ITEM_INSTANCE_PROPERTY should return 'HIGHLIGHTED' as the bevel value when appropriate.

Registry Settings

This section describes Forms registry settings.

These notes apply to the following table:

- Required: An environment variable is required if Forms requires a non-null value in order to function correctly.
- Valid values: In a few cases, these are listed explicitly, but in most cases they can be inferred from the description.
  If the description implies that the value is Boolean (for example, it states Specifies whether), the valid values are TRUE, True, true, T, t, YES, Yes, yes, Y, y, 1, or FALSE, False, false, F, f, NO, No, no N, n, 0, (case-sensitive), unless indicated otherwise. Numeric values should be integers specified as decimal numbers, optionally followed by K, M, G, or T [powers of 1024, not 1000], unless indicated otherwise. The elements in lists of files or directories should be separated by ':' [Unix] or ';' [Windows] unless indicated otherwise.
- Default: If specified, this is the non-null value that produces the same behavior as not specifying a value. If omitted, the implication is that the default value is null.

Table 25   Additional Registry.dat Settings

Property	Default Value	Description
`default.tabLabel.selectedColor`		Specifies the label color of the selected tab using RGB color values; for example: 255,255,0 (yellow). Each of the R, G, or B primaries can have a value in the range from 0 to 255. If no values are specified, or the setting is invalid, the selected tab uses the default color. **Note:** This setting is only supported when using the Oracle look-and-feel, and will be ignored when using the generic look-and-feel.

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description
`default.tabLabel.unselectedColor`		Specifies the label color of the unselected tabs using RGB color values; for example: 255,255,0 (yellow). Each of the R, G, or B primaries can have a value in the range from 0 to 255. If no values are specified, or the setting is invalid, unselected tabs uses the default color. **Note:** This setting is only supported when using the Oracle look-and-feel, and will be ignored when using the generic look-and-feel.

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description

`default.border.right`
`default.border.left`
`default.border.top`
`default.border.bottom`
	True	Specifies whether the sides of an item (edit items, display items, poplists, combo-boxes, tlists, image items) display a border. These settings take Boolean values.

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description
`default.concealedData.character`	* (asterisk)	Specifies the character to display in a conceal-data field, including the password field in the Logon dialog, to represent the hidden characters being typed. If the value specified is longer than one character, the first character in the value is used. For special characters, specify the Unicode character value for the character preceded by `\u`. For example, to use a small dot, enter `\u2022`. For a large dot, enter `\u25cf`. If the property isn't specified in Registry.dat, or has an empty value, then * (asterisk) is used. If you change the value in Registry.dat, you'll need to restart your browser in order to get a new JVM for the changes to take effect. **Note:** "Default" in the registry setting, `default.concealedData.character` refers to the application

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description
		name. Specifying a different application name can allow for a different character to be used for each application.

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description
`default.fontMap.defaultMapping`	FULL	Controls how font handling is managed. Values are FULL, PARTIAL, and NONE. • If set to FULL, the fontmapping mechanism is used. If a specified font is not found in the `appFontnames` list, the default font is used. • If set to PARTIAL, the fontmapping mechanism is used. If a specified font is *not* found in the `appFontnames` list, the requested font is passed directly to Java, and Java attempts to use the specified font. • If set to NONE, the fontmapping mechanism is not used. The requested font is passed directly to Java, and Java attempts to use the specified font.

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description
`default.icons.style`		Specifies which set of icons and images is used by Forms. A new set of PNG images, including for the Forms runtime smart bar and alert icons, have been included with 14.1.2.0.0. To revert to the older GIF images stored in the Forms jarfile (`frmall.jar`), set `default.icons.style` to "legacy". In addition to new PNG files for the GIFs that are used in the Forms smart bat, new PNG files have also been provided for the Oracle logos that are provided for the menu bar, the splash screen, and the background image. Additionally, PNG files have also been provided for EWT alerts (error, warning, info) so that if the new style icons are being used, alerts will also have the new appearance. **Note:** Some icons are loaded before the registry is read. For this reason, these icons use the

Table 25 (Cont.) Additional Registry.dat Settings

Property	Default Value	Description
		new PNG files even if the new registry value is set to "legacy". These include the splash screen, the logo in the menu bar, and the background image.

Enabling Language Detection

Oracle Forms architecture supports deployment in multiple languages. The purpose of this feature is to automatically select the appropriate configuration to match a user's preferred language.

In this way, all users can run Oracle Forms applications using the same URL, yet have the application run in their preferred language. As Oracle Forms Services do not provide an integrated translation tool, you must have translated application source files.

The following sections are included:

- Specifying Language Detection
- How Language Detection Works
- Inline IME Support

Specifying Language Detection

For each configuration section in the **Web Configuration** page, you can create language-specific sections with names like `<config_name>.<language-code>`. For example, if you created a configuration section "hr", and wanted to create French and Chinese languages, your configuration section might look like the following:

```
[hr]
lookAndFeel=oracle
width=600
height=500
envFile=default.env
[hr.fr]
envFile=french.env
[hr.zh]
envFile=chinese.env
```

How Language Detection Works

When the Forms servlet receives a request for a particular configuration (for example, `http://myserv/servlet/frmservlet?config=hr`) it gets the client language setting from the request header "`accept-language`". This gives a list of languages in order of preference. For example, accept-language: de, fr, en_us means the order of preference is German, French, then US English. The servlet looks for a language-specific configuration section matching the first language. If one is not found, it looks for the next and so on. If no language-specific configuration is found, it uses the base configuration.

When the Forms servlet receives a request with no particular configuration specified (with no "`config=`" URL parameter, for example, `http://myserv/servlet/frmservlet`), it looks for a language-specific section in the default section matching the first language (for example, [.fr]).

Multi-Level Inheritance

For ease of use, to avoid duplication of common values across all language-specific variants of a given base configuration, only parameters which are language-specific to be defined in the language-specific sections are allowed. Four levels of inheritance are now supported:

1. If a particular configuration is requested, using a URL query parameter like `config=myconfig`, the value for each parameter is looked for in the language-specific configuration section which best matches the user's browser language settings (for example in section [myconfig.fr]).
2. Then, if not found, the value is looked for in the base configuration section ([myconfig]).
3. Then, failing that, in the language-specific default section (for example, [.fr]).
4. And finally in the default section.

Typically, the parameter which is most likely to vary from one language to another is `envFile`. Using a different `envFile` setting for each language lets you have different values of `NLS_LANG` (to allow for different character sets, date and number formats) and `FORMS_PATH` (to pick up language-specific fmx files).

Inline IME Support

Inline IME support enables Forms Web applications to properly display the composing text in which each character may not be directly represented by a single keystroke (for example, Asian characters) near the insertion cursor (so called inline, or on-the-spot). It is enabled by default. To disable, set the applet parameter "inlineIME" to "false" in the baseHTML file:

```
<HTML>
<!-- FILE: base.htm (Oracle Forms) -->
 <BODY>
 ...
 <OBJECT classid=...
>
<PARAM NAME="inlineIME" VALUE="false">
<EMBED SRC="" ...
inlineIME="false"
>
...
.
</BODY>
</HTML>
```

Enabling Key Mappings

A key binding connects a key to an application function. When you bind a key to a function, the program performs that function when you type that keystroke.

The Forms Key Bindings files can be customized from within Fusion Middleware Control. Alternatively, you can define key bindings manually in the `fmrweb.res` file in the `$DOMAIN_HOME/config/fmwconfig/components/FORMS/instances/<FORMS Instance Name>/admin/resource/<lang` directory in UNIX, for example `$DOMAIN_HOME/config/fmwconfig/components/FORMS/forms1/admin/resource/US`. For Windows, the location is `$DOMAIN_HOME/config/fmwconfig/components/FORMS/<FORMS Instance Name>`.

By defining key bindings, you can integrate a variety of keyboards to make an application feel similar on each of them. On some platforms not all keys are able to be re-mapped. For example, on Microsoft Windows, because keys are defined in the Windows keyboard device driver, certain keys cannot be re-mapped. Key combinations integral to Windows, such as Alt-F4 (Close Window) and F1 (Help) cannot be re-mapped. As a general rule, keys which are part of the "extended" keyboard also cannot be re-mapped. These keys include the number pad, gray arrow and editing keys, Print Screen, Scroll Lock, and Pause.

Note:

If running with different `NLS_LANG` settings, for example, `NLS_LANG=GERMAN_GERMANY=WE8ISO8859P15`, a different resource file, `fmrweb.res`, is used. There is a resource file for each supported language. To override it, pass parameter `term=fullpath\filename.res` to the Oracle Forms Runtime process.

It is possible to pass this parameter directly within the URL. For example:

```
http://hostname:port/forms/frmservlet?Form=test.fmx&term=fullpath/filename.res
```

You can also set this parameter in the `formsweb.cfg` file, for example:

```
otherParams=term=fullpath\filename.res
```

Managing Configuration Template and Key Binding Files

The Forms Enterprise Manager Fusion Middleware Control, Advanced Configuration page is used to add, edit, and delete configuration template and key binding files using a free text editor.

On the Forms Enterprise Manager Fusion Middleware Control Home page, click **Advanced Configuration** to the opens the Advanced Configuration page.

The Advanced Configuration page shows the **Select Category** and **Select File** list box. If you select a category in **Select Category** list box, all the related files are displayed in the **Select File** list box.

The following topics are included:

- Add, Edit, and Delete a Configuration Template File
- Editing Key Binding Files

Add, Edit, and Delete a Configuration Template File

Add or modify an existing configuration template file and delete a file on the Advanced Configuration page.

To create a file:

1. Click **Create Like** button to open a dialog box.
   Category and **Source File Name** fields are preselected based the file selected in **Select File** list box. If a file is not selected, you have to select a specific category.
2. Insert a name in **New File Name** field.
3. Click **Create** button, to create a new file.

> **Note:**
>
> If a file exists with the same file name, an error message is displayed.

To edit a file:

1. Select a file in **Select File** list box, to displays its content in Edit box.

> **Note:**
>
> The Edit box uses a free text editor.

2. In the Edit box update the content of a file and click **Apply**.
   Or,

   Click **Revert**, to discard the changes.

   > **Note:**
   >
   > The file contents are read and saved with the help of an mBean operation.

To delete a file:

1. Select a category in **Select Category** list box.
2. Select a file in **Select File** list box.
3. Click **Delete** button to delete a specific file.

> **Note:**
>
> If the selected file is a user created file, a **Delete** confirmation dialog box is displayed. If the selected file is a shipped file, an error message showing `Standard files cannot be deleted. Only user created files can be deleted`, is displayed.

Editing Key Binding Files

Edit Key Binding files from the Advanced Configuration page.

1. Click **Select Key Binding Files**.

   The **Select language** field is preselected based on the Key Binding file selected in **Select File** list box. If a file is not selected, you have to select a specific language in the **Select language** field. Files related to a specific language are displayed.

2. Select or clear check boxes in the **Select file(s)** list box as required.

   Standard and non-standard files are displayed in the **Select file(s)** list box. Standard files and files whose MBeans already exist are shown with a selected check box. Non-standard files are shown with an deselected check box.

   Select a different language to refresh the populated files list in the **Select file(s)** list box.

3. Click **Save** to create MBeans for any newly-selected items and delete MBeans for any newly-deselected items. MBeans are not deleted/added for the files whose selection are not changed.

4. Repeat the selection process for additional languages.

Customizing fmrweb.res Example

Example: Swapping Enter and Execute Mappings

In the section marked `USER-READABLE STRINGS`, find the entries with

```
122 : 0 : "F11" : 76 : "Enter Query"
122 : 2 : "Ctrl+F11" : 77 : "Execute Query"
```

and change them to:

```
122 : 2 : "Ctrl+F11" : 76 : "Enter Query"
122 : 0 : "F11" : 77 : "Execute Query"
```

> **Note:**
>
> By default, `fmrweb.res` does *not* reflect the Microsoft Windows client/server keyboard mappings. It reflects the key mapping if running client/server on UNIX X-Windows/ Motif.

A file named `fmrpcweb.res` has also been provided which gives the Microsoft Windows client/ server keyboard mappings. To use this file, set the term parameter in the Web Configuration to reference the desired file name. Do not include the file extension; for example, "term=fmrpcweb".

Exceptions or Special Key Mappings

To map special key like F2, ENTER, Number Keys and ESC keys follow the instructions and examples provided in the section for each special key.

These sections are included:

- Mapping F2
- Mapping for ENTER to Fire KEY-ENTER-TRIGGER
- Mapping Number Keys

Mapping F2

To map **F2**, change the default entry for **F2**, "List Tab Pages", to another key. Here is an example of the default entry:

```
113: 0 : "F2" : 95 : "List Tab Pages"
```

This must be explicitly changed to another key mapping such as the following:

```
113: 8 : "F2" : 95 : "List Tab Pages"
```

To map the **F2** function to the **F2** key, comment out the lines that begin with "113 : 0" and "113: 8" with a # symbol and add the following lines to the bottom of the resource file:

```
113: 0 : "F2" : 84 : "Function 2"
113: 8 : " " : 95 : " "
```

Since a new function has been added which uses **F2** by default, it is necessary to explicitly map this new function to something else to map the **F2** key. This function was added to allow for keyboard navigation between the tab canvas pages and it defaults to **F2**. Even if it is commented out and not assigned to **F2**, the **F2** key cannot be mapped unless this function, Forms Function Number 95, is mapped to another key.

Mapping for ENTER to Fire KEY-ENTER-TRIGGER

By default, whether deploying client/server or over the Web pressing the **ENTER** key takes the cursor to the next navigable item in the block. To override this default behavior it is necessary to modify the forms resource file to revise the key mapping details.

Modify `fmrweb.res` and change the Forms Function Number (FFN) from 27 to 75 for the Return Key. The line should be changed to the following:

```
10 : 0 : "Return" : 75 : "Return"
```

By default, the line is displayed with an FFN of 27 and looks as follows:

```
10 : 0 : "Return" : 27 : "Return"
```

This line should NOT fire the Key-Enter trigger since the Return or Enter key is actually returning the Return function represented by the FFN of 27. The FFN of 75 represents the Enter function and fires the Key-Enter trigger.

Mapping Number Keys

The objective is to map CTRL+<number> keys in fmrweb.res for numbers 0 to 9 and there are no Java Function keys mentioned for the numbers in `fmrweb.res`. Perform the following steps along with an example that shows the steps needed to map CTRL+1 to 'Next Record':

1. List the Java function key numbers that could be implemented in `fmrweb.res` file for the Key Mapping. For example:

   ```
   public static final int VK_1 = 0x31;
   ```

2. The hexadecimal values have to be converted to their decimal equivalents before their use in `fmrweb.res`.

   In step (1), 0x31 is a hexadecimal value that has to be converted to its decimal equivalent. (Note:1019580.6). For example,

   ```
   SQL> select hextodec('31') from dual;
   HEXTODEC('31')
   --------------
   49
   ```

3. Use this decimal value for mapping the number key 1 in `fmrweb.res`. For example, CTRL+1 can be mapped to 'Next Record' as:

   ```
   49 : 2 : "CTRL+1" : 67 : "Next Record"
   ```

Oracle Forms Configuration Helper Script

The Oracle Forms Configuration Helper script `frmconfighelper` helps administrators easily perform typically complex post install, Forms configuration tasks.

Before using the Helper script, it is assumed that you have successfully installed Oracle Forms 14*c* and completed its initial configuration using the `config.sh` or `config.bat` script. Use of this script is preferred over attempted to make these changes manually.

The Oracle Forms Helper Script is located in `MW_HOME\forms\provision` directory.

For more information about using this script, run the script without any arguments. This will present detailed usage information.

The script includes the following functions:

- **enable_ohs**: Enables routing for the location/forms from OHS to the Forms managed server(s) under cluster cluster_forms.

  > **Note:**
  >
  > When enable routing to multiple Forms managed servers, provide the managed server host port information in sequence.

- **deploy_app**: Deploys `formsapp ear` file, overriding the context-root and servlet alias to the specified managed server.
- **update_app**: Updates a deployed app (with an overridden context root) after applying FMW Forms Services patches.
- **enable_webgate**: Used only if partner app is registered via OAM console. This should not be used if `enable_sso` was previously used.
- **create_machine**: Creates a new (remote) WLS machine for custom Forms application deployment (see create_managed_server).
- **create_managed_server**: Creates a new managed server for custom Forms application deployment (see deploy_app).
- **enable_sso**: Enables Webgate configuration in the OHS instance, performs partner application registration using OHS SSL and non-SSL ports and copies over the Webgate artifacts to the OHS instance.

> **Note:**
>
> Make sure to back up the Domain before performing any administration tasks on it using this script.

Table 26   `frmconfighelper` script

Option	When to use it	What it does	Components Requiring Restart
`enable_webgate`	When you have added any new OHS instances and you want to individually enable Webgate configuration on the OHS instance. This command should not be used if the *enable_sso* command was previously used.	Enables Webgate configuration on the OHS instance.	- Admin Server - OHS

`deploy_app`	After you have run the Config Wizard and, want to deploy the Forms javaEE application again to override the default context-root and the Forms servlet alias.
	Example: The default Forms JavaEE application access URL is:
	`http(s)://host:port/forms/frmservlet`
	If you override the context-root to sales and Forms Servlet alias to salesservlet, the application access URL will be:
	`http(s)://host:port/sales/salesservlet`
		- Overrides the Forms JavaEE application context-root, Forms Servlet alias and packages the Forms JavaEE application into a new ear file.
		- Deploys the ear file to the Weblogic Domain and activates it the Managed Server.
		You need to create the Managed Server before you run `deploy_app` option.
			- Admin Server
			- Managed Server associated with deployment

`update_app`	When you ran deploy_app option and you want to update the custom application after a patch release.	Updates the JavaEE custom ear file created with deploy_app option after a patch release.	- Admin Server - Managed Server associated with deployment
`enable_ohs`	When you have created a new OHS instance and you want to enable routing to a Forms Managed server.	Adds Managed server routing directives to the template forms.conf, copies it over to the OHS instance.	- Admin Server - OHS
`create_machine`	Used when working with a remote node or if a default machine is not desirable for adding managed servers for custom deployments created by this utility.	Creates a new WLS machine.	

Table 26 (Cont.) `frmconfighelper` script

Option	When to use it	What it does	Components Requiring Restart
`create_managed_server`	Used to create a custom managed server that will host a customized Forms J2EE app deployment. This managed server is intended to be used with the custom app deployed created by this utility (see deploy_app). This function should not be used for creating generic, non-Forms servers.	Creates a custom managed server.	
`enable_sso`	Used to enable SSO with SSL in the Forms environment.	• Enables OHS-Forms Managed Server routing. • Enables Webgate configuration on the OHS instance. • Performs partner application registration on the OHS instance. • Creates a policy on the OAM server to protect the Forms and Reports application.	• Admin Server • WLS_FORMS • OHS

- **Syntax**
  `frmconfighelper.cmd` *`<option> <arguments>`*
- **Options**
  - `enable_ohs` *`<domain-home> <ohs-instance> <forms-managed-server1-host> <forms-managed-server1-port> <forms-managed-server2-host> <forms-managed-server2-port>`*
  - `deploy_app` *`<new-context-root> <new-servlet-alias> <managed-server>`*
  - `update_app` *`<Forms-context-root> <Forms-servlet-alias>`*
  - `enable_webgate` *`<domain-home> <ohs-instance>`*
  - `create_machine` *`<wls-machine-name> <machine-host-name>`*
  - `create_managed_server` *`<mananged-server-name> <wls-machine-name> <managed-server-port> <standalone>`*
  - `enable_sso` *`<oam-host> <oam-port> <ohs-host> <ohs-ssl-port> <ohs-non-ssl-port> <domain-home> <ohs-instance>`*

Argument Description

The argument description of each of the functions included in the script are provided in this section.

The details of argument description are as follows:

- **enable_ohs**
  - *domain-home*: Domain Home directory
  - *ohs-instance*: OHS instance name (example ohs1)
  - *forms-managed-server(n)-host*: Forms managed server host
  - *forms-managed-server(n)-port*: Forms managed server port
- **deploy_app or update_app**
  - *new-context-root*: new context root for the formsapp
  - *new-servlet-alias*: new servlet alias for the formsservlet
  - *managed-server*: target managed server for the new application
- **enable_webgate**
  - *domain-home*: Domain Home directory
  - *ohs-instance*: OHS instance name (example ohs1)
- **create_machine**
  - *machine-name*: WLS machine name
  - *host-name*: Remote WLS machine host name
- **create_managed_server**
  - *managed-server-name*: Managed server name
  - *wls-machine-name*: WLS machine name
  - *managed-server-port*: Managed server port number
  - *standalone* (optional): Indicates standalone managed server which is not part of any cluster.
- **enable_sso**
  - *oam-host*: OAM Server host name
  - *oam-port*: OAM Server port number
  - *ohs-host*: OHS host name
  - *ohs-ssl-port*: OHS SSL port number
  - *ohs-non-ssl-port*: OHS NON-SSL port number
  - *domain-home*: Domain Home directory
  - *ohs-instance*: OHS instance name (example ohs1)

Examples of each Function

Follow the examples provided for each of the functions include in the script.

- **enable_ohs**

  ```
  frmconfighelper.sh enable_ohs /middleware/user_projects/domain/base_domain
  ohs1 wlshost.example.com 9001 wlshost.example.com 9010
  ```

- **deploy_app**

  ```
  frmconfighelper.sh deploy_app sales salesservlet WLS_FORMS3
  ```

- **update_app**

  ```
  frmconfighelper.sh update_app sales salesservlet
  ```

- **enable_webgate**.

  ```
  frmconfighelper.sh enable_webgate /middleware/user_projects/domain/
  base_domain ohs1
  ```

- **create_machine**

  ```
  frmconfighelper.cmd create_machine SalesRemoteMachine remotehostname
  ```

- **create_managed_server** (2 examples)

  ```
  frmconfighelper.cmd create_managed_server WLS_SALES AdminServerMachine 9010
  ```

  ```
  frmconfighelper.cmd create_managed_server WLS_FINANCE AdminServerMachine
  9020 standalone
  ```

- **enable_sso**

  ```
  frmconfighelper.cmd enable_sso_ssl oamhost.example.com 7001
  ohshost.example.com 4443 7777
  /middleware/user_projects/domain/base_domain ohs1
  ```

Part V

Integration

This part covers the integrations of Forms services with other applications, such as Oracle Access Manager, Java virtual machines and Oracle WebLogic Server.

Specifically, this part contains the following chapters:

- Using Oracle Forms Services with the HTTP Listener and Oracle WebLogic Server
- Oracle Forms and JavaScript Integration
- Enhanced Java Support
- Working with Server and System Events
- Using Forms Services with Oracle Access Manager
- Integrating Oracle Forms with IAM Cloud Service
- Configuring and Managing Java Virtual Machines

Using Oracle Forms Services with the HTTP Listener and Oracle WebLogic Server

Oracle WebLogic Server is a scalable, enterprise-ready Java EE application server. It implements the full range of Java EE technologies, and provides many more additional features such as advanced management, clustering, and Web services. It forms the core of the Oracle Fusion Middleware platform, and provides a stable framework for building scalable, highly available, and secure applications.
Oracle HTTP Server (OHS) is the Web server component for Oracle Fusion Middleware. It can be used as a front-end listener (e.g. proxy) for Oracle WebLogic Server and the framework for hosting static pages, dynamic pages, and applications over the Web.

This chapter contains the following sections:

- About WebLogic Managed Servers and HTTP Server
- Using HTTPS with the Forms Listener Servlet
- Enabling SSL
- Enabling SSL with a Load Balancing Router
- Work with Forms Managed Server
- Performance/Scalability Tuning
- Load Balancing Oracle WebLogic Server
- Using an Authenticating Proxy to Run Oracle Forms Services Applications

About WebLogic Managed Servers and HTTP Server

Managed Servers host business applications, application components, Web services, and their associated resources.

To optimize performance, managed servers maintain a read-only copy of the domain's configuration document. When a managed server starts up, it connects to the domain's administration server to synchronize its configuration document with the document that the administration server maintains. Oracle Fusion Middleware system components (such as Forms, Reports, Oracle HTTP Server, and Identity Management components), as well as customer-deployed applications, are deployed to managed servers in the domain. During configuration, some managed servers are created specifically to host the Oracle Fusion Middleware applications (for example, Forms JavaEE applications).

Figure 14   Oracle WebLogic Managed Server and Forms Services

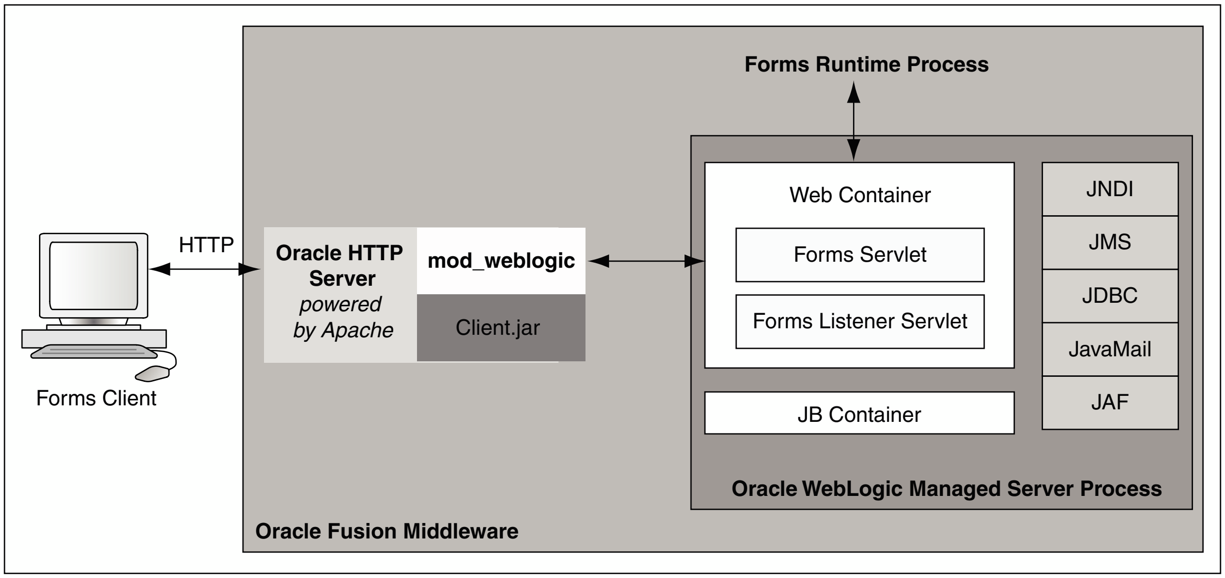


Figure 14 shows the HTTP request flow from WebLogic Managed Server to Forms Runtime Process. On the left of the image, resides the Forms client. The Forms servlet renders the start HTML file and provides the information about the Forms Listener servlet to the client. The client passes this HTTP request to the Oracle HTTP Server Listener in the middle. The Oracle HTTP Server Listener, in turn, passes the HTTP request to the Forms Listener servlet running inside WebLogic Managed Server. The WebLogic Managed Server Process includes a Web Container, JB Container, and other services, such as Java Naming and Directory Interface (JNDI), JMS, JavaMail, and so forth. The Forms Listener servlet establishes a Forms Server runtime process and is responsible for on-going communication between the client browser and the runtime process.

Enabling Oracle HTTP Server with Oracle Forms Services

In Oracle Fusion Middleware, to enable Oracle HTTP Server to route requests to the Forms Managed Server manual, post installation steps are required.

Users have two options to enable this configuration.

- Use the Forms Configuration Helper Script, as described in Oracle Forms Configuration Helper Script and, pass it the enable_ohs option.
- Manually edit forms.conf, as described in About Editing forms.conf.

About Editing forms.conf

`forms.conf` is an Oracle HTTP Server directives file. In Oracle Fusion Middleware, the `forms.conf` file should be included in the Oracle HTTP Server configuration directory at `$DOMAIN_HOME/config/fmwconfig/components/OHS/<OHS INSTANCE NAME>/moduleconf`.

If you add any custom Oracle HTTP Server directives to `forms.conf`, you must restart the Oracle HTTP Server node where it resides.

Configuring OHS

If you choose to configure Oracle HTTP Server, then perform the following tasks:

1. Copy the Forms OHS directives file `forms.conf` from the Forms configuration files templates directory to the OHS instance `moduleconf` directory

   Source location (on Forms tier):

   `$FMW_HOME/forms/templates/config/forms.conf`

   Destination location (on OHS tier):

   `$DOMAIN_HOME/config/fmwconfig/components/OHS/<OHS INSTANCE NAME>/moduleconf`

2. Specify the appropriate managed server cluster or the managed server for the default forms Java EE application context root (`/forms`).

   Example of cluster entry:

   ```
   <Location /forms/>
           SetHandler weblogic-handler
           WebLogicCluster    <HOSTNAME>:<WLS_PORT>,<HOSTNAME>:<WLS_PORT>
           DynamicServerList OFF
   </Location>
   ```

   Example of non-cluster entry:

   ```
   <Location /forms/>
           SetHandler weblogic-handler
           WebLogicHost = <HOSTNAME>
           WebLogicPort = <PORT>
   </Location>
   ```

3. Make sure that any directories referenced in user-added directives are accessible on the OHS tier.
4. Restart the Admin Server.
5. Restart OHS instance on the OHS tier.

Note:

Use the Oracle Fusion Middleware Control to make further changes to forms.conf once it is setup. You can access it using OHS Instance page Menu: Oracle HTTP Server > Administration > Advanced Configuration. On the Advanced Server Configuration page, choose forms.conf in the Choose a File pull down Menu.

When including any user-defined `aliasMatch` with the prefix `/forms/` in `forms.conf`, add the directive `WLExcludePathOrMimeType`. For example, in Linux, when defining the `aliasMatch` for `/forms/usericons` in `forms.conf`, the directive `WLExcludePathOrMimeType` is defined as following:

```
AliasMatch /forms/usericons/(..*) "/home/userx/myicons/$1"
```

```
WLExcludePathOrMimeType /forms/usericons/
```

Using HTTPS with the Forms Listener Servlet

Using HTTPS with Oracle Forms is no different than using HTTPS with any other Web-based application.

HTTPS requires the use of digital certificates (for example, DigiCert or VeriSign). Because Forms Services servlets are accessed via your Web server, you do not need to purchase special certificates for communications between the Oracle Forms client and the server. You only need to purchase a certificate for your Web server from a recognized certificate authority.

Oracle recommends that for ensuring the highest level of security between your end-user and middle tier, Secure Socket Layer (SSL) should be configured. For details on how to enable SSL in your environment, see Managing Application Security in *Administering Oracle HTTP Server* and Configuring SSL in Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

Enabling SSL

You have to perform specific steps to run Oracle Forms Services applications in SSL mode.

Perform the following steps:

- Create a Wallet to manage certificates.
- Enable the HTTPS port in Oracle HTTP Server. By default, Oracle HTTP Server has one SSL Port enabled.
- Optionally, consider enabling HTTPS in WebLogic Server for the Forms managed server (for example, WLS_FORMS).

Note:

See Configuring SSL in Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

Enabling SSL with a Load Balancing Router

Running a Forms application that uses an HTTPS port requires a certificate to be imported. If Oracle Forms is behind a load balancing router, and SSL terminates at it, you need to import the certificate from the load balancing router.

To enable SSL with your Forms applications over a load balancing router:

1. Start a Web browser and enter the Forms application HTTPS URL containing the fully qualified host name (including port number if required) used by your own Oracle installation. For example: `https://example.com:443/forms/frmservlet`

   The Security Alert dialog box is displayed.
2. Click **View Certificate**.
3. Click the **Details** tab in the Certificate dialog.
4. Click **Copy to File...**
5. In the Welcome page of the Certificate Export Wizard, click **Next**.
6. In the Export File Format page, select **Base-64 encoded X.509 (.CER)**, then click Next.
7. Enter a file name such as `c:\temp\forms`, then click Next.
8. Click **Finish**.

   A message appears saying that the export was successful.
9. Click **OK**.
10. Close the Certificate Export Wizard, but keep the Security Alert dialog open.
11. Import the security certificate file that you saved earlier into the certificate store of the JVM you are using.
12. At the Security Alert dialog, click Yes to accept the security certificate and start the Forms application.

To import the certificate into Java Plugin:

1. On the client machine, open the Control Panel.
2. Open Java.
3. Navigate to Securities tab.
4. Click Certificate.
5. Import the certificate that was exported in the previous section.
6. Click Apply.

Work with Forms Managed Server

By default (out-of-the-box installation), the Forms Services Java EE application (`formsapp.ear`) is deployed on Forms Managed Server (`WLS_FORMS`).

You can manage `WLS_FORMS` and `formsapp.ear` using Oracle WebLogic Administration Console or Oracle Fusion Middleware Control. Refer to the following links:

- Starting and Stopping Forms Managed Server, as described in Overview of Starting and Stopping Procedures in *Administering Oracle Fusion Middleware*

- Deploying Forms Application to Forms Managed Server, as described in Configuring Forms Using the Configuration Wizard.
- Custom deployment of Forms Java EE application, as described in Custom Deployment of Forms Java EE Application.
- Expanding Forms Managed Server Clusters, as described in Expanding Forms Managed Server Clusters.
- Modifying `weblogic.xml`, `web.xml`, `application.xml` and `weblogic-application.xml` post deployment, as described in Modifying of Forms J2EE Application Deployment Descriptors.
- Starting Forms Managed Server as a Windows Service, see Setting Up a WebLogic Server Instance as a Windows Service in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

Custom Deployment of Forms Java EE Application

Users can override the default Forms JavaEE application context root (/forms) and the default Forms servlet alias (frmservlet) and customize it.

The default Forms applications access URL: `http://host:port/forms/frmservlet` can be changed to `http://host:port/<user-context>/<user-servlet-alias>`.

To create a custom managed server and deploy Forms application on it, perform the following steps:

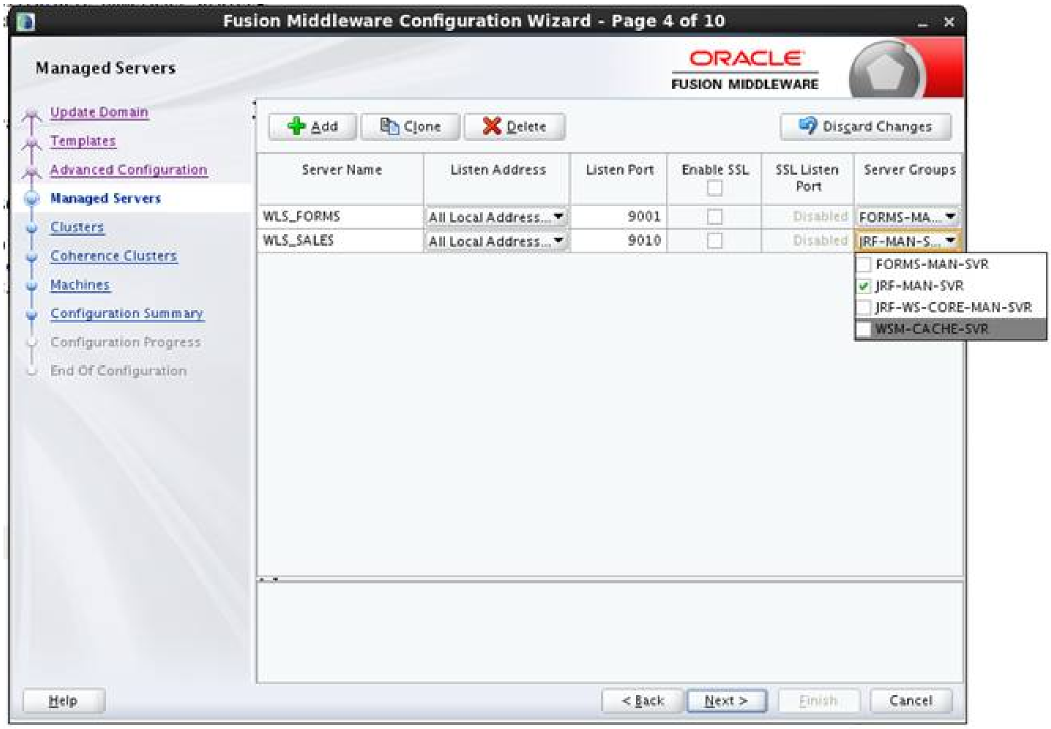
- Creating and deploying custom application
- Post-Patching Tasks
- Testing the Custom Deployment

Creating and deploying custom application

To create and deploy custom application perform the following steps:

1. Create a separate managed server using the config wizard. This managed server should not be a part of the default Forms cluster (cluster_forms) and it should use the `JRF_MAN_SRV` server group selected.

Figure 15 Create a Separate Managed Server



Server Name	Listen Address	Listen Port	Enable SSL ☐	SSL Listen Port	Server Groups
WLS_FORMS	All Local Address...	9001	☐	Disabled	FORMS-MA...
WLS_SALES	All Local Address...	9010	☐	Disabled	JRF-MAN-S...

☐ FORMS-MAN-SVR
☑ JRF-MAN-SVR
☐ JRF-WS-CORE-MAN-SVR
☐ WSM-CACHE-SVR

2. Run the frmconfighelper script using the deploy_app option, .

[Oracle Forms Configuration Helper Script]

Post-Patching Tasks

If additional managed servers for Forms have been created and any Oracle Forms specific patches have been applied, follow the steps below. This will ensure that any updates to the Forms servlet included in the patch(es) are applied to the added managed servers.

1. Ensure that the servers in the Domain have been stopped.
2. Run the `frmconfighelper` script using the `update_app` option after applying the patch.
3. The managed server has to be re-started after running the `update_app` option to take effect.

For information on the `frmconfighelper` script, see Oracle Forms Configuration Helper Script.

Testing the Custom Deployment

Test the deployment using the URL: `http://<Host>:<Port Number>/<context root>/<servlet name>`.

For the example in this section, the URL would be `http://<Host>:<Port Number>/customapp/customservlet`. In the case that you are running form with SSO (`ssoMode=true` or `webgate`), additional settings with permissions are needed in: `DOMAIN_HOME/config/fmwconfig/system-jazn-data.xml` file.

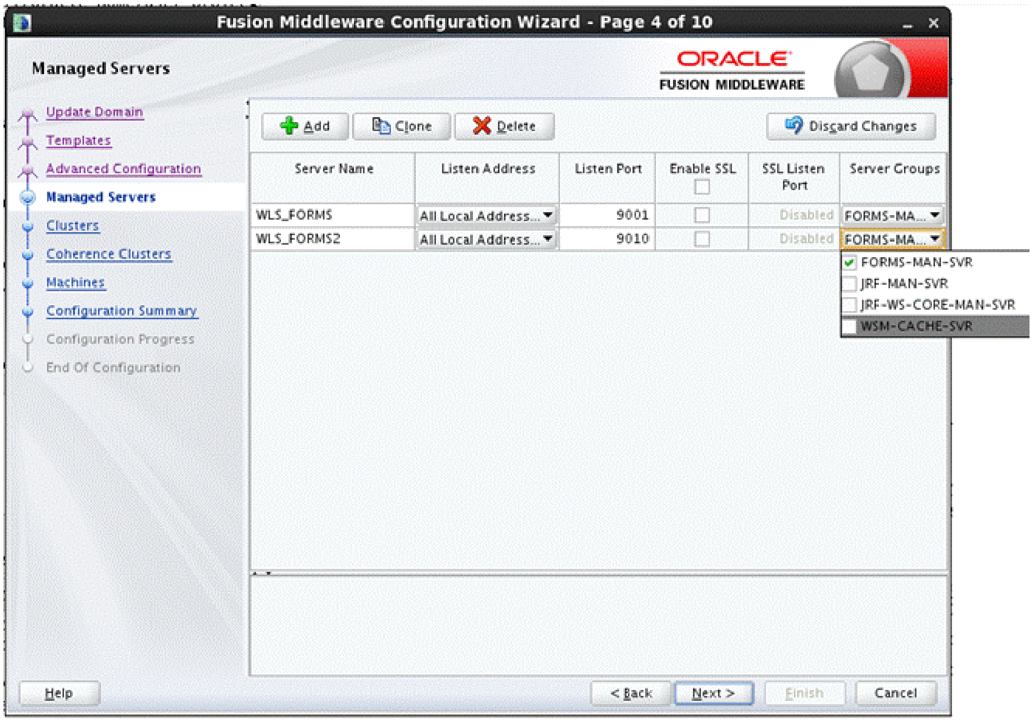
Expanding Forms Managed Server Clusters

To improve the scalability and performance of Forms deployments on high-end machines (multiprocessor and high-memory configuration machines), expand the Forms Managed Server cluster (`cluster_forms`). Perform the following manual steps to expand the Forms Managed Server cluster:

1. Perform the following steps to add a new Managed Server to the default Forms application cluster (`cluster_forms`):

   a. Add additional Managed Server(s) using the config wizard. Make sure that you select the FORMS-MAN-SRV Server Group.

Figure 16   Adding Managed Server(s)



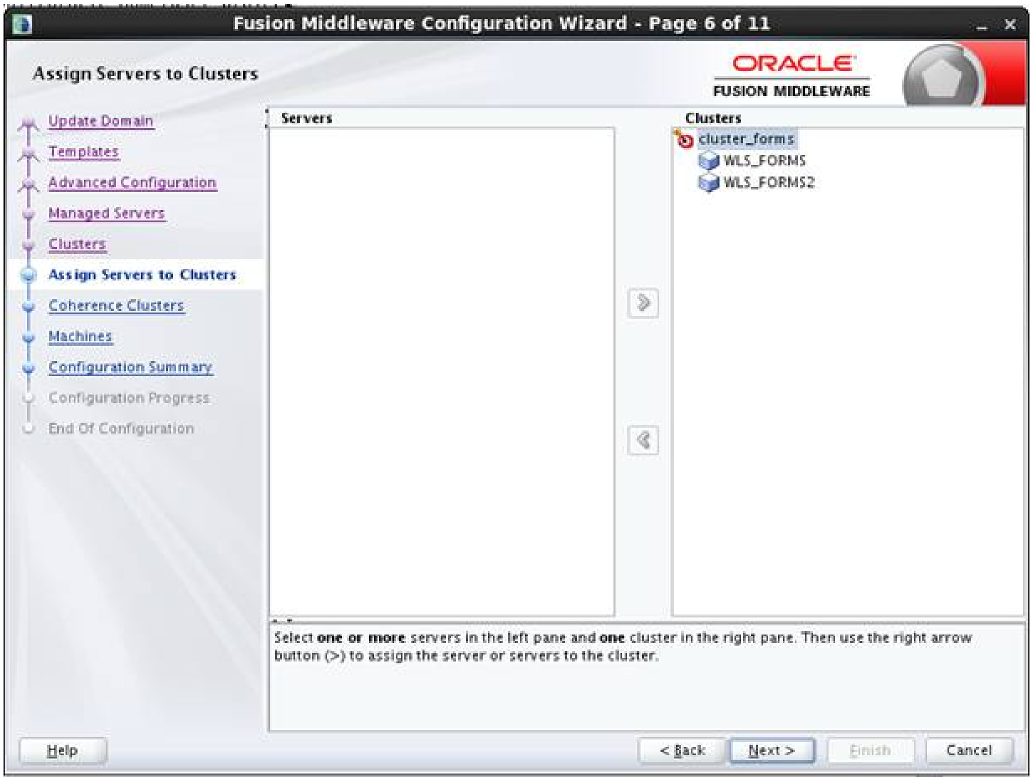
Server Name	Listen Address	Listen Port	Enable SSL	SSL Listen Port	Server Groups
WLS_FORMS	All Local Address...	9001	□	Disabled	FORMS-MA...
WLS_FORMS2	All Local Address...	9010	□	Disabled	FORMS-MA...

☑ FORMS-MAN-SVR
□ JRF-MAN-SVR
□ JRF-WS-CORE-MAN-SVR
□ WSM-CACHE-SVR

   Adding a new manager server.

   b. Ensure that it is added to the cluster_forms after creating the Managed Server.

Figure 17   Assigning Servers to Clusters



Select **one or more** servers in the left pane and **one** cluster in the right pane. Then use the right arrow button (>) to assign the server or servers to the cluster.

Adding a new managed server.

c. Start the newly created Managed Server.

2. Add the new Managed Server's host and port information to the WebLogicCluster entry in `forms.conf`:

   ```
   <Location /forms>

   SetHandler weblogic-handler

   WebLogicCluster <HostName>:9001, <HostName>:9010

   DynamicServerList OFF

   </Location>
   ```

3. Restart OHS.

Creating Multiple Forms System Component Instances on the Same Physical Machine

If you set up more than one Forms System Component Instance on the same physical machine, then the Forms managed server should be associated with its respective Forms System Component Instance.

This setup can be created by defining `forms.instance` system property on the Forms managed server and setting it to Forms System Component Instance name.

For example:

```
Machine 1     forms1     WLS_FORMS
        forms2     WLS_FORMS2
```

Set the `forms.instance` system property on `WLS_FORMS1` to `forms1`. Similarly, set `forms.instance` system property on `WLS_FORMS2` to `forms2`. This can be done using the Managed Server setting in the Oracle WebLogic Remote Console.

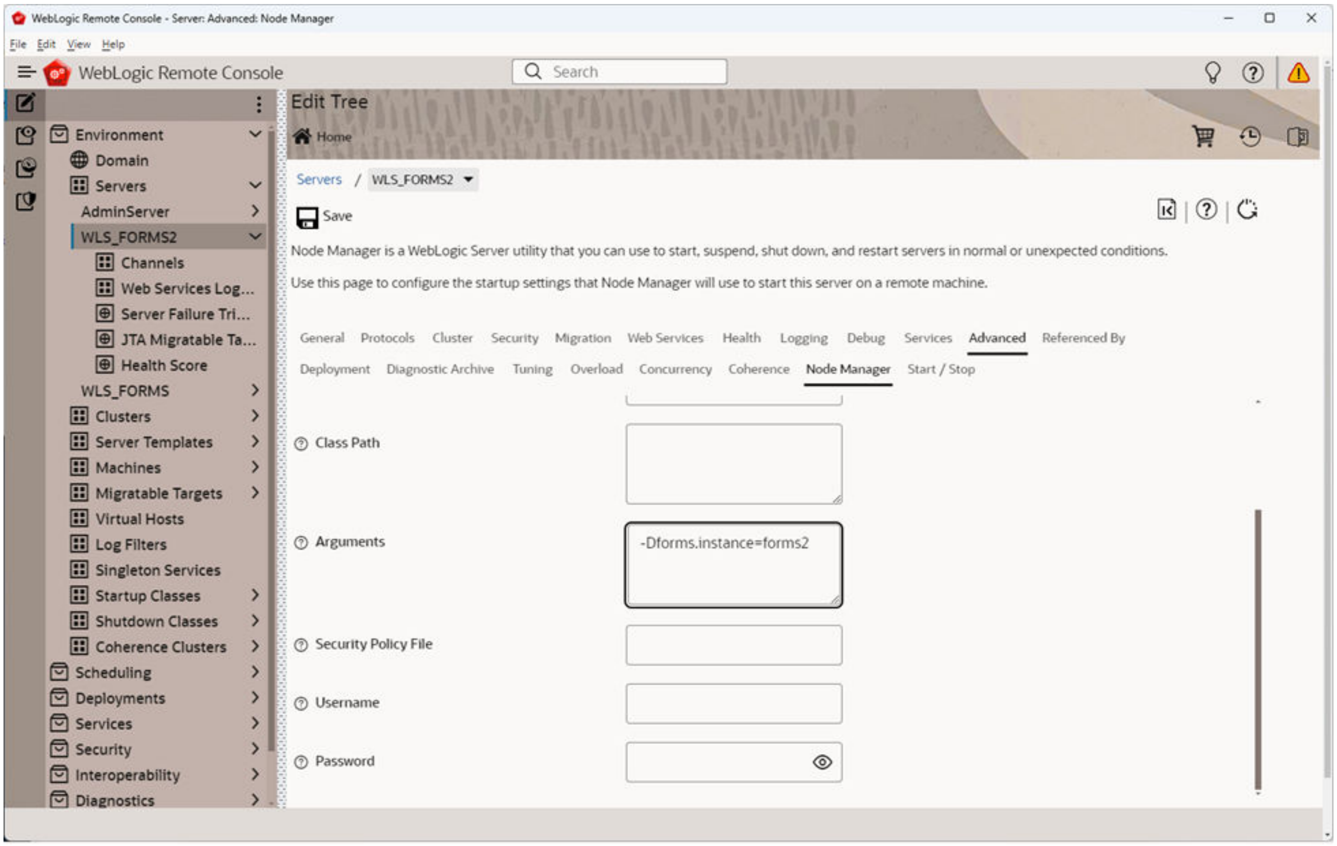
Perform these steps:

1. From the WebLogic Remote Console, select the **Edit Tree** vertical tab on the left pane.
2. Click **Environment**, then **Servers** to expand both nodes.
3. Select the managed server (for example, `WLS_FORMS2`) for editing.
4. Click the **Advanced** tab, then the **Node Manager** subtab.
5. In the **Arguments** field, enter desired instance setting and name as follows:

   ```
   -Dforms.instance=forms2
   ```

6. Click **Save** to accept the changes.
7. Click the shopping cart icon on the upper right, then **Commit Changes**.
8. Restart any managed servers that have been updated.

Figure 18 Managed Server setting in the Oracle WebLogic Server Administration Console



Node Manager is a WebLogic Server utility that you can use to start, suspend, shut down, and restart servers in normal or unexpected conditions.

Use this page to configure the startup settings that Node Manager will use to start this server on a remote machine.

Class Path

Arguments: -Dforms.instance=forms2

Security Policy File

Username

Password

Modifying of Forms J2EE Application Deployment Descriptors

To customize the Forms J2EE application deployment descriptors, add the deployment descriptors customizations to the Forms J2EE application's deployment plan and update the application in place with the new deployment plan changes.

Post-deployment, Forms J2EE application deployment descriptors (`weblogic.xml`, `web.xml`, `application.xml` and `weblogic-application.xml`) cannot be modified in Oracle WebLogic Server.

As a workaround, perform these steps:

1. Back up the default formsapp deployment plan, `$DOMAIN_HOME/config/fmwconfig/deployment-plans/formsapp/14.1.2/plan.xml`.
2. Add the deployment descriptors customizations to the Forms J2EE application's deployment plan.

   > **Note:**
   >
   > See the following example on how to modify a deployment plan.
   >
   > To update the deployment plan, see *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server.*

3. Using the WebLogic Administration Console, update the Forms application (redeploy) and select the option **Update this application in place with new deployment plan changes**.
4. Restart the Forms J2EE application using the WebLogic Administration Console.

Example: Modifying a Deployment Plan

In this example, the deployment plan is modified to override the Forms Servlet `testMode` parameter and set it to true. To do this:

1. Enter the following commands:

   ```
   mkdir -p $FMW_HOME/forms/j2ee/backup
   cd $FMW_HOME/forms/j2ee
   cp $DOMAIN_HOME/config/fmwconfig/deployment-plans/formsapp/14.1.2/plan.xml
   vi $DOMAIN_HOME/config/fmwconfig/deployment-plans/formsapp/14.1.2/plan.xml
   ```

2. Modify the deployment plan. The following is a sample of the deployment plan with the added entries highlighted in bold:

   ```
   <deployment-plan xmlns="http://xmlns.oracle.com/weblogic/deployment-plan"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://
   xmlns.oracle.com/weblogic/deployment-plan http://xmlns.oracle.com/weblogic/
   deployment-plan/1.0/deployment-plan.xsd">
     <variable-definition>
   <variable>
         <name>vd-/scratch/t_work/Oracle/Middleware/Oracle_Home/forms</name>
         <value>/scratch/t_work/Oracle/Middleware/Oracle_Home/forms</value>
      </variable>
       <variable>
          <name>FormsServlet_InitParam_testMode</name>
          <value>true</value>
       </variable>
     </variable-definition>
   ```

```
  <application-name>formsapp</application-name>
  <module-override>
    <module-name>formsapp.ear</module-name>
    <module-type>ear</module-type>
    <module-descriptor external="false">
      <root-element>weblogic-application</root-element>
      <uri>META-INF/weblogic-application.xml</uri>
    </module-descriptor>
    <module-descriptor external="false">
      <root-element>application</root-element>
      <uri>META-INF/application.xml</uri>
    </module-descriptor>
    <module-descriptor external="true">
      <root-element>wldf-resource</root-element>
      <uri>META-INF/weblogic-diagnostics.xml</uri>
    </module-descriptor>
  </module-override>
  <module-override>
    <module-name>formsweb.war</module-name>
    <module-type>war</module-type>
    <module-descriptor external="false">
      <root-element>weblogic-web-app</root-element>
      <uri>WEB-INF/weblogic.xml</uri>
                    <variable-assignment>
       <name>vd-/scratch/t_work/Oracle/Middleware/Oracle_Home/forms</name>
                    <xpath>/weblogic-web-app/virtual-directory-mapping/[url-
pattern="java/*"]/local-path</xpath>
                </variable-assignment>
              </variable-assignment>
        <name>vd-/scratch/t_work/Oracle/Middleware/Oracle_Home/forms</name> <xpath>/
weblogic-web-app/virtual-directory-mapping/[url-pattern="webutil/*"]/local-path</
xpath>
      </variable-assignment>
    </module-descriptor>
    <module-descriptor external="false">
      <root-element>web-app</root-element>
      <uri>WEB-INF/web.xml</uri>
      <variable-assignment>
        <name>FormsServlet_InitParam_testMode</name>
<xpath>/web-app/servlet/[servlet-name="frmservlet"]/init-param/[param-
name="testMode"]/param-value</xpath>
      </variable-assignment>
    </module-descriptor>
  </module-override>
</deployment-plan>
```

3. Restart the Forms J2EE application using the WebLogic Administration Console.

Performance/Scalability Tuning

The steps for tuning the Forms Listener servlet are similar to steps for tuning any high throughput servlet application.

You have to take into account resource management and user needs for optimal tuning of your particular Forms Services configuration, see Monitoring in *Tuning Performance*.

Load Balancing Oracle WebLogic Server

The Forms Listener servlet architecture allows you to load balance the system using any of the standard HTTP load balancing techniques available.

The Oracle HTTP Server Listener provides a load balancing mechanism that allows you to run multiple WebLogic instances on the same host as the HTTP process, on multiple, different hosts, or on any combination of hosts. The HTTP Listener then routes HTTP requests to Oracle WebLogic Managed Server instances.

The following scenarios are just a few of the possible combinations available and are intended to show you some possibilities. The best choice for your site will depend on many factors. For a complete description of this feature, see Monitoring in *Tuning Performance*.

The following images show four possible deployment scenarios.

Figure 19 Multiple Oracle WebLogic Servers on the same host as the Oracle HTTP Listener

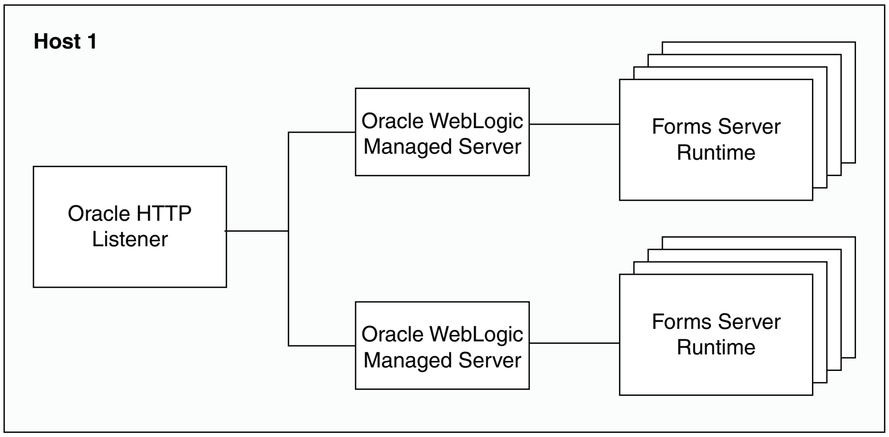


Figure 20 Multiple Oracle WebLogic Servers on a different host to the Oracle HTTP Listener

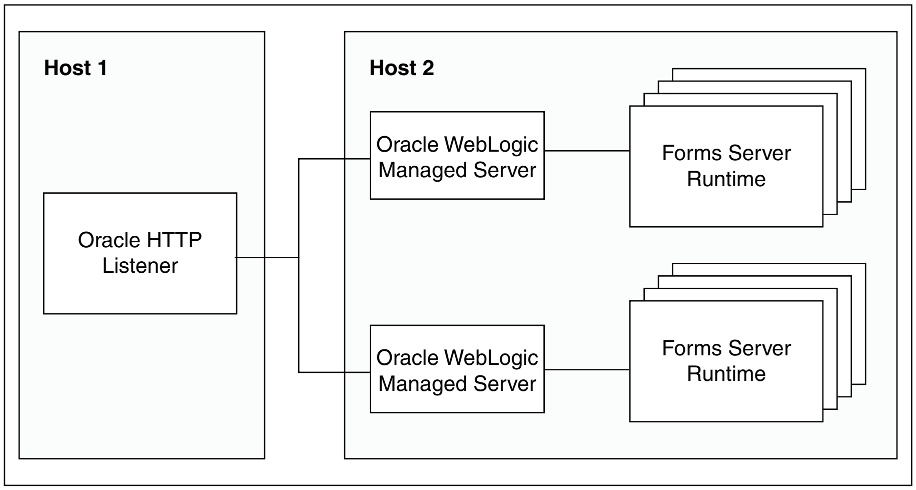


Figure 21   Multiple Oracle WebLogic Servers and multiple Oracle HTTP Listeners on different hosts

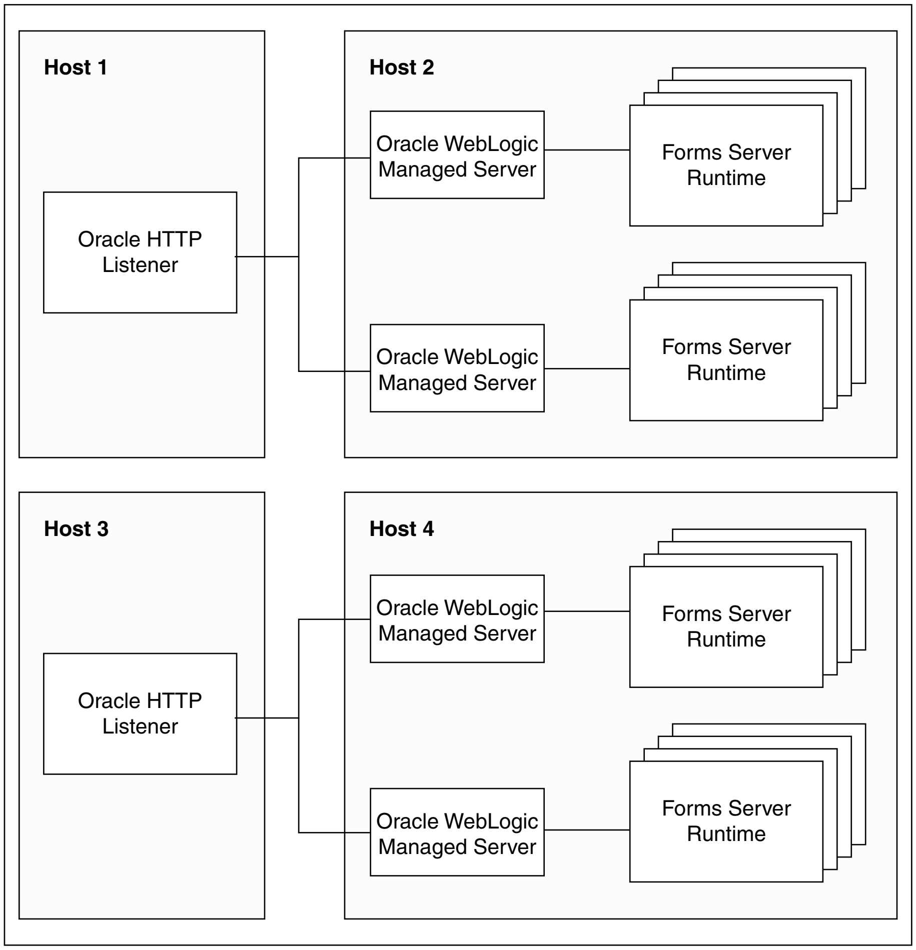
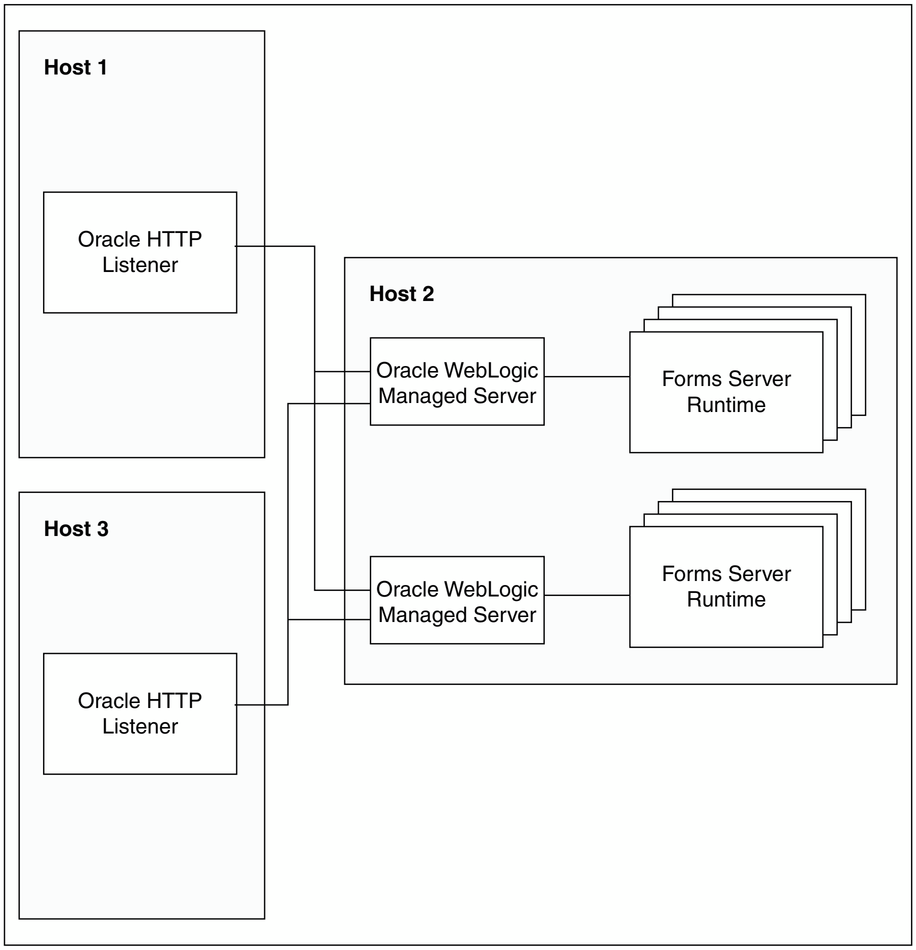


Figure 22 Multiple Oracle HTTP Listeners on different hosts with multiple Oracle WebLogic Servers on one host



Note:

To tune and optimize Forms Services with the HTTP Listener and Oracle WebLogic Server, see Tuning Oracle HTTP Server in *Tuning Performance Guide*.

Using an Authenticating Proxy to Run Oracle Forms Services Applications

The default configuration as set up by the Oracle Fusion Middleware installation process supports authenticating proxies.

An authenticating proxy is one that requires the user to supply a username and password to access the destination server where the application is running. Typically, authenticating proxies set a cookie to detect whether the user has logged on (or been authenticated). The cookie is sent in all subsequent network requests to avoid further logon prompts.

The codebase and server URL values that are set up by the Oracle WebLogic Server installation process include `$FMW_HOME/forms/java` and `/forms/lservlet`. As these are under the document base of the page (`$FMW_HOME/forms`), authenticating proxies will work.

Oracle Forms and JavaScript Integration

Learn how to integrate JavaScript in Oracle Forms application with an example, Oracle Forms Calling External Events, JavaScript Events Calling into Oracle Forms. You can also enable or disable JavaScript integration by configuring formsweb.cfg and Environment Variables.
This chapter contains the following sections:

- About Oracle Forms Calling External Events
- About JavaScript Events Calling into Oracle Forms
- Integrating JavaScript and Oracle Forms
- Forms and JavaScript Integration for Java Web Start and Forms Standalone Launcher
- Web Configuration for JavaScript Integration
- Configuring Environment Variables

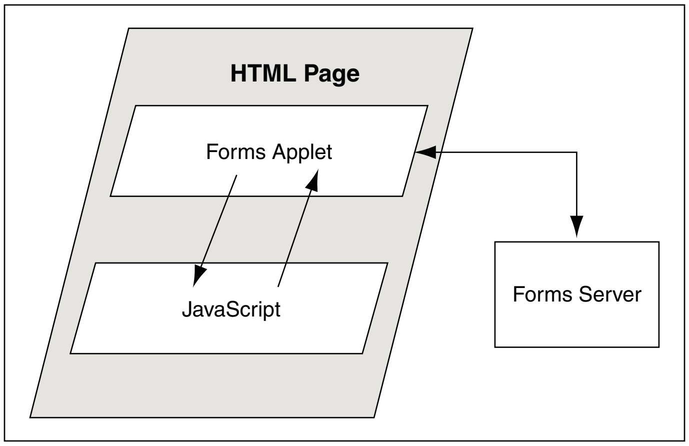
About Oracle Forms Calling External Events

In previous releases of Oracle Forms, you had to implement OLE and DDE to interact with a limited number of event types outside of Forms. In later versions, Forms offered `web.show_document` and Java integration to interface with external application sources.

But in terms of calling out to the Web page where Forms is displayed, there was no easy solution. It was also not possible to call from the Web page into Forms, perhaps to update a value acquired from an HTML form.

JavaScript integration provides the ability to have JavaScript events call into Forms, or have Forms execute JavaScript events. The following figure shows how JavaScript and Oracle Forms work together. In the left side of the image, JavaScript is executed in the page in which the Forms applet is hosted. Oracle Forms now has the capability to call JavaScript functions using native built-ins. Also, JavaScript functions can now trigger a Oracle Forms trigger by using a new API that has been provided.

Figure 23 Oracle Forms and JavaScript



Two new calls are available in the web Built-in package:

- web.javascript_eval_expr
- web.javascript_eval_function

The first call web.javascript_eval_expr is a procedure which takes two arguments: an expression and a target, both of data type varchar2. This legal JavaScript expression is interpreted in the Web page in which the Forms applet is embedded. The expression can be a call to a function that is defined in the target page or any valid JavaScript expression that can be executed on the target page, for example, document.bgColor='red'. The expression is executed, using LiveConnect's JSObject.eval() method, in the context of the page or frame that is named in the target argument. If the target argument is null, then it is executed in the page or frame in which the Forms applet is embedded.

The second call, web.javascript_eval_function is a function and returns a varchar2 value. Both web.javascript_eval_expr and web.javascript_eval_function have the same functionality except that javascript_eval_expr does not send any return value from the Forms client to the Forms Services. If your application does not need a return value, use web.javascript_eval_expr. The additional network trip that is required to carry the return value from the Forms client to the Forms Services is eliminated.

To set the value of an HTML text item with the ID outside_field_id to the value of the Forms field called inside, you could write this PL/SQL code:

```
web.javascript_eval_expr('
document.getElementById("outside_field_id").value='
||:inside
);
```

Notice that the PL/SQL string must use single quotes while JavaScript is flexible enough to use single or double quotes. Using double quotes inside the expression works without having to use escape sequences. You could also write a function in the Web page:

```
<SCRIPT>
    function set_field(field_id, myvalue){
        document.getElementById(field_id).value=myvalue;
    };
</SCRIPT>
```

To get the value of the outside field and assign it to the inside field, you could write the following PL/SQL code:

```
:inside:=web.javascript_eval_function('
     document.getElementById("outside_field_id").value
     ');
```

Reason for Calling Events Outside of Oracle Forms

JavaScript functionality allows you to integrate Forms with HTML-based application technologies in the Web browser. For example you can use JavaScript integration when the Forms-based application is required to integrate on the page with new functionality based on an HTML front end.

About JavaScript Events Calling into Oracle Forms

You can also allow JavaScript calls into Oracle Forms by using JavaScript in the Web page that hosts the Forms applet.

There is new functionality available on the embedded Forms object in the DOM (Document Object Model) tree. You use JavaScript to do:

```
document.forms_applet.raiseEvent(event_name, payload);
```

The assumption here is that you have set the ID configuration variable to `forms_applet`.

When the surrounding Web page executes this JavaScript code, Oracle Forms fires a new type of trigger called `WHEN-CUSTOM-JAVASCRIPT-EVENT`. In this trigger there are only two valid system variables: `system.javascript_event_value` and `system.javascript_event_name`. These variables contain the payload and event name that were passed into Forms through the `raiseEvent` method. On calling the `raiseEvent` method, a trigger named `WHEN-CUSTOM-JAVASCRIPT-EVENT` is fired on the server side.

```
declare
    event_val varchar2(300):= :system.javascript_event_value;
begin
    if (:system.javascript_event_name='show') then
        handleShowEvent(event_val);
    elsif(:system.javascript_event_name='grab') then
      handleGrabEvent(event_val);
    else
        null;
    end if;
end;
```

This PL/SQL code recognizes two events: 'show' and 'grab'. Any other name is ignored.

Reason to Let Events Call into Oracle Forms

You can synchronize an HTML based application, whether it is Java-based or otherwise, with a Forms-based application in the same hosting Web page. For example, you can use the HTML-based application to query data and use Forms to update it if, and only if, the user has the correct access privileges.

Integrating JavaScript and Oracle Forms

This section describes an example for integrating JavaScript in Oracle Forms application.

To build a Forms application using JavaScript events, see:

- About Oracle Forms Calling External Events
- About JavaScript Events Calling into Oracle Forms
- Also refer Forms Builder Online Help

To integrate JavaScript in Oracle Forms applications:

1. Build a Forms application using the JavaScript events. Use the `:system.javascript_event_name` and `:system.javascript_event_value` in the `WHEN-CUSTOM-JAVASCRIPT-EVENT` trigger. Compile the module.

2. Create an html file (for example, `test.html`) that the Forms servlet will use as a template when generating the HTML page used to start an Oracle Forms application. Copy the file to the Forms configuration directory: `$ORACLE_INSTANCE/config/FormsComponent/forms/server`
3. Copy any required images, html files, JavaScript files, and css files to the following directory: `$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_14.1.2/<random_string2>/war/`
4. Create an html file that uses the JavaScripts (for example, `js.html`) and invokes the servlet URL.
5. Using Enterprise Manager, create a new configuration section or modify an existing one and enable `enableJavascriptEvent`. Set baseHTMLjpi to `test.html`.
6. Using Enterprise Manager, edit the `default.env` file and add the directory where you saved the forms application to the environment variable `FORMS_PATH`.
7. Run the application by using the URL in your browser: `http://<localhost>:9001/forms/js.html`

Forms and JavaScript Integration for Java Web Start and Forms Standalone Launcher

When using Java Web Start or Forms Standalone Launcher, it is possible to integrate with JavaScript using Forms Websocket JavaScript Integration (WJSI).

The ability to integrate a Forms application with a web page through JavaScript was introduced in Oracle Forms 11*g* R2. This feature allowed developers to blend Forms applications with HTML-based web applications. Forms applications were able to communicate with HTML pages, which resulted in more creative application designs. In releases earlier than 12.2.1.3, the ability to integrate through JavaScript was not possible when using Java Web Start or the Forms Standalone Launcher. This is because those configurations do not have a parent browser, thereby exposing no way to connect between the two technologies.

Leveraging Eclipse/Jetty, an extremely lightweight web listener, a Forms application running with Java Web Start or the Forms Standalone Launcher can communicate with a web page through Web Socket connections.

This feature requires the download and signing of Jetty jar file, `jetty-all-9.4.5.v20170502-uber.jar` or newer from Eclipse. Download the jar file from the jetty-all directory at maven.org. For more information about Jetty, see The Eclipse Jetty Project.

> **Note:**
>
> Running Forms with Jetty versions newer than the 9.4 family is not supported at this time.

In order to facilitate integration between a web page and a Forms application, in addition to the use of Eclipse or Jetty, each web page that is to be integrated must include the following HTML to the BODY of the web page:

```
<script src="/forms/java/frmwebsocketjsi.js"></script>
```

If the web page is running from a different server than Forms, use a fully qualified path to address the Forms environment.

> **Note:**
>
> Altering the contents of frmwebsocketjsi.js is not supported or recommended. It is also not recommended that this file be moved from its default location, as doing so will prevent patching from properly updating the file if necessary.

Web Configuration for JavaScript Integration

The administrator of the Forms application can enable or disable JavaScript integration by setting the parameter `enableJavascriptEvent` in `formsweb.cfg` to "true" or "false".

If `enableJavascriptEvent` is not set to true, then calls from JavaScript would be ignored. The applet_name parameter must be set to the value that is used by the HTML developer to reference the forms applet via document.<applet_name>.

The administrator can also set `JavaScriptBlocksHeartBeat` (default value is false) in `formsweb.cfg` to true. This blocks Form's `HEARTBEAT` during the time JavaScript is executed. If the JavaScript calls complete execution before the FORMS_TIMEOUT period, setting JavaScriptBlocksHeartBeat to true provides an increase in performance by avoiding additional network messages.

Notice that if JavaScriptBlocksHeartBeat is set to true, Forms would abnormally terminate if the time taken for executing a JavaScript is more than FORMS_TIMEOUT.

The following Web configuration parameters can be optionally used with the WJSI configuration explained in the section above.

- `websocketJSILogging`
- `websocketJSIServerTimeout`
- `websocketJSISessionTimeout`

Refer to the Web Configuration Parameters section for more information on using these parameters.

Configuring Environment Variables

An environment variable called `FORMS_ALLOW_JAVASCRIPT_EVENTS` in `default.env` is also used to enable or disable JavaScript integration.

By default, the value of the variable is true. If this is set to false, then JavaScript integration is not enabled for any Forms application that uses that instance of `default.env`, no matter what value is set for `enableJavascriptEvent` in `formsweb.cfg`.

The `FORMS_WJSI_OVERRIDE_TIMEOUT` environment variable can be optionally used with the WJSI configuration explained in the section above. Refer to Environment Variables for more information on using this environment variable.

Enhanced Java Support

Oracle Forms provides Java classes that define the appearance and behavior of standard user interface components such as buttons, text areas, radio groups, list items, and so on. A Forms pluggable Java component (PJC) can be thought of as an extension of the default Forms client

component. When you create a PJC, you write your own Java code to extend the functionality of any of the provided default classes.
This chapter contains the following sections:

- Dispatching Events from Forms Developer
- Dispatching Events to Forms Services
- About Custom Item Event Triggers

Dispatching Events from Forms Developer

In addition to extending the standard Forms user interface components, you can also create a PJC that includes Java Swing user interface components in your form.

A pluggable Java component extends a class provided by Forms, that is, `oracle.forms.ui.VBean`, and lives in the Bean Area as seen on the Forms canvas. The Bean Area does not have its own user interface, but rather is a container. On the layout editor or on a canvas, you see only an empty rectangle until you associate an implementation class with it and add some user interface components.

In earlier releases of Oracle Forms, Forms user interface components implemented the IView interface. However, it did not have any special method to add or remove CustomListener from the pluggable Java component or the view. Beginning with Oracle Forms 12*c*, you can add or remove CustomListener in the IView interface.

Dispatching Events to Forms Services

You can dispatch Custom Events along with parameters and payloads. Since JavaBean classes do this by exposing the public method dispatchCustomEvent, you need to add the same method for your PJC.

You call the dispatchCustomEvent method from the PJC to dispatch the CustomEvent.

Since CustomEvent is usually associated with parameters, Forms provides a way to add them. In a JavaBean, you can use the getHandler().setProperty() method to set the parameters. Users must be able to do the same for PJC, see About the Custom Item Event Trigger at Runtime.

About Custom Item Event Triggers

You can add the `WHEN-CUSTOM-ITEM-EVENT` trigger to items at design time and code the pluggable Java components so that the trigger can be fired at runtime.

This trigger fires whenever a JavaBean custom component in the form causes the occurrence of an event. You can use a `WHEN-CUSTOM-ITEM-EVENT` trigger to respond to a selection or change of value of a custom component. The system variable `SYSTEM.CUSTOM_ITEM_EVENT_PARAMETERS` stores a parameter name that contains the supplementary arguments for an event that is fired by a custom control. Control event names are case sensitive.

Adding the When-Custom-Item-Event Trigger at Design Time

The most common way of adding a trigger to an item is by clicking the Create button in the Object Navigator toolbar in Oracle Forms Developer, while the focus is on the Trigger node, or by pressing the corresponding shortcut key. Forms Developer presents to you a list of available triggers at that level or for that item.

Another way of adding some commonly used triggers is by right-clicking the trigger node of the item in the Object Navigator. Then, select one of the triggers listed in the smart Triggers menu.

For information about working with triggers, see Oracle Forms Developer Online Help.

About the Custom Item Event Trigger at Runtime

Pluggable Java components can raise the `WHEN-CUSTOM-ITEM-EVENT` trigger. This enhanced trigger provides greater control over the content of the communication between the client and server.

The Forms client dispatches `CustomEvent` through the pluggable Java component, which fires the `WHEN-CUSTOM-ITEM-EVENT` trigger on the Forms Services. The `WHEN-CUSTOM-ITEM-EVENT` trigger provides a simple way to retrieve the event name and parameter values that are passed from the client pluggable Java component through CustomEvent. The event name is stored in `SYSTEM.CUSTOM_ITEM_EVENT`; parameters (name and value) are stored in `SYSTEM.CUSTOM_ITEM_EVENT_PARAMETERS`.

The Forms Built-in `get_parameter_attr` helps to retrieve the values and different parameters from `SYSTEM.CUSTOM_ITEM_EVENT_PARAMETERS`. The supported datatype for the values or payloads that are returned from get_parameter_attr is a `VARCHAR2` string.

Example: A Java class for a Push Button

In this example, a Java class is created for a push button that enables selecting a client file using the File Open option and returns the path to the server.

1. Create a Java class for a push button with simple PJC code such as:

```
  // MyButtonPJC.java
 import java.awt.event.ActionEvent;
 import java.awt.event.ActionListener;
 import javax.swing.JFileChooser;
 import oracle.forms.ui.CustomEvent;
 import oracle.forms.ui.VButton;
 import oracle.forms.properties.ID;
 public class MyButtonPJC extends VButton implements ActionListener
 {
   private static final ID CLIENT_SELECTED_FILE =
ID.registerProperty("CLIENT_SELECTED_FILE");
   public MyButtonPJC()
   {
     addActionListener(this);
   }
   public void actionPerformed(ActionEvent event)
   {
     JFileChooser fc = new JFileChooser();
     if(fc.showOpenDialog(getHandler().getApplet()) == JFileChooser.APPROVE_OPTION)
     {
       CustomEvent ce = new CustomEvent(getHandler(), "MyButtonPJC_Event");
       ce.setProperty(CLIENT_SELECTED_FILE, fc.getSelectedFile().getAbsolutePath());
       this.dispatchCustomEvent(ce);
     }
   }
   public void destroy()
   {
     removeActionListener(this);
     super.destroy();
    }
}
```

2. Ensure CLASSPATH variable is defined in the environment and `$ORACLE_HOME/forms/java/frmall.jar` is added to it.

3. Compile the Java class. For ease of creating the jar later, place the output class files in a separate directory by using the `-d <output-directory>` option of the javac (java compiler).

4. Navigate to the output directory and create a jar file, for example, MyButtonPJC.jar, containing the generated class files by using the command

   ```
   jar cvf <jar-file-path> *
   ```

5. MyButtonPJC.jar needs to be signed with a trusted certificate before deploying in Forms applet.

6. Copy MyButtonPJC.jar to `$ORACLE_HOME/forms/java` directory.

7. Add the path of MyButtonPJC.jar to the FORMS_BUILDER_CLASSPATH. This makes the class files in that jar available in Forms Builder.

8. Add the push button on the layout in the Forms application.

9. In Property Palette of the push button, set MyButtonPJC as the implementation class.

10. Add `WHEN-CUSTOM-ITEM-EVENT` trigger to the push button.

11. Add the following PL/SQL code to the `WHEN-CUSTOM-ITEM-EVENT` trigger of the push button. This code handles the CustomEvent dispatched by the PJC and then extracts the parameters in the event.

    ```
     declare
        filePath VARCHAR2(1024);
        dataType      PLS_INTEGER;
      begin
        Message('Custom Event Name='||:SYSTEM.CUSTOM_ITEM_EVENT);

    get_parameter_attr(:SYSTEM.CUSTOM_ITEM_EVENT_PARAMETERS,'CLIENT_SELECTED_FILE',dataTy
    pe, filePath);
        Message('The selected client file path is '|| filePath);
      end;
    ```

12. Add MyButtonPJC.jar to the list of comma-separated jars (only jar file name, not the full path) in the `archive` parameter in Forms configuration file (`formsweb.cfg`). This ensures that the jar file is loaded in Forms applet on the client side.

> **Note:**
>
> For information about how to sign a Java jar file, see the Signing and Verifying JAR Files tutorial.

Working with Server and System Events

This chapter tells you about Oracle forms and server events, how to create, manage, subscribe to events and event propagation. It also provides information about application integration between Forms and system events.

The following sections are included:

- Oracle Forms and Server Events
- About Creating Events
- About Subscribing to Events
- Event Propagation
- Publishing Database Events
- Application Integration Between Forms
- System Events

Oracle Forms and Server Events

With the exception of timers, most events in Oracle Forms occur from some kind of user interaction.

In versions prior to 11g of Oracle Forms, there was no easy support to receive an external event if it could not be bound to the Form's graphical user interface. Forms clients had to use techniques such as polling through a great deal of coding to respond to these events to deal with external events that it did not initiate.

Beginning with Oracle Forms 11*g* and Oracle Database, you can handle external events, such as asynchronous events, by using the database queue. Oracle Streams Advanced Queuing (AQ), an asynchronous queuing feature, enables messages to be exchanged between different programs. AQ functionality is implemented by using interfaces such as DBMS_AQ, DBMS_AQADM, and DBMS_AQELM, which are PL/SQL packages. For information about Advanced Queuing, see Transactional Event Queues and Advanced Queuing User's Guide.

In general, the steps required to integrate events and database queues are:

Database

- Create a queue table: Define the administration and access privileges (AQ_ADMINISTRATOR_ROLE, AQ_USER_ROLE) for a user to set up advanced queuing. Define the object type for the payload and the payload of a message that uses the object type. Using the payload, define the queue table.
- Create a queue: Define the queue for the queue table. A queue table can hold multiple queues with the same payload type.
- Start the queue: Enable enqueue/dequeue on the queue.
- Enqueue a message: Write messages to the queue using the DBMS_AQ.ENQUEUE procedure.

> **Note:**
>
> Forms integration with Advanced Queuing is not supported with Autonomous Database.

Form Builder

- Create an event object: Create a new event in the Events node in the Object Navigator in the Form Builder.

- Subscribe the event object to the queue: The name of the queue is specified in the Subscription Name property.
- Code necessary notification: Write the event handling function, which is queued up for execution by Forms and is executed when the server receives a request from the client. Write the trigger code for the When-Event-Raised trigger that is attached to the Event node.

Forms Services

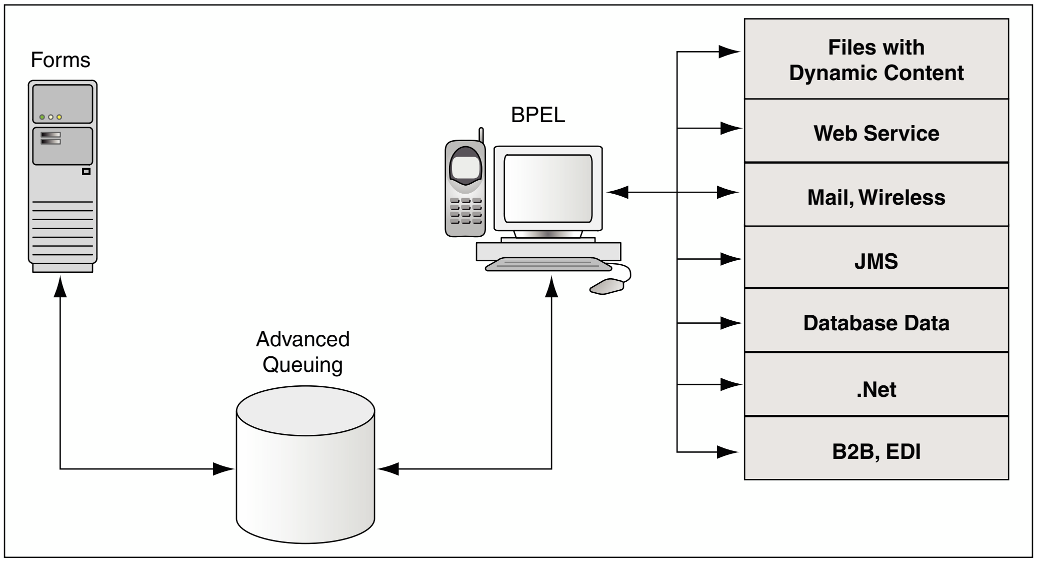
- Run the form and register the subscription
- Invoke the When-Event-Raised trigger upon event notification

In earlier versions of Forms, handling external events was only possible through custom programming, usually done in Java with the help of Forms' Java Bean support. Beginning in Oracle Forms 11g, it is possible to call into Forms from any technology that can interface with Advanced Queuing (AQ), for example Java Messaging (JMS).

As shown in the following image the flow of events that take advantage of the improved integration of the different components your application might work with. In the left side of the image, the Oracle Forms has two-way communication with the AQ functionality of Oracle Database. In the center of the image, the AQ function of Oracle Database also has two-way communication with the possible outside events that can trigger internal Forms events. In the right side of the image, these external events can include technologies such as files with dynamic content, Web services, mail, JMS, or database content that interact with BPEL processes which in turn interact with AQ. BPEL, however, is not necessary. JMS, as an example, can interact with AQ directly without having to go through BPEL.

> **Note:**
>
> Third party tools such as antivirus and security software may prevent Advanced Queuing from working correctly with Oracle Forms. As a workaround, turn off any third party security tools.

Figure 24 Oracle Forms Handles Outside Events with Advanced Queuing in Oracle Database



Single-Sign-Off

Oracle Forms is not a true single sign-on partner application. Therefore, historically for Oracle Forms applications that are authenticated using SSO were not be able to determine if a sign-off request occurred. With the addition of this System Single-Sign-Off event, this is no longer the case. The sign-off action can occur because of the user explicitly logging out of SSO or if the SSO session expires. It may be desirable to have the Oracle Forms application know about such a sign-off condition so it can react to it. Application developers can then decide how to react to this event, if at all.

Following are some limitations:

- The event may not appear to be raised immediately because control may be on server (e.g. due to a long running database query) when logout occurred.
- The event may not be sent to the server until the next scheduled exchange (e.g. Heartbeat) or user interaction.

About Creating Events

Oracle Forms Developer provides a declarative environment for creating and managing event objects.

For known external events, Forms Developer provides a list of available events that can be subscribed to. The property of the event object can be set at runtime or at design time. The ability to end a subscription to a particular external event is also provided through a dynamic setting of the event object property.

Most of the new event functionality is also available through standard Oracle interfaces. Both client and server-side PL/SQL provide all the necessary functionality to create, subscribe, and publish a database event. Oracle Forms provides a declarative and user-friendly way of registering a database event. Oracle Forms provides a standard way of responding to the event by hiding most of the complexity from end-users.

About Subscribing to Events

The Forms Services gets notified when events it has registered interest in are added to the event queue.

Registration is done either when the runtime starts up or when connecting to the database, depending on the type of the event. For database events, the type of the event queue (persistent or non-persistent) is also saved as part of the event creation.

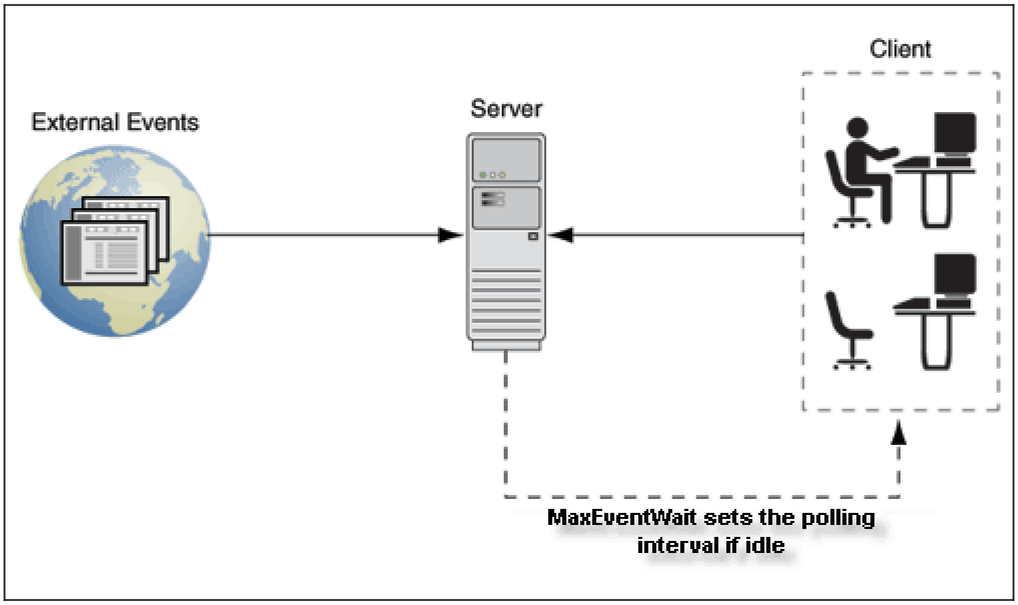
Event Propagation

In a situation where a Forms client is idle, since Oracle Forms is driven by the HTTP protocol, which is a request/response protocol only, nothing can change on the client if the client is idle.

An applet parameter `MaxEventWait`, expressed in milliseconds, governs how long the application should wait before checking for an event. In other words, you can specify how often the client should send a request to the server, thus causing the execution of the PL/SQL that is specified as a response to an event.

Note, however, that, on the server-side, Forms Services receives all the events without polling. However, the server does not start running the `WHEN_EVENT_RAISED` triggers until it receives the notification from the Forms Client (because of the HTTP request/reply paradigm of the Forms Client and hence the need for the MaxEventWait property).

Figure 25 Notification flow with idle or active clients



When-Event-Raised Trigger

Oracle Forms responds to or fires a trigger in response to a variety of events. For both Forms Developer and internal events, Forms provides entry points in terms of triggers so that an application developer can associate and execute some code in response to an event.

For example, a defined trigger is attached to a specific object in a form. The object to which a trigger is attached defines the *scope* of the trigger. For example, the WHEN-BUTTON-PRESSED trigger corresponds to the Button Pressed event which occurs when an operator selects a button. The name of the trigger establishes the association between the event and the trigger code. When a user clicks on a button, Forms responds by executing the code in the WHEN-BUTTON-PRESSED trigger.

This new event object has a corresponding trigger defined at the event object level. The WHEN-EVENT-RAISED trigger fires in response to the occurrence of a database event for which it has a subscription. The firing of the new trigger is similar to the internal processing of triggers. However, the source of the event is, in this case, an external event such as a database event (firing because of an operation) and not the result of any user interaction with forms or because of an internal form processing.

Trigger Definition Level and Scope

Oracle Forms triggers are usually attached to a specific object, such as an item, block, or Form. The object to which a trigger is attached determines the trigger's *definition level* in the object hierarchy. A trigger's definition level determines the trigger's *scope*. The scope of a trigger is its domain within the Forms object hierarchy, and determines where an event must occur for the trigger to respond to it. Although the WHEN-EVENT-RAISED trigger is attached to an event object, it has an application level scope because of the nature of the server-centric events. When the event notification is invoked because of an asynchronous callback mechanism for registered database events, any number of forms running within that application and with a subscription for that event receive the notification. This alleviates the need for the application developer to code complex logic to deal with the event.

There is also a Form-level scope so that the event will only be handled if the application is running the specific form from where the event is defined.

Publishing Database Events

You use the standard PL/SQL interface for publishing a database event from Forms.

For example, you can publish the `SalaryExceed` event by calling the enqueue interface and providing all the necessary arguments. You can also call a stored procedure to perform this task.

The following program unit can be called from a WHEN-BUTTON-PRESSED trigger by passing the queue name. Depending on how you have defined the queue in the database, a commit might or might not be necessary to actually publish the event. The following sample code will not actually publish the event since there is no commit issued.

```
Declare
  msgprop      dbms_aq.message_properties_t;
  enqopt       dbms_aq.enqueue_options_t;
  enq_msgid    raw(16);
  payload      raw(10);
  correlation  varchar2(60);
begin
    payload := hextoraw('123');
    correlation := 'Jones';
    enqopt.visibility := dbms_aq.IMMEDIATE;
    msgprop.correlation := correlation;
    DBMS_AQ.ENQUEUE( queue, enqopt, msgprop, payload, enq_msgid);
end;
```

> **Note:**
>
> For informations about database events, see PL/SQL Triggers .

Application Integration Between Forms

Many enterprise applications are made of a large number of forms which are defined to perform specific tasks such as purchasing, accounting, and sales force management. These applications may also interact with other non-Forms based applications as part of performing a task.

The need to provide an integration model where an enterprise can easily integrate its applications (including passing data) with those of its partners, suppliers, and distributors is extremely important.

In previous releases, Oracle Forms attempted to integrate loosely coupled applications through mechanisms ranging from using user_exit calls and some polling via timers to using pluggable Java components. These methods are all useful in some limited circumstances, but they do not provide a formal infrastructure for enterprise application integration.

Apart from the deployment concerns and performance issues, the main reason why these methods do not fully integrate applications is that the integration is only provided through Forms Developer as almost all events are bound to Forms visual components. Also, the communication with the Forms Services is always initiated by the Forms client via a request-reply model.

To provide better support for application integration, Oracle Forms supports synchronous and asynchronous server-centric events.

Synchronous Communication

Synchronous communication follows a request-reply paradigm, where a program sends a request to another program and waits until the reply arrives. HTTP follows this paradigm. This model of communication (also called online or connected) is suitable for programs that need to get the reply before they can proceed with their work. Traditional client-server architectures are based on this model. Earlier releases of Oracle Forms client-server architecture is also an example of this model. One of the drawbacks of the synchronous model of communication is that all the programs must be available and running for the application to work. In the event of network or machine failure, programs cease to function. For example, if the Forms Services dies, the Forms client ceases to function as well. The synchronous communication model is also in use when the Forms Services interacts with other systems such as PL/SQL or the database. The Forms system would be blocked waiting for the current operation to end before continuing with its work. Another drawback of synchronous communication is that the calling program has to wait for a response and unexpected events cannot be handled without first polling for them.

Asynchronous Communication

Asynchronous communication is when a user or form places a request in a queue and then proceeds with its work without waiting for a reply or when an asynchronous event is received without any initial request. Programs in the role of consumers retrieve requests from the queue and act on them. This model is well-suited for applications that can continue with their work after placing a request in the queue because they are not blocked waiting for a reply. It is also suited to applications that can continue with their work until there is a message to retrieve.

Oracle Forms supports asynchronous communication with the help of database events. A thin queuing mechanism provides the mechanism for asynchronous events. The queue is checked for messages once there are no more current operations to be performed.

For example, an application might require data to be entered or an operation executed at a later time, after specific conditions are met. The recipient program retrieves the request from the queue and acts on it.

Configuring Asynchronous Communication

Oracle Forms uses a polling technique at the application level. The client polls the server for an update after specified intervals of time. The frequency of polling can be modified using the parameters - `MaxEventWait` and `HEARTBEAT`. A higher frequency of polling may ensure that a client polls the server more frequently for updates; however, this may result in consumption of considerable resources.

The frequency value for polling is set in `formsweb.cfg`. The value assigned to this constant is in milliseconds and is a positive number.

In the absence of the configuration file setting, the current Oracle Forms `HEARTBEAT` setting is used. However, special attention and care should be made with regards setting and using of

`MaxEventWait`. In a default setting where `MaxEventWait` is not set, the `HEARTBEAT` mechanism is used for polling. The default delay when the HEARTBEAT mechanism is used is two minutes. You can set the `MaxEventWait` (which is in milliseconds) to a value smaller than the `HEARTBEAT` for faster response.

For information about configuring these parameters using the Enterprise Manager, see Managing Parameters.

System Events

Often it is required to have an application be aware of events that occur on the system hosting it and, have an ability to react to these actions.

In most cases, such events are not directly caused by the running Forms application, but knowledge of their occurrence could provide valuable information to it. This is most common on the client tier of an Oracle Forms application. An example of such a system event might be an indication that the end-user has been idle for an extended period of time. Knowledge of such a condition would allow the application developer to react and take appropriate actions. System Events can provide that desired knowledge.

There are five System Events. For all of these Events, little to no administrative configuration is required to use them. To use any of these events the application developer will be required to create the appropriate event object and code the `WHEN-EVENT-RAISED` trigger to perform the desired action. However, a brief description of each will be provided here. More information can be found in the Form Builder Help.

> **Note:**
>
> Because System Events rely on actions outside of Oracle Forms or the successful completion of an action within the application, they should not be used as the only means for implementing application security. If for any reason the action associated with the System Event does not complete successfully or is undetected by the application, the related application trigger may not fire. Although this is likely to be rare, it should be considered when developing with System Events.

Client-Idle

The System Client-Idle event monitors for end-user activity, on the client tier within the running Oracle Forms application. This event can be enabled in one of two ways. One way to enable this event, is for an administrator to set the applet parameter `idleTimeout` to a whole number. This will represent the time in seconds to wait before raising this event. To use this applet parameter, it must first be added to the appropriate Forms template base.htm file and formsweb.cfg.

This event can also be enabled/disabled programmatically using a new argument added to `SET_APPLICATION_PROPERTY`, `CLIENT_IDLE_TIME`. As with the applet parameter, the value of CLIENT_IDLE_TIME is represented in whole seconds. Refer to the Form Builder Help for more details on how to use this in PL/SQL.

Following are some limitations:

- Client-Idle will be ignored while the client applet is waiting on a response from the server or if a modal dialog is open (e.g. Alert, File Open dialog, etc). However, if the server responds immediately before the idle time has lapsed, the event may be raised. Although this condition should be rare, developers should consider this when developing the application. Adjusting the idle time programmatically may be necessary to avoid this condition when it is expected that the server may take an extended period to complete its task.

Client-Idle events can be fired if the `FORMS_ON_MODAL_DETECT_IDLE` environment variable is set to `1` (`TRUE`). When set to `TRUE`, the Client-Idle event executes immediately after the blocked condition is released and any previously executing trigger(s) completes processing. This does not apply to custom Java Beans that expose modal Java dialogs or any other blocked condition not originating from native Forms.

- Although the amount of idle time is set in seconds, the application may not react until the next exchange with the server. This is caused by either `Heartbeat` or `MaxEventWait`.

DB-Idle

The System DB-Idle event monitors activity between the Oracle Forms Runtime and the database to which it is connected. This event will monitor any interaction with the database executed by the associated application. This event can be enabled in one of two ways. The first way is to set the environment variable `FORMS_DB_IDLE_TIME` to a whole number, which represents the number of seconds to wait before raising this event.

This event can also be enabled/disabled programmatically using a new argument added to `SET_APPLICATION_PROPERTY`, `DB_IDLE_TIME`. The value for `DB_IDLE_TIME` will be a whole number, in seconds. Refer to the Form Builder Help for more details on how to use this in PL/SQL.

Following are some limitations:

- The internal timer for the DB-Idle event begins immediately after completing a database action. This does not include the Forms default login action that occurs during application startup.
- By default, this event will only be raised one time. For example if a `COMMIT` is executed then activity against the db no longer occurs and the preset time is reached, this event will be raised. If after that raising of the event, the idle condition remains, the event will not be raised again. To cause this event to continue monitoring, it must be set programmatically by setting the Application property `DB_IDLE_REPEAT` to `TRUE`.

Single-Sign-Off

Oracle Forms is not a true single sign-on partner application. Therefore, historically for Oracle Forms applications that are authenticated using SSO were not be able to determine if a sign-off request occurred. With the addition of this System Single-Sign-Off event, this is no longer the case. The sign-off action can occur because of the user explicitly logging out of SSO or if the SSO session expires. It may be desirable to have the Oracle Forms application know about such a sign-off condition so it can react to it. Application developers can then decide how to react to this event, if at all.

Following are some limitations:

- The event may not appear to be raised immediately because control may be on server (e.g. due to a long running database query) when logout occurred.
- The event may not be sent to the server until the next scheduled exchange (e.g. Heartbeat) or user interaction.

Notification

System Notifications allow an administrator, from Fusion Middleware Control, to raise various event levels. This will further allow developers to create specific tasks based on the message or notification level received from Fusion Middleware Control. For example, the application may be designed to display a message to the user when Notification level 3 is received. This message may have been predefined to tell users that the system will be shutting down for maintenance and they need to exit the application. Five notification levels are being provided (1-5).

> **Note:**
>
> For information about how to send User Session Notifications, see Managing User Sessions.

Media Completion

Oracle Forms supports the playing of audio files. The System Media Completion event will be raised when the playing of an audio file reaches the end of the track.

MDI Window Resize

The MDI Window Resize event monitors the application's multiple document interface (MDI) or parent window and detects if its size (height or width) has changed. If so, this event will fire.

This applies only to applications running with Forms Standalone Launcher or Java Web Start. This event can be helpful when attempting to dynamically resize and/or position objects on a canvas.

Using Forms Services with Oracle Access Manager

Oracle Access Manager, a component of Oracle Fusion Middleware, is a Single Sign-On solution for authentication and authorization. Information is provided about enable Single Sign-On protection for Forms applications, Forms services features with authentication server protection, protecting forms applications with single sign-on and integrating Oracle Forms. The following sections are included in this chapter:

- Oracle Access Manager and Single Sign-On
- Setup Process
- Forms Services Features with Authentication Server Protection
- Protecting Forms applications with Single Sign-On
- Enabling and Configuring Proxy Users
- Post installation Configuration

Oracle Access Manager and Single Sign-On

Oracle Access Manager is a Java Platform, Enterprise Edition (Java EE)-based enterprise-level security application that provides restricted access to confidential information and centralized authentication and authorization services. Oracle Access Manager, a component of Oracle Fusion Middleware, is a Single Sign-On solution for authentication and authorization.

Authentication servers enable an application to authenticate users by means of a shared authentication token or authentication authority. That means that a user authenticated for one application is automatically authenticated for all other applications within the same authentication domain.

Forms applications use a single sign-on solution only for obtaining database connection information from Oracle Internet Directory or Oracle Platforms Security Services (OPSS). Exiting a Forms application does not perform a single sign-on logout unless the application has been coded with one of the SSO logout features introduced in Oracle Forms 12*c*. Similarly, logging out of a single sign-on session does not terminate an active Forms session unless the application has been coded with one of the Forms SSO logout features. The database session exists until the Forms Runtime (for example, `frmweb.exe`) on the server terminates, usually by explicitly exiting the form.

Oracle Forms Services provides out-of-the box support for single sign-on for as many Forms applications as run by the server instance with no additional coding required in the Forms application.

> **Note:**
>
> Oracle Forms Services applications run in a single sign-on environment using the OID (or OPSS) and authentication server combinations. Supported versions can be found in the Product Certification Guide.
>
> For information about:
>
> - Certifications, see Oracle Fusion Middleware Supported System Configurations.
> - Oracle Access Manager, see Understanding Single Sign-On with Access Manager.
> - Oracle Internet Directory, see Configuring SSO Providers for Oracle Identity Manager.
> - Oracle Platform Security Services, see Introduction to Oracle Platform Security Services.

If you redeploy the forms JavaEE application and override its context root servlet alias when running the custom Forms application in single sign on mode, you might run into the following error:

```
FRM-60209 error obtaining credentials from Oracle Platform Security Services:
missing resource FRM-60209 error obtaining credentials from Oracle Platform
Security Services in oracle.forms.servlet.LBServletBundle
```

To resolve this error, follow these steps to run the app in SSO mode.

1. Deploy the custom Forms JavaEE applications.

In this example, we use sales and salesservlet as the context root and application, respectively.

```
context-root forms-> sales servlet-alias frmservlet-> salesservlet
```

2. Create the mappings in the forms.conf file.
   Here is an example of how to do this for the sales application.

   ```
   <Location /sales/> SetHandler weblogic-handler WebLogicCluster
   example.com:9010 DynamicServerList OFF </Location>
   ```

3. Add the following in the Forms OAM Registration meta-data files, `FormsOAMRegRequest2Ports.xml` and `FormsOAMRegRequest.xml`, stored in `$ORACLE_HOME/forms/provision` the directory.

Entry	Change this...	To this...
`protectedResourcesList`	`<protectedResourcesList> <resource>/forms/frmservlet?*oamMode=true*</resource> <resource>/reports/rwservlet/*</resource> </protectedResourcesList>`	`<protectedResourcesList> <resource>/forms/frmservlet?*oamMode=true*</resource> <resource>/sales/salesservlet?*oamMode=true*</resource> <resource>/reports/rwservlet/*</resource> </protectedResourcesList>`
`excludedResourcesList`	`<excludedResourcesList> <resource>/forms/frmservlet?*ifcmd=startsession*</resource> <resource>/forms/lservlet*</resource> <resource>/forms/lservlet/**</resource> <resource>/forms/java/**</resource> <resource>/forms/html/**</resource> </excludedResourcesList>`	`<excludedResourcesList> <resource>/forms/frmservlet?*ifcmd=startsession*</resource> <resource>/forms/lservlet*</resource> <resource>/forms/lservlet/**</resource> <resource>/forms/java/**</resource> <resource>/forms/html/**</resource> <resource>/sales/salesservlet?*ifcmd=startsession*</resource> <resource>/sales/lservlet*</resource> <resource>/sales/lservlet/**</resource> <resource>/sales/java/**</resource> <resource>/sales/html/**</resource> </excludedResourcesList>`

4. Perform the partner app registration using the `frmconfighelper` scripts.
5. Grant the following OPSS grants. Connect to WLST and run the following:

   ```
   grantPermission(codeBaseURL="file:${domain.home}/servers/${weblogic.Name}/tmp/
   _WL_user/salesapp_14.1.2/-",
   permClass="oracle.security.jps.service.keystore.KeyStoreAccessPermission",
   permTarget="stripeName=salesapp,keystore=formsks,alias=*,Action=*")
   ```

6. Restart the WebLogic servers.

Single Sign-On Components used by Oracle Forms

There are various Single Sign-On components in Oracle Fusion Middleware that are involved when running Forms applications in single sign-on mode with an authentication server.

The following figures, describes the high level overview of the various components involved in the single sign-on deployment setup of Forms Services.

Figure 26   Components involved in the Single Sign-On Deployment Setup of Forms Services with OPSS as the Forms Identity Store

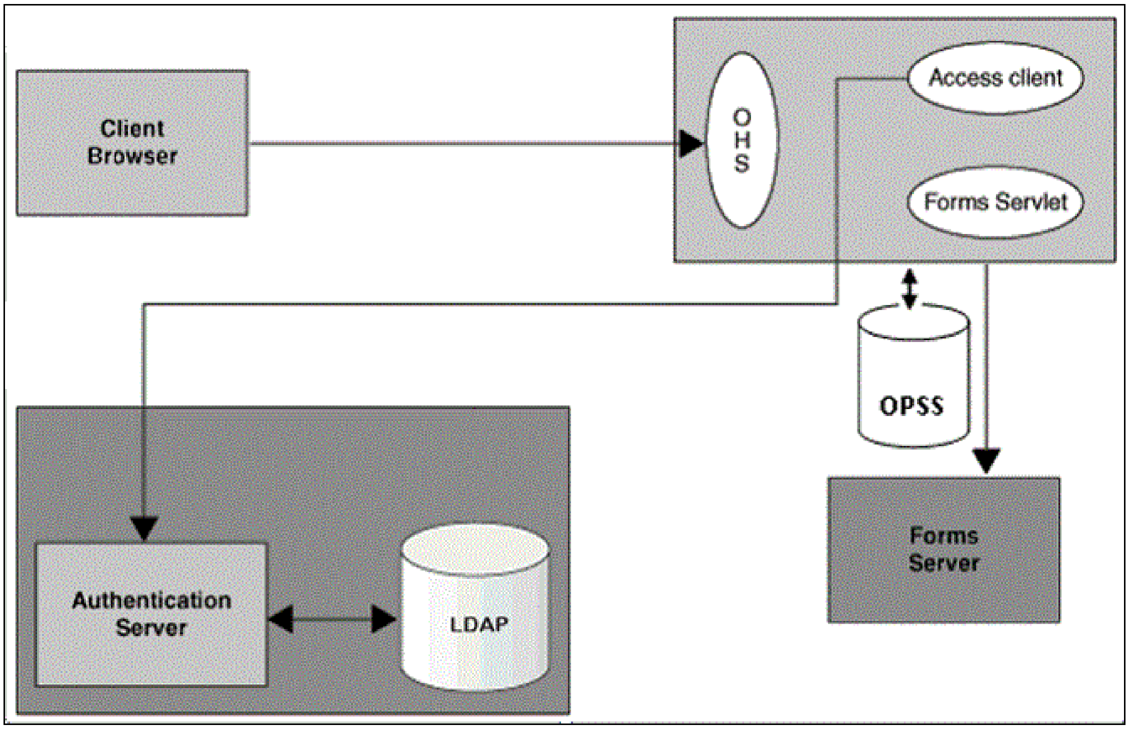
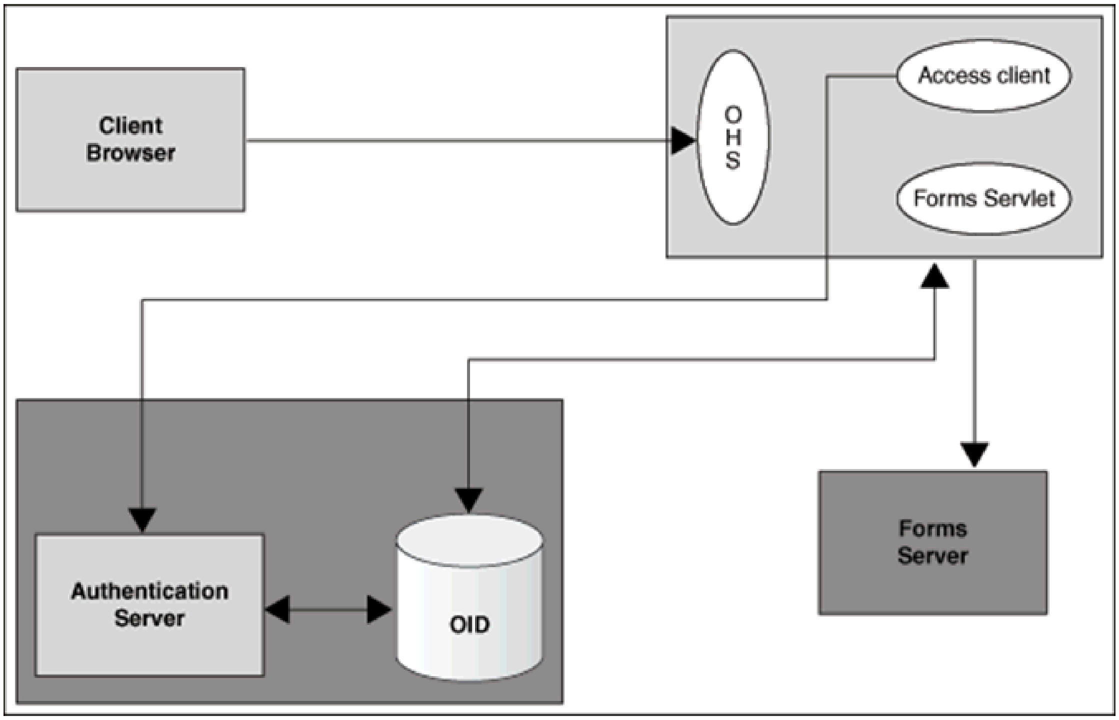


Figure 27 Components involved in the Single Sign-On Deployment Setup of Forms Services with (Oracle Internet Directory) OID Identity as the Forms Identity Store



Following is the description of the components mentioned in the above figure:

- **Authentication Server**
  - Oracle Access Manager (OAM Server) - Oracle FMW authentication server that provides a full range of security functions, including Web single sign-on, authentication and authorization. When running Forms Services, Oracle Internet Directory can be used as the Identity Store. Oracle Access Manager can use `webgate` as the access client configured with Oracle HTTP Server.
- **Access Client**
  - `webgate` - WebGate provides single sign-on support. It intercepts incoming HTTP requests and forwards them to the Access Server for authentication. Oracle Forms Services can use `webgate` as an access client with OAM server.
- **Forms Identity Store**
  - It is the storage for Forms Resource Access Descriptors, which contains the Forms Server database connection information. Oracle Platform Security Services (OPSS) or Oracle Internet Directory (OID) can be used as a Forms Identity Store. Oracle Platform Security Services (OPSS) is set as the default Forms Identity Store, but Forms administrators can use Oracle Enterprise Manager to change the Forms Identity Store to Oracle Internet Directory (OID) and back to Oracle Platform Security Services.
- **OAM Server Identity Store** - Oracle Internet Directory (OID) is an LDAP server that is used as the Identity store by the Oracle Access Manager (OAM) authentication server and the Forms applications. Any LDAP server certified for use with OAM can be used in an Oracle Forms environment when the Identity Store for Forms is OPSS and not OID.

Note:

When Oracle Internet Directory (OID) is used as the Forms Identity Store, the same Oracle Internet Directory (OID) instance should be set as the Oracle Access Manager's primary identity store.

- **Forms Servlet** - The Oracle Forms Services component accepts the initial user request to start a Forms application. The Forms servlet detects if an application requires authentication, directs the request to the authentication server and accesses the Oracle Internet Directory to obtain the database connect information.

Authentication Flow

The following figures describes the authentication flow of authentication server support in Oracle Forms, the first time the user requests an application URL that is protected by authentication server:

Figure 28 Authentication Flow for First Time Client Request

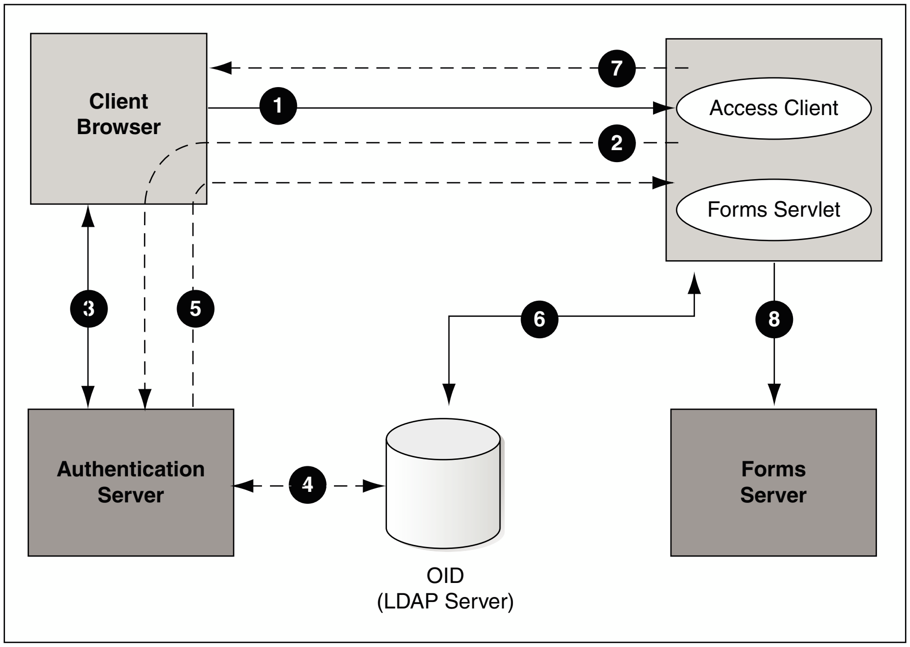
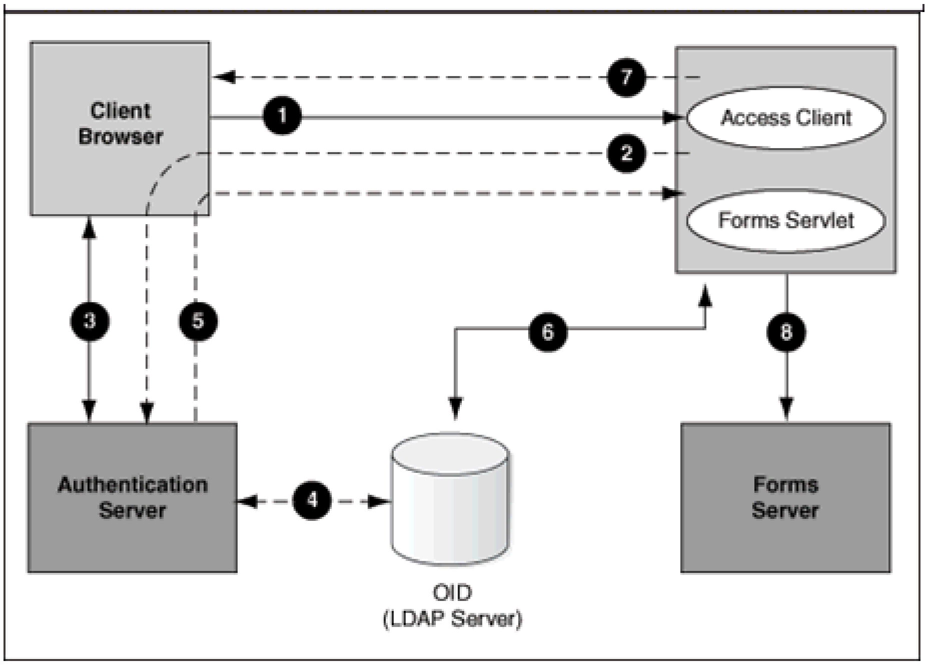


Figure 29   Authentication Flow for First Time Client Request



These steps describe the authentication flow mentioned in the above figure:

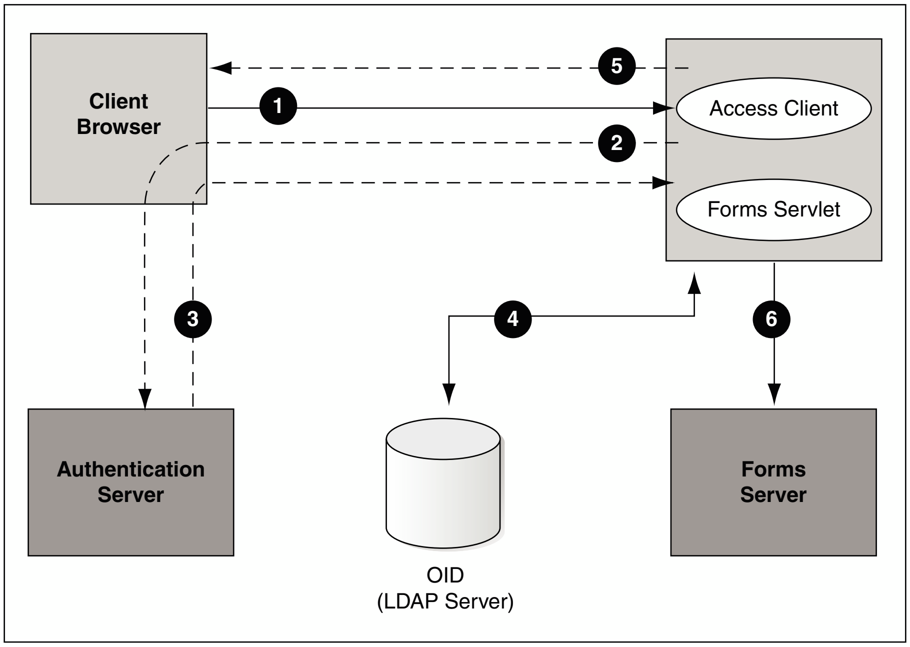
1. The user requests a Forms URL similar to `http(s)://<hostname>:<port>/forms/frmservlet?config= <application>&...`

   > **Note:**
   >
   > Use the HTTP port number in the Forms URL for Forms applications that use single sign-on. The Forms URL is similar to `http://<host name>:<http port>/forms/frmservlet?config=ssoapp` where `<ssoapp>` is the name of the section in forms configuration file with single sign-on (`ssoMode`) enabled.

2. The Forms servlet redirects the user to the authentication server login page.
3. The user provides user name and password through the login form.
4. The password is verified through Oracle Internet Directory (LDAP Server).
5. The user is redirected to the URL with `sso_userid` information.
6. The Forms servlet retrieves the database credentials from Forms Identity Store.
7. The Forms servlet sets the `sso_userid` parameter in the Run form session and permits the applet to connect to the Forms listener servlet.
8. The Forms servlet starts the Forms server.

Figure 30 describes the authentication flow of single sign-on support in Oracle Forms Services when a user, authenticated through another partner application, requests an application that is protected by authentication server.

Figure 30 Authentication Flow for Subsequent Client Requests



These steps describe the authentication flow mentioned in the above figure:

1. The user requests the Forms URL.
2. The Forms servlet redirects the user to the authentication server and its login page.
3. The user is redirected to the URL with the `sso_userid` information.
4. The Forms servlet retrieves the database credentials from the Forms Identity Store.
5. The Forms servlet sets the `sso_userid` parameter in the Runform session and the applet connects to the Forms listener servlet.
6. The Forms servlet starts the Forms server.

Setup Process

Single Sign-On is not enabled out of the box for Forms applications.

The following step is required to enable Single Sign-On protection for Forms applications.

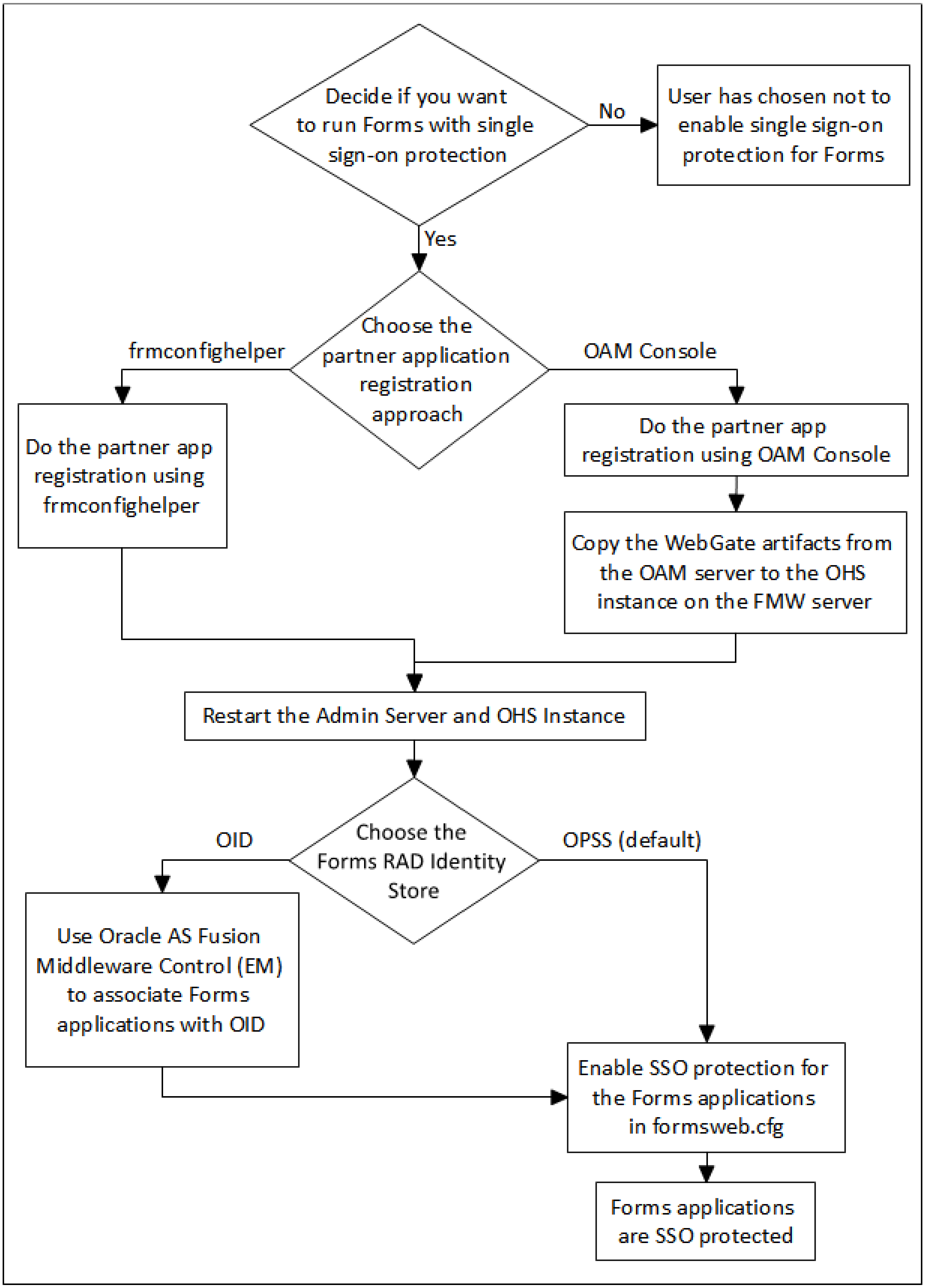
- Enabling SSO for Forms Application after Configuring a Forms Services Weblogic Domain

Enabling SSO for Forms Application after Configuring a Forms Services Weblogic Domain

Single sign-on (SSO) can be enabled for Forms Applications after setting up the Forms Services Weblogic Domain and after configuring a Web-tier instance in the Domain.

The following flowchart describes the steps to enable SSO for Forms application post installation.

Figure 31 Enabling SSO for Forms application post installation


Decide if you want to run Forms with single sign-on protection
No
User has chosen not to enable single sign-on protection for Forms
Yes
Choose the partner application registration approach
frmconfighelper
OAM Console
Do the partner app registration using frmconfighelper
Do the partner app registration using OAM Console
Copy the WebGate artifacts from the OAM server to the OHS instance on the FMW server
Restart the Admin Server and OHS Instance
Choose the Forms RAD Identity Store
OID
OPSS (default)
Use Oracle AS Fusion Middleware Control (EM) to associate Forms applications with OID
Enable SSO protection for the Forms applications in formsweb.cfg
Forms applications are SSO protected

The steps depicted in the flowchart are described in the following table:

Table 27 Tasks to Enable Single Sign-On for Forms Application Post installation

Tasks	Options	Description	Comments
Prerequisite	No	Create a Web-tier (OHS) instance in the Weblogic Domain and enable Web-tier (OHS) to Forms managed server routing.	
Task 1: Make a decision if you want to enable single sign-on Protection for Forms applications.	No	User has opted to run Forms applications without single sign-on protection.	
	Yes	User has opted to run Forms with single sign-On server with Oracle Access Manager (OAM Server) as the authentication server.	For detailed steps for installing OAM, see Oracle Fusion Middleware Installation Guide for Oracle Forms and Reports.
Task2: Select the partner application registration approach.	Use frmconfighelper script	User has opted to use frmconfighelper script to register the web-tier instance as the partner application with Oracle Access Manager (OAM Server).	For detailed steps, see Registering web-tier instance as OAM partner application and OAM policy configuration.
	Use OAM Admin Console	User has opted to use OAM Console to do register the web-tier instance as the partner application with Oracle Access Manager (OAM Server).	For detailed steps, see Registering web-tier instance as OAM partner application and OAM policy configuration.
Task 3: Restart the Web-tier instance and Admin Server instance		The Web-tier instance and the WLS Admin server have to be restarted to replicate WebGate configuration to the web-tier runtime instances.	
Task 4: Choose the Forms Identity Store type for storing Resource Access descriptors.	Oracle Platform Security Services (OPSS)	Oracle Platform Security Services (OPSS) is configured as the default Forms Identity Store, so no action is required.	For detailed steps see Selecting Oracle Internet Directory or Oracle Platform Security as the Forms Identity Store.
	Oracle Internet Directory (OID)	The user opted to use Oracle Internet Directory (OID) as the Forms Identity Store.	For detailed steps on Forms Oracle Internet Directory (OID) association and enabling Oracle Internet Directory (OID) as the Forms Identity store see Configuring Forms J2EE application with Oracle Internet Directory.
Task 5: Enable SSO for Forms applications in formsweb.cfg	This task is mandatory.	After having registered the Access client with the authentication server, the user must enable SSO for Forms applications.	For detailed steps for enabling SSO for Forms applications in formsweb.cfg, see Protecting Forms applications with Single Sign-On.

Forms Services Features with Authentication Server Protection

In this release of Oracle Forms Services specific features and enhancements are available for Authentication Server Protection.

The following are the features and enhancements:

- Dynamic Resource Creation
- Support for Dynamic Directives
- Support for Database Password Expiration

Dynamic Resource Creation

In single-sign on mode, when a user tries to connect to a Forms application, the user is authenticated by `webgate` in combination with an authentication server and Forms Identity Store. Once the user is authenticated, the user is directed to the Forms servlet which takes the user's request information containing the single sign-on user name. The user name and the application name build a unique pair that identifies the user's resource information for this application in Forms Identity Store.

When an authorized Forms user has neither the resource for a particular application that is being requested nor a default resource in Forms Identity Store, then the user is redirected to the Forms RAD Servlet for the creation of the Resource Access Descriptor. After creating the resource, the user is redirected to the original Forms request URL.

The way Oracle Forms Services handles the missing resource information can be customized by the application or Oracle Forms Services administrator. The following options are available:

- Allow dynamic resource creation (default)
- Redirect the user to a pre-defined URL as specified by the ssoErrorUrl parameter
- Display the Forms error message

The redirection URL is provided by the system administrator in the Forms configuration files and should be either absolute or relative.

Support for Dynamic Directives

Enforcing single sign-on in Forms is done within the `formsweb.cfg` file. The single sign-on parameter, `ssoMode`, when set to a valid value other than `FALSE`, indicates that the application requires authentication by authentication server.

This parameter allows a Forms Services instance to handle both application types, those that rely or do not rely on single sign-on for retrieving the database password. Because single sign-on is configured in the `formsweb.cfg` file, Fusion Middleware Control users can use to manage this aspect of authentication.

Support for Database Password Expiration

In Oracle Forms Services 12*c*, if the database password has expired and the *Forms Services* application, running in single sign-on mode, helps to renew it, the new password entered by the user updates the Resource Access Descriptor (RAD) in Forms Identity Store for this application. This feature ensures that authenticating a Forms user via authentication server with Forms continues to work even when the user's database password has changed.

However, if password changes are made in SQL*Plus, and not in Oracle Forms, the database connect string is not updated in the Forms Identity Store.

Protecting Forms applications with Single Sign-On

Oracle Forms applications are configured using a central configuration file, the `formsweb.cfg` file in the `$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config` directory. The recommended method of managing `formsweb.cfg` file is using Fusion Middleware Control.

The following parameters defined in Oracle Forms Services configuration file `formsweb.cfg` is necessary for the users to enable Single Sign-On in individual or collective Forms applications. It is recommended that this file should be managed using the Fusion Middleware Control.

Table 28 Parameters used to enable single Sign-On

Parameter Name	Valid values	Default Value
ssoMode	true webgate false	false
ssoProxyConnect	yes no	yes
ssoDynamicResourceCreate	true false	true
ssoErrorUrl	String URL	
ssoCancelUrl	String URL	

> **Note:**
>
> A detailed description of these parameters along with their possible values are discussed below.

These Oracle Forms parameters in the `formsweb.cfg` file are set in the **User Parameter** section, which define the behavior for all Forms applications run by the server. These parameters can also be set in a **Named Configuration**, which define the settings for a particular application only. A single sign-on parameter set in a Named Configuration section overrides the same parameter set in the **User Parameter** section.

To enable single sign-on for an application:

1. Start Fusion Middleware Control.
2. Select **Web Configuration** from the **Forms** menu.
3. Select the row that lists the configuration section for your application.
4. In the Section region, select **sso** in the **Show** drop down list.
5. In the Section region, select the row containing `ssoMode`.
6. In the **Value** field, enter `webgate` or `TRUE`.
7. Click **Apply** to update the `formsweb.cfg` file.

Single sign-on is now enabled for the selected application.

To disable single sign-on for an application:

1. Select **Web Configuration** from the **Forms** menu.
2. Select the row that lists the configuration section for your application.
3. In the Section region, select **sso** in the **Show** drop down list.
4. In the Section region, select the row containing `ssoMode`.
5. In the **Value** column, enter `FALSE`.
6. Click **Apply**.

Single sign-on is now disabled for the selected application.

ssoMode

The `ssoMode` parameter enables a Oracle Forms Services application to connect to an authentication server. Following are the values that the single sign-on parameter, `ssoMode` can assume:

- `ssoMode`, when set to `TRUE` or `webgate` indicates that the application requires authentication by OAM Server using webgate as the access client. Webgate must be manually configured.
- `ssoMode`, when set to `FALSE` indicates that the application does not require authentication with an authentication server.

By default, Oracle Forms applications are not configured to run in single sign-on mode. The ssoMode parameter can be set in two places in the formsweb.cfg file:

- By setting `ssoMode` in the default section of `formsweb.cfg` with a value of `true` or `webgate` which allows all applications to run in single sign-on mode by this Oracle Forms Services instance
- By setting the `ssoMode` parameter in a named configuration of an Oracle Forms application which enables or disables single sign-on only for this particular application, for example:

  ```
  [myApp]

  form=myFmx

  ssoMode=true
  ```

ssoProxyConnect

The `ssoProxyConnect` parameter enables a user to control when Oracle Forms should use a proxy connection to the database and when it should not. The `ssoProxyConnect` parameter can be set in two ways:

- By setting `ssoProxyConnect` in the default section of `formsweb.cfg` with a value of `yes` which allows all applications to run in single sign-on mode by this Oracle Forms Services instance
- By passing the `ssoProxyConnect` parameter in the URL at runtime, for example `http://<host>:<port>/?config=myapp&……&ssoProxyConnect=yes`

ssoDynamicResourceCreate

The `ssoDynamicResourceCreate` parameter is set to `true` by default which allows the user to create a Resource Access Descriptor (RAD) entry in OPSS (depending on how you have configured) to run the application if this resource entry does not exist.

Allowing dynamic resource creation simplifies administration because there is no longer the need for an administrator to create user RAD information in advance. The `ssoDynamicResourceCreate` parameter can be set as a system parameter in the `formsweb.cfg` file or as a parameter of a named configuration. Because the default is set to `true`, this parameter may be used in a named configuration for a specific application to handle a missing RAD entry differently from the default.

Notice that enabling an application for single sign-on with the value of the `ssoDynamicResourceCreate` parameter set to `false`, while not specifying a value for the `ssoErrorURL`, causes Oracle Forms to show an error message if no RAD resource exists for the authenticated user and this application.

Since not all administrators want their users to create resources for themselves these parameters allow administrators to control Forms Identity Store resource creation. Although the default behavior is to direct users to an HTML form that allows them to create the resource, the administrator can change the setting and redirect the user to a custom URL.

For the configuration section for the Forms application, you need to set these parameters:

```
[myApp]

form=myFmx

ssoMode=true

ssoDynamicResourceCreate=false
```

For information about setting these parameters through Enterprise Manager Fusion Middleware Control, see Managing Parameters.

ssoErrorURL

The `ssoErrorURL` parameter allows an administrator to specify a redirection URL that handles the case where a user RAD entry is missing for a particular application. This parameter has effect only if the `ssoDynamicResourceCreate` parameter is set to `false`, which disables the dynamic resource creation behavior. The `ssoErrorURL` parameter can be defined in the default section and as a parameter in a named configuration section. The URL can be of any kind of application, a static HTML file, or a custom Servlet (JSP) application handling the RAD creation, as in the example below.

```
[myApp]
form=myFmx
ssoMode=true
ssoDynamicResourceCreate=false
ssoErrorURL=http://example.com:7779/servlet/handleCustomRADcreation.jsp
…
```

ssoCancelUrl

The `ssoCancelURL` parameter is used in combination with the dynamic RAD creation feature (`ssoDynamicResourceCreate= true`) and defines the URL that a user is redirected to if the user presses the cancel button in the HTML form that is used to dynamically create the RAD entry for the requested application.

Accessing Single Sign-on Information From Forms

Optionally, if you need to work with authentication server to authenticate information in a Forms application, the GET_APPLICATION_PROPERTY() Built-in you can use to retrieve the following login information: single sign-on user ID, the user distinguished name (dn), and the subscriber distinguished name (subscriber dn)

```
authenticated_username := get_application_property(SSO_USERID);
userDistinguishedName := get_application_property(SSO_USRDN);
subscriberName := get_application_property(SSO_SUBDN);
config := get_application_property(CONFIG).
```

The Forms application developer can obtain the SSO information such as single sign-on user ID, subscriber distinguished name (subscriber dn), and user distinguished name (dn) in SSO mode with either OracleAS Single Sign-On server or Oracle Access Manager when using `webgate` as the access client.

When using Oracle Platform Security Services (OPSS) as the Forms Identity Store and if SSO_USERDN or SSO_SUBDN parameter is passed to `get_application_property` built-in, it will return an empty String. These parameters are valid only when running with Oracle Internet Directory as the Forms Identity store.

> **Note:**
>
> `config` can be obtained even in non-SSO mode.

Enabling and Configuring Proxy Users

Oracle Database supports proxy user authentication, which allows a client user to connect to the database through an application server, as a proxy user.

The users connecting through a Forms application as proxy users must also be defined in authentication server and Oracle Internet Directory. Oracle Forms authenticates the user via authentication server (using authentication server with Forms is a requirement when using a proxy user). Oracle Forms then connects to the database as the proxy user with a username and password that is in the RAD for the Oracle Internet Directory entry for the application user.

This section contains the following:

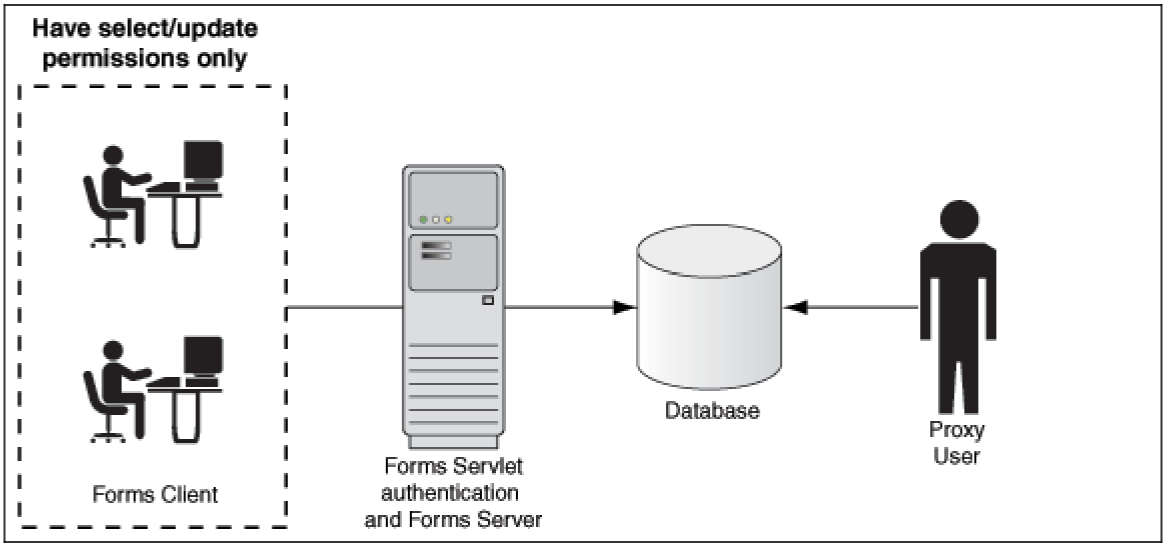
- Proxy User Overview
- Enabling Proxy User Connections When Enabling SSO with Oracle Internet Directory
- Enabling SSO for Proxy Users
- Accessing the Forms Application
- Changes in Forms Built-ins

Proxy User Overview

Many large applications, including Oracle's own E-Business Suite, use a single username for all connections. This makes it possible to manage users in a way that often suits large companies better but it creates a problem with auditing. All inserts, updates and removals of records appear, from the database's perspective, to have been done by a single user. To restore auditing, the application developers must write and implement customized auditing code in the database that requires a user name to be passed to the database from the application. This step not only takes development time, but also duplicates functionality that is already implemented in the Oracle Database.The second issue is security. If that single user access is ever compromised, the compromised user will have access to the entire application schema.To address these two issues, Oracle Database supports proxy user authentication, which allows a client user to connect to the database through an application server, as a proxy user.

The following figure describes the authentication of a Forms proxy user.

Figure 32   Proxy User Authentication


Have select/update permissions only
Forms Client
Forms Servlet authentication and Forms Server
Database
Proxy User

- Oracle Forms authenticates the user through Oracle Internet Directory or LDAP, as shown in the center of the image.
- Forms then connects as the proxy user with or without a password, passing in the real username from the Oracle Internet Directory repository.
- Typically, the proxy user is configured with least set of privileges. In the following procedure, the proxy user has "connect" and "create session" privileges.
- The database accepts the `create` session action for the proxy user and uses the real username in audits and access control.
- The Oracle Internet Directory user cannot connect to the database independently without configuration of the proxy user account.
- The proxy user account isolates the client from direct SQL*Plus connections.

Enabling Proxy User Connections When Enabling SSO with Oracle Internet Directory

To use a proxy support in Forms, you first need to create a proxy user.

In this example, the proxy user is called `midtier`:

1. Create a proxy user in the database.

   ```
   SQL> CREATE USER midtier IDENTIFIED BY midtierPW;
   ```

2. Assign connect and create session privileges to midtier:

   ```
   SQL> GRANT CONNECT,CREATE SESSION TO midtier;
   ```

   At this point, this proxy user has connect and create session privileges and has no grants on any of the user schemas.

3. Create a database user which has one-to-one mapping with a SSO username (that is, if `appuser` is the SSO username create database user `appuser`).

   ```
   SQL> CREATE USER appuser IDENTIFIED BY appuserPW;
   ```

4. Assign create session privileges to appuser.

   ```
   SQL> GRANT CREATE SESSION TO appuser;
   ```

5. To make it possible to connect through the midtier user you need to alter the database user:

   ```
   SQL> ALTER USER appuser GRANT CONNECT THROUGH midtier;
   ```

   The user `appuser` can now connect through the midtier account.

   Alternatively, you can define the roles that the proxy user can connect to the database as

   ```
   SQL> ALTER USER appuser GRANT CONNECT THROUGH midtier WITH ROLE <role_name>;
   ```

   Repeat Step 3 and 4 for all database users who need to use the proxy user account.

It is also possible to set up the database users in Oracle Internet Directory with the help of the database functionality called Enterprise User Security. If you choose this method, the proxy user is the only user defined in the database and the additional benefit of easy administration is gained, see Configuring Directory Server Chaining in *Administering Oracle Internet Directory*.

The application user's password is not presented to the database; only the user name and the proxy user's user name and password. Forms, with the help of OCI calls, issues the equivalent of:

```
SQL> connect midtier[appuser]/midtierPW@databaseTnsName
```

For example, suppose your application always connects to the database using midtier. This midtier now informs the database that the actual user is `appuser`. Without using proxy users, the SQL command `select USER from DUAL` would return midtier, but, using proxy users, this query returns `appuser`. This essentially tells the database to trust that the user is authenticated elsewhere and to let the user connect without a password and to grant the connect role.

Note:

- In the Step 3 of the above procedure, the database users are typically configured to have a subset of permissions granted to a schema. For example, appuser is granted `CREATE` permissions to the schema `app_schema` with the SQL command:

  ```
  SQL> GRANT CREATE ON SCHEMA app_schema TO appuser
  ```

  Thus, the appuser is restricted to perform only a set of actions in proxy user mode.
- When the database user (for example, appuser) is connected in proxy mode, user actions of the database users are audited rather than that of the proxy user.

Enabling SSO for Proxy Users

Create a configuration section in `formweb.cfg` for single sign-on (for example, `ssoapp`) and set `SSOProxyConnect` to `yes` and `ssoMode` to `true` or `webgate`.

The username and password that is used for the proxy connection is defined in the RAD entry for the user that is logging on. If `ssoProxyConnect=yes`, the connect string equivalent issued by Forms is in effect:

```
SQL> connect RADUsername[appuserName]/RADPassword@databaseTnsName
```

Accessing the Forms Application

After enabling proxy user connections and single sign-on, perform the following steps to access the forms applications:

1. Run the forms application with the URL `https://<host name>:<https port>/forms/frmservlet?config=ssoapp` where `ssoapp` is the name of the configuration section with single sign-on (`ssoMode`) is enabled.
2. Use the single sign-on user name and password to log in.

Enabling Proxy User Connections When Enabling SSO with Oracle Internet Directory`appuserappuserPW`

Changes in Forms Built-ins

The Built-in `get_application_property` now takes a new parameter called `IS_PROXY_CONNECTION` (a Boolean). When this parameter is supplied, the call returns `true` if the form is running in proxy user mode, `false` otherwise.

Post installation Configuration

This section describes specific post installation steps.

These steps are required to perform depending on the choices made in Setup Process.

The following sections are included:

- Configuring Forms J2EE application with Oracle Internet Directory

- Selecting Oracle Internet Directory or Oracle Platform Security as the Forms Identity Store
- Registering web-tier instance as OAM partner application and OAM policy configuration

Configuring Forms J2EE application with Oracle Internet Directory

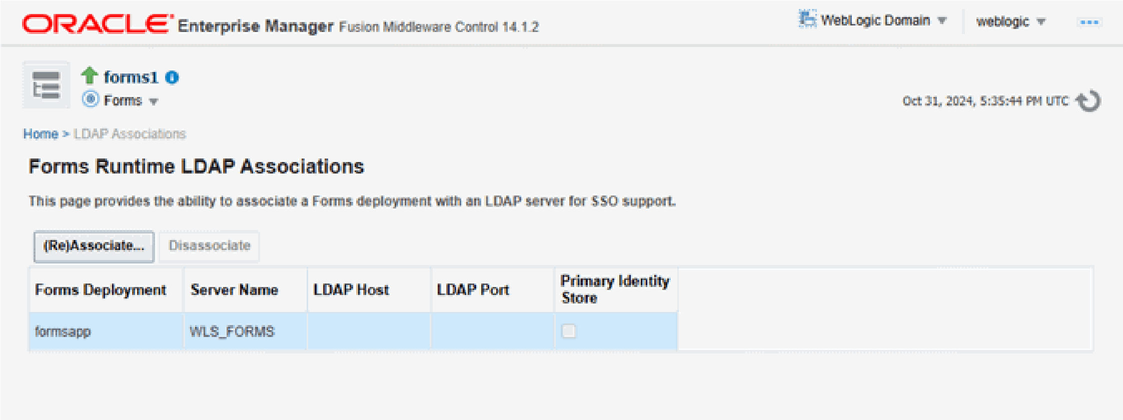
The topics describes how to configure a Forms application to work with Oracle Internet Directory.

To access the Associate/Disassociate page:

1. Start Fusion Middleware Control.
2. Navigate to the Forms **Home** page.
3. From the Forms menu, select **Forms Runtime LDAP Associations**.

   The **Forms Runtime LDAP Associations** page is displayed.

Figure 33 Forms Runtime LDAP Associations


ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Oct 31, 2024, 5:35:44 PM UTC
Home > LDAP Associations
(Re)Associate...
Disassociate

Forms Runtime LDAP Associations

This page provides the ability to associate a Forms deployment with an LDAP server for SSO support.

Forms Deployment	Server Name	LDAP Host	LDAP Port	Primary Identity Store
formsapp	WLS_FORMS			□

To associate OID Host with a Forms Application:

1. To associate an Oracle Internet Directory host with a Forms application for the first time, from the **Associate/Disassociate OID** page, select the Forms application. Click **Associate**.

   The Associate dialog appears.
2. Enter the Oracle Internet Directory Host details as described in the following table.
3. Click **Associate**.

   The **Associate/Disassociate OID** page reappears.

Table 29 Oracle Internet Directory Host Details

Parameter	Description
OID Host	Select the Oracle Internet Directory Host from the list or select New Oracle Internet Directory (OID) host to add new host details.
New OID host	Host name of the Oracle Internet Directory server. This field is enabled if you have selected to add new Oracle Internet Directory (OID) Host.

Table 29   (Cont.) Oracle Internet Directory Host Details

Parameter	Description
New OID Port	Port number on which Oracle Internet Directory is listening. This field is enabled if you have selected to add new Oracle Internet Directory Host.
Username	Oracle Internet Directory Administrator username
Password	Oracle Internet Directory Administrator password
Use SSL Port	Select this box if the connection to the Oracle Internet Directory Host should use SSL (in which case the port number provided should be the SSL port).

To Disassociate OID Host from a Forms Application:

1. From the **Associate/Disassociate OID** page, select the Forms application. Click **Disassociate**.

   A confirmation box appears.

2. Click **Yes**.

   The Oracle Internet Directory host is disassociated from the Forms application.

3. Restart the Oracle WebLogic Managed Server and the front-end OHS for the changes to take effect.

   To prevent users from being inadvertently disconnected from active forms sessions, ensure you choose to restart Oracle WebLogic Managed Server and the front-end OHS at a convenient time when users are not running any forms sessions.

To re-associate an OID Host with a Forms Application:

1. From the **Associate/Disassociate OID** page, select the Forms application. Click **Disassociate**.
2. From the **Associate/Disassociate OID** page, select the Forms application. Click **Associate**.

   Enter the Oracle Internet Directory Host details as described in the above table.

3. Generate and apply the access client file.

Access client file, as described in Selecting Oracle Internet Directory or Oracle Platform Security as the Forms Identity Store

Selecting Oracle Internet Directory or Oracle Platform Security as the Forms Identity Store

Oracle Platform Security Services (OPSS) is the set default Forms Identity Store. If the administrator performs the Forms OID association, it will set Oracle Internet Directory as the Forms Identity Store. Users can switch back to Oracle Platform Security Services (OPSS) as Forms Identity Store by un-checking the check box in the Primary Identity Store column for each deployment on Forms Runtime LDAP Associations page.

Registering web-tier instance as OAM partner application and OAM policy configuration

Users have two choices for registering the web-tier instance as the Oracle Access Manager (OAM) partner application and configure the resulting OAM policy.

- frmconfighelper script
- OAM console

> **Note:**
>
> The Web-tier and its managing Weblogic Admin Server must be restarted after either of the configuration options.

Using frmconfighelper Script for the Web-tier Partner Application Registration and Configuring Policy

Run the `frmconfighelper` script to perform partner application registration and subsequently configure the policy on the Oracle Access Manager.

The `frmconfighelper` script uses the Oracle Access Manager's RREG tool to perform these tasks. All the policy configuration details are included in the Forms OAM policy configuration file, `$FMW_HOME/forms/provision/FormsOAMRegRequest.xml`.

Users need to:

1. Download `RREG.tar` located on the Oracle Access Manager Server and untar under the Oracle Forms and Reports Home directory (for example, "`Oracle_Home`").
2. Run the `frmconfighelper` script and pass it `enable_sso` option.

Using Oracle Access Manager (OAM) console for doing the web-tier partner application registration and configuring policy

Users need to perform these steps:

1. Configure Webgate on the web-tier instance.

   Webgate is installed with Oracle HTTP Server, but not configured in the OHS instance. Users can follow the instructions in the Oracle HTTP Server Webgate documentation or run the `frmconfighelper` script and pass the `enable_Webgate` option.
2. Creating Webgates on the OAM console and configure the resulting policy.

   Use OAM console to create a Webgate agent, pass in the OHS host and port information and add the following to the Protected Resource List:

   ```
   /forms/frmservlet?*oamMode=true*
   ```

   Edit resources in the generated policy using the OAM console and all the following the Excluded List.

   ```
   /* and /.../*
   ```
3. Copy ObAccessClient.xml and cwallet.sso from the OAM server to the relevant OHS under the directory `DOMAIN_HOME/config/fmwconfig/components/OHS/<ohs instance>/Webgate/config`.

Unsupported Client Configurations

For information on the `frmconfighelper` script, see Oracle Forms Configuration Helper Script.

If you try to use an unsupported client configuration like an "embedded applet" or "embedded jnlp", you may see a Java exception error the first time you attempt to access an SSO-enabled Oracle Forms application . To avoid this, have the administrator disable the `HTTPOnly` parameter, which is set in OAM.

> **Note:**
>
> Understand that this change, as well as the use of an embedded applet, is not recommended. Disabling `HTTPOnly` may create a security issue. Also, the use of Microsoft Internet Explorer or Microsoft Edge with IE-mode is no longer supported. This information has been included for backward compatibility reference only.

To achieve this, perform these steps:

1. Log in to the OAM Administration Console.
2. Select **Authentication Schemes** and navigate to **LDAPScheme**.
3. Set the `ssoCookie` parameter value to *disablehttponly*.
4. Click **Apply**.

Administering Resource Access Descriptors

Resource Access Descriptors or RADs are used by Oracle Forms to allow its runtime to connect to an Oracle Database when applications are SSO-enabled. They are managed from the Resource Administration pages of Fusion Middleware Control.

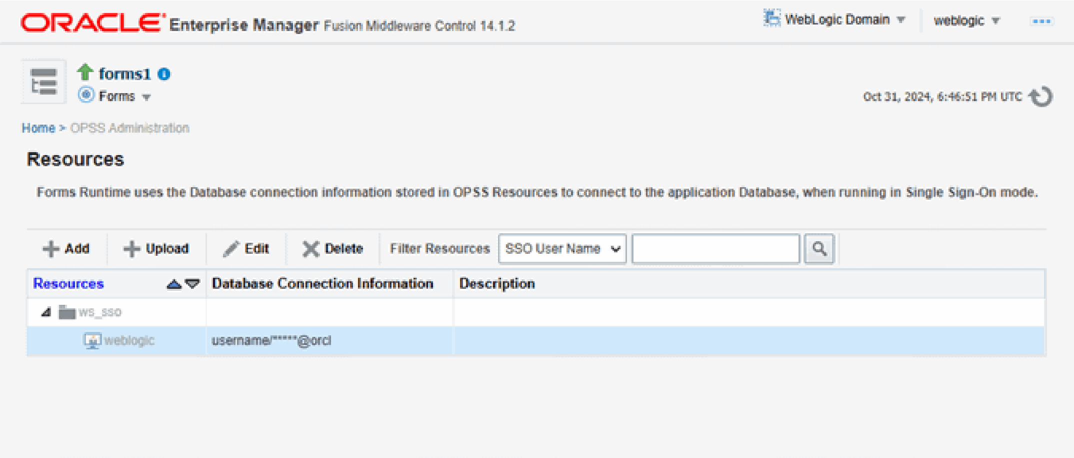
To manage RADs:

1. Log in to Fusion Middleware Control.
2. Expand the sidebar by clicking on the **Target Navigation** icon (☰) near the upper left corner next to the domain name.
3. Expand the **Forms** node then click the desired Forms instance, for example "forms1".
4. Expand the **Forms** drop-down near the upper left.
5. Select **Security**, then either **Forms OPSS Resource Administration** or **Forms LDAP Resource Administration**, depending on whether you are using OPSS or Oracle Internet Directory (LDAP) to store RAD information.

   > **Note:**
   >
   > RADs are stored in OPSS by default.

   Figure 34 shows the OPSS Administration Resources page:

Figure 34   Resources page



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain ▾
weblogic ▾
forms1
Forms ▾
Oct 31, 2024, 6:46:51 PM UTC
Home > OPSS Administration
Add
Upload
Edit
Delete
Filter Resources
SSO User Name

Resources

Forms Runtime uses the Database connection information stored in OPSS Resources to connect to the application Database, when running in Single Sign-On mode.

Resources	Database Connection Information	Description
ws_sso		
weblogic	username/*****@orcl	

6. To administer individual users or groups, use the **Add**, **Edit**, or **Delete** buttons on the **Resources** page as required.

7. To upload one or more RADs at a time, click **Upload** and provide a properly formatted text file with the desired RAD entries using this syntax:

> **Note:**
>
> Each RAD entry must be on a new line within the text file and each value must be quoted.

For individual user RADs:

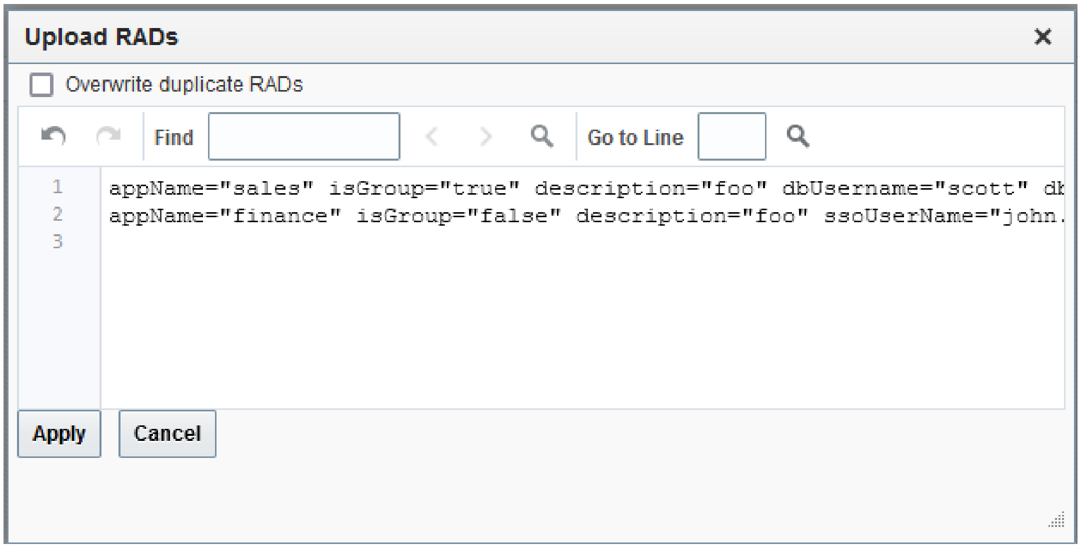
```
appName="<application name>" isGroup="<true/false>" "
description="<description>" dbUsername="<database username>"
dbPassword="<database password>" dbName="<database alias>"
```

For group RADs:

```
appName="<application name>" isGroup="<true/false>"
description="<description>" ssoUserName="<SSO username>" dbUsername="<database
username>" dbPassword="<database password>" dbName="<database alias>"
```

Figure 35 shows the Upload RADs dialog with two RAD entries:

Figure 35   Upload RADs dialog



```
appName="sales" isGroup="true" description="foo" dbUsername="scott" db
appName="finance" isGroup="false" description="foo" ssoUserName="john.
```

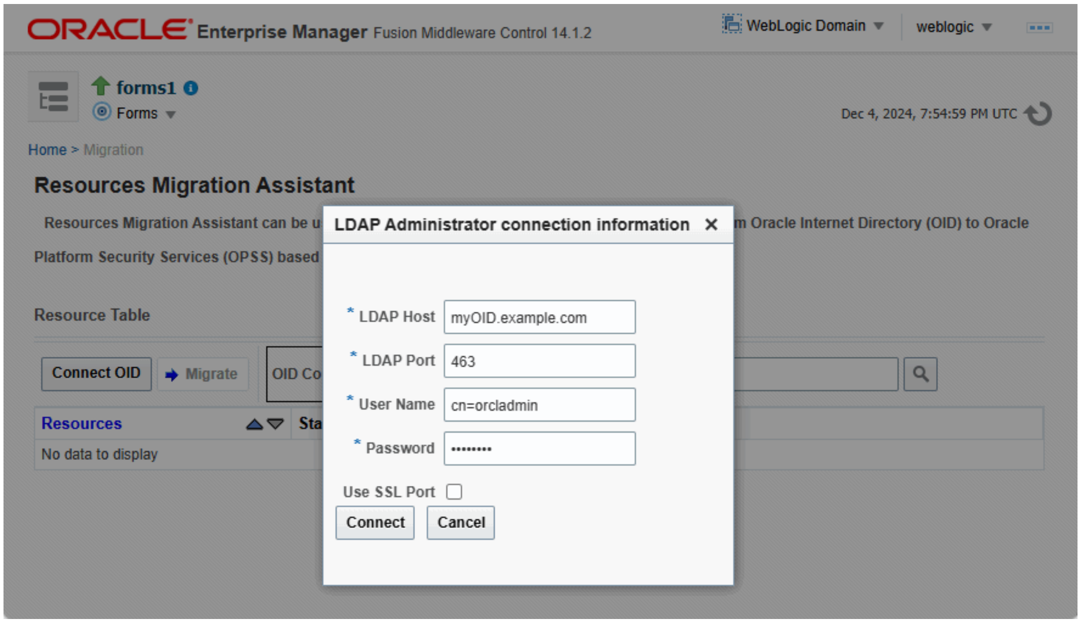
Resource Migration Assistant

The Resource Migration Assistant page allows for the migration of Oracle Forms RADs stored in Oracle Internet Directory (OID) to be moved to Oracle Platform Security Services (OPSS). This utility is intended for the purpose of migration from OID to OPPS only.

To access the Resource Migration Assistant page, perform the following steps:

1. Log into Fusion Middleware Control.
2. Expand the sidebar by clicking on the icon near the upper left corner, next to the domain name
3. Expand the **Forms** node then click the desired Forms instance, for example "forms1".
4. Expand the **Forms** drop-down near the upper left.
5. Select **Security** then select **Resource Migration**.
6. The Resource Migration page will be displayed. You will be required to enter information about the Oracle Internet Directory (OID) server to be accessed for the migration process. This selection can be changed after accessing the page by clicking on the Connect OID button.

Figure 36 Resource Migration Assistant



Resources Migration Assistant

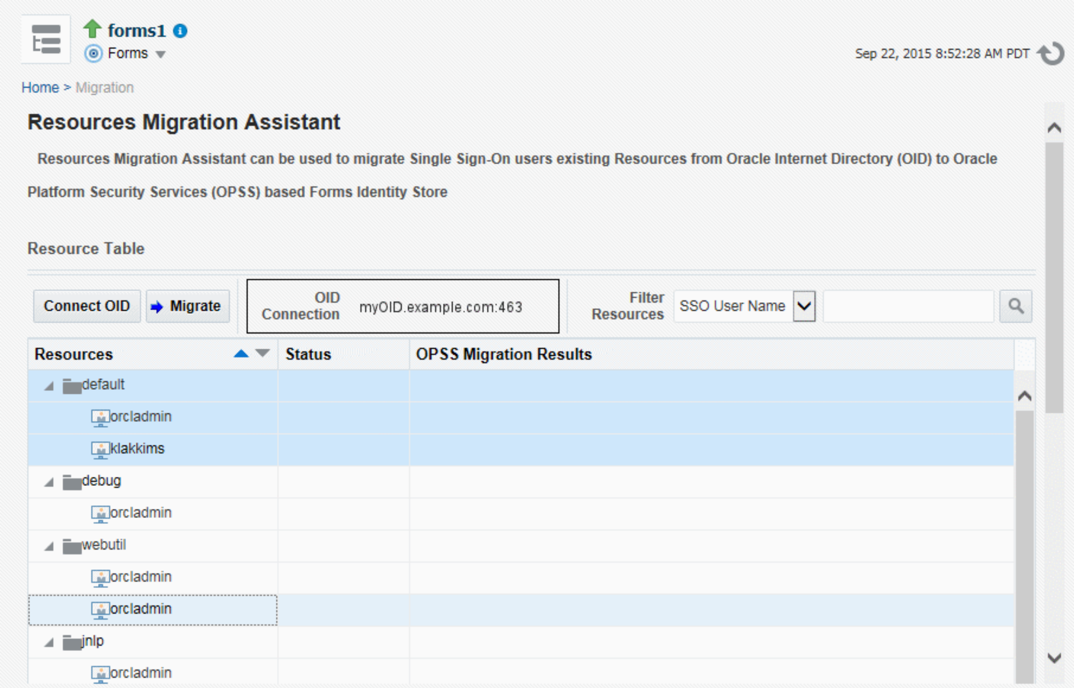
Resources Migration Assistant can be u... m Oracle Internet Directory (OID) to Oracle Platform Security Services (OPSS) based

LDAP Administrator connection information ✕

* LDAP Host	myOID.example.com
* LDAP Port	463
* User Name	cn=orcladmin
* Password	••••••••
Use SSL Port	□

7. Once on the Resource Migration page, the table will display all the resources found in the OID selected. Select the entries in the table that should be migrated to OPSS then click **Migrate**. The status of the transfer will be displayed in a popup dialog.

Figure 37 Resource Migration Assistant page



Resources Migration Assistant

Resources Migration Assistant can be used to migrate Single Sign-On users existing Resources from Oracle Internet Directory (OID) to Oracle Platform Security Services (OPSS) based Forms Identity Store

OID Connection: myOID.example.com:463

Filter Resources: SSO User Name

Resources	Status	OPSS Migration Results
default		
orcladmin		
klakkims		
debug		
orcladmin		
webutil		
orcladmin		
orcladmin		
jnlp		
orcladmin		

Integrating Oracle Forms with IAM Cloud Service

Oracle Forms supports using Oracle Identity and Access Management (IAM) Cloud Service to provide identity management, single sign-on (SSO), and identity governance for Forms applications.

To take advantage of IAM functionality, Oracle Forms applications must be launched using Java Web Start or Forms Standalone Launcher. Launching applications using the embedded applet configuration in Microsoft Internet Explorer or Edge (with IE-mode) is not supported.

To configure Oracle Forms with IAM, you'll need to perform these tasks:

- Create an application on the IAM server
- Create an App Gateway
- Set up the App Gateway Client using docker
- Enable SSO in an Oracle Forms application

These instructions assume that the IAM App Gateway will be configured on a Linux operating system.

Refer to the IAM documentation for additional details.

Before You Begin

Before you begin, make sure you have the necessary privileges and certificates.

You'll need:

- Administration privileges in the IAM Service environment
- Privileges to download and configure the IAM App Gateway on the middle tier or the desired alternative server where App Gateway will be configured
- Privileges to access and make administrative changes to the middle tier software—in this case, Oracle Forms
- A current SSL/TLS certificate (from a trusted CA) associated with the host that will be running the App Gateway client and/or the Forms middle tier. (Recommended but not required)

Enable IAM Integration in Oracle Forms

Now that the IAM App Gateway has been configured, configure Forms to use it.

To enable IAM for Forms:

1. Using Fusion Middleware Control, access the **Forms Web Configuration** settings page.
2. From the menu on the upper left side, expand the **Forms** node, then click the forms instance to be modified (for example, `forms1`).
3. Click **Web Configuration** to access the administration screen.
4. On the upper right side, click the padlock icon then select **Lock & Edit**.

5. In the upper table, select the desired Section Name to be used with IAM. If enabling for all, select `default`.

6. Scroll down to the lower table and expand the pop-list labeled "Show" and select `sso` from the list.

7. Set the value of `ssoMode` to `TRUE`.

8. If you plan to use Java Web Start to launch the applications, click **Add** at the top of the parameters table, then add the parameter `webstart_codebase`.

   The value is the URL pointing to the Forms codebase, but use the IAM host and port. For example:

   ```
   https://<IAM AppGateway host>:<port>/forms/java
   ```

9. Click **Apply** above the current table.

10. On the upper right side, click the padlock icon then select **Activate Changes**.

11. To run your Forms app, use this format:

    ```
    https://<IAM AppGateway host>:<port>/forms/frmservlet?config=<Forms
    config name>
    ```

12. Perform these steps to enable the JWS setting to automatically remove the downloaded jnlp file after it has been launched. Since the Java Web Start jnlp file cannot be reused, there is no reason to retain it after it has been used.

    This will help to improve the user’s experience by not creating duplicate files, as well as improve security.

    > **Note:**
    >
    > These steps are recommended if this is an upgraded domain. For new installations and new domains, this is configured by default.

    a. In **Fusion Middleware Control**, navigate to the forms instance (for example, `forms1`) associated with this IAM configuration.

    b. From the **Forms** drop-down list, select **Advanced Configuration**.

    c. From the **Activate Edit Session** switch (padlock upper right side), select **Lock & Edit**.

    d. From **Select Category**, select **Client Templates**.

    e. From **Select File**, select either:

       - `base.jnlp` or `webutil.jnlp` if using a WebUtil enabled app
       - Your own custom jnlp template if one was created and is in use

    f. Add the following to the <resources> section of the template exactly as it appears here:

       ```
       <property name="jnlp.delete.jnlp.file" value="true"/>
       ```

13. Click **Apply** to save the change.

14. Using the **Activate Edit Session** switch (padlock upper right side), select **Activate Changes**.

Notes and Limitations

Review these notes when using IAM with Oracle Forms:

- The use of Microsoft Internet Explorer or Microsoft Edge with IE-mode are not supported with this configuration.
- The Oracle Cloud tenancy used for this configuration must offer Identity and Access Management (IAM) and not its predecessor Identity Cloud Service (IDCS). If your tenancy requires the creation of an Identity Domain, you are using IAM.
- The use of a self-generated SSL/TLS certificate, as instructed in this document will result in various warnings during application startup. It is therefore recommended that a certificate be obtained from a known and trusted certificate authority.
- The use of the provided Fusion Middleware SSL/TLS Demo/Example certificate is not supported for use with this configuration.
- It may not be possible to use a self-generated SSL/TLS certificate if launching applications using the Forms Standalone Launcher (FSAL). A certificate provided by a known and trusted CA, as described above should be used.

Troubleshooting

If you encounter issues with the implementation, review these issues and causes before contacting Oracle Support.

When Did the Issue Occur?	Issue/Error	Possible Causes	Corrective Actions
Setup	The `PATCH` payload is invalid. The "op" or "Operations" attribute can't be null or empty.	• The Edit panel **Save** button was pressed, but no changes were made. • You attempted to add a Resource or Policy without first saving previous changes.	• Use the **Cancel** link rather than **Save** if no changes were entered. • Cancel current changes and save previous changes. Reattempt to add new changes.
Runtime	`HTTP-502` (Bad Gateway)	• Forms Managed Server is not running. • WLS Managed Server is not accessible by the App Gateway client.	• Verify the Managed Server is running. • Verify firewall (OS and/or external) is allowing communication between the App Gateway client and middle tier server and IAM Cloud Service and App Gateway client. (e.g. ports). • Verify the setting for the Origin server is correct.

When Did the Issue Occur?	Issue/Error	Possible Causes	Corrective Actions
Runtime	`HTTP-504` (Gateway Time-out)	Origin server (Forms middle tier) not accessible.	• Verify the setting for the Origin server is correct. • Verify the managed server is running. • Verify the managed server port is accessible through the firewall.
Runtime	• "No subject alternative names present." or • "FRM-92575: SSL/TLS hostname verification failed."	• You are using a self-generated SSL certificate or a certificate that is not properly formatted. • The server name (App Gateway client machine) and the server name used in the SSL certificate do not match.	• Obtain a certificate from a trusted Certificate Authority (CA). • Ensure the server name used with the certificate request matches the server name (in the URL) on which the IAM App Gateway is running.
Runtime	"FRM-93261: JNLP file launched from unexpected IP address"	• You have attempted to copy the downloaded jnlp file (if configured to use Java Webstart) and moved the file to another machine. • The administrator has enabled the `jnlpMatchIP` setting (set to `TRUE`).	• Be sure to not copy/move the downloaded jnlp file to another machine, as this is not supported. • Disable (set to FALSE) the Forms Web Configuration parameter `jnlpMatchIP`.

When Did the Issue Occur?	Issue/Error	Possible Causes	Corrective Actions
Runtime	Java error indicating that `extensions.jnlp` cannot be downloaded when trying to run with Java Web Start.	App Gateway client not running on same host with middle tier.	• Verify that `extensions.jnlp` exists on the server and its file permissions are appropriate for accessing. • Verify the file can be downloaded from a web browser. Use the same protocol, IAM server hostname, and port used by the App Gateway client. • Set Forms Web Configuration parameter `webstart_codebase`. The value should reflect the same protocol, IAM server, and port used by the App Gateway client.

Configure the IAM Server for Oracle Forms

Create an IAM application and an App Gateway on the IAM server.

- Create an IAM Application
- Create the App Gateway

Create an IAM Application

Create the IAM Application, as well as create the needed resource settings and authentication policies.

To create an IAM application;

1. Log in to your Oracle Cloud tenancy where you plan to set up your IAM environment.
2. From the menu on the left pane, select **Identity and Security**.
3. Under **Identity**, click **Domains**.
4. Select the desired domain.

   Any Domain Type can be used, however carefully consider the limitations associated with each when selecting.

   If you have not already created a domain for IAM, create one now by clicking **Create Domain**.
5. Click **Integrated Applications** on the left, then click **Add Application**.

6. From the Add Application dialog, select **Enterprise Application**, then click **Launch Workflow**.
7. Enter the following:
   - **Name**: Provide a name for this IAM Application
   - **Description**: (Optional) Provide a description for this IAM application
   - **Application Icon**: (Optional) Provide an icon that will appear in the Applications table
8. In the URLs section, type the fully-qualified URL for the IAM App Gateway environment in the **Application URL** field.

   This URL should include the Forms context root and app name, such as `/forms/frmservlet`. Do not include application arguments.

   For example:

   ```
   https://<APP GATEWAY HOST.DOMAIN>:<IAM SSL PORT>/forms/frmservlet
   ```

   > **Note:**
   >
   > An IP address can be used in place of the *host.domain*.

9. Select **Display in My Apps**, then click **Next**.
10. Click **Next** to skip the OAuth configuration and proceed to the next screen.
11. Click **Add Resource** to add resources for each numbered row. Adding a description is optional, but recommended.

Priority	Resource Name	Resource URL	URL Query String	Use Regex Expressions	Description
1	Protected resource	`/forms/frmservlet.*`	`oamMode=true`	Yes	Forms application protected resource
2	Anonymous access resource	`/forms/frmservlet.*`		Yes	Forms application anonymous access
3	Anonymous access resource 2	`/forms/frmservlet`	`ifcmd=startsession`		Forms application anonymous resource 2
4	Protected RAD resource	`/forms/radservlet.*`		Yes	Forms application protected RAD resource
5	Public resource	`/.*`		Yes	Forms application public resource

12. Ensure these check boxes are selected:
    - **Audience Validation**
    - **Require Secure Cookies**
    - **Add Managed Resources**

13. Click **Add Managed Resource** to add managed resources for each numbered row. Be sure the Priority order matches the table shown.

Priority	Resource	Authentication Method	Enable Audience Validation
1	Protected resource	Form or Access Token	Yes
2	Anonymous access resource	Anonymous	
3	Anonymous access resource 2	Anonymous	
4	Protected RAD resource	Form or Access Token	Yes
5	Public resource	Public	

14. Click **Finish** to return to this Integrated Application's Details screen.
15. Click **Activate** above the Application Information tab.

Create the App Gateway

Create an App Gateway associated with your domain.

> **Note:**
>
> The IAM Service environment must be accessible from the server where the IAM App Gateway (Client) is running.

To create an App Gateway:

1. Navigate to your domain's Overview screen using the breadcrumb at the top left of the page.
2. From the menu on the left, click **Security**, then **App Gateways**.
3. Click **Create App Gateway**.
4. Type an easily identifiable name and description in the **Name** and **Description** fields.
5. Click the **Add Host** link on the left side, then click the **Add Host** button.
6. Enter these two entries, replacing the Host Identifier, Host, and Port with those that match your system:

   > **Note:**
   >
   > Note that the first row is for requests received as non-SSL and the second is for handling SSL requests. Non-SSL requests will be redirected to SSL to ensure the highest degree of security.

Host Identifier	Host	App Gateway Port	SSL Enabled	Additional Properties
`appgateway-nonssl`	*<App Gateway client hostname>*	`7777` (non-SSL port)	NO	`return 301 https://$host:4443$request_uri;`
`appgateway-ssl`	*<App Gateway client hostname>*	`4443` (SSL port)	YES	`ssl_certificate /home/oracle/keys/ssl.cert; ssl_certificate_key /home/oracle/keys/ssl.key; ssl_protocols TLSv1 TLSv1.1 TLSv1.2; ssl_ciphers HIGH:!aNULL:!MD5;`

Before you continue, review the Additional Properties values and ensure they are appropriate for your environment. References to host, IP address, or port should reflect those which are user-accessible.

The certificate and key paths presented in the Additional Properties column must be as shown regardless of whether you are using a certificate provided from a trusted CA or a self-generated certificate.

Refer toSetting up App Gateway in the *Oracle Cloud Infrastructure Documentation*.

7. Click **Add Host** to accept the changes, then click **Next**.
8. Click **Add App** and do the following:
   a. Select the app you just created from the **Application** list.
   b. Select the SSL entry created earlier from the **Select a Host** list.
   c. Type a forward slash (`/`) in the **Resource Prefix** field.
   d. Type the fully-qualified non-SSL URL that points to the middle tier (Forms) host.domain:port in the **Origin Server** field.

      Use the SSL URL if not using non-SSL.
9. Type the following string in the **Additional Properties** field, replacing *<port_number>* with the port used by your Forms managed server:

```
proxy_set_header Host $host:<port_number>;
```

> **Note:**
>
> If the App Gateway client is running on a different machine than the Forms middle tier, replace `$host` with the user-accessible host.domain (or IP address) of the Forms middle tier.

10. Click **Add App** at the bottom, then click **Close**.

    Make note of the Client ID and Client Secret now seen in the App Gateway Information panel. Do not share this information with others. This information will be needed in the following section.

11. Click **Activate App Gateway**.

Create and Configure an App Gateway Client

Create the App Gateway client and start the server.

This requires you to set up the App Gateway client, install the docker, and run the docker image.

Before you begin, review these requirements:

- Administrative access to the machine where the App Gateway will be running is required.
- Because the App Gateway client will be using ports 7777 and 4443 in this example, stop all processes currently running on those ports. For example, Oracle HTTP Server typically uses these ports by default. Be sure to stop that server or any others that might use those ports before continuing.

  > **Note:**
  >
  > The port numbers used here are only examples. Use whatever valid port numbers are desired. However, be sure the ports selected do not create a port conflict with other listeners on the machine and be sure to use the chosen values throughout the configuration.

- Ensure you have the Client ID and Client Secret previously obtained in an earlier step above.
- Ensure that Docker is properly installed and running on this machine. If not, do so now. Installation will require “root” access.

Set up the App Gateway Client on the Middle Tier

Create the App Gateway client using the App Gateway Docker Image for Identity Cloud Service as well as a wallet using the App Gateway Wallet Tool.

To set up the App Gateway client:

1. Navigate to the Identity Domain Overview screen for the domain created in Create an IAM Application.
2. Click **Settings** from the menu on the left.
3. Click **Downloads**.

4. Download the latest version of these software packages onto the machine where the App Gateway client will be configured (the middle tier machine).

   - App Gateway Docker Image for Identity Cloud Service
   - App Gateway Wallet Tool

5. Create directories for the Docker image, Wallet Tool, and for storing the key and certificate files that will be created in the following steps.

   For example, enter the following commands:

   ```
   mkdir /u01/oracle/appgateway/docker
   mkdir /u01/oracle/appgateway/wallettool
   mkdir /u01/oracle/appgateway/keys
   ```

6. Extract the contents of the Docker Image downloaded earlier into the ‘docker’ directory and the contents of the Wallet Tool into the ‘wallettool’ directory.

   The docker extraction will include a `.gz` file. Do not extract it. Leave it in the `gz` format.

7. Change the current directory to enter the ‘keys’ directory created earlier.

   ```
   cd /u01/oracle/appgateway/keys
   ```

8. Create a self-generated certificate with this command. Make changes in the example to accommodate your needs.

   ```
   openssl req -new -newkey rsa:2048 -days 730 -nodes -x509 -subj
   /C=US/ST=CA/L=RedwoodShores/O=OracleUSAInc/OU=Forms/CN=yourserver.com/
   emailAddress=youradministrator@example.com
   -keyout ssl.key -out ssl.cert
   ```

   > **Note:**
   >
   > Using a self-generated certificate is not recommended for production use. Further, such a configuration may not work correctly when attempting to run an Oracle Forms application using Forms Standalone Launcher. Using a certificate provided by a known and trusted certificate authority is always recommended.

9. Change directory to the location of the Wallet Tool:

   ```
   cd /u01/oracle/appgateway/wallettool
   ```

10. Execute each of the following one line at a time in the shell:

    ```
    export CLIENT_ID=<ENTER YOUR CLIENT ID FROM THE EARLIER STEP ABOVE>
    export CLIENT_SECRET=<ENTER YOUR CLIENT SECRET FROM THE EARLIER STEP ABOVE>
    echo $CLIENT_SECRET | env LD_LIBRARY_PATH=./lib ./cgwallettool --create -
    i $CLIENT_ID
    ```

11. Move the created wallet to the ‘keys’ directory:

    ```
    mv cwallet.sso ../keys/cwallet.sso
    ```

12. Change directories to the ‘appgateway’ directory.

For example:

```
cd /u01/oracle/appgateway
```

13. In the ‘appgateway’ directory, create a text file and name it `appgateway-env` (do not include an extension). Add the following contents to the file and save it. Replace < > entries with those that are appropriate for your system.

```
CG_APP_TENANT=<tenant name>
IDCS_INSTANCE_URL=<idcs instance url>
NGINX_DNS_RESOLVER=<resolver ip>
```

CG_APP_TENANT: The tenant name identifier of the Identity Domains instance. This is not your Cloud account/tenancy name. The IAM tenant name is the first portion of the IDCS Instance URL (up to the first dot, but not including the protocol) you will use below. On the Domain Overview page, the “Domain URL” can be found in the Domain Information panel (See Figure 6 below). Copy the first portion of the URL. For example: idcs-1234567890abcdefghijk

IDCS_INSTANCE_URL: The URL required to access the IAM instance. Enter the entire URL. This can be found on the Identity Domain Overview screen in the Domain Information tab.

NGINX_DNS_RESOLVER: Enter the nameserver found in the file `/etc/resolv.conf`. The default value is `127.0.0.1`.

Run the Oracle App Gateway Docker Container

Install and run Docker and the App Gateway Docker Image.

Because the required App Gateway client is only available as a Docker Image or full Virtual Machine (VM), familiarity with using either Docker or running a Virtual Machine will be helpful. This document will only cover the use of the Docker Image. A similar configuration approach can be used for using the full VM. Refer to the IAM documentation for assistance with configuring and using the full VM.

Install and Start Docker

Install and start Docker.

On newer Linux versions `podman` may be used in place of `docker`. However, the command arguments should be the same for both. If necessary, refer to the official Docker or Podman documentation for more guidance on using these utilities.

To install Docker:

1. If Docker is not already installed on the machine, as the “root” user install Docker with this command:

```
yum install docker-engine
```

> **Note:**
>
> For Linux 8 or newer, use `dnf` instead of `yum`.

2. As the “root” user add the current user to the Docker group. Skip the first command if a “docker” group already exists.

```
groupadd docker
usermod -aG docker oracle
```

> **Note:**
>
> Replace “oracle” above with the user that will own the AppGateway Docker environment.

3. As the “root” user, start Docker with the following command then return to your user:

```
systemctl start docker
```

4. Verify that the file `/var/run/docker.sock` has 766 permissions. If not, as the root user change the permissions on the file using the chmod command.

> **Note:**
>
> If this permission is not set properly, the user (for example, `oracle`) will not be able to connect to the Docker image or start it.

5. Exit from the ‘root’ user and verify that Docker is running and can be accessed by the current user (for example, `oracle`).

   The following command should return the contents of the local repository, which will be empty if this is a new installation. Otherwise, it will list the images previously created. No errors should be presented.

   It may be necessary to reboot the machine if Docker does not start without error and you just installed it. If any permissions errors are presented, refer to the previous step.

```
docker images
```

Import the App Gateway Docker Image

Import and load the App Gateway Docker Image

To import and load the App Gateway Docker Image:

1. Open a shell and change directory to the directory where you extracted the App Gateway Docker file (`.tar.gz`) in Set up the App Gateway Client on the Middle Tier.
2. Execute the following in order to load the image:

```
docker load --input appgateway-<VERSION NUMER>.tar.gz
```

Run the Docker Image

Run the Docker image for the first time.

To run the Docker image for the first time:

1. List the available images by executing this command. Some of this information will be needed in order to start the image:

   ```
   docker images
   ```

   This will output something similar to the following:

   ```
   REPOSITORY              TAG                     IMAGE ID      CREATED       SIZE
   idcs/idcs-appgateway   23.2.101-2304020420   ba1249c25dc3  9 months ago
   498MB
   ```

2. Execute the following command.

   This command is a single line that wraps visually. Be sure to enter it as a single entry, adjusting the values to match your system and desired settings. Replace the `idcs-appgateway` version with the version you downloaded. The version number is part of the downloaded file name.

   ```
   docker run -it -p 4443:4443 -p 7777:7777 --name appgateway --env-file
   /u01/oracle/appgateway/appgateway-env --env HOST_MACHINE=`hostname -f` --
   volume
   /u01/oracle/appgateway/keys/cwallet.sso:/usr/local/nginx/conf/cwallet.sso
   --volume
   /u01/oracle/appgateway/keys:/home/oracle/keys --net=host idcs/idcs-
   appgateway:23.2.101-2304020420
   ```

   > **Note:**
   >
   > Refer to the IAM Setup documentation for more information.

3. Given the above example settings, the App Gateway client listener in the container can be accessed using the `hostname:4443` or `hostname:7777`. For example: `https://<hostname>:4443`

4. To verify IAM and the App Gateway are responding, run the following from a browser:

   `https://<app gateway host>:<port>/cloudgate/v1/about`

   This will return a JSON page indicating a “RUNNING” status. If you see any other result, likely the App Gateway is not configured correctly and/or is not accessible from the client browser.

   If the above is not accessible, verify that the value of “HOST_MACHINE” provided above is an accessible host name by users. If you do not have a registered domain name you can alternatively use the machine’s public facing IP address.

   The above command can only be used the first time the contain image is started. Refer to Stop the Docker Container for information on restarting after having stopped previously.

Stop the Docker Container

To stop the running Docker container:

1. Determine the Container ID by running this command:

   ```
   docker ps
   ```

   The output will resemble the following, although only two columns are shown below:

   ```
   CONTAINER ID    IMAGE
   ebe97115c307              idcs/idcs-appgateway:22.2.61-2203221920
   ```

2. Using the Container ID from above, execute the “stop” command.

   ```
   docker stop <container ID>
   ```

   You can alternatively use the container name:

   ```
   docker stop <container name>
   ```

After the container has been stopped, it can be restarted using the same settings previously used, with the following Docker command:

```
docker start <contain name>
```

Configuring and Managing Java Virtual Machines

When an Oracle Forms application calls out to Java on the server, a Java Virtual Machine (JVM) is spawned and attached to its Runtime process the first time the call is made. This chapter provides information about java virtual machine pooling, JVM processes, multiple JVM controllers, JVM configuration and controllers.
This chapter contains the following sections:

- Java Virtual Machine Pooling
- Child JVM Processes
- Multiple JVM Controllers
- JVM Pooling Usage Examples
- Design-time Considerations
- Configuring JVM using Fusion Middleware Control
- Manage JVM Controllers from the Command Line
- Managing JVM Pooling from Fusion Middleware Control
- JVM Controller Logging

Java Virtual Machine Pooling

JVM remains attached to the Runtime process, the first time Oracle Forms application calls out to Java on the server, for the remainder of the process's life, even though the process may never call out to Java again. Given that each Forms Runtime session creates its own JVM instance, the amount of resources consumed on the server can become significant.

In a JVM Pooling environment, Forms Runtime processes share JVMs. A single JVM is capable of handling multiple Forms sessions from different Runtime processes. The pooling environment helps in greatly reducing the memory footprint on the hosted machine by eliminating the need of attaching one JVM per Runtime process. This environment is configurable. Forms administrators can set various parameters and tune the environment based on their needs and requirements. A limit on the number of Runtime sessions managed by a single JVM process in the pool can also be set.

When using JVM pooling, a JVM Controller is created. This is a JVM with specific responsibilities. Besides accepting connections from Runtime processes, it is also responsible for creating new JVMs when necessary. A new JVM process (also referred to as a Child JVM) is created only when the Controller finds that the existing JVMs in the pool (including itself) are unable to accommodate further sessions from Runtime processes.

Oracle Forms JVM pooling works with applications that used the Forms Java Importer during development. The Java Importer allows developers to reference Java classes from PL/SQL at design time. At runtime, the Java classes are loaded and executed by the JVM(s) as needed. Forms JVM pooling also works with Forms' ability to call out to Oracle BI-Publisher and Oracle Reports. Enabling the JVM pooling feature is recommended if any applications being executed fall into one of the aforementioned cases.

For information on the Java Importer, see Oracle Form Builder Online Help.

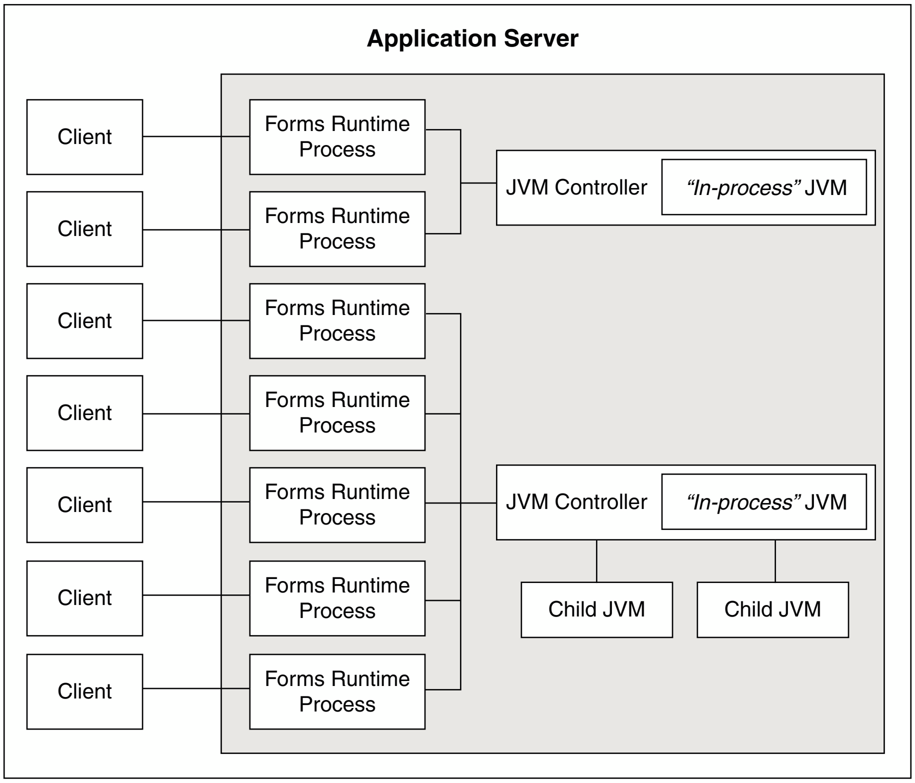
Child JVM Processes

Since each Forms runtime process has its own thread within the JVM, there is concurrency. If the JVM reaches a specified number of concurrent requests, it will spawn a child JVM to share the load. Moreover, it's possible to have multiple JVM controllers, each of which may have multiple child JVMs.

For example, different Forms applications need to use different JVMs with different options or classpath. You can specify which JVM controller and Forms application is necessary in the named sections of the Forms configuration file (formsweb.cfg). Alternatively, this information can also be passed as a parameter in the URL for invoking the Forms Application. A form can be configured to use a specific JVM controller using the jvmcontroller parameter. The jvmcontroller parameter indicates to the Forms Runtime process which JVM controller to use. The parameters that need to be used during startup of the jvmcontroller have to be specified in the the JVM controller's configuration file, jvmcontrollers.cfg, see Forms Configuration File Settings.

The following figure shows an example of what an environment might look like using JVM pooling. There are two JVM controllers: the first one is using only its in-process JVM, the second one is using three JVMs.

Figure 38 Multiple JVM Controllers with Child Processes



Although it's not shown in Figure 38, each JVM controller has a unique name which is used in starting and stopping, or for referencing in the Forms configuration file.

Figure 38 is conceptual only in that it shows different Forms applications using different JVM controllers. However, the Forms runtime process does not communicate with the JVM controller, but directly with one of the available JVMs. Therefore, the first two clients in the diagram can only use the in-process JVM; the rest have three available JVMs to work with.

When the performance of a JVM degrades significantly, it probably means it is servicing too many requests. In that case, it is possible to have multiple child JVMs for the same JVM controller which get created dynamically as needed.

The JVM parameter `maxsessions` specifies how many Forms runtime processes are allowed to attach to a JVM before a new child JVM is created. When a child JVM is started, it inherits the same parameters as the JVM controller.

If any JVM has `maxsessions` connections, it does not take any request from new Forms runtime processes. When a new Forms runtime process first attempts to execute Java code, it attaches to a JVM that is available, that is, has fewer than `maxsessions` connections.

If a JVM reaches maxsessions connections, but another JVM has not, no new JVM is created. If all JVMs have simultaneously reached maxsessions connections, another child JVM is created, and so on.

The scope of a child JVM is within the context of a JVM controller namespace. For example, if you have two JVM controllers, `ordersJVM` and `hrJVM`, ordersJVM and its child JVMs do not affect – nor are affected by – `hrJVM` or its child JVMs.

Child JVM Example

Suppose the JVM controller called ordersJVM has `maxsessions=50`. Each Orders application that runs sends requests to ordersJVM. Each time a new Forms runtime process sends a request to ordersJVM, a new thread is created that communicates with the Forms runtime process. The JVM controller then returns to listening for new requests. As users end their sessions, the threads in the JVM are also terminated.

When the ordersJVM controller receives the 50th concurrent request (not necessarily the first 50 users because some of them may have quit before the later users started) it will spawn a child JVM. Since it inherits its parent's settings, `maxsessions` for this child JVM will also be `50`. At this stage, the JVM controller has 50 connections, and the child JVM has none.

As new users start this Oracle Forms application and execute Java code, the Forms runtime process attaches to a JVM that is listening within the JVM controller namespace. Since the JVM controller has 50 connections, it is unavailable and the child JVM receives the request. Later, when the parent JVM controller has fewer connections because some users have quit their applications, it is available to receive new requests as long as it has not reached `maxsessions` connections.

While all this is going on, the `hrJVM` is operating independently. Overflow connections from `ordersJVM` will not connect to `hrJVM`, only to child JVMs of `ordersJVM`.

Child JVM Management

The JVM Controller can monitor usage and cleanup or remove unused processes when appropriate. This can help to prevent wasting valuable server resources and further improve performance.

By default, this cleanup feature is not enabled. To enable it, set the parameter `autoremoval` in the JVM Controller configuration. Valid values are listed in the following table.

Table 30 Child JVM Management

Value	Description
OFF (default)	Auto-removal feature is `disabled`. JVMs would not be removed automatically by JVM Controller. The pool size will continue to grow, but not shrink. Child JVMs will continue to live until terminated manually or terminated by the Controller upon its exit.
AGGRESSIVE	Auto-removal feature is `enabled`. The frequency of removing Child JVMs is at its highest. Assuming the JVM can accommodate a maximum of M sessions, in this configuration, the Controller will keep a buffer (spares) of M/2 for accommodating future session requests. One advantage of this setting is that the pool would always have the least possible number of child JVMs to serve the current load plus a maximum of one spare for accommodating future requests. A disadvantage of this setting is that in an active environment (frequent session starts and stops) the frequency of JVMs exiting/ creating would be higher. The Controller may become excessively busy managing the children. If all sessions are closed, the pool size would shrink to 1 (the JVM Controller).

Table 30   (Cont.) Child JVM Management

Value	Description
MODERATE	Auto-removal feature is `enabled`. The frequency of removing Child JVMs is lower than AGGRESSIVE. Assuming the JVM can accommodate a maximum of M sessions, in this configuration, the Controller would keep a buffer (spares) of M for accommodating future session requests. If all sessions are closed, the pool size would shrink to 1 (the JVM Controller).
CONSERVATIVE	Auto-removal is `enabled`. The frequency of removing Child JVMs is lower than the previous two options. Assuming the JVM can accommodate a maximum of M sessions, in this configuration, the Controller would keep a buffer of 3*M/2 for accommodating future session requests. If all sessions are closed, the pool size would shrink to 2 (the JVM Controller and 1 child).

JVM Load Balancing

To allow connections to be more organized and uniform, a load distribution technique like Round Robin or Least Loaded First or both can be incorporated in the JVM Controller. This load balancing feature is optional and can be configured in the JVM controller configuration file. To use this feature, a set the parameter '`loadbalance`' in the configuration file. It can be set with any of the following options:

- Least Loaded First
- Round Robin
- Random

Valid values are described in the following table.

Table 31   JVM Load Balancing

Value	Description
RANDOM (default)	In Random mode, the JVM Controller operates as it did in previous versions. All children created by the Controller are free to accept new connections. Assuming a JVM is available to receive a new connection, it will.
LEASTLOADEDFIRST	In Least Loaded First mode, the JVM Controller monitors and controls the connection accepting behavior of the children JVMs. Only one child JVM would be allowed to listen for new connection requests at a time. To schedule a child JVM, the JVM Controller would iterate though all the child JVMs in the pool and select a child JVM which is serving the least number of sessions. It would instruct the selected child JVM to listen for the next connection request. The scheduled child JVM would acknowledge back to the JVM Controllers after accepting the session request. The JVM Controller would initiate the load balancing sequence again and look for the next least loaded child JVM from the pool.

Table 31 (Cont.) JVM Load Balancing

Value	Description
ROUNDROBIN	In Round Robin mode, the JVM Controller monitors and controls the connection accepting behavior of the children JVMs. To distribute the load, the Controller iterates through the list of JVMs and gives each a fair chance to accept new connection requests. Initially, the Controller would start with the first JVM in the list and instruct it to start accepting connection requests. The JVM Controller would receive the acknowledgment from currently schedule child JVM then move to the next available child available child and initiate the load balancing sequence again. The Controller will cycle through all available JVMs.

Multiple JVM Controllers

The JVM pooling architecture allows you to have multiple JVM controllers, each of which may have child JVMs.

You would use multiple JVM controllers if:

- You want each application to have its own JVM controller so that it can be started and stopped independently of others.
- Different applications require different settings. For example, you may not want to mix classpaths or JVM settings between different controllers.
- You want to monitor resource usage of the JVM controllers from Fusion Middleware Control. If different JVM controllers are used by different applications and/or groups of users, you can determine how resources are being consumed by your Java Importer code.
- You have multiple development, test, or production environments on the same computer.
- You do not want different applications to share static data.

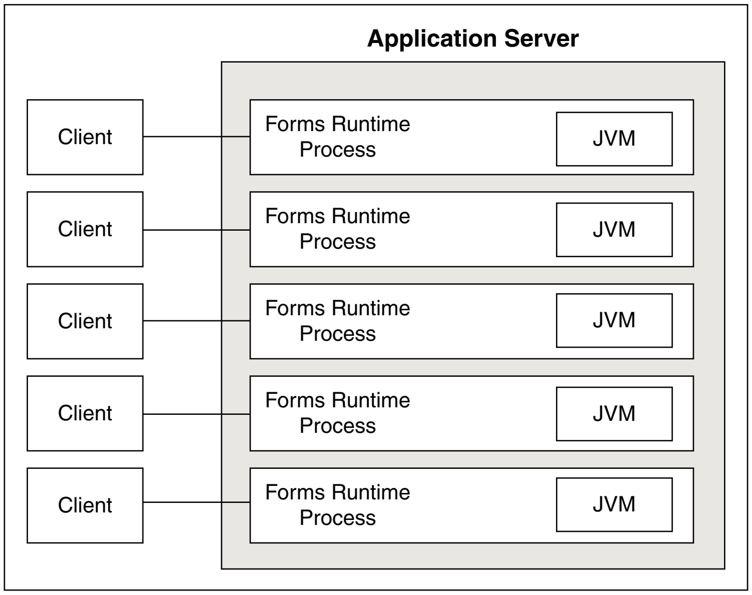
JVM Pooling Usage Examples

In this example, consider a Oracle Forms application that has a user interface button.

When a user presses the button, Oracle Forms takes the value from a field on the screen, and passes it to Java (using the Java Importer feature) to do some complex calculation which cannot be done in PL/SQL. The result is then returned and displayed in a field in the Form. One JVM process is running to execute this Forms session.

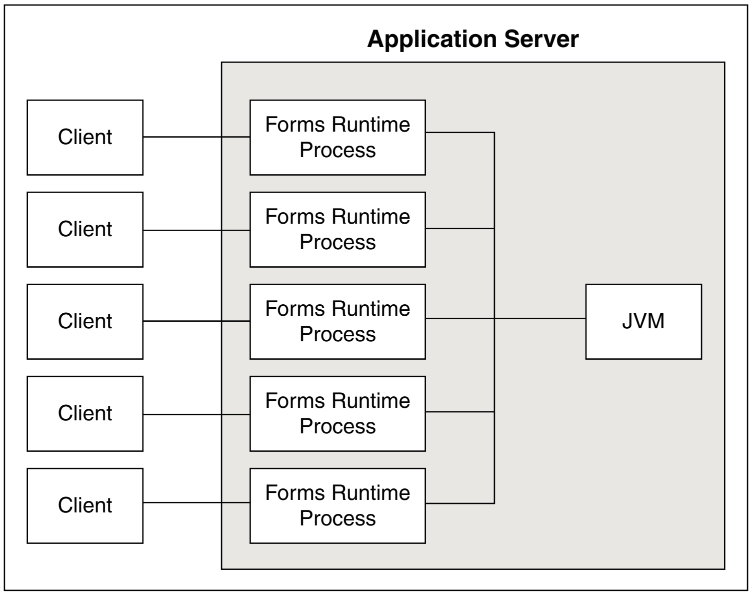
The following figure shows how this Oracle Forms session has its own in-process JVM because JVM pooling is not enabled. In the left side of the image, there are multiple clients running their own Forms session. In the center of the image, each client makes a call to its own Forms Runtime process, which contains its own JVM process.

Figure 39 Forms Runtime with no JVM Pooling



The following figure shows the Forms Runtime processes sharing a single JVM process when JVM pooling is enabled, as shown in the right side of the image.

Figure 40 Forms Runtime with JVM Pooling Enabled



In this example as shown in the above figure, five clients working in the same application through their own runtime processes are using a pooled JVM process instead of each Forms Runtime process spawning its own JVM instance. This can be a significant savings in memory usage and system resources.

Design-time Considerations

This section describes some of the design-time considerations.

The following sections are included:

- Re-importing Your Java Code
- About Sharing Static Variables Across Multiple JVMs

Re-importing Your Java Code

If you used the Java Importer feature of Oracle Forms prior to the availability of JVM Pooling, you will need to reimport your Java classes before using JVM pooling. When you originally imported your Java classes, PL/SQL wrappers for the Java classes were generated, which you can see in the Program Units that were created in your Form. However, the PL/SQL wrappers that are generated by the Java Importer to utilize JVM pooling are different.

From Oracle Forms Services 10*g* and later, the Java Importer generates the new PL/SQL wrappers. If you want to use the Java Importer, but do not wish to take advantage of JVM pooling, the in-process JVM will work with the new PL/SQL wrappers. It will also continue to work with the older-style PL/SQL wrappers.

About Sharing Static Variables Across Multiple JVMs

One advantage of JVM pooling is the ability to share data between instances of a class by using static variables. However, static variables will be shared between instances of the same class within a JVM, but not across JVMs. You will need to plan accordingly.

For example, suppose your loan class has a static variable called `interestRate` because all instances use the same interest rate in calculations. If you are using only one JVM, and one of the instances of your loan class changes `interestRate`, all of the other instances will be affected (which is what you want).

However, if the JVM controller has one or more child JVMs, there may be at least two JVMs. If interestRate changes in one JVM, the loan instances in the other JVMs won't see this new value, see Child JVM Processes. Prior to JVM pooling, if you changed interestRate it would not affect any other instances because each Oracle Forms Runtime process had its own in-process JVM.

If you rely on static variables to share information between instances of your class, ensure that no child JVM is spawned by setting `maxsessions` to `65535`.

Configuring JVM using Fusion Middleware Control

You have to perform specific steps to configure JVM using Fusion Middleware Control.

Perform the following steps:

1. Using Fusion Middleware Control, add a new configuration section or modify an existing section in `formsweb.cfg` to enable or disable use of JVM controller for applications.
2. Ensure CLASSPATH is updated in `default.env` or in `jvmcontrollers.cfg`.
3. Using Fusion Middleware Control, configure the JVM parameters.
4. Start the JVM controller.

For information about:

- JVM pooling parameters that are used in the Forms configuration file, see Forms Configuration File Settings.
- JVM parameters, see Managing Parameters.
- JVM controller, see Starting and Stopping JVM Controllers with Fusion Middleware Control.

Network Proxies and Java Calls Using JVM Controller

When JVM pooling is enabled and the JVM Controller runs Java, it may be necessary to set the '`jvmoptions`' parameter in jvmcontrollers.cfg file. This parameter users can use to set the java properties related to network proxies. For applications calling Oracle Reports, Oracle BI-Publisher, or using Imported Java, and these calls require access through network proxy; use the appropriate parameters to configure the environment as appropriate for proxy in use

- http.proxyHost
- http.proxyPort
- https.proxyHost
- https.proxyPort
- http.nonProxyHosts

For information about these properties, see

`https://docs.oracle.com/javase/8/docs/api/java/net/doc-files/net-properties.html` in Java documentation.

Manage JVM Controllers from the Command Line

If you manage JVM controllers from the command line, you must know the options to start and stop them, as well as specify the environment.

> **Caution:**
>
> Starting the JVM Controller on the command line is strongly discouraged. Controlling and configuring the JVM Controller from within Fusion Middleware Control is recommended. See Managing JVM Pooling from Fusion Middleware Control.
>
> The recommended way to start the JVM Controller is to allow the application to initiate the startup. It will start when an associated application is run for the first time.
>
> To start the controller automatically, add `otherparams=jvmcontroller=`*`<YOUR CONTROLLER NAME>`* to the URL used to start the application. This will start the controller if it is not already running.
>
> If the controller is already running, this setting simply instructs the controller to add this session to the existing pool.

You can only access the JVM controllers on the same computer from which they are running.

The mechanics for controlling the JVM controller as described in this chapter are mostly relevant at the command line. It is easier to use Fusion Middleware Control with its user-friendly screens and online help. Fusion Middleware Control users are still urged to read through the following information, however, to understand what the different fields and options mean, and how the JVM controller works.

JVM Controller Command Examples

This section provides some sample JVM controller commands. For a detailed explanation on the example, see Startup Example.

You must set the environment variables in Table 32 before attempting to start the controller form the command line:

Table 32 Required Environment Variables

Environment Variable	Value
`ORACLE_HOME`	The fully-qualified path that points to the Oracle Home containing the Oracle Forms installation. For example: `C:\oracle\middleware\Oracle_Home`
`FORMS_INSTANCE`	The fully-qualified path that points to the Forms Instance containing the configurations associated with the JVM Controller to be used. For example: `C:\oracle\middleware\user_projects\domains\base_domain\config\fmwconfig\components\FORMS\instances\forms1`
`FORMS_INSTANCE_NAME`	The name of the Forms instance that contains the configurations associated with the JVM Controller to be used. For example: `forms1`
`DOMAIN_HOME`	The fully-qualified path that points to the Domain containing the configurations associated with the JVM Controller to be used. For example: `C:\oracle\middleware\user_projects\domains\base_domain`

Here are some sample JVM controller commands:

Table 33 Sample JVM Controller Commands

JVM Controller Command	Description
`dejvm -start jvmcontroller=hrJVM`	Starts a JVM controller with ID `hrJVM`. The controller name, `hrJVM`, is defined as a named section in the `jvmcontrollers.cfg` configuration file. Therefore, JVM options and classpath parameters are taken from the configuration file. `maxsessions` is `50` as defined in the Default section, and other parameters take their default values.

Table 33 (Cont.) Sample JVM Controller Commands

JVM Controller Command	Description
`dejvm -start jvmcontroller=myJVM`	Starts a JVM controller with ID `myJVM`. Since no option was specified, and there is no named section in the `jvmcontrollers.cfg` file, the JVM options parameter is `"-Xms512m -Xmx1024m"` and `maxsessions=50` as set in the Default section. The other parameters take on their default values. For instance, the `CLASSPATH` value is the system `CLASSPATH`.
`dejvm -start jvmcontroller=hrJVM jvmoptions="-Xms128m -Xmx256m" maxsessions=75`	Sets the classpath to `/myJava/hrClasses` as defined in the named section. JVM options are `"-Xms128m -Xmx256m"` because the command line overrides the `jvmcontrollers.cfg` file. Similarly, `maxsessions` is `75`. All other parameters take on their default values.
`dejvm -start jvmcontroller=myJVM maxsessions=100 classpath=/myJava/myClasses;/moreJava/moreClasses`	The controller has `jvmoptions="-Xms512m -Xmx1024m"` as defined in the default section of the `jvmcontrollers.cfg` file. `maxsessions` is `100` which overrides the default section, and the classpath is `/myJava/myClasses;/moreJava/moreClasses`. All other parameters take on their default values.
`dejvm -stop jvmcontroller=hrJVM`	Stops the `hrJVM` controller. It must already be started for you to issue this command successfully.

Command Restrictions

Keep these command restrictions in mind:

- The commands are case sensitive.
- You can only issue one command at a time to a JVM controller.
- You can only issue a command to one JVM controller at a time.

The available commands for the JVM controller (or the dejvm process) are specified in Table 34. If you are using Enterprise Manager, there are screens that have an interface for issuing these commands. If you are using the command line, you may not be able to manage the JVM controller using the Enterprise Manager.

Start Command Parameters

The following table describes the JVM parameters used to start the JVM from the command line.

Table 34 JVM Parameters

Parameter	Description
jvmcontroller	Enter a name for this JVM. This name must contain a legal Oracle identifier that starts with a letter and contains an alphanumeric character, '_', '$' or '#' . An Oracle identifier has a maximum length of 30 bytes. Hint: You may want to enter a name based on the application that will be accessing it. You cannot change the name of this JVM controller later.
maxsessions	Specifies the maximum number of concurrent Oracle Forms sessions this JVM will serve before a new JVM is spawned. This value will override any set for the default JVM controller.
classpath	When you specify a classpath, it will override the system classpath or any classpath specified in your environment or any classpath set for the default JVM controller.
jvmoptions	Enter any valid options to pass to the JVM. This value will override any set for the default JVM controller. Refer to the Oracle Java documentation for a list of valid JVM startup options.
logdir	Leave Log Directory blank to use the log location for the default JVM controller. If any other directory is set, the log file may not be accessible through Enterprise Manager.
logging	• off - Logging not use • info - Reports general information about JVM Controller activity. (default) • warn - Reports potentially harmful conditions that may require further investigation. • error - Reports errors that have occurred. The application may continue running, but functionality may be reduced. • crit - Reports critical failures that resulted in aborting the JVM Controller. • debug - Reports verbose debugging information

Managing JVM Pooling from Fusion Middleware Control

Fusion Middleware Control provides a Web-based environment to manage all available JVM pooling options. It also lists all JVM controllers in your environment and allows you to (remotely) manage them.

For example, you can start and stop JVM controllers; add new ones; or reconfigure existing ones. In addition, Fusion Middleware Control also provides metric information such as resources (memory and CPU) that are consumed by JVM controllers, number of Forms connected, total JVMs, and so on.

While the Forms runtime process interacts directly with a JVM, the JVM controller manages the JVM, such as starting and stopping a JVM, or getting the state of one, and so on. For example, when an administrator stops the JVM controller, the JVM controller ensures that all child JVMs are terminated. You use Fusion Middleware Control to manage the JVM controller.

The JVM controller can be started in three ways:

- From Fusion Middleware Control
- When a Forms application that is bound to an existing JVM controller requests that the controller start up

- From the command line

Fusion Middleware Control reads the JVM controller configuration file. The configuration contains name-value pairs, has a default section, and has named sections. The parameters contained in `jvmcontrollers.cfg` correspond to the start parameters of the JVM controller.

> **Note:**
>
> You cannot change the location or name of the JVM controllers configuration file.

When you start a JVM controller, it takes its settings from the configuration file. You may specify none, some, or all options in this file, both in the default section and in named sections.

Use the JVM Configuration and JVM Controller pages in Fusion Middleware Control to manage JVM pooling tasks:

- Common Tasks in the JVM Configuration Page
- Managing JVM Configuration Sections
- Managing Parameters
- JVM Configuration Parameters and Default Values
- Starting and Stopping JVM Controllers with Fusion Middleware Control
- Forms Configuration File Settings
- Startup Example

Common Tasks in the JVM Configuration Page

This section describes the common tasks that you can do to edit configuration with the sections of a JVM configuration file and their parameters.

The following table describes the tasks you can do with the configuration sections within a JVM configuration file:

Table 35 Tasks for Working with Configuration Sections

Task	Description	Comment
Create Like	Creates a copy of a configuration section.	Use to create a configuration section based on the parameters of an existing configuration section.
Edit	Opens the Edit Description dialog.	Allows editing the text description of a configuration section.
Delete	Opens the Confirmation dialog when deleting a configuration section.	Irrevocably deletes a configuration section and its contents when you press Delete in the Confirmation dialog.
Create	Opens the Create Section dialog.	Creates a new configuration section. You must supply a required name and an optional description for it.

The following table describes the tasks that you can do to modify the parameters within a named configuration section:

Table 36   Tasks for Working with Parameters in a Named Configuration Section

Task	Description	Comment
Revert	Allows you to revert back to the previous version of the configuration section.	Does not allow you to revert individual changes in a configuration section.
Apply	Applies and activates all changes made to parameters in a configuration section.	Once applied, you cannot revert changes to individual parameters.
Add	Opens the Add Parameter dialog.	Add a parameter to a configuration section based on a mandatory name and an optional value and description.
Delete	Deletes a parameter.	Use Apply to save changes or Revert to discard them. Once applied, you cannot revert changes to individual parameters.

Managing JVM Configuration Sections

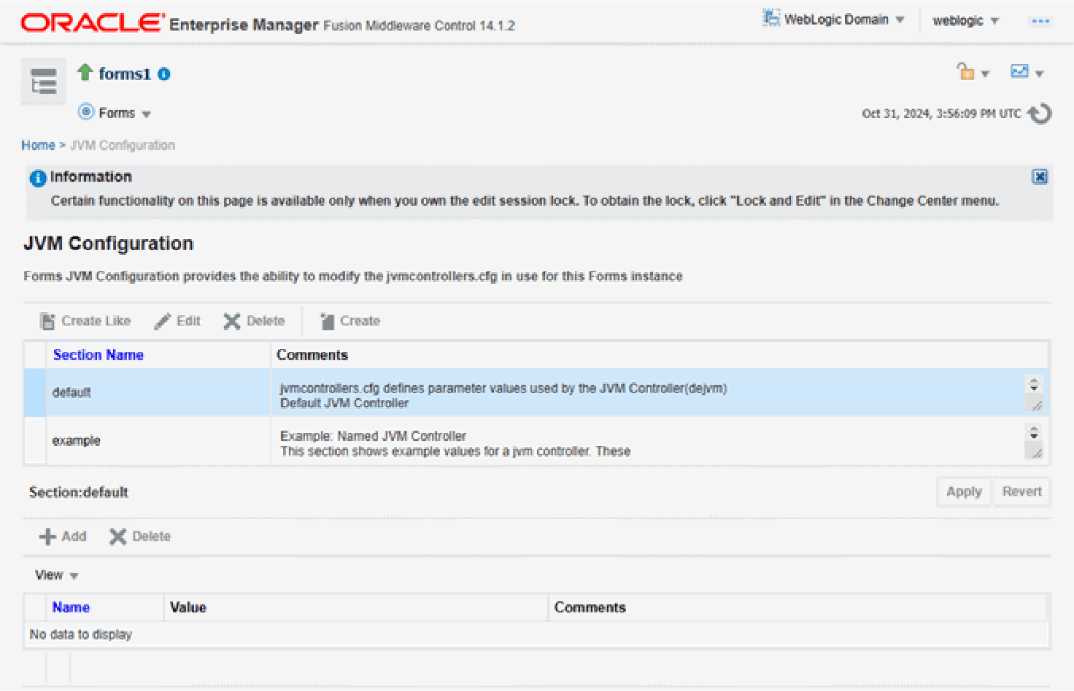
This section describes creating, editing, duplicating, and deleting named JVM configuration sections.

Accessing the JVM Configuration Page

To access the JVM configuration page:

1. Start the Enterprise Manager Fusion Middleware Control.
2. From the Fusion Middleware Control main page, click the link to the Forms Services instance that you want to configure.
3. From the Forms menu list, select the **JVM Configuration** menu item. The **JVM Configuration** page is displayed.

Figure 41   JVM Configuration Page



Information

Certain functionality on this page is available only when you own the edit session lock. To obtain the lock, click "Lock and Edit" in the Change Center menu.

JVM Configuration

Forms JVM Configuration provides the ability to modify the jvmcontrollers.cfg in use for this Forms instance

Section Name	Comments
default	jvmcontrollers.cfg defines parameter values used by the JVM Controller(dejvm) Default JVM Controller
example	Example: Named JVM Controller This section shows example values for a jvm controller. These

Section:default

Name	Value	Comments
No data to display		

Creating a New Configuration Section

You can create new configuration sections in `jvmcontrollers.cfg` from the **JVM Configuration** page of Fusion Middleware Control. These configurations can be requested in the end-user's query string of the URL that is used to run a form.

To create a new configuration section:

1. From the Fusion Middleware Control main page, click the link to the Forms Services instance that you want to configure.
2. From the Forms menu list, select **JVM Configuration**.
3. Click **Create**.

   The **Create** dialog appears.
4. Enter a name and description for your new configuration section and click **Create**.

   The new configuration section is added.

Editing a Named Configuration Description

You can edit the description (comments) for a named configuration from the **JVM Configuration** page.

To edit a named configuration description:

1. In the **JVM Configuration** region, select the row containing the named configuration for which you want to edit the description.
2. Click **Edit**.
3. The **Edit Description** dialog appears.
4. Enter the description in the Comments field.

5. Click **Save**.

   The **Edit Description** dialog box is dismissed, and your changes are saved and displayed.

Duplicating a Named Configuration

You can make a copy of a named configuration for backup purposes, or create new configuration sections from existing configuration sections.

To duplicate a named configuration:

1. In the **JVM Configuration** region, select **Create Like**.
2. In the Create Like dialog, from the **Section to Duplicate** menu, select the name of an existing configuration section you want to duplicate.
3. In the **New Section Name** field, enter a name for the new configuration section. The name for the new configuration section must be unique.
4. Click **Create**.

   A new section with exactly the same parameters, parameter values and comments of the section you are duplicating is created.

Deleting a Named Configuration

When you delete a named configuration section, you delete *all* the information within it. If you only want to delete specific parameters, see Managing Parameters.

To delete a named configuration:

1. From the **JVM Configuration** region, select the row of the configuration section you want to delete.
2. Click **Delete**.

   The **Confirmation** dialog appears.
3. Click **Delete**.

   The configuration section is deleted.

   Oracle Enterprise Manager returns to the **JVM Configuration** page and displays the remaining configurations.

> **Note:**
>
> You cannot delete the Default configuration section.

Managing Parameters

Use Fusion Middleware Control to manage parameters within a named configuration. You can add, edit, or delete parameters using Fusion Middleware Control.

To edit a parameter in a configuration section:

1. From the **JVM Configuration** region, select the row of the configuration section that contains the parameter(s) you want to edit.
2. Select the row of the parameter you want to edit. Enter the Value and Comments.

3. Click **Apply** to save the changes or **Revert** to discard them.

To add a parameter to a configuration section:

1. In Fusion Middleware Control, from the **JVM Configuration** region, select the configuration section row for which you want to add a parameter.
2. Click Add to add a new parameter.

   The Add dialog box is displayed.
3. Enter the Name, Value and Comments for the parameter.
4. Click **Create** to add the parameter.
5. Click **Apply** to save the changes or **Revert** to discard them.

To delete a parameter in a configuration section:

1. In Fusion Middleware Control, from the **JVM Configuration** region, select the configuration section from which you want to delete a parameter.
2. Select the row that contains the parameter you want to delete.
3. Click **Delete**.
4. Click **Apply** to save the changes or **Revert** to discard them.

JVM Configuration Parameters and Default Values

The following table describes the JVM configuration parameters and their default values.

Table 37 List of JVM Configuration Parameters

Parameters	Description	Default Value
maxsessions	Specifies the maximum number of concurrent Oracle Forms sessions the default JVM will serve before a new JVM is spawned.	65535
classpath	When you specify a `classpath`, it will override the system classpath or any classpath specified in your environment.	`$ORACLE_HOME/oracle_common/jdk/bin`
jvmoptions	Enter any valid options to pass to the JVM. Refer to the Oracle Java documentation for a list of valid JVM startup parameters.	Null
logdir	Leave Log Directory blank to use the log location for the default JVM controller. If any other directory is set, the log file cannot be viewed through Enterprise Manager.	`$DOMAIN_HOME/system_components/FORMS/forms1/tools/jvm/log`
logging	Specifies whether `logging` is enabled or not. Valid values: off, debug, warn, error, critical, info.	info

Table 37 (Cont.) List of JVM Configuration Parameters

Parameters	Description	Default Value
autoremoval	When enabled, `autoremoval` will allow the JVM Controller to monitor and manage child JVM and determine whether they are needed. As child JVM processes become unneeded the `autoremoval` feature will cleanly terminate those JVMs.	Off
loadbalance	When enabled, `loadbalance` will allow the JVM Controller to monitor the status of each child JVM process. Based on the `loadbalance` setting selected, the JVM Controller will determine where to send requests for processing.	Random

Starting and Stopping JVM Controllers with Fusion Middleware Control

Fusion Middleware Control is the recommended tool for managing Oracle Forms Services, such as starting, stopping, and restarting a JVM controller.

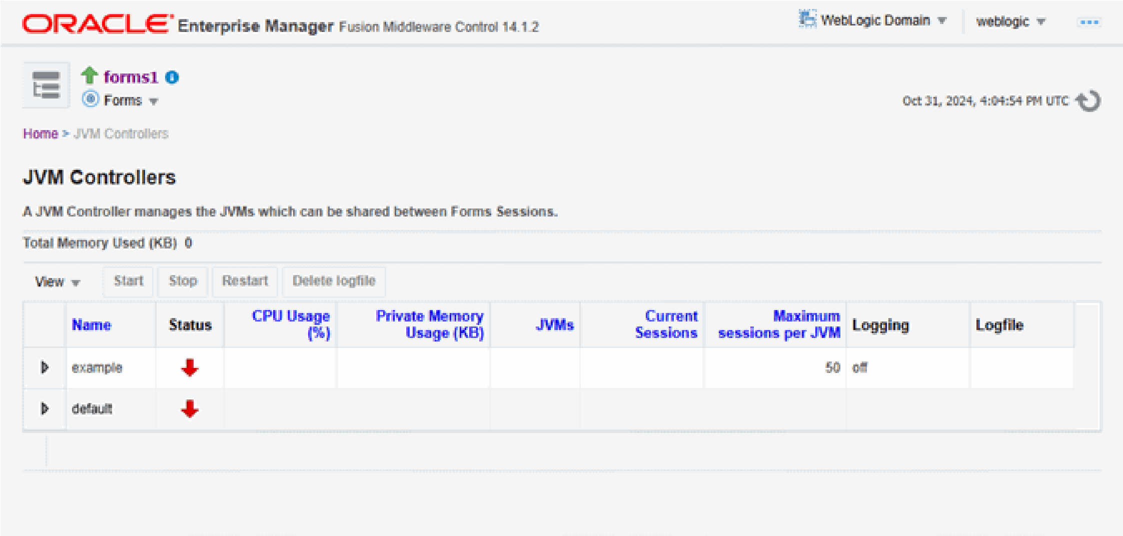
If a JVM controller is down, you can start it. If a JVM controller is already running, you can restart it without first having to manually stop it. Fusion Middleware Control does this step for you.

> **Note:**
>
> Ensure that users have stopped the forms sessions that are using the JVM controller before you stop or restart the JVM. Users may want to restart sessions when the JVM is restarted.

To access the JVM Controller page:

1. Start the Enterprise Manager Fusion Middleware Control.
2. From the Forms home page, select **JVM Controllers**. The **JVM Controllers** page is displayed.

Figure 42   JVM Controller page



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Oct 31, 2024, 4:04:54 PM UTC
Home > JVM Controllers

JVM Controllers

A JVM Controller manages the JVMs which can be shared between Forms Sessions.

Total Memory Used (KB) 0

Name	Status	CPU Usage (%)	Private Memory Usage (KB)	JVMs	Current Sessions	Maximum sessions per JVM	Logging	Logfile
example	↓					50	off	
default	↓							

To start a JVM controller that is not running:

1. From the Forms menu, select **JVM Controllers**.

   The **JVM Controllers** page is displayed.

2. Select the JVM controller that you want to start. A JVM that is not running is indicated by a red, down arrow.

3. Click **Start**.

   When the JVM controller has started, a green, up arrow is displayed in the **Status** column.

To restart a running JVM controller:

1. From the Forms menu, select **JVM Controllers**.

   The **JVM Controllers** page is displayed.

2. Select the JVM controller to be restarted.

3. Click **Restart**.

4. Click **Yes** on the Confirmation dialog.

   The **JVM Controller page** reappears.

   When the JVM controller has restarted, a green, up arrow is displayed in the Status.

To stop a JVM Controller:

1. From the Forms menu, select **JVM Controllers**.

   The **JVM Controllers** page is displayed.

2. Select the running JVM controller that you want to stop, indicated by a green, up arrow.

3. Click **Stop**.

4. Click **Yes** on the Confirmation dialog.

   When the JVM controller has been stopped, a red, down arrow is displayed in the **Status** column.

To view additional details of a JVM Controller:

1. From the Forms menu, select **JVM Controllers**.

   The **JVM Controllers** page is displayed.

2. Click the plus symbol next to the JVM controller. The row is expanded to display additional details of the JVM controller.

Forms Configuration File Settings

This section describes the JVM pooling parameters that are used in the Forms configuration file (`formsweb.cfg`) to enable or disable use of JVM controller for applications. The parameter names are not case-sensitive. You can use Fusion Middleware Control to administer the Forms configuration file.

For information about modifying the parameters in `formsweb.cfg`, see Managing Parameters.

The following table describes the startup options that you specify in the `formsweb.cfg` file.

Table 38   Oracle Forms JVM Controller Startup Parameters

Parameter	Description
`jvmcontroller`	Valid values: name of `jvmcontroller`. In addition, you can specify no JVM by leaving it blank. Default value: none **Note**: To specify this parameter in `formsweb.cfg`, you must first specify this parameter in `otherparams` in the form `jvmcontroller=%jvmcontroller%`. See otherparams in Web Configuration Parameters. This parameter can be set globally in the default section, or any application section can choose to override it. This tells the Forms runtime process which JVM controller to use. It corresponds to the `jvmcontroller` parameter for the dejvm executable. If `jvmcontroller` does not have a value (`jvmcontroller=`), then the Forms runtime process will start its own in-process JVM, which means that the Java Importer uses pre-10*g* behavior.
`allowJVMControllerAutoStart`	Valid values: `true`, `false` Default value: true This parameter enables Oracle Forms to run the JVM controller if Forms is configured to use the JVM controller which is not already running.

Startup Example

This example shows an environment of multiple JVMs for multiple applications.

As shown in following table, `formsweb.cfg` is configured with four configuration sections.

Table 39   Multiple JVMs for Multiple Applications

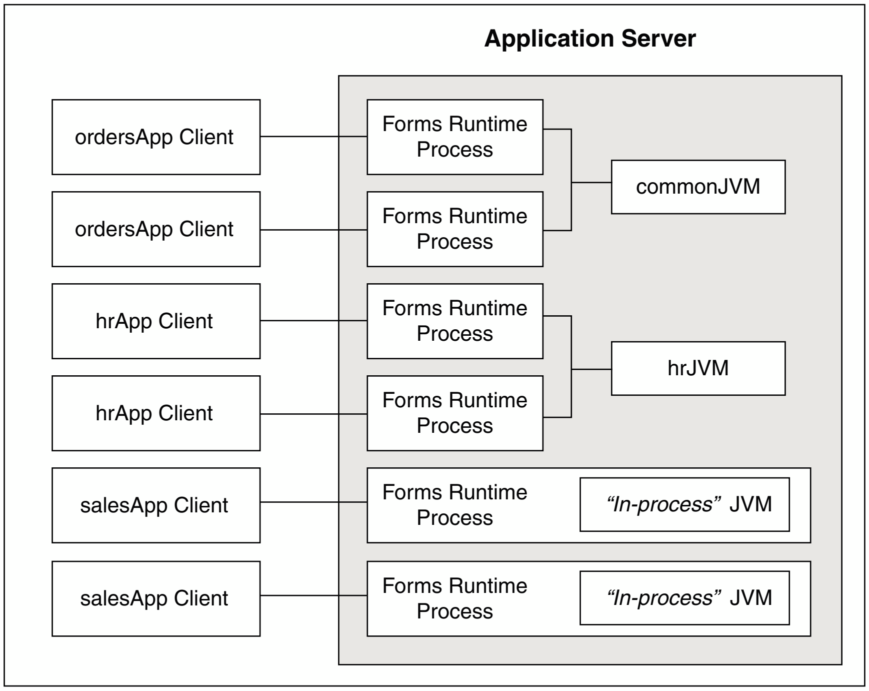
Named Configuration Section	JVM Configuration
`default`	`jvmcontroller=commonJVM`
ordersApp	None
hrApp	`jvmcontroller=hrJVM`
salesApp	`jvmcontroller=`

If a user starts an `ordersApp` application, and the application executes Java code, the Forms runtime process will route the request to the JVM controller named `commonJVM`. Because the `[ordersApp]` application section does not specify which JVM controller to use, the Forms runtime process uses the global one. If the JVM controller is not started, it will be dynamically started. If a second user starts the same application, it too will attach to `commonJVM`.

When a user starts an `hrApp` application and it executes Java code, the Forms runtime process sends the request to the JVM controller named `hrJVM` because the `[hrApp]` application section overrides the global setting. If the JVM controller is not started, it will be dynamically started. When a second user starts the same application, it too will attach to `hrJVM`.

When a user starts a `salesApp` application and it executes Java code, the Forms runtime process starts an in-process JVM in the same way the Java Importer works without JVM pooling. When a second user starts the same application, the application will get their own in-process JVM, thus consuming more memory, as shown in following figure:

Figure 43 Multiple JVMs for multiple applications



JVM Controller Logging

When logging is enabled, the JVM controller logs specific information to the log file.

The information are logged:

- The values of the JVM parameters (maxsessions, classpath, and so on);
- When a JVM controller starts and stops;
- When a child JVM is spawned;
- When an Forms runtime process starts a new connection, along with its process ID

  This is useful for knowing which Forms runtime processes are connected to which JVM controller for diagnostics or administration;

- When an Forms runtime process session ends and disconnects from the JVM.

The following section are included:

- Specifying JVM Default Logging Properties
- Specifying the JVM Log Directory Location
- Accessing Log Files
- Deleting a Log File for a JVM Controller

Specifying JVM Default Logging Properties

Use Fusion Middleware Control to manage the properties for JVM controller logging.

1. In the **JVM Configuration** page, select the the JVM configuration section.
2. For the *logging* parameter, enter a valid logging value, as described in Table 34.
3. Click **Apply**.

Specifying the JVM Log Directory Location

You can specify the log file directory in the JVM controller. You can also specify the default JVM controller log file location for other JVM controllers to use.

To specify the log file directory location:

1. Create a JVM controller.
2. Add the **Log Directory** parameter.

   If you have duplicated a named configuration section that has **Log Directory** parameter defined in it, you can edit the existing parameter.
3. Click **Apply** to save the changes.

The **JVM Configuration** page reappears.
For information about:

- JVM controller, see Creating a New Configuration Section or Duplicating a Named Configuration.
- Managing parameter, see Managing Parameters.

Accessing Log Files

When the log file exists, an icon is displayed in the Logfile column.

To access a log file:

- Click the Log File link in the Logfile column that is available for that JVM controller.

  The Log File page appears and displays the log information.

Deleting a Log File for a JVM Controller

Use Fusion Middleware Control to delete log files.

To delete a log file for a JVM controller:

1. From the **JVM Controllers** page, select the target JVM.

2. Click **Delete Logfile**.

   The Delete Confirmation dialog appears.

3. Click **Delete**.

   The logfile is deleted and the **JVM Controllers** page reappears.

> **Note:**
>
> If you delete a log file of a JVM that is running, the log file will be available again when the JVM is restarted. Logging is possible only when the JVM is restarted.

Integrating Oracle Forms with REST Services

Using or calling REST from Oracle Forms allows the application developer to access data from sources other than an Oracle database. To get started using REST with Forms, you will need to first understand the REST API(s) you plan to use.

In modern applications, data is often gathered from various resources in order to present a singular representation of that data. For Forms, the ability to access data through a REST service call offers the ability to obtain lightweight data quickly and easily. It also provides a way to access data from third-party services with minimal effort.

When registering with a REST service, the REST service typically asks for two pieces of information:

- The type of client or application. Oracle Forms runs as a confidential client in a web application.
- The redirect URI. This is the listener servlet URL such as `https://example.com:9443/forms/lservlet`. The port number—in this example, `9443`—should specify a port that accesses WebLogic directly (without going through Oracle HTTP Server) even when REST services will be accessed in SSO mode.

Once this information is provided, the REST service provides REST server URL(s) and various authorization tokens that may be required when accessing REST service operations. The authorization tokens, and potentially REST server URLs, need to be stored as credentials in Oracle Platform Security Services [OPSS].

Specifying REST Authorization Credentials in OPSS

For some REST services, authorization is required in order to make requests against it. For an Oracle Forms application, such authorization credentials are stored in OPSS with an associated "key" name. This key name is referenced from the Forms application as a "Credential ID".

When the Forms application runs and attempts to make a REST call, it uses the Credential ID provided by the application developer. If this Credential ID (or key) is found in OPSS, its details are included with the call. If credentials are required by the service but are not provided, the call fails. Details of this failure are found in the ODL (Oracle Diagnostic Logging) for Forms, but a failure may also be presented to the user as a typical Forms error alert, unless the application code is able to catch or handle this failure condition.

In some cases, the REST service may provide multiple variants of an authorization token or REST server URL—such as one for a production version of the REST service and one for a beta version of the REST service with improved or additional functionality. The OPSS credentials may be constructed so as to enable an end user to select a variant by specifying a specific value for the `config` parameter in the URL that invokes the Forms application that accesses the REST service. When `config=default` is specified—or the `config` parameter is omitted—the variant which is designated as the default variant is selected. The default variant is also selected if there is no OPSS credential corresponding to the `config` parameter specified in the URL.

The key for the OPSS credential for a default variant must consist of alphanumeric characters and the special characters: underscore [_], dot [`.`], and hyphen [`-`]. OPSS credential keys are case-insensitive. The key should be chosen jointly by the administrator and the Forms application developer whose application needs to access the REST service. The Forms application developer may suggest a key. The sysadmin must ensure that it is not already in use.

The key for the OPSS credential for a non-default variant is formed by appending the equals sign [`=`] and the desired config section name to the OPSS credential for the default variant. The config section name must also consist of alphanumeric characters and the special characters: underscore [_], dot [`.`], and hyphen [`-`].

Storing Credentials for REST Authorization

An authorization token contains a private part and an optional public part.

When there is no public part of the credential, the private part (the entire authorization token) may be stored either as the password component of an OPSS password credential (the user name component is ignored) or as an OPSS generic credential.

When there is a public part, the authorization token may be stored as an OPSS password credential with the public part as the user name component and the private part as the password component. The token can also be stored as an OPSS generic credential consisting of the public part followed by the private part, separated by newline (`0x0A`) or carriage return/newline (`0x0D0A`).

Here are the OPSS credentials that should be created for various types of authorization:

- **Basic Authentication**: Normally, a user name and password are provided programmatically or by the end-user, in which case no OPSS credentials should be created. However, in certain cases, a fixed user name and password may be appropriate. In this case, an OPSS credential (or credentials, if there are variants) may be created. The user name is the public part and the password is the private part.
- **Bearer Token**: OPSS credential(s) should be created with the bearer token as the private part. There is no public part.
- **API Key**: OPSS credential(s) should be created with the API key as the private part. There is no public part.
- **OAuth2**: OPSS credential(s) should be created with the client ID as the public part and the client secret (or client password) as the private part. Both the client ID and the client secret/password must be URL-encoded, using the URL-encoding semantics expected by the REST service.

For non-OAuth2 types of authorization, Oracle Forms does the URL-encoding of the authorization token. Although there may be more than one way of URL-encoding a specific authorization token, the expectation is that the REST service will produce the same result when it URL-decodes the URL-encoded value produced by Oracle Forms.

For OAuth2, RFC 6749 (The OAuth 2.0 Authorization Framework) recommends using Basic Authentication for Client Authentication, and it requires that the client ID and client password be URL-encoded before they are used to construct a Basic Authorization header. Only one of the possible URL-encodings will produce an acceptable Basic Authorization header, and Oracle Forms has no way of determining which one that is. Therefore, Oracle Forms requires the administrator (or application developer) to determine the correct URL-encoding when creating the OPSS credential.

So how should the administrator determine the correct URL-encoding? First of all, alphanumeric characters never require URL-encoding, so if the client ID and client password consist entirely of alphanumeric characters, no URL-encoding is required. In other cases, the administrator may be able contact the REST service to determine the URL-encoding the REST service is expecting. If that is not possible, trial and error may be required. Construct trial URL-encodings using the following guidelines:

- Underscore [_], dot [`.`], hyphen [`-`], and tilde [`~`] almost never require URL-encoding.
- Bang [`!`], dollar sign [`$`], apostrophe [`'`], left and right parentheses [`(` and `)`], asterisk [`*`], comma [`,`], semicolon [`;`], colon [`:`], and the at sign [`@`] may or may not require URL-encoding.
- Other characters almost always require URL-encoding.

URL-encoding a character consists of producing its UTF-8 representation and then replacing each byte in this representation by its 3-byte percent-encoding (`%` followed by the upper-case hex digits for the byte), with the exception that certain REST services may expect a space to be URL-encoded as ‘`+`’ rather than ‘`%20`’.

Refer to the Forms References part of this guide for configuration settings that may be used when making REST calls from Forms. Specifically refer to Environment Variables and Web Configuration Parameters.

Creating a Key in OPSS

Create a key in OPSS using the OPSS Credential screen for the domain in Fusion Middleware Control. When you create a key, you can choose either the "password" or "generic" type.

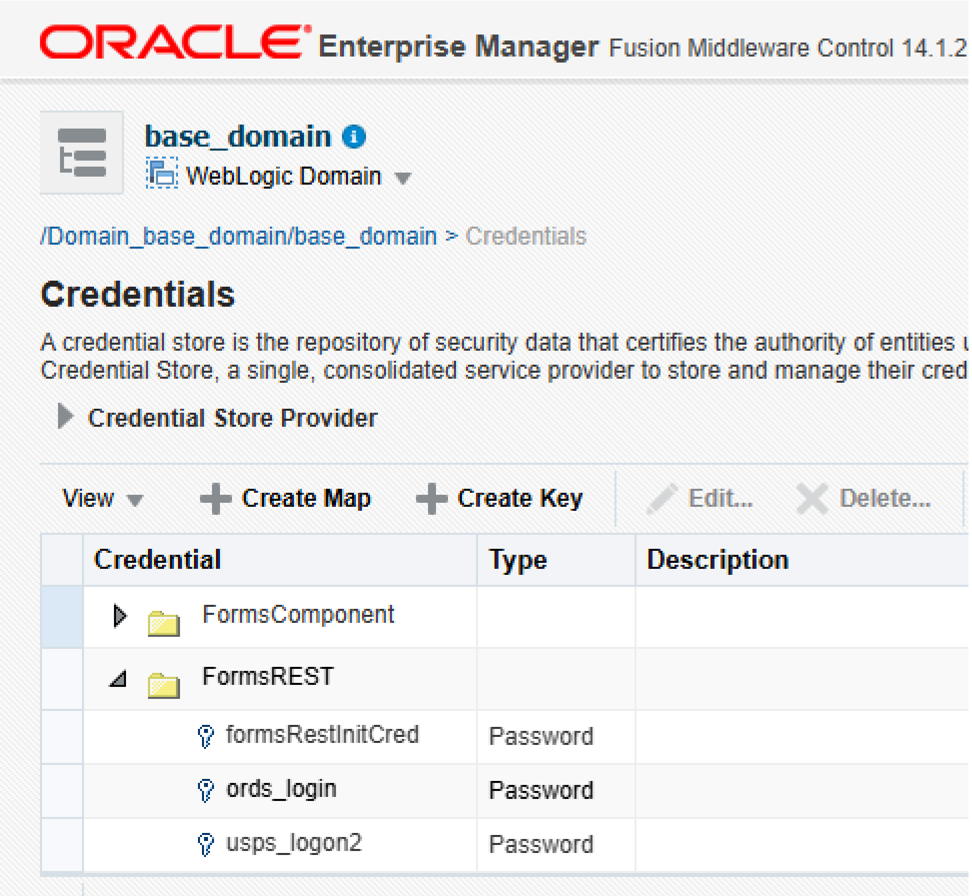
To create a key:

1. From the **WebLogic Domain** menu, click **Security**, then **Credentials**.

   This screen includes a pre-seeded mapping for "FormsREST" along with a pre-seeded credential entry. Do not delete this entry.

   Any credentials for use with Forms REST must be stored as keys under this mapping node as shown in Figure 44.

Figure 44   Credential Store Provider Page



/Domain_base_domain/base_domain > Credentials

Credentials

A credential store is the repository of security data that certifies the authority of entities u
Credential Store, a single, consolidated service provider to store and manage their cred

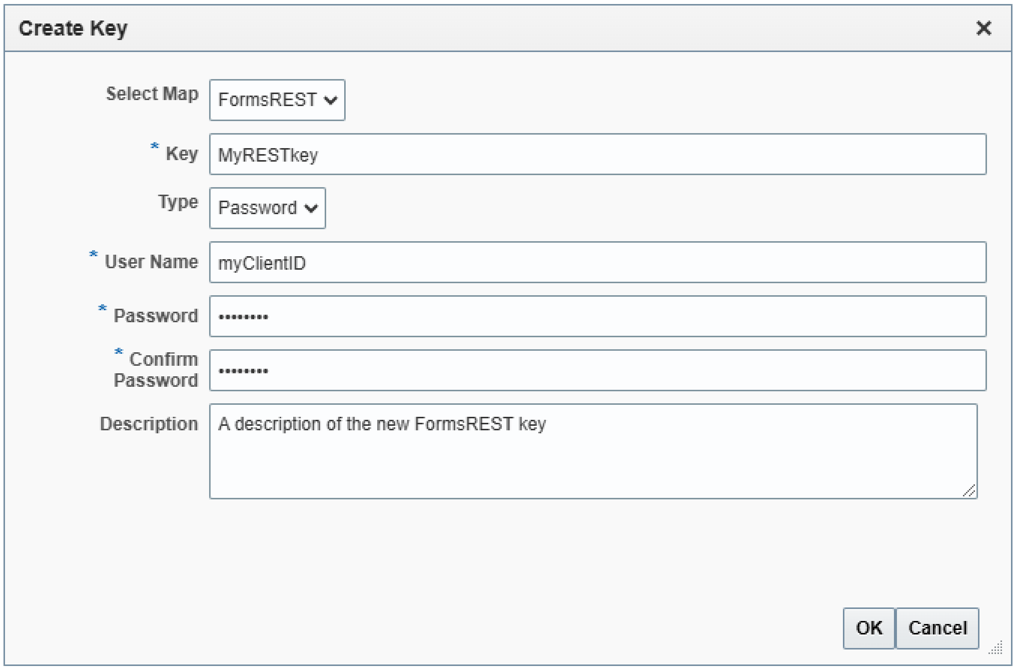
Credential Store Provider

Credential	Type	Description
FormsComponent		
FormsREST		
formsRestInitCred	Password	
ords_login	Password	
usps_logon2	Password	

2. Click **Create Key** to open the **Create Key** dialog.

   Figure 45 shows the dialog with the fields available for a password-type key. This configuration includes **User Name**, **Password**, and **Confirm Password** fields.

Figure 45 Create Key dialog for a Password-type Key



Create Key

Select Map: FormsREST

* Key: MyRESTkey

Type: Password

* User Name: myClientID

* Password: ••••••••

* Confirm Password: ••••••••

Description: A description of the new FormsREST key

3. In the **Key** field, type the Forms application Credential ID.

   The key value is used in your Forms application to indicate which credentials are needed for a particular REST call.

4. Select either **Password** or **Generic** for Forms REST entries from the **Type** dropdown list.

   The "Password" type is preferred as it is more secure.

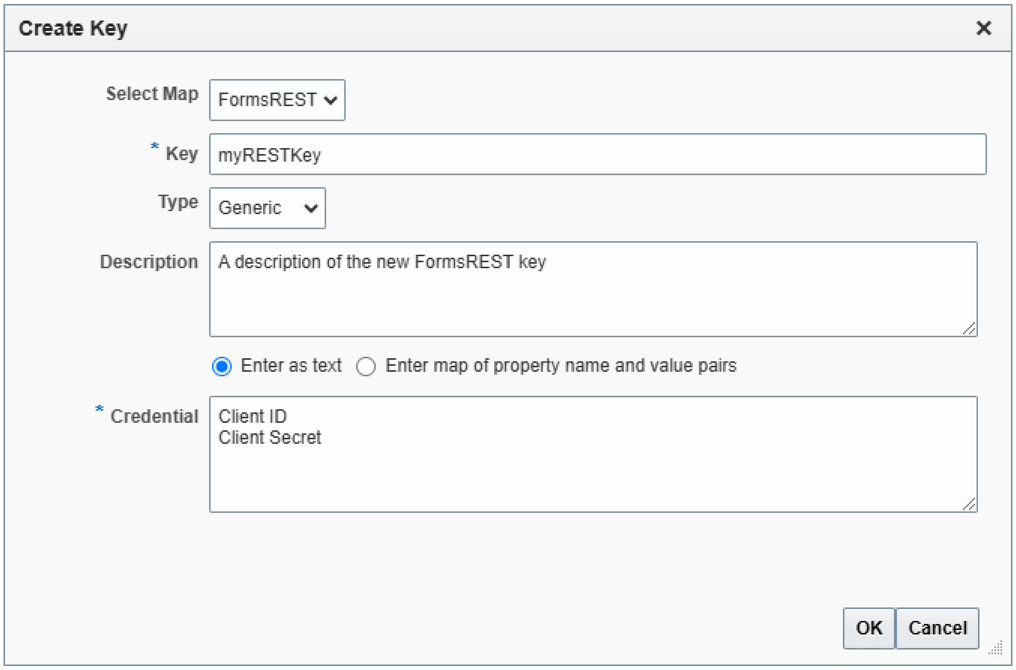
5. If using the "Password" type, enter the public part of the authorization token in the **User Name** field and the private part of the authorization token in the **Password** field.

   If there is no public part, leave the **User Name** field blank.

6. If using the "Generic" type, enter the public and private parts of the authorization token on separate lines of the **Credential** field as shown in Figure 46.

   If there is no public part, enter the private part on a single line in the **Credential** field.

Figure 46   Create Key dialog for a Generic-type Key



Create Key

Select Map: FormsREST

* Key: myRESTKey

Type: Generic

Description: A description of the new FormsREST key

● Enter as text ○ Enter map of property name and value pairs

* Credential: Client ID
Client Secret

7. Type a description for the key in the **Description** field.
8. Click **OK**.

Part VI

Security

The ability to control user access to web content and to protect your site against people breaking into your system is critical. This part describes the architecture and configuration of security for Oracle Forms Services.

This part contains the following chapters:

For information on how to enable SSL/TLS on WebLogic Server or Oracle HTTP Server, refer to Configuring SSL in Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

Single Sign-On

Single Sign-on in Oracle Forms Services is available through `webgate`, Oracle modules for the Oracle HTTP Server. The `webgate` access client authenticates a user against Oracle Access Manager (OAM).

Review Using Forms Services with Oracle Access Manager for more informtaion.

Forms applications expect a database connect string to be passed along with the application request, otherwise a logon dialog is shown. To retrieve the database connect information in the Single Sign-On environment, the Forms servlet queries Oracle Internet Directory for the value of the combined unique key that is constructed from the user's single sign-on server name, the authenticated user name, and the name of the application that the user is requesting to start.

Resource Access Descriptors (RAD) are entries in Oracle Internet Directory that are defined for each user and application which contain the required database connect information. The Forms servlet reads the database connect information from the RAD and passes it along with the command line that starts the Forms Web application. Although the Forms authentication is still database-centric, or `webgate` and the Forms servlet are now integrated in a Web-based authentication server environment.

For more information, see:

- Introduction to Oracle Platform Security Services
- Getting Started With Oracle Internet Directory
- Understanding the Concepts and Architecture of Oracle Internet Directory

Classes of Users and Their Privileges

Historically, Forms applications use the database to authenticate application users. To use Oracle Forms Services with Single Sign-On (SSO), the user account and its connect information must be available in Oracle Internet Directory. Oracle Internet Directory provides several ways of provisioning user data, using PL/SQL, Java or the Oracle Delegated

Administration Services. Oracle Delegated Administration Services is a Web-based user interface for Oracle Single Sign-On users and delegated administrators to administer self-service data in Oracle Internet Directory for which they are authorized.

Once a user account is created in Oracle Internet Directory, the Resource Access Descriptors (RAD) entries can be created dynamically the first time that a user requests a Forms application, assuming the user knows about the database connect information required for this application.

Another option is to use the RAD entries that can be created using Oracle Delegated Administration Services. The default RAD entries are accessible for all users that are authenticated through Oracle Single Sign-On. Use the default RAD if all users share the same database connect information when running a particular Forms application on the Web. This way, users are authenticated individually by their Oracle Single Sign-On credentials; however, all users share a common database connect (information) for the application defined by a default RAD entry.

Default Single Sign-On Behavior for User Accounts

By default, the authentication server is enabled and no proxy user is involved. Oracle Forms users need to authenticate with an authentication server, retrieve Resource Access Descriptors from the identity store (which is usually Oracle Internet Directory) and use these credentials to connect to the database.

Users Using Database Proxy Functionality

There is a new Single Sign-On parameter, `ssoProxyConnect`. Setting this to `true` allows users to connect as proxy users. The user is then required to authenticate with an authentication server; a Resource Access Descriptor is configured which holds the proxy user's username and password. There is additional database configuration that needs to be implemented by the database administrator to allow for proxy connections.

Resources that are Protected

When you enable Single Sign-On for your Forms applications, you can secure your Forms applications with these features:

- **Dynamic Resource Creation in Oracle Internet Directory**: In some previous releases of Oracle Forms Services, if no resource access descriptor (RAD) definition was found for a specific application and user, an error message was displayed which locked out the user from running that Forms application, despite having authentication to do so. In this release of Oracle Forms Services, you can now configure Oracle Forms Services to allow users to create the RAD for this application on the fly if it does not exist. The functionality to redirect to the appropriate creation pages is achieved with the single sign-on parameter *ssoDynamicResourceCreate*.
- **Database Password Expiration when Using Single Sign-On**: In some previous releases of Oracle Forms Services, the RAD information in Oracle Internet Directory was not updated if the database password had expired, and users then renewed them when connecting to a Forms application. In this release, Oracle Forms Services automatically updates the RAD information in Oracle Internet Directory whenever a database password is updated through Forms. There is no extra configuration necessary to enable this feature in Oracle Forms Services.

Authentication and Access Enforcement

For detailed information about the authentication flow in Single Sign-On support in Oracle Forms Services, such as when the first time the user requests an Oracle Forms Services URL, or from a partner application, see Authentication Flow.

Security Configuration

Configuring security for Oracle Forms Services is done through Oracle Fusion Middleware Control.

To configure Oracle Forms options for Oracle Fusion Middleware Security Framework, see the Administration part of this guide.

To configure Oracle Identity Management options for Oracle Forms, see Using Forms Services with Oracle Access Manager. Oracle Forms Services can be configured to create resources dynamically in Oracle Platform Security Services or Oracle Internet Directory.

To enable, configure, and manage Forms Trace and to view Forms Trace output, see Tracing and Diagnostics.

Online Help is also provided for the Oracle Fusion Middleware Control screens.

Securing RADs Stored in Oracle Internet Directory

To increase the security of RADs and prevent them from being viewable by the Oracle Internet Directory (OID) administrator, perform the following steps:

1. Copy the contents enclosed by `---aci-change.ldif---` into the file `aci-change.ldif`

```
---aci-change.ldif---
dn: cn=Extended Properties,%s_OracleContextDN%
changetype: modify
delete: orclaci
orclaci: access to attr=(orclUserIDAttribute,orclPasswordAttribute) by
guidattr=(orclOwnerGUID)(read,search,compare,write) by
dnattr=(orclresourceviewers) (read,search, compare, write) by
groupattr=(orclresourceviewers) (read,search, write) by * (none)
-
add: orclaci
orclaci: access to attr=(orclUserIDAttribute,orclPasswordAttribute)
DenyGroupOverride by guidattr=(orclOwnerGUID)(read,search,compare,write) by
dnattr=(orclresourceviewers) (read,search, compare, write) by
groupattr=(orclresourceviewers) (read,search, write) by * (none)
---aci-change.ldif---
```

> **Note:**
>
> In `aci-change.ldif`, the line beginning with `orclaci: access to attr=` is a single line ending with `by * (none)` and should not have any line breaks in the middle.

2. In the LDIF file, replace `%s_OracleContextDN%` with the distinguished name (DN) of the realm-specific Oracle Context.

   For example, if the DN in the deployment is `dc=acme,dc=com`, then the realm-specific Oracle Context is `cn=OracleContext,dc=acme,dc=com`.

3. Execute the following command on the OID tier:

   ```
   ldapmodify -p <port> -h <host> -D cn=orcladmin -q -v -f aci-change.ldif
   ```

4. When this command is run, it will prompt for the `cn=orcladmin` password since the password is not included as a command-line parameter.

To undo these changes, issue the same command (subject to the notes as above), but using the following contents in the `.ldif` file:

```
---aci-revert.ldif---
dn: cn=Extended Properties,%s_OracleContextDN%
changetype: modify
delete: orclaci
orclaci: access to attr=(orclUserIDAttribute,orclPasswordAttribute)
DenyGroupOverride by guidattr=(orclOwnerGUID)(read,search,compare,write) by
dnattr=(orclresourceviewers) (read,search, compare, write) by
groupattr=(orclresourceviewers) (read,search, write) by * (none)
-
add: orclaci
orclaci: access to attr=(orclUserIDAttribute,orclPasswordAttribute) by
guidattr=(orclOwnerGUID)(read,search,compare,write) by
dnattr=(orclresourceviewers) (read,search, compare, write) by
groupattr=(orclresourceviewers) (read,search, write) by * (none)
---aci-revert.ldif---
```

Managing URL Security for Applications

Oracle Forms applications are web-deployed solutions that users access using a URL. Oracle Forms architecture allows Forms developers two ways to choose and configure how a Forms application runs. One option is to set the parameter and the value in the URL. The second option is to set the parameter and its value(s) in the Web Configuration (`formsweb.cfg`).

The parameter that is set in the Web Configuration can be overridden by the parameter set in the URL. A Forms administrator can override this default behavior, and give the Forms administrator full control over what parameter can be used in the URL.

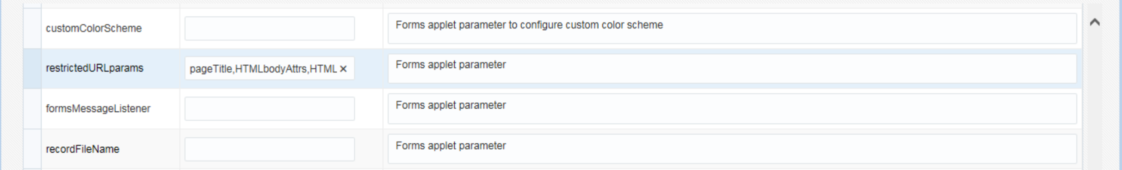
Here are two scenarios to consider when deciding which parameters to allow or not allow in a URL. The first scenario is when an administrator just wants to restrict the usage of the USERID parameter in the URL that forces the end-user to always log in using the default login window. The second scenario is when an administrator disables all parameters except a few, such as `CONFIG=MyApp` in a URL.

The parameter `restrictedURLparams` allows flexibility for the Forms administrator to consider any URL-accessible parameter in the `formsweb.cfg` file as restricted to a user. An administrator can specify this parameter in a named configuration section to override the one specified in the default configuration section. The `restrictedURLparams` parameter itself cannot be set in the URL.

By design, command line arguments passed in a URL always override similar definitions in the formsweb.cfg.

In this example, the `userid` is defined as `user1/user1pwd` and `debug` is set to `false`. An application that is configured to connect to the database as `user1/user1pwd` can connect as a different user with the `userid` parameter added as a URL parameter. To prevent it, define the `userid` parameter in the `restrictedURLparams` parameter.

Figure 47 Defining the restrictedURLparams Parameter



customColorScheme		Forms applet parameter to configure custom color scheme
restrictedURLparams	pageTitle,HTMLbodyAttrs,HTML ×	Forms applet parameter
formsMessageListener		Forms applet parameter
recordFileName		Forms applet parameter

Similarly, an administrator can use the `restrictedURLparams` parameter to redirect a user to a page which lists the restricted parameters that were used. The `restrictedURLparams` parameter can be set to '`all`' which will prevent any parameters (other than config section) from being specified in the URL.

Securing the Oracle Forms Test Form

The test form runs when you access an Oracle Forms URL but do not specify an application to run. For example, normally you call an Oracle Forms application with the following syntax:

```
https://<host>:<port>/forms/frmservlet?config=myApp
```

The Forms servlet locates `[myApp]` in the formsweb.cfg file and launches that application. However, when no application is specified, for example:

```
https://<host>:<port>/forms/frmservlet
```

The Forms servlet uses the settings in the default section of the formsweb.cfg file. These settings are located under `[default]` in the Forms Configuration file (anytime an application does not override any of these settings, the defaults are used). The default section has the following setting:

```
form=test.fmx
```

This is the test form which enables you to test your Oracle Forms Services installation and configuration. Thus if you do not specify an application, Forms launches the test.fmx file. You could change this to:

```
form=
```

And the form does not run. However, this is not optimal; the Forms servlet still sends the dynamically generated HTML file to the client, from which a curious user could obtain information. The optimally secure solution is to redirect requests to an informational HTML page that is presented to the client instead. Some parameters in the `formsweb.cfg` file must be changed.

Here are the parameters to change, along with their default values when you install Oracle Forms Services:

```
# System parameter: default base HTML file
baseHTML=base.htm
# System parameter: base HTML file for use with Oracle's Java Plug-In
baseHTMLjpi=basejpi.htm
```

These parameters are templates for the HTML information that are sent to the client. Create an informational HTML page and have these variables point to that instead. For example, in the `$DOMAIN_HOME/config/fmwconfig/components/FORMS/instances/<Forms Instance Name>/server` directory, create a simple HTML page called `forbidden.html` with the following content:

```
<html>
  <head>
    <title>Forbidden</title>
  </head>
  <body>
   <h1>Forbidden!</h1>
    <h2>You may not access this Forms application.</h2>
  </body>
</html>
```

> **Note:**
>
> This message page was displayed because redirecting of client information is different from the page that the Web server returns when the requested content has restricted permissions on it.

Next, modify the `formsweb.cfg` parameters by commenting out or modifying the original parameters:

```
# System parameter: default base HTML file
#baseHTML=base.htm
baseHTML=forbidden.html
# System parameter: base HTML file for use with Oracle's Java Plug-In
#baseHTMLjpi=basejpi.htm
baseHTMLjpi=forbidden.html
# System parameter: base HTML file for use with Microsoft Internet Explorer
#  (when using the native JVM)
```

When a user enters the URL

```
https://<host>:<port>/forms/frmservlet
```

the customized Web page is presented. Of course, you can customize `forbidden.html`, including its contents, its file name, and its location if you make the corresponding changes to these parameters in the `formsweb.cfg` file. Administrators can put any information, such as warnings, errors, time stamps, IP logging, or contact information in this information Web page with minimal impact on the server configuration.

Note:

Overriding the base HTML template entries in the default section of `formsweb.cfg` requires that you add the same entries pointing to the original values (or some other valid HTML file) in your application-specific named configuration:

```
[myApp]
form=myApplication.fmx
lookandfeel=oracle
baseHTML=base.htm
baseHTMLjpi=basejpi.htm
```

If you do not specify these base HTML values, and when a user runs an application, the `forbidden.html` page is displayed because the application-specific configuration section has not overridden the default values.

Part VII

Troubleshooting, Diagnostics, and Performance Tuning

This part covers how to troubleshoot Oracle Forms services and details Forms services tracing and diagnostics, and performance tuning considerations.

Specifically, this part contains the following chapter:

Tracing and Diagnostics

Oracle Forms Trace allows you to record information about a precisely defined part of Forms functionality or a class of user actions. This chapter provides information about enabling, configuring, managing Forms Trace and viewing Forms Trace output.
The following sections are included:

Forms Trace

Forms Trace allows you to record information about a precisely defined part of forms functionality or a class of user actions. This is accomplished by defining events for which you want to collect trace information.

For example, you can record information about trigger execution, mouse-clicks, or both. From the Enterprise Manager Fusion Middleware Control, you can use trace output to diagnose performance and other problems with Oracle Forms applications.

Forms Trace replaces the functionality that was provided with Forms Runtime Diagnostics (FRD) and Performance Event Collection Services (PECS), which were available in earlier releases of Oracle Forms. Forms Trace allows you to trace the execution path through a form, for example, the steps the user took while using the form.

Difference between Tracing and Debugging

You use Forms debugging to find out what happens when a user presses a button. Debugging allows a remote developer to connect to an existing Forms user session and to trace the user actions as the application runs or to debug on a local machine. Forms Trace provides information about the timing of specific events. Oracle Support uses tracing to isolate and analyze issues. For example, you use Forms trace to find out which query takes the longest time to execute, or which trigger causes performance issues with Oracle Forms.

Enable and Configure Forms Trace

An *event* is something that happens inside a Forms application as a direct or indirect result of a user action. An *event set* specifies a group of events that you can trace simply by specifying the event set name rather than each event number individually when you start the trace.

An example is when a user presses a button that executes a query. Use the **Trace Configuration** selection in the **Forms** menu of Fusion Middleware Control page to define the events that you want to trace. This page manages all changes in the `ftrace.cfg` file for you.

Note the following items when working with Forms Trace:

- If you first switch off trace, and then switch it on again with new settings, then trace is enabled with the new trace group.
- Backup the `ftrace.cfg` and `default.env` files before editing them with Fusion Middleware Control.
- As with most Web applications, it is easy to lose unsaved changes by switching pages. Be sure to save any changes you make through Fusion Middleware Control to Forms configuration, trace, or environment files before proceeding to other pages.

  The length of time it takes for changes to be saved is affected by the number of lines you have changed. For example, an additional fifty lines of comments will take longer to save than just deleting a single entry.

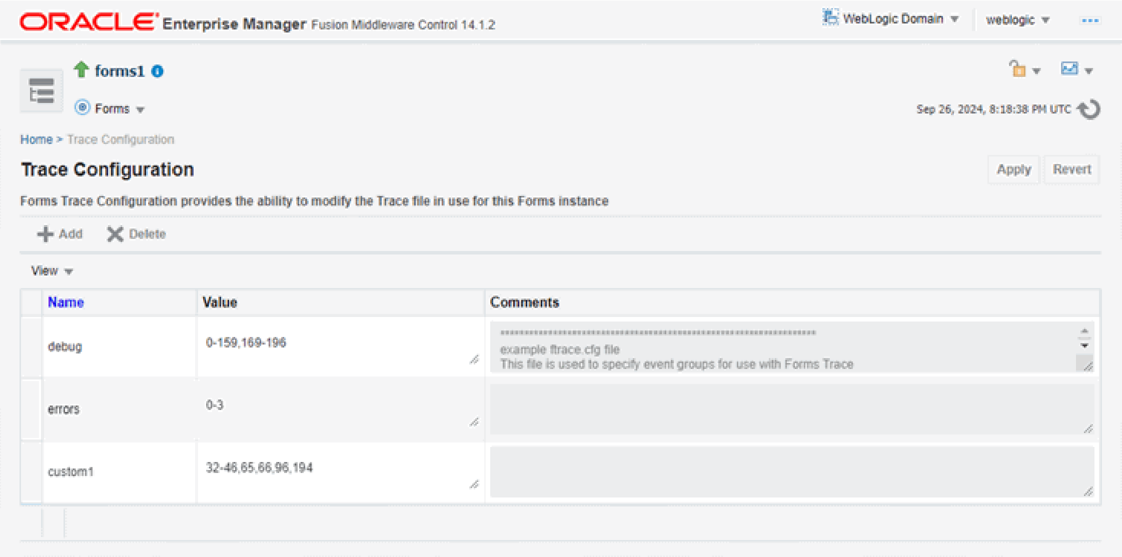
For a list of events and their corresponding event numbers, see List of Traceable Events.

Configuring Forms Trace

To access the Trace Configuration page:

1. Start Fusion Middleware Control.
2. From the Fusion Middleware Control main page, click the Oracle Forms Services instance link that you want to configure.
3. From the Forms menu list, select **Trace Configuration**. The **Trace Configuration** page is displayed.

Figure 48   Trace Configuration Page


ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Sep 26, 2024, 8:18:38 PM UTC
Home > Trace Configuration
Apply
Revert
Add
Delete
View

Trace Configuration

Forms Trace Configuration provides the ability to modify the Trace file in use for this Forms instance

Name	Value	Comments
debug	0-159,169-196	** example ftrace.cfg file This file is used to specify event groups for use with Forms Trace
errors	0-3	
custom1	32-46,65,66,96,194	

To create a new trace group:

1. From the Fusion Middleware Control main page, click the link to the Oracle Forms Services instance that you want to configure.
2. From the Forms menu list, select **Trace Configuration**.

   The **Trace Configuration** page is displayed.
3. Click **Add**.

   The **Add** dialog is displayed.
4. Enter the information for the new trace group:

   Name: Enter a name for the trace group.

   Value: See Table 41 for the values of traceable events.

   Comment : Enter a comment.

   - The trace group name must not contain spaces. For example, a_b_c is an acceptable trace group name.
   - There must be a comma between each event number you specify in the Value. For example, 65,66,96,194 is an acceptable value.
   - You can use a range of numbers. For example, 32-46 is an acceptable range.
5. Click **Add**.

   The new trace group is added.
6. Click **Apply** to save the changes, or **Revert** to discard them.

To delete a trace group:

1. In the **Trace Configuration** page, select the group you want to delete.
2. Click **Delete**.

   The trace group is deleted and the **Trace Configuration** page reappears.
3. Click **Apply** to save the changes, or **Revert** to discard them.

To edit an existing trace group:

1. In the **Trace Configuration** page, select the group you want to edit.
2. Enter the value and description for the trace group.
3. Click **Apply** to save the changes, or **Revert** to discard them.

Specify URL Parameter Options

The following command line parameters are used to configure Forms Trace:

```
Record =
Tracegroup =
Log = <filename>
```

The following table describes the parameter values:

Table 40 Forms Trace Command Line Parameters

Parameter	Values	Description
`Record`	forms	Enables Forms Trace.

`Tracegroup`	Name, event number, or event range	Indicates which events should be recorded and logged.
		• If Tracegroup is not specified, only error and Startup messages are collected.
		• Tracegroup is ignored if Forms Trace is not switched on at the command line.
		• You can create a named set of events using the Tracegroup keyword, for example
		Tracegroup=<keyword>, where <keyword> is specified in ftrace.cfg (for example, Tracegroup=MyEvents).
		This lets you log the events in the named set MyEvents.
		• You can log all events in a specified range using the Tracegroup keyword, for example
		`Tracegroup = 0-3`
		This lets you log all events in the range defined by 0 <= event <=3.
		• You can log individual events using the Tracegroup keyword, for example
		`Tracegroup = 34,67`
		• You can combine event sets using the Tracegroup keyword, for example
		`Tracegroup = 0-3,34,67,SQLInfo`

Starting and Stopping Forms Trace

You start a trace by specifying trace entries in the URL or from Fusion Middleware Control. Entries should include the grouping of events to collect and the trace file name. Trace collection starts when the form executes.

The following are sample URLs to start a trace:

```
http://example.com/forms/frmservlet?form=cxl&record=forms&tracegroup=0-199
http://example.com/forms/frmservlet?form=cxl&record=forms&tracegroup=mysql
```

To start tracing a session from Fusion Middleware Control:

1. From the Forms menu, select **User Sessions**.

   The **User Sessions** page appears.
2. Select the row containing the Forms user session for which you want to enable tracing.
3. Click **Enable Tracing**.

   The Enable Tracing dialog appears.
4. From the Select Trace Group list, select an available trace group and click **OK**.

   The Enable Tracing dialog is dismissed and tracing is now enabled for the selected Forms user session.

To stop tracing a session from Fusion Middleware Control:

1. From the Forms menu, select **User Sessions**.

   The User Sessions page appears.
2. Select the row containing the Forms user session for which you want to disable tracing.
3. Click **Disable Tracing**.

   The Disable Tracing dialog is displayed.
4. Click **OK**.

   The Disable Tracing dialog is dismissed and tracing is now stopped for the selected Forms user session.

To switch between trace groups for a session:

1. Select the row containing the Forms user session for which you want to change the trace group.
2. Click **Enable Tracing**.

   The Enable Tracing dialog is displayed.
3. From the Select Trace Group list, select the new trace group and click **OK**.

   The Enable Tracing dialog is dismissed. Refresh the page.

Viewing Forms Trace Output

Only administrators or a user belonging to administrators' group can view trace log files. Once the user has logged in, he or she does not have to log in again in the same browser session to view trace log files for different sessions.

Trace data is stored in a binary file with a `*.trc` extension. The default location of the trace log is `$DOMAIN_HOME/system_components/FORMS/<FORMS INSTANCE NAME>/trace/forms_<PID>.trc` where *pid* is the process ID of the user session. If you are not using Enterprise Manager Fusion Middleware Control, you need to use the Translate utility, as described in Running the Translate Utility.

To view trace data:

1. From the Forms menu in Fusion Middleware Control, select the **User Sessions** menu item.
2. Select a User Session row and click **Trace Log** to see the contents of the trace log.
3. Log in to view the trace file.

Running the Translate Utility

The Translate utility converts trace data to XML, HTML, or text formats. You need to specify an additional parameter "OutputClass" which has three legal values: "WriteOutTEXT", "WriteOutXML" and "WriteOutHTML". If you do not specify the `outputclass`, the output file is in text format. These values are case-sensitive.

> **Note:**
>
> To use the Translate Utility:
>
> 1. Set the PATH variable to include the path to the directory containing the Java executable.
> 2. Set the CLASSPATH variable to include the path to frmxlate.jar.

To convert trace data to Text format:

- At the command line, enter:

  ```
  java oracle.forms.diagnostics.Xlate datafile=a.trc outputfile=myfile.txt
  outputclass=WriteOutTEXT
  ```

  This creates a file called myfile.txt in text format.

To convert trace data to HTML format:

- At the command line, enter:

  ```
  java oracle.forms.diagnostics.Xlate datafile=a.trc outputfile=myfile.html
  outputclass=WriteOutHTML
  ```

  This creates a file called myfile.html in HTML format.

To convert trace data to XML format:

- To create `myfile.xml`, at the command line, enter:

```
java oracle.forms.diagnostics.Xlate datafile=a.trc outputfile=myfile.xml
outputclass=WriteOutXML
```

This creates a file called myfile.xml in XML format.

List of Traceable Events

The table provided in this section, lists the events that can be defined for tracing. In future releases of Forms, more events may be added to this list.

Event types are as follows:

- Point event: An event that happens in Oracle Forms as the result of a user action or internal signal for which there is no discernible duration, for example, displaying an error message on the status line. Each instance of this event type creates one entry in the log file.
- Duration event: An event with a start and end, for example, a trigger. Each instance of this event type creates a pair of entries in the log file (a start and end event).
- Built-in event: An event associated with a built-in. Each instance of this event type provides a greater quantity of information about the event (for example, argument values).

Table 41 List of Traceable Events

Event Number	Definition	Type
0	Abnormal Error	point
1	Error during open form	point
2	Forms Died Error	point
3	Error messages on the status bar	point
4-31	Reserved for future use	NA
32	Startup	point
33	Menu	point
34	Key	point
35	Click	point
36	Double-click	point
37	Value	point
38	Scroll	point
39	LOV Selection	point
40	not used	not used
41	Window Close	point
42	Window Activate	point
43	Window Deactivate	point
44	Window Resize	point
45	Tab Page	point
46	Timer	point
47	DB Event	point
48	Reserved for future use	NA

Table 41   (Cont.) List of Traceable Events

Event Number	Definition	Type
49-63	Reserved for future use	NA
64	Form (Start & End)	duration
65	Program Unit (Start & End)	duration
66	Trigger (Start & End)	duration
67	LOV (Start & End)	duration
68	Opening a Editor	point
69	Canvas	point
70	Alert	duration
71	GetFile	point
72-95	Reserved for future use	NA
96	Builtin (Start & End)	builtin
97	User Exit (Start & End)	duration
98	SQL (Start & End)	duration
99	MenuCreate (Start & End)	duration
100	DB PU (Start & End)	duration
101	Execute Query	duration
102-127	Reserved for future use	NA
128	Client Connect	point
129	Client Handshake	point
130	Heartbeat	point
131	HTTP Reconnect	point
132	Socket (Start & End)	duration
133	HTTP (Start & End)	duration
134	SSL (Start & End)	duration
135	DB Processing (Start & End)	duration
136	DB Logon (Start & End)	duration
137	DB Logoff (Start & End)	duration
138-159	Reserved for future use	NA
160-168	Reserved for internal use	NA
169-191	Reserved for future use	NA
192*	Environment Dump	N/A
193*	State Delta	N/A
194*	Builtin Arguments	N/A
195*	UserExit Arguments	N/A
196*	Program Unit Arguments	N/A
256 and higher	User defined	NA

Table 41   (Cont.) List of Traceable Events

Event Number	Definition	Type
1024 and higher	Reserved for internal use	NA

> **Note:**
>
> These event numbers do not have a `TYPE` because they are not really events, but rather details for events. For example, the State Delta is something you can choose to see - it is triggered by a real action or event.

List of Event Details

The following tables list event details that can be defined for tracing.

> **NOT_SUPPORTED:**
>
> Event names are case sensitive.

Table 42   User Action Event Details

Action	Details	Number
Menu Selection	Menu Name, Selection	33
Key	Key Pressed, Form, Block, Item	34
Click	Mouse/Key, Form, Block, Item	35
DoubleClick	Form, Block, Item	36
Value	Form, Block, Item	37
Scroll	Form, Up, Down, Page, Row	38
LOV Selection	LOV Name, Selection Item	39
Alert	AlertName, Selection	40
Tab	Form	45
DB Event	Queue Name	47
Window Activate, Deactivate,Close, Resize	WindowName, FormName, Size	41,42,43,44

Table 43   Forms Services Event Details

Event Name	Details	Number
Form	Form ID, Name, Path, Attached Libraries, Attached Menus	64
Program Unit	Program Unit Name, FormID	65

Table 43 (Cont.) Forms Services Event Details

Event Name	Details	Number
Trigger	TriggerName, FormName, BlockName, ItemName, FormID	66
LOV	LOV name, FormId	67
Editor	FormId, Editor Name	68
Canvas	FormId, Canvas Name	69

Table 44 Detailed Events

Event Name	Details	Number
Builtin	BuiltinName, FormId	96
User Exit	UserExitName, FormId	97
MenuCreate	MenuName, FormID	99
PLSQL	PLSQLSTmt, FormID	100
ExecQuery	Block Name	101

Table 45 Three-Tier Event Details

Event Name	Details	Number
Client Connect	Timestamp	128
Client Handshake	Timestamp	129
Heartbeat	Timestamp	130
HTTP Reconnect	NA	131
Socket	FormId, Packets, Bytes	132
HTTP	FormId, Packets, Bytes	133
HTTPS	FormId, Packets, Bytes	134
DB Processing	FormId, Statement	135
DB Logon	FormId	136
DB Logoff	FormId	137

Table 46 Miscellaneous Event Details

Event Name	Details	Number
Environment Dump	Selected environment information	192
State Delta	Changes to internal state caused by last action/event	193
Builtin Args	Argument values to a builtin	194
Userexit args	Arguments passed to a userexit	195
Procedure Args	Arguments (in|out) passed to a procedure.	196

Taking Advantage of Oracle Diagnostics and Logging Tools

Oracle Diagnostics and Logging (ODL) is a feature of Oracle Fusion Middleware that enables administrators to keep a record of all Oracle Forms sessions, monitor Oracle Forms-related network traffic, and debug site configuration problems.

Some features of Oracle Diagnostics and Logging available to Forms Services include:

- Recording of all Oracle Forms sessions, including session start and end times, and the user's IP address and host name (session-level logging)
- Monitoring of Oracle Forms-related network traffic and performance (session-performance and request-performance-level logging)
- Generating debugging information for site configuration issues (debug-level logging)
- Logging handled through Fusion Middleware Control
- Correlating events in these log files with events in the database
- Automatic handling of log file rotation.
- Handling of log size restriction by the mechanism rather than by OS level scripts as was done previously

The following sections are included:

- Enabling Oracle Diagnostics and Logging
- Viewing Diagnostics Logs
- Using the Servlet Page
- Location of Log Files
- Example Output for Each Level of Servlet Logging

Enabling Oracle Diagnostics and Logging

When you turn on logging, the Listener Servlet writes log messages to the servlet log file.

To view examples of output for the various levels of logging, see Example Output for Each Level of Servlet Logging.

The following table describes the supported logging capabilities. If no string is appended to serverURL, then default logging is supported. To start other loggers, they must be specified in serverURL as described in the next section.

Table 47 Supported logging capabilities

String appended to serverURL client parameter	Description of logging
(none)	During Forms servlet initialization, a message is written to the log file stating the name and path of the configuration file being used. Messages of levels higher and equal to the log level set for the default logger in logging.xml are logged. Default Value is set to NOTIFICATION:1 and levels NOTIFICATION:1, WARNING:1, ERROR:1 and INTERNAL_ERROR are logged.

Table 47 (Cont.) Supported logging capabilities

String appended to serverURL client parameter	Description of logging
/session	Log messages are written whenever a Forms session starts or ends. These give the host name and IP address of the client (the computer on which the user's Web browser is running), the runtime process id, and a unique internal session id number.
/sessionperf	Performance summary statistics are included with the session end message.
/perf	A performance message is written for every request from the client.
/debug	Full debug messages. Other debug messages are written in addition to the messages mentioned above. This logging level is verbose and is intended mainly for debugging and support purposes.

Specifying Logging

To specify logging for all users, change the serverURL entry, in the **Web Configuration** page default section, to the following entry:

```
serverURL=/forms/lservlet/<string>
```

The <string>, the above entry, specifies the logging capability, as described in Table 47. If no string is provided, the default logging, for example, if you want to start session-level logging, modify the serverURL as follows:

```
serverURL=/forms/lservlet/session
```

Specifying Logging Levels Using Fusion Middleware Control

To set the log levels for Forms servlet logging using Fusion Middleware Control, perform the following:

1. From the Fusion Middleware Control, select the managed server (for example WLS_FORMS).
2. From the WebLogic Server menu, select Logs, then Log Configuration.
3. In the Logger Name field, expand Root Logger. Expand each of the following: oracle, oracle.forms. The Logger name defined in serverURL, as described in previous section, is displayed. For example, oracle.forms.servlet.debug.
4. Choose the Log level as required from the list in the Oracle Diagnostic Logging Level field. For the mapping of the internal Forms log level to the Java levels, see the following table.

Table 48 Oracle Diagnostic Logging Levels

Internal Forms Log Levels	Java Log Levels
DEBUG	TRACE:32
REQUEST_PERFORMANCE	TRACE:16
SESSION_PERFORMANCE	TRACE:1
SESSION_START_END	NOTIFICATION:16

Table 48   (Cont.) Oracle Diagnostic Logging Levels

Internal Forms Log Levels	Java Log Levels
NOTIFICATION	NOTIFICATION:1
WARNING	WARNING:1
ERROR	ERROR:1
INTERNAL_ERROR	INTERNAL_ERROR

> **Note:**
>
> This configuration modifies the logging.xml file for the managed server.

Specifying Full Diagnostics in the URL that Invokes the Forms Servlet

To start full diagnostics, specify the parameter serverURL in `formsweb.cfg` as follows:

```
serverURL=/forms/lservlet/debug
```

Start the Oracle Forms application using a URL as follows:

```
http://example.com/forms/frmservlet/debug?
```

Viewing Diagnostics Logs

You view the contents of diagnostics logs from Fusion Middleware Control.

To view the contents of diagnostics logs:

1. From the Forms menu, select Home.

   The Fusion Middleware Control home page is displayed.
2. In the Forms Deployment region, scroll to the Servlet Logs column.
3. Click the corresponding Logs link for the target deployed application.

   The Log Messages page is displayed.

Using the Servlet Page

From the Forms menu, select Monitoring and then Servlet Logs. Use this page to search, sort, view, download, and export collected server diagnostics logs.

For information on managing and viewing the log files, see Managing Log Files and Diagnostic Data in *Administering Oracle Fusion Middleware*.

Location of Log Files

The default servlet log file is named `formsapp-diagnostic.log`. It is written to the `WLS_FORMS/logs` directory of the Oracle WebLogic Managed Server to which Forms is deployed.

In Oracle Forms Services, the full path is:

```
$DOMAIN_HOME/servers/WLS_FORMS/logs/<application name>-diagnostic.log
```

The trace logs are stored in files named `forms_pid.trc` by default, where `pid` is the process ID of the user session. The default location of the trace log is:

```
$DOMAIN_HOME/system_components/FORMS/forms1/trace/forms_pid.trc
```

Use the Translate Utility, as described in Running the Translate Utility to view them.

Example Output for Each Level of Servlet Logging

The following are examples of the type of output you get when you use the following levels of logging.

(none)

```
[2008-09-10T06:58:47.106-07:00] [WLS_FORMS] [NOTIFICATION] [FRM-93100]
 [oracle.forms.servlet] [tid: 11] [ecid: 0000HlCYKnmD4i8nvgy0V118lx4u000000,0]
 [APP: formsapp] [arg:
configFileName:     <configfilename>
testMode:           false] Initializing the Forms Servlet.  Initialization
 parameters are:[[
    configFileName:    <configfilename>
    testMode:
false
]]
[2008-09-10T06:58:53.517-07:00] [WLS_FORMS] [NOTIFICATION] [FRM-93180]
 [oracle.forms.servlet] [tid: 11] [ecid: 0000HlCZfTDD4i8nvgy0V118lx4u000003,0]
 [APP: formsapp] [arg:
envFile:            null
executable:         null
WaitTime:           500
MaxBlockTime:       1000]
Initializing ListenerServlet.  Initialization parameters
 are:[[
    envFile:            null
    executable:         null
    WaitTime:           500
    MaxBlockTime:       1000
]]
```

/session

```
[2008-09-11T07:35:01.507-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93251]
[oracle.forms.servlet.session] [tid: 14] [ecid:
0000HlHpYGDD4i8nvgy0V118mFuv00000V,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession] [APP: formsapp] [SRC_METHOD: <init>]
[FORMS
SESSION_ID: ..8] [arg: sup-pc1] [arg: xxx.xxx.xxx.xxx] Runtime session started
for client <pc1> (IP address <ip address>).
2008-09-11T07:35:01.798-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93548]
[oracle.forms.servlet.session] [tid: 14] [ecid:
0000HlHpYGDD4i8nvgy0V118mFuv00000V,0] [SRC_CLASS:
oracle.forms.servlet.RunformProcess] [APP: formsapp] [SRC_METHOD: connect]
[FORMS
SESSION_ID: ..8] [arg: 7765] Runtime process ID is 7765.
2008-09-11T07:38:11.372-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93252]
```

```
[oracle.forms.servlet.session] [tid: 14] [ecid:
0000HlHpYGDD4i8nvgy0V118mFuv00000V,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession] [APP: formsapp] [SRC_METHOD: stop] [FORMS
SESSION_ID: ..8] Forms session ended.
```

/sessionperf

```
[2008-09-11T07:40:25.923-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93251]
[oracle.forms.servlet.sessionperf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession] [APP: formsapp] [SRC_METHOD: <init>]
[FORMS
SESSION_ID: ..9] [arg: <pc1>] [arg: xxx.xxx.xxx.xxx] Runtime session started
for client <pc1> (IP address xxx.xxx.xxx.xxx).
2008-09-11T07:40:26.223-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93548]
 [oracle.forms.servlet.sessionperf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0] [SRC_CLASS:
oracle.forms.servlet.RunformProcess] [APP: formsapp] [SRC_METHOD: connect]
[FORMS
SESSION_ID: ..9] [arg: 8023] Runtime process ID is 8023.
2008-09-11T07:40:43.593-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93252]
[oracle.forms.servlet.sessionperf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession] [APP: formsapp] [SRC_METHOD: stop] [FORMS
SESSION_ID: ..9] Forms session ended.
[2008-09-11T07:40:43.594-07:00] [WLS_FORMS] [TRACE] [FRM-93710]
[oracle.forms.servlet.sessionperf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession] [APP: formsapp] [SRC_METHOD: stop] [FORMS
SESSION_ID: ..9] [arg: 1.557] [arg: 6] [arg: 0] [arg: 1.000] [arg: 0.259]
[arg:
5106] [arg: 352] Total duration of network exchanges is 1.557.[[
Total number of network exchanges is 6 (0 long ones over 1.000 sec).
Average time for one network exchange (excluding long ones) is 0.259.
Total number of bytes sent is 5106.
Total number of bytes received is 352.
]]
```

/perf

```
[2008-09-11T07:42:46.560-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93251]
[oracle.forms.servlet.perf] [tid: 14] [ecid:
0000HlHrJmWD4i8nvgy0V118mFuv00000^,0]
[SRC_CLASS: oracle.forms.servlet.RunformSession] [APP: formsapp] [SRC_METHOD:
<init>] [FORMS_SESSION_ID: ..10] [arg: <pc1>] [arg: xxx.xxx.xxx.xxx] Runtime
session started for client <pc1> (IP address <ip address>).
[2008-09-11T07:42:46.854-07:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93548]
[oracle.forms.servlet.perf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0]
[SRC_CLASS: oracle.forms.servlet.RunformProcess] [APP: formsapp] [SRC_METHOD:
connect] [FORMS_SESSION_ID: ..10] [arg: 8149] Runtime process ID is 8149.
[2008-09-11T07:42:46.865-07:00] [WLS_FORMS] [TRACE:16] [FRM-93700]
[oracle.forms.servlet.perf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0]
[SRC_CLASS: oracle.forms.servlet.ListenerServlet] [APP: formsapp] [SRC_METHOD:
```

```
doPost] [FORMS_SESSION_ID: ..10] [arg: 0.011] [arg: 8] [arg: 8] [arg: null]
Request duration is 0.011 seconds.  Request size is 8 bytes; response size is
8
bytes.
[2008-09-11T07:42:47.921-07:00] [WLS_FORMS] [TRACE:16] [FRM-93700]
[oracle.forms.servlet.perf] [tid: 17] [ecid:
0000HlHqlS9D4i8nvgy0V118mFuv00000Y,0]
[SRC_CLASS: oracle.forms.servlet.ListenerServlet] [APP: formsapp] [SRC_METHOD:
doPost] [FORMS_SESSION_ID: ..10] [arg: 0.438] [arg: 272] [arg: 5022] [arg:
null]
Request duration is 0.438 seconds.  Request size is 272 bytes; response size
is
5022 bytes.
```

/debug

```
[2009-02-11T14:39:03.016+00:00] [WLS_FORMS] [NOTIFICATION:16] [FRM-93250]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_lhDcD4i8nvgy0V119Xz350000HZ,0] [APP: formsapp#12.2.1] Forms session started.
[2009-02-11T14:39:03.017+00:00] [WLS_FORMS] [TRACE:32] [FRM-94200]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_lhDcD4i8nvgy0V119Xz350000HZ,0] [SRC_CLASS: oracle.forms.servlet.FormsServlet]
 [APP: formsapp#12.2.1] [SRC_METHOD: doRequest] [FORMS_SESSION_ID: ..43] [arg:
 GET] [arg:
cmd:                frmservlet
config:             null
requestCharset:     null
QueryString:        null
Content-Type:       null
Accept-Charset:     null
responseCharset:    null] FormsServlet receiving GET request.  Details:[[
    cmd:                frmservlet
    config:             null
    requestCharset:     null
    QueryString:        null
    Content-Type:       null
    Accept-Charset:     null
    responseCharset:    null
]]
[2009-02-11T14:39:03.017+00:00] [WLS_FORMS] [TRACE:32] [FRM-94281]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_lhDcD4i8nvgy0V119Xz350000HZ,0] [SRC_CLASS:
oracle.forms.servlet.ListenerServlet]
 [APP: formsapp#12.2.1] [SRC_METHOD: printSessionDetails]
[FORMS_SESSION_ID: ..43]
 No current servlet session ID.
[2009-02-11T14:39:03.017+00:00] [WLS_FORMS] [TRACE:32] [FRM-94170]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_lhDcD4i8nvgy0V119Xz350000HZ,0] [SRC_CLASS: oracle.forms.servlet.FormsServlet]
 [APP: formsapp#12.2.1] [SRC_METHOD: findFile] [FORMS_SESSION_ID: ..43] [arg:
 basejpi.htm] [arg: <config folder>] File basejpi.htm is missing from the
 current directory, looking in <config folder>
[2009-02-11T14:39:21.460+00:00] [WLS_FORMS] [TRACE:32] [FRM-94200]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llhoD4i8nvgy0V119Xz350000Hd,0] [SRC_CLASS: oracle.forms.servlet.FormsServlet]
 [APP: formsapp#12.2.1] [SRC_METHOD: doRequest] [FORMS_SESSION_ID: ..43] [arg:
 GET] [arg:
cmd:                startsession
config:             null
requestCharset:     null
QueryString:       ifsessid=..43&acceptLanguage=en-
us&ifcmd=startsession&iflocale=en-US
    Content-Type:       null
Accept-Charset:     null
responseCharset:    null]
FormsServlet receiving GET request.  Details:[[
    cmd:                startsession
    config:             null
    requestCharset:     null
    QueryString:        ifsessid=..43&acceptLanguage=en-
us&ifcmd=startsession&iflocale=en-US
    Content-Type:       null
    Accept-Charset:     null
    responseCharset:    null
]]


.
.
.
.

[2009-02-11T14:39:21.716+00:00] [WLS_FORMS] [TRACE:32] [FRM-94201]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.ListenerServlet]
[APP: formsapp#12.2.1] [SRC_METHOD: doGet] [FORMS_SESSION_ID: ..43] [arg: GET]
 [arg:
cmd:                getinfo
QueryString:        ifcmd=getinfo&ifhost=sup-pc1&ifip=xxx.xxx.xxx.xxx]
 ListenerServlet receiving GET request.  Details:[[
    cmd:                getinfo
    QueryString:        ifcmd=getinfo&ifhost=sup-pc1&ifip=xxx.xxx.xxx.xxx
]]

[2009-02-11T14:39:21.717+00:00] [WLS_FORMS] [TRACE:32] [FRM-94282]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.ListenerServlet]
 [APP: formsapp#12.2.1] [SRC_METHOD: printSessionDetails]
[FORMS_SESSION_ID: ..43]
 [arg:
 HyLhJSjZ85F5GWbZLDgwp1MY02FK5tC6yVDP1LylbCvgmv9y3CfK!126690176!1234363161461]
 Existing servlet session, ID =
 HyLhJSjZ85F5GWbZLDgwp1MY02FK5tC6yVDP1LylbCvgmv9y3CfK!126690176!1234363161461
[2009-02-11T14:39:21.717+00:00] [WLS_FORMS] [TRACE:32] [FRM-94286]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.ListenerServlet]
 [APP: formsapp#12.2.1] [SRC_METHOD: printSessionDetails]
[FORMS_SESSION_ID: ..43]
 Session ID is not from cookie.
[2009-02-11T14:39:21.717+00:00] [WLS_FORMS] [TRACE:32] [FRM-94430]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession]
 [APP: formsapp#12.2.1] [SRC_METHOD: <init>] [FORMS_SESSION_ID: ..43] Trying
to
 get a prestarted process.
[2009-02-11T14:39:21.717+00:00] [WLS_FORMS] [TRACE:32] [FRM-94432]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession]
 [APP: formsapp#12.2.1] [SRC_METHOD: <init>] [FORMS_SESSION_ID: ..43]
Prestarted
 process is not available.
[2009-02-11T14:39:21.718+00:00] [WLS_FORMS] [TRACE:32] [FRM-94522]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.RunformSession]
 [APP: formsapp#12.2.1] [SRC_METHOD: <init>] [FORMS_SESSION_ID: ..43] [arg:
null]
 Creating new runtime process using default executable.
[2009-02-11T14:39:21.718+00:00] [WLS_FORMS] [TRACE:32] [FRM-94532]
 [oracle.forms.servlet] [tid: [ACTIVE].ExecuteThread: '2' for queue:
 'weblogic.kernel.Default (self-tuning)'] [userId: <anonymous>] [ecid: 0000Hx
_llloD4i8nvgy0V119Xz350000Hf,0] [SRC_CLASS:
oracle.forms.servlet.RunformProcess]
 [APP: formsapp#12.2.1] [SRC_METHOD: startProcess] [FORMS_SESSION_ID: ..43]
[arg:
 frmweb webfile=HTTP-0,default] RunformProcess.startProcess():  executing
frmweb
 webfile=HTTP-0,default

.
.
.
.
```

Performance Tuning Considerations

This chapter provides information about built-in optimization features for Oracle Forms services, improving performance by tuning applications, and techniques that reduce the resources required to execute an application.
The following sections are included:

- Built-in Optimization Features of Forms Services
- Oracle Forms Services Applications Tuning

> **Note:**
>
> Tuning the connection between Oracle Forms Services and the Oracle Database Server is beyond the scope of this chapter.

Built-in Optimization Features of Forms Services

Several optimizations are included in Oracle Forms Services and Java client.

The optimizations fits broadly into the following categories:

- Monitor Forms Services
- Forms Services Web Runtime Pooling
- Minimizing Client Resource Requirements
- Minimizing Forms Services Resource Requirements
- Minimizing Network Usage
- Maximizing the Efficiency of Packets Sent Over the Network
- Rendering Application Displays Efficiently on the Client

Monitor Forms Services

Use Fusion Middleware Control to monitor Oracle Forms and review metrics information, including:

- Forms Services Instances
- Events
- User Sessions
- Forms Trace

Monitoring Forms Services Instances

Use the Forms Home page to monitor metrics for a Forms Services instance.

1. Start Enterprise Manager Fusion Middleware Control.
2. From the Enterprise Manager Fusion Middleware Control main page, select the link to the Forms Services instance that you want to monitor.

The Forms Home page for the Forms Services instance displays the following:

- Status of Forms application instance (up, down, unknown)
- URL of the Forms Services instance being monitored
- Number of Forms sessions

Additionally, you can navigate to the following detail pages:

- Performance Summary
- Servlet Logs
- Session Details
- Web Configuration
- Environment Configuration
- Trace Configuration
- User Sessions
- JVM Configuration
- JVM Controllers

  In the Performance Summary page, you can add charts for other Forms metrics to the page dynamically by using the Show Metric Palette. You can also overlay metrics to compare them. For example, drag and drop Private Memory consumed by two JVM Controllers into one chart to compare them, see Monitoring in *Tuning Performance*.

Monitoring Forms Events

Use the Enterprise Manager Fusion Middleware Control to enable tracing for all events or specific ones. The follows table provides the list of tasks you can perform on this page.

Table 49 Tasks for Monitoring Forms Events

Task	See
Monitoring metrics for user sessions	To view Forms user sessions:
Sorting metrics information	To sort the list of Forms user sessions:
Searching for metrics information	To search for a Forms user sessions:

Forms Services Web Runtime Pooling

Forms Runtime Pooling (or Forms Runtime prestart) enables the startup of a configurable number of application runtime engines prior to their usage. Runtime Pooling provides quick connections at server peak times, which shortens the server-side application startup time. Runtime pooling is useful for situations where server configurations have a small window in which many users connect to a Forms application. All prestarted runtime engines run in the same environment serving the same application.

Configuring Prestart Parameters

Use Enterprise Manager Fusion Middleware Control to configure runtime pooling for Forms Services with the following parameters, as described in the following table.

Table 50   Forms Runtime Pooling Parameters

Parameter Name	Data type	Description	Default Value
`prestartRuntimes`	boolean	Runtime pre starting or pooling is enabled only if true	false
`prestartInit`	integer	Number of the runtime processes that should be spawned initially	1
`prestartTimeout`	integer	Time in minutes after which all the prestarted processes of this pool (configuration section) will be stopped. A runtime process is removed from the prestart pool once client connection is made and thus will not be stopped.	0 (When set to zero the timer never starts)
`prestartMin`	integer	Minimum number of runtime processes to exist in the pool.	0
`prestartIncrement`	integer	The number of runtime processes to be created when the number of prestarted runtime processes is less than minRuntimes.	0

> **Note:**
>
> See that `prestartMin` defines the minimum number of pre-started runtimes that must exist at any time while runtime pooling is still active for a specific application. The minimum value must be less than or equal to what's defined for the `prestartInit` parameter. The `prestartMin` parameter can be modified at any time and does not require the application server to be restarted. The new entries will be picked up when a client requests a connection to a pre-started runtime process and the prestarted runtime processes have not timed out. Once they have timed out, an application uses default behavior and a minimum threshold is not maintained.

Each configuration section can specify values for these parameter. If the `prestartRuntimes = true` entry is found, but there is no associating prestart parameter, then default values are used.

In a load balanced system that has multiple instances of Oracle WebLogic Managed Server, the various values provided for the above parameters are on a per JVM basis, and not the total for the application.

Starting Runtime Pooling

An Administrator can configure specific application(s), from the Enterprise Manager Fusion Middleware Control, to enable Runtime Pooling. On the startup of the application server (Oracle WebLogic Managed Server), the configured number of Forms Runtime processes are pre-started for each application.

In the initialization phase of the Forms servlet, the configuration file (`formsweb.cfg`) is read and the server pre-starts the applications which have the `prestartRuntimes` parameter enabled.

Scheduling Runtime Pooling

Scheduling Runtime Pooling (or Scheduling Runtime Prestart) is a feature that enables you to schedule the prestart of Forms Runtime engines. In addition to managing the startup of a configurable number of Forms Runtime engines prior to their usage, Oracle Forms now allows you to schedule the prestarting of Forms Runtime processes on a more flexible basis, at any appropriate time. You can schedule a Forms Runtime prestart, view existing schedules, delete any existing schedule, and export and import a schedule using the Enterprise Manager Fusion Middleware Control.

Setting the prestartRuntimes Parameter

The parameter, `prestartRuntimes`, must be set to `TRUE` for the configuration section that will use pre-started processes. If this parameter is not correctly set, attempts to create a schedule will fail silently, even though the message, `Prestart job has been created`, is displayed.

`prestartRuntimes` must be set in the Forms Web Configuration. It is recommended not setting the parameter in the [default] section. Doing so will result in processes being started for all other configurations unless overridden with a value of `FALSE` for configuration sections where this is not desired.

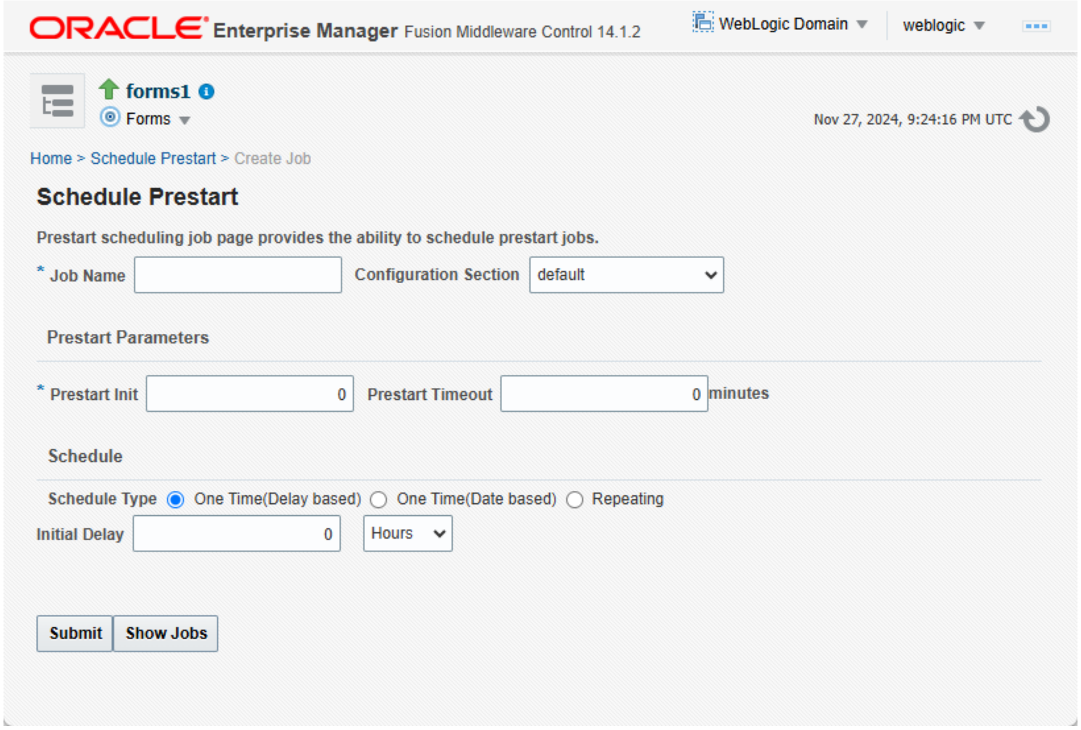
Creating a Prestart Schedule

To create a prestart schedule, perform the following steps:

1. From the Forms Menu, select **Schedule Prestart**.

   The **Schedule Prestart** page is displayed.
2. From the **Scheduled Jobs** region, click **Create**. The **Create Job** page is displayed for scheduling a prestart.

Figure 49   Scheduling Forms Runtime Prestart



ORACLE Enterprise Manager Fusion Middleware Control 14.1.2

WebLogic Domain ▼   weblogic ▼

forms1

Forms ▼

Nov 27, 2024, 9:24:16 PM UTC

Home > Schedule Prestart > Create Job

Schedule Prestart

Prestart scheduling job page provides the ability to schedule prestart jobs.

* Job Name:

Configuration Section: default

Prestart Parameters

* Prestart Init: 0

Prestart Timeout: 0 minutes

Schedule

Schedule Type: ● One Time(Delay based) ○ One Time(Date based) ○ Repeating

Initial Delay: 0 Hours

3. In **Job Name**, enter a name for the schedule.

   Maximum length of the job name must not exceed 100 characters. The name must not contain any special characters such as ampersand (&).

4. From the **Configuration Section** list, choose a configuration type.

   This list contains a logical set of parameters.

5. In **Prestart Init**, enter a numerical value for the number of runtime processes that must be spawned initially. Ensure that the value is greater than or equal to 1.

6. In **Prestart Timeout**, enter a numerical value for the time in minutes after which the unused prestart process will be stopped. If this value is set to zero, the timer never starts and thus the processes do not time out.

7. From the **Schedule Type** options, select the appropriate schedule type.

   Following are the three types of schedules that you can set:

   - **One Time(Delay based)**: Select this option if you want to schedule a single occurrence prestart based on the initial delay. Initial delay is a time based parameter that specifies the number of hours or minutes after which the prestart will begin. If you select this option, you must enter the initial delay time (in hours or minutes) in the **Initial delay** field that appears below the schedule type.

   - **One Time(Date based)**: Select this option if you want to schedule a single occurrence prestart based on a date. If you select this option, you must enter the date and time in the **Start Date** field that appears below the Schedule type.

   - **Repeating**: Select this option if you want to schedule a repeat prestart. From the **Frequency** list, you can select one of the following options:

     – **Repeat date and interval**: If you select this option, you must specify the start date and interval after which you want the prestart to repeat.

  - **Repeat initial delay and interval**: If you select this option, you must specify the initial delay and interval after which you want the prestart to repeat.
8. Click **Submit**.

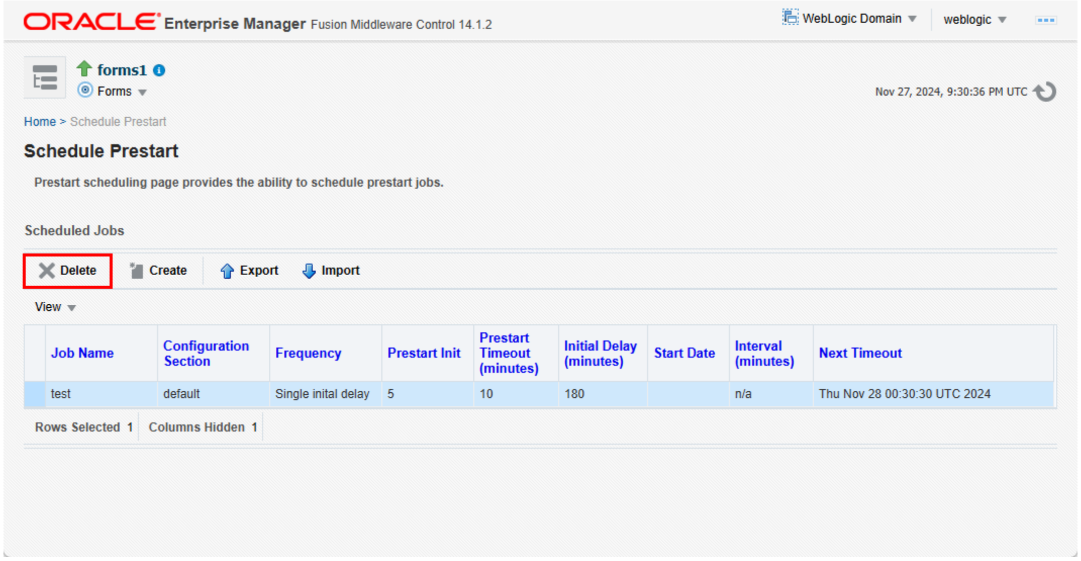
   You can click **Show Jobs** to view all the prestart schedules that are created.

Deleting a Prestart Schedule

To delete a prestart schedule, perform the following steps:

1. From the Forms Menu, select **Schedule Prestart**. The **Schedule Prestart** page is displayed.
2. From the **Scheduled Jobs** region, select the row of the scheduled prestart that you want to delete.

Figure 50   Deleting a Prestart Schedule


ORACLE Enterprise Manager Fusion Middleware Control 14.1.2
WebLogic Domain
weblogic
forms1
Forms
Nov 27, 2024, 9:30:36 PM UTC
Home > Schedule Prestart

Schedule Prestart

Prestart scheduling page provides the ability to schedule prestart jobs.

Scheduled Jobs

Delete
Create
Export
Import
View

Job Name	Configuration Section	Frequency	Prestart Init	Prestart Timeout (minutes)	Initial Delay (minutes)	Start Date	Interval (minutes)	Next Timeout
test	default	Single inital delay	5	10	180		n/a	Thu Nov 28 00:30:30 UTC 2024

Rows Selected 1 Columns Hidden 1

> **Tip:**
>
> You can select multiple rows to delete at the same time.

3. Click **Delete**. The selected prestart schedule is deleted.

Exporting and Importing Prestart Schedules

This utility allows you to export all existing schedules from a particular managed server and import it to some other managed server. This feature does not allow the users to export schedules selectively; the user must export all existing schedules. To export and import prestart schedules, perform the following steps:

1. From the Forms Menu, select **Schedule Prestart**. The **Schedule Prestart** page is displayed.
2. From the **Scheduled Jobs** region, click **Export**. A dialog box appears.
3. Enter a new file name in the dialog box.

> **Note:**
>
> The file name must contain `.xml` extension.

4. Click **OK**. This will create an XML file that contains the attributes of all prestart schedules. Depending on your browser settings, you will either be prompted to choose a location to save the file in, or the file will be saved at a default location on you local machine.
5. To import the schedules (that you have exported in the above steps) to some other managed server, go to the **Schedule Prestart** page of that server, and click **Import**. A dialog box appears.
6. Enter the name of the file that you created while exporting the schedules. You can also click **Browse** on the dialog box and upload the `xml` file from you local machine.
7. Click **OK**.

> **Note:**
>
> If there is an error in any schedule entry in the `xml` file, the server skips that particular schedule and imports the next schedule entry. If there are any expired schedules in the `xml` file, they are ignored and not imported.

The imported schedules will now be listed in the **Scheduled Jobs** region.

Minimizing Client Resource Requirements

The Java client is primarily responsible for rendering the application display. It has no embedded application logic. Once loaded, a Java client can display multiple forms simultaneously. Using a generic Java client for all Oracle Forms applications requires fewer resources on the client when compared to having a customized Java client for each application.

The Java client is structured around many Java classes. These classes are grouped into functional subcomponents, such as displaying the splash screen, communicating with the network, and changing the look-and-feel. Functional subcomponents allow the Oracle Forms Developer and the Java Virtual Machine (JVM) to load functionality as it is needed, rather than downloading all of the functionality classes at once.

Minimizing Forms Services Resource Requirements

When a form definition is loaded from an FMX file, the profile of the executing process can be summarized as:

- Encoded Program Units
- Boilerplate Objects/Images
- Data Segments

Of these, only the data segments section is unique to a given instance of an application. The encoded program units and boilerplate objects/images are common to all application users. Oracle Forms Services maps the shared components into physical memory, and then shares them between all processes accessing the same FMX file.

The first user to load a given FMX file will use the full memory requirement for that form. However, subsequent users will have a greatly reduced memory requirement, which is dependent only on the extent of local data. This method of mapping shared components reduces the average memory required per user for a given application.

Minimizing Network Usage

Bandwidth is a valuable resource, and the general growth of Internet computing puts an ever increasing strain on the infrastructure. Therefore, it is critical that applications use the network's capacity sparingly.

Oracle Forms Services communicates with the Java client using metadata messages. Metadata messages are a collection of name-value pairs that tell the client which object to act upon and how. By sending only parameters to generic objects on the Java client, there is approximately 90-percent less traffic (when compared to sending new code to achieve the same effect).

Oracle Forms Services intelligently condenses the data stream in three ways:

- When sets of similar messages (collections of name-value pairs) are sent, the second and subsequent messages include only the differences from the previous message. This results in significant reductions in network traffic. This process is called *message diff-ing*.
- When the same string is to be repeated on the client display (for example, when displaying multiple rows of data with the same company name), Oracle Forms Services sends the string only once, and then references the string in subsequent messages. Passing strings by reference increases bandwidth efficiency.
- Data types are transmitted in the lowest number of bytes required for their value.

Maximizing the Efficiency of Packets Sent Over the Network

The extensive use of triggers within the Oracle Forms Developer model is a strength, but they can increase the effect of latency by requiring a network round trip for each trigger. Latency can be the most significant factor that influences the responsiveness of an application. Notice that latency is not the same as network speed. Network speed involves a measure of the bits that can be transported per time unit whereas latency is the time taken for one bit to travel from one end-point to the other. One of the best ways to reduce the effects of latency is to minimize the number of network packets sent during a conversation between the Java client and the Forms Services.

Oracle Forms Services implements event bundling by grouping trigger events together through Event Bundling. Event Bundling gathers all of the events triggered while navigating between the two objects, and delivers them as a single packet to Oracle Forms Services for processing.

For example, when a user navigates from item A to item B (such as when tabbing from one entry field to another), a range of pre- and post-triggers may fire, each of which requires processing on the Forms Services. When navigation involves traversing many objects (such as when a mouse click is on a distant object), Event Bundling gathers all events from all of the objects that were traversed, and delivers the group to Oracle Forms Services as a single network message.

Applet/servlet parameters like `gzipCompressApplet` and `removeCommentLinesFromApplet` and others can help in improving startup performance. Refer to Web Configuration Parameters for details on these and many other useful parameters.

Rendering Application Displays Efficiently on the Client

All boilerplate objects in a given form are part of a Virtual Graphics System (VGS) tree. VGS is the graphical subcomponent that is common to all Oracle Forms Developer products. VGS tree objects are described using attributes such as coordinates, colors, line width, and font. When sending a VGS tree for an object to the Java client, the only attributes that are sent are those that differ from the defaults for the given object type.

Images are transmitted and stored as compressed JPEG images. This reduces both network overhead and client memory requirements.

Minimizing resources includes minimizing the memory overhead of the client and server processes. Optimal use of the network requires that bandwidth be kept to a minimum and that the number of packets used to communicate between the client and Oracle Forms Services be minimized to contain the latency effects of the network.

Oracle Forms Services Applications Tuning

An application developer can take steps to ensure that maximum benefits are gained from Forms Services' built-in architectural optimizations.

The following sections discusses key performance issues that affect many applications and how developers can improve performance by tuning applications to exploit Forms Services features.

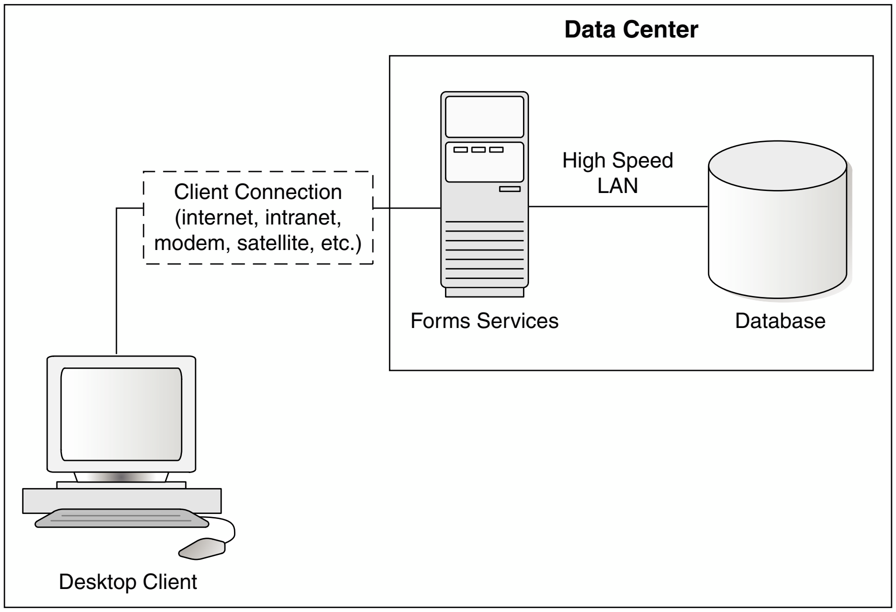
Location of the Oracle Forms Services with Respect to the Data Server

The Forms Java client is only responsible to display the GUI objects. All of the Oracle Forms logic runs in Oracle Forms Services, on the middle tier. This includes inserting or updating the data to the database, querying data from the database, executing stored procedures on the database, and so on. Therefore, it is important to have a high-speed connection (high bandwidth and not low latency) between the application server and the database server.

All of this interaction takes place without any communication to the Forms Java client. Only when there is a change on the screen is there any traffic between the client and Forms Services. This allows Oracle Forms applications to run across slower networks (high latency networks), such as with modems or satellites.

The configuration in the following figure shows how Forms Services and the database server are located together in a data center.

Figure 51 Co-Locating the Oracle Application Server Forms Services and Database Server


Data Center
High Speed LAN
Client Connection (internet, intranet, modem, satellite, etc.)
Forms Services
Database
Desktop Client

Minimizing the Application Startup Time

First impressions are important, and a key criterion for any user is the time it takes to load an application. Startup time is regarded as overhead. It also sets an expectation of future performance. When a business uses thin-client technologies, the required additional overhead of loading client code may have a negative impact on users. Therefore, it is important to minimize load time wherever possible.

After requesting an Oracle Forms application, several steps must be completed before the application is ready for use:

1. Invoke Java Virtual Machine (JVM).
2. Load all initial Java client classes, and authenticate security of classes.
3. Display splash screen.
4. Initialize form:
    a. Load additional Java classes, as required.
    b. Authenticate security of classes.
    c. Render boilerplate objects and images.
    d. Render all elements on the initial screen.
5. Remove splash screen.
6. Form is ready for use.

An application developer has little influence on the time it takes to launch the JVM. However, the Java deployment model and the structure of the Oracle Forms Developer Java client allow the developer to decide which Java classes to load and how. This, in turn, minimizes the load time required for Java classes.

The Java client requires a core set of classes for basic functionality (such as opening a window) and additional classes for specific display objects (such as LOV items). These classes must initially reside on the server. See Using Java Files for more information about how to improve the time it takes to load these classes into the client's JVM.

Using Java Files

Java provides the Java Archive (JAR) mechanism to create files that allow classes to be grouped together and then compressed (zipped) for efficient delivery across the network to the client. Once used on the client, the files are cached for future use.

Table 51 lists Forms JAR files provided to support the client side functionality of a running application.

Table 51 Forms JAR Files

File Name	Description
`frmall.jar`	Contains all the Java class files needed to run most modules. This file should not be used if using Deferred JAR Loading.
`frmwebutil.jar`	Contains the class files required for running WebUtil enabled modules
`frmwebsocketjsi.jar`	Contains the class files required for running WJSI enabled modules
`frmgeneric_laf.jar`	Contains the class files required for using the "generic" Look and Feel option. This is only supported with Oracle Java 8.
`frmoracle_laf.jar`	Contains the class files required for using the "oracle" Look and Feel option. This is only supported with Oracle Java 8.
`frmmain.jar`	Contains a minimal number of Java class files needed to run most modules written for English reading users. This file should only be used if using Deferred JAR Loading. This is only supported with Oracle Java 8.
`frmresources.jar`	Contains the class files required to support languages other than English. This is only supported with Oracle Java 8.

Reducing the Required Network Bandwidth

The developer can design the application to maximize the data stream compression, called message-diffing, that Forms automatically performs. This means that Forms sends along data stream compression by using message diff-ing, which sends along only the information that differs from one message to another. The following steps can be taken to reduce the differences between messages:

- **Promote similarities between objects.** Using similar objects improves *message diff-ing* effectiveness (in addition to being more visually appealing to the user). The following steps encourage consistency between objects:
  - Accept default values for properties, and change only those attributes needed for the object.
  - Use Smart Classes to describe groups of objects.
  - Lock the look-and-feel into a small number of visual attributes.

- **Reduce the use of boilerplate text.** As a developer, you should use the PROMPT item property rather than boilerplate text wherever applicable. Forms Developer 6.0 and higher includes the Associate Prompt feature, which allows boilerplate text to be re-designated as the prompt for a given item.
- **Reduce the use of boilerplate items (such as arcs, circles, and polygons).** All boilerplate items for a given form are loaded at form initialization. Boilerplate items take time to load and use resources on the client whether they are displayed or not. Common boilerplate items, namely rectangles and lines, are optimized. Therefore, restricting the application to these basic boilerplate items reduces network bandwidth and client resources while improving startup times.
- **Keep navigation to a minimum.** An Event Bundle is sent each time a navigation event finishes, whether the navigation extends over two objects or many more. Design forms that do not require the user to navigate through fields when default values are being accepted. A form should encourage the user to quickly exit once the form is complete, which causes all additional navigation events to fire as one Event Bundle.
- **Reduce the time to draw the initial screen.** Once the Java client has loaded the required classes, it must load and initialize all of the objects to be displayed before it can display the initial screen. By keeping the number of items to a minimum, the initial screen is populated and displayed to the user more promptly. Techniques that reduce the time to draw the initial screen include:
  - Providing a login screen for the application with a restricted set of objects (such as a title, small logo, username, and password).
  - On the form's initial display, hiding elements not immediately required. Use the canvas properties:
    `RAISE ON ENTRY = YES` (Canvas only)

    `VISIBLE = NO`

    Pay attention to TAB canvases that consist of several sheets where only one will ever be displayed. For responsive switching between tabs, all items for all sheets on the canvas are loaded, including those that are hidden behind the initial tab. Consequently, the time taken to load and initialize a TAB canvas is related to all objects on the canvas and not just to those initially visible.

    > **Tip:**
    >
    > When using Tab canvases, use stacked canvases and display the right canvas in the when-tab-page-changed trigger. Remember to set the properties `RAISE ON ENTRY = YES` and `VISIBLE = NO` for all the canvases not displayed in the first screen.
- **Disable MENU_BUFFERING.** By default, MENU_BUFFERING is set to True. This means that changes to a menu are buffered for a future "synchronize" event when the altered menu is re-transmitted in full. (Most applications make either many simultaneous changes to a menu or none at all. Therefore, sending the entire menu at once is the most efficient method of updating the menu on the client.) However, a given application may make only minimal changes to a menu. In this case, it may be more efficient to send each change as it happens. You can achieve this using the statement:

  ```
  Set_Application_Property (MENU_BUFFERING, 'false');
  ```

  Menu buffering applies only to the menu properties of `LABEL`, `ICON`, `VISIBLE`, and `CHECKED`. An `ENABLE`/`DISABLE` event is always sent and does not entail the retransmission of an entire menu.

Other Techniques to Improve Performance

The following techniques may further reduce the resources required to execute an application:

- **Examine timers and replace with JavaBeans.** When a timer fires, an asynchronous event is generated. There may not be other events in the queue to bundle with this event. Although a timer is only a few bytes in size, a timer firing every second generates 60 network trips a minute and almost 30,000 packets in a typical working day. Many timers are used to provide clocks or animation. Replace these components with self-contained JavaBeans that achieve the same effect without requiring the intervention of Oracle Forms Services and the network.
- **Consider localizing the validation of input items.** It is common practice to process input to an item using a When-Validate-Item trigger. The trigger itself is processed on the Oracle Forms Services. You should consider using pluggable Java components to replace the default functionality of standard client items, such as text boxes. Then, validation of items, such as date or max/min values, are contained within the item. This technique opens up opportunities for more complex, application-specific validation like automatic formatting of input, such as telephone numbers with the format (XXX) XXX-XXXX.
- **Reduce the application to many smaller forms, rather than one large form.** By providing a fine-grained application, the user's navigation defines which objects are loaded and initialized from the Oracle Forms Services. With large Forms, the danger is that the application is delayed while objects are initialized, many of which may never be referenced. When chaining Forms together, consider using the built-ins OPEN_FORM and NEW_FORM:
  - With OPEN_FORM, the calling Form is left open on the client and the server, so that the additional Form on both the client and the server consumes more memory. However, if the Form is already in use by another user, then the increase in server memory is limited to just the data segments. When the user returns to the initial Form, it already resides in local memory and requires no additional network traffic to redisplay.
  - With NEW_FORM, the calling Form is closed on the client and the server, and all object properties are destroyed. Consequently, it consumes less memory on the server and client. Returning to the initial Form requires that it be downloaded again to the client, which requires network resources and startup time delays. Use OPEN_FORM to display the next Form in an application unless it is unlikely that the initial form will be called again (such as a login form).
- **Avoid unnecessary graphics and images.** Wherever possible, reduce the number of image items and background images displayed in your applications. Each time an image is displayed to application users, the image must be downloaded from the application server to the user's Web browser. To display a company logo with your Web application, include the image in the HTML file that downloads at application startup. Do this instead of including it as a background image in the application. As a background image, it must be retrieved from the database or file system and downloaded repeatedly to users' computers.

Forms Diagnostics Agent

Forms Diagnostics Agent or Forms Metrics Agent enables the user to analyze various performance-related information about Forms applications running in your environment.

Note:

For Forms Diagnostics Agent to work, see Installing and Configuring Oracle Forms.

This agent accesses the metrics data (available in DMS) at regular time intervals and populates the database tables. This process allows the user to access the data collected as historical data. The deployment of Forms Diagnostics agent is optional. The agent application provides an interactive interface where the user can specify the frequency of data collection and also control the starting and stopping of data collection. This can be achieved by performing the tasks in the following sections:

- Setting up the Database Schema
- Setting up a Data Source in WebLogic
- Deploying Forms Diagnostics Agent
- Managing the Data Collection
- Use the Agent Application
- Limitations of the Agent Application

Setting up the Database Schema

To set up DB schema for Forms Diagnostics Agent, you must create a user and schema in the database. The user can choose a database instance of their choice. There is no special database that is installed with Forms or the diagnostic agent.

Create a User in Database

Note:

Before creating a user in the database, ensure that the user name provided by you is new and does not already exist. This is because the `.sql` script (used to create the user in the database) overwrites the user (user name provided during the creation) with the new user.

To create a user in the database, perform the following steps:

1. Log in to the database as `sysdba` as shown below:

   ```
   sqlplus sys/<sys-password>@<DB> as sysdba
   ```

2. Run the following script:

   ```
   @ORACLE_HOME/forms/forms_create_diagnostics_user.sql.
   ```

3. The user must enter the userID and password.

The user is created in the database.

Create a Schema in Database

> **Note:**
>
> Before creating a schema in the database, ensure that the user name provided by you is new and does not already exist. This is because the `.sql` script (used to create the schema in the database) overwrites the schema (user name provided during the creation) with the new schema.

To create a schema in the database, perform the following steps:

1. Log in to the database as the user that you created in the above steps:

   ```
   sqlplus <user>/<password>@<DB>
   ```

2. Run the following script:

   ```
   @ORACLE_HOME/forms/forms_create_diagnostics_schema.sql
   ```

The schema is created in the database.

Setting up a Data Source in WebLogic

After setting up the database to work with the Forms Diagnostics Agent, you must set up a data source using the Weblogic console.

To setup a data source using the Weblogic console, perform the following steps:

1. Log into the WebLogic console.
2. In the left navigation panel, select **Services** and navigate to **Data Sources**.

   Click **Lock and Edit** in the Change Center window to make changes.
3. In the **Summary of JDBC Data Sources** page, click **Configuration**.

   In the **Data Sources** table, click **New** and select **Generic Data Sources** from the list.
4. Enter the values for the following parameters:

   name for the JDBC Data Sources

   The user can enter any name.

   JNDI name

   oracle/forms/agentDS

   Database Type

   Choose type of database that you used to create user and schema in the previous steps.

   Click **Next**. The **Create a new JDBC data sources** page appears.
5. Select **Database driver** from the list of drivers available for the type of database you have selected. Click **Next**.
6. Enter the values for the following parameters:

   Database Name

   Host Name

Port

Database User Name

Enter the user name that you used while creating a user in the database in the steps above.

Password

Enter the password that you used while creating a user in the database in the steps above.

Click **Next**.

7. In the next page, click **Test Configurations** at the top left corner to check if the database has been configured successfully.

   Click **Next**.

8. Select **Admin Server** as a target to deploy the data source.

   Click **Finish**.

9. Click **Activate Changes** in the Change Center window to save changes.

You have now set up a JDBC data source.

Deploying Forms Diagnostics Agent

After setting up a data source in Weblogic, Forms Diagnostics agent must be deployed to the Weblogic Admin Server.

To deploy Forms Diagnostics agent, perform the following steps in the Weblogic console:

1. Log into the Weblogic Console.
2. In the left navigation panel, select **Deployments**.

   Click **Lock and Edit** in the Change Center window to make changes.

3. In the Summary of Deployments page, click **Install**.

   The **Install Application Assistant** page appears.

4. Enter the path of the .war file as shown below:

   ```
   ORACLE_HOME/forms/j2ee
   ```

   This is the location of the `formsagentapp.war` file.

5. Select the `formsagentapp.war` file. Click **Next**.

   The **Choose Targeting Style** page appears.

6. Select **Install this deployment as an application**.
7. Select **Admin Server** as a target to deploy Forms Diagnostics Agent. Click **Next**.
8. Leave the optional settings at their default values and click **Finish**.

   Click **Activate Changes** in the Change Center window to save changes.

The Forms Diagnostics agent has been successfully deployed to the Weblogic Admin Server. To start the application, select **formsagentapp** from the list of deployed applications and Click **Start**.

Managing the Data Collection

The Forms Diagnostics agent allows the users to manage data collection using an interface.

The user can specify the frequency of data collection and control the starting or stopping of data collection. This can be achieved by performing the following steps:

1. Log in to the agent console by using the following url:

   ```
   http://<host>:<admin port>/formsagent/AgentConsole.jsp
   ```

2. Enter the user ID and password.

   Any user with administrator's privileges can log in to the console.

3. Enter a value for the **Frequency of Data Collection**. This parameter is the time difference between two consecutive data collections.

   The default value is 10 minutes. The minimum value should be one minute.

4. Click **Start**.

You will see a message indicating that the Forms Diagnostics Agent is running. Click **Stop** whenever you want to stop the collection of metrics by the agent.

Use the Agent Application

When the user prompts the agent application to start the collection of metrics, the agent collects the metrics from DMS and populates the database tables. The user can access this collected metrics in the database tables.

To bring this collected metrics into use, the user can create a frontend application which will be able to read this data and analyze the historical performance of Forms applications running in your environment by preparing charts, graphs, etc.

The primary and foreign key in each table has been mentioned in the respective tables. The following are the database tables that get populated during the collection of metrics by the Forms Diagnostics agent:

Table 52 ADMIN_SERVER Database Table

Serial Number	Column Name	Sample Value	Description
01	AGENT_ID	1	AGENT_ID is the primary key in the ADMIN_SERVER database table. ID of the agent application. Any integer value beginning with 1
02	ADMIN_HOSTNAME	myhost.mydomain.com	Name of the machine where admin server is deployed
03	ADMIN_PORT	7001	Port of the admin server

Table 53 AGENT Database Table

Serial Number	Column Name	Sample Value	Description
01	AGENT_STATUS	Running	Status of the Agent

Table 53   (Cont.) AGENT Database Table

Serial Number	Column Name	Sample Value	Description
02	REAL_TIME	2009.07.23 at 17:11:41	Date and time when status is recorded All time entries are in UTC/GMT
03	SEQUENCE_ID	205	ID created by the agent each time the agent status is recorded
04	FREQUENCY	40	Time difference (in minutes) between two consecutive data collections
05	AGENT_ID	1	AGENT_ID is a foreign key in this database table. It refers to AGENT_ID in the ADMIN_SERVER database table ID of the agent application. Any integer value beginning with 1

Table 54   FRM_DB Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_DB_ID	8769	FRM_DB_ID is the primary key in the FRM_DB database table. ID assigned to the database that is used by the Forms application
02	DB_NAME	v11g	The database to which frmweb is connected. If it is connected to v11g, the this field will display v11g. This can be NULL when frmweb is not connected to any database
03	TNS_ENTRY	(DESCRIPTION = (ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = sample.host.com)(PORT = 1521))) (CONNECT_DATA = (SERVICE_NAME = v11g)))	TNS entry of the database to which frmweb is connected
04	USER_NAME	scott	The database user who has logged in

Table 55   FRM_DB_LOGIN Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_DB_LOGIN_ID	345	FRM_DB_LOGIN_ID is the primary key in the FRM_DB_LOGIN database table. ID assigned to each row in the table
02	FRM_RUNTIME_ID	107	FRM_RUNTIME_ID is a foreign key in this database table. It refers to FRM_RUNTIME_ID in the FRM_RUNTIME database table
03	FRM_DB_ID	1025	FRM_DB_ID is a foreign key in this database table. It refers to FRM_DB_ID in the FRM_DB database table ID assigned to the database that is used by the Forms application
04	REAL_TIME	2009.07.23 at 17:11:41	Date and time when status is recorded and metric data collected to update the table All time entries are in UTC/GMT

Table 56   FRM_RUNTIME Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_RUNTIME_ID	107	FRM_RUNTIME_ID is a primary key in the FRM_RUNTIME database table. ID that identifies the Forms process
02	WLS_APP_ID	5005	WLS_APP_ID is a foreign key in this database table. It refers to WLS_APP_ID in the WLS_APP database table Determines the Forms application in the WLS_APP table
03	FRM_USER_ID	1310	FRM_USER_ID is a foreign key in this database table. It refers to FRM_USER_ID in the FRM_USER database table ID assigned to the Forms client or client instance
04	CONFIG_VALUE		Configuration section name from `formsweb.cfg`

Table 56 (Cont.) FRM_RUNTIME Database Table

Serial Number	Column Name	Sample Value	Description
05	CONNECT_TIME	2009.07.23 at 17:11:41	Date and time when frmweb is spawned All time entries are in UTC/GMT
06	DISCONNECT_TIME	2009.07.23 at 18:15:43	Date and time when frmweb terminates All time entries are in UTC/GMT
07	STARTING_FORM_NAME	emp	Name of starting form
08	PROCESS_ID	8020	Process Id of frmweb on the middle tier machine
09	FRM_STATUS	Running / Exited	Status of frmweb
10	FRM_CPU_TIME_ON_EXIT	256	CPU time on exit of frmweb
11	FRM_PRIVATE_MEMORY_ON_EXIT	6385	Memory used by the Forms process at the time of exit
12	FRM_EXIT_CODE		

Table 57 FRM_TRACE Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_TRACE_ID	2679	FRM_TRACE_ID is the primary key in this database table. ID assigned to the rows in the FRM_TRACE database table
02	TRACE_FILE	forms_1055.trc	Name of the trace file when `ftrace` is enabled. It can be NULL if `ftrace` is disabled
03	TRACING	mytrace	The name of the trace group selected by the application

Table 58 FRM_TRACE_USE Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_TRACE_USE_ID	1906	FRM_TRACE_USE_ID is the primary key in this database table. ID assigned to the rows in the database table

Table 58 (Cont.) FRM_TRACE_USE Database Table

Serial Number	Column Name	Sample Value	Description
02	FRM_RUNTIME_ID	107	FRM_RUNTIME_ID is a foreign key in this database table. It refers to FRM_RUNTIME_ID in the FRM_RUNTIME database table ID that identifies the Forms process
03	FRM_TRACE_ID	2679	FRM_TRACE_ID is a foreign key in this database table. It refers to FRM_TRACE_ID in the FRM_TRACE database table ID assigned to the row in the database table
04	REAL_TIME	2009.07.23 at 17:11:41	Date and time when status is recorded and metric data collected to update the table All time entries are in UTC/GMT

Table 59 FRM_USER Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_USER_ID	1310	FRM_USER_ID is the primary key in this database table. ID assigned to the Forms client or client instance
02	CLIENT_IP	255.255.255.255	IP address of client machine from where the browser was launched and through which the user connected to the middle tier
03	SSO_USERID	fname.lname@myapp.com	Single Sign-On ID of the user who logged in.

Table 60 HISTORY Database Table

Serial Number	Column Name	Sample Value	Description
01	FRM_RUNTIME_ID	107	FRM_RUNTIME_ID is a foreign key in this database table. It refers to FRM_RUNTIME_ID in the FRM_RUNTIME database table ID that identifies the Forms process

Table 60   (Cont.) HISTORY Database Table

Serial Number	Column Name	Sample Value	Description
02	REAL_TIME	2009.07.23 at 17:11:41	Date and time when the snapshot is taken All time entries are in UTC/GMT
03	SEQUENCE_ID	205	ID created by the agent each time agent status is recorded.
04	FRM_BYTES_SENT	400	Number of bytes sent from the server to the client for this process so far
05	FRM_BYTES_SENT_DELTA	37	Difference in the number of bytes sent from the server to the client for this process since the previous reading of the agent was taken
06	FRM_BYTES_RECEIVED	200	Number of bytes sent from the client to the sever for this process so far
07	FRM_BYTES_RECEIVED_DELTA	23	Difference in the number of bytes sent from the client to the server for this process since the previous reading of the agent was taken
08	FRM_NETWORK_ROUND_TRIPS	30	Number of network round trips between the client and the server for this process so far
09	FRM_NETWORK_ROUND_TRIPS_DELTA	3	Difference in the number of network roundtrips between the client and the server for this process since the previous reading of the agent was taken
10	FRM_CPU_TIME	230	Total processing time taken by frmweb (in milliseconds) for this process so far
11	FRM_CPU_TIME_DELTA	47	Difference in the value of FRM_CPU_TIME since the previous reading of the agent was taken
12	FRM_PRIVATE_MEMORY	7998	Memory used by the Forms process at the time when snapshot was taken.
13	ITERATION	50	Number of times data is collected into the database table.

Table 61   WLS_APP Database Table

Serial Number	Column Name	Sample Value	Description
01	WLS_APP_ID	5005	WLS_APP_ID is the primary key in this database table. Determines the Forms application in the WLS_APP table
02	SERVER_TYPE	MANAGED	Type of the server (For example, MANAGED or ADMIN)
03	SERVER_NAME	WLS_FORMS	Name of the server
04	DEPLOYED_APPLN_NAME	formsapp	Forms Application name
05	FORMS_HOSTNAME	host52.example.com	Middle tier machine on which Forms runtime is running
06	INSTANCE_HOME_NAME	asinst_1	Name of the FMW instance home, where Forms runtime is deployed
07	CLUSTER_NAME	cluster_xyz	Name of the cluster where the Forms application is deployed
08	AGENT_ID	1	AGENT_ID is a foreign key in this database table. It refers to AGENT_ID in the ADMIN_SERVER database table. ID of the agent application. Any integer value beginning with 1

Limitations of the Agent Application

Forms Diagnostics agent has certain limitations on its deployment and usage.

The limitations are as follows:

- The deployment of the Forms Diagnostics Agent application is optional. In case you want to analyze performance-related information about Forms applications, you must deploy Forms Diagnostics Agent manually post installation.
- The agent application must be deployed to the Admin Server only. The agent application collects information about all Forms sessions that are running in the WLS domain of the Admin Server.
- For the agent to be able to access the metrics data (available in DMS), the DMS application must be up and running.
- The schema is designed to be functional only on one domain at any given time. You cannot use the same schema for multiple agents (running in separate domains).
- Do not set the frequency of data collection to a small value. Setting the frequency of data collection to a small value slows down the production environment and causes excessive, needless data collection.

- This utility only provides the database objects and the agent needed to perform the collection. It does not provide a user interface (UI) for exposing the collected data. Data can be retrieved by querying the tables outlined in the documentation above. Alternatively, a user interface can be developed using a preferred technology.

Troubleshooting

This chapter describes problems that you might encounter when you run an application over the Web using Oracle Forms, and explains how to solve them.
It contains an outline of common causes for errors, the method you can use to verify your installation, and the tools and techniques provided to diagnose problems. The appendix is also a subset of the Oracle9i Forms Diagnostic Techniques white paper.

The following topics are included:

- Accessing the Listener Servlet Administration Page
- Verifying the Installation
- Diagnose FRM-XXXXX Errors
- Diagnosing Server Crashes with Stack Traces
- Diagnosing Client Crashes
- Forms Trace and Servlet Logging Tools
- Resolving Memory Problems
- Troubleshooting Tips

Accessing the Listener Servlet Administration Page

You can display a test page for the Listener Servlet by accessing the following URL:

```
http://<hostname>:<port>/forms/frmservlet/admin
```

The Servlet Administration Test utility (`/forms/frmservlet/admin`) has always been helpful for troubleshooting, but was previously difficult or risky to enable because it required modifying the Forms J2EE deployment plan. Enable this utility by setting `allow_debug=testmode` in the Forms Web Configuration (`formsweb.cfg`). Valid values for `allow_debug` are:

Table 62 allow_debug Values

Value	Description
`FALSE` (default)	Blocks Servlet Administration pages and the commands associated with it. Servlet debugging and logging is disabled. The Forms Listener Servlet status page is presented.
`TRUE`	Enables various debug and ODL logging functions exposed by Forms, as well as enabling Servlet Administration specific commands. Servlet Administration test page is not accessible. Forms Listener Servlet status page is presented.
`TESTMODE`	Same as setting to `TRUE`, however allows access to the Servlet Administration test page.

Table 62 (Cont.) allow_debug Values

Value	Description
`NONE`	Blocks Forms Listener Servlet status, Servlet Administration page, and the commands associated with it. Servlet debugging and logging is disabled.

> **Note:**
>
> By default `allow_debug` is disabled (`FALSE`). It is recommended that this parameter only be set to `TRUE` or `TESTMODE` during troubleshooting. It should not be set to `TRUE` or `TESTMODE` unless explicitly being used by administrators during testing.

Verifying the Installation

If there is something wrong with the installation, then it will result in faulty configuration and Oracle Forms will not run correctly.

After the Oracle Universal Installer indicates that Fusion Middleware was successfully installed, you can verify whether Oracle Forms Services is correctly configured or not.

You can use the following tool:

- Web Form Tester

Using the Web Form Tester

The Web Form Tester is available with your Oracle Fusion Middleware installation.

> **Note:**
>
> The Web Form Tester is a simple static html page. It can be customized as needed to meet your needs. However, keep in mind that the file is stored with the Forms Servlet deployment (in the WLS Domain) therefore it may be replaced/overwritten when applying patches or upgrading.

To verify whether the Oracle installation and configuration of Forms Services is correct, run the Web Form Tester. To verify an installation, this should be performed on the middle tier where the installation occurred.

These steps assume that WebLogic Server and Oracle Forms have already been installed, the Configuration Wizard run successfully, and Node Manager, Admin Server, and WLS_FORMS have been started. It is also assumed that a Java Plugin has been installed and configured for the browser.

1. Open a browser certified for use with this Oracle Forms release.
2. In the browser enter the following URL and press Enter. Be sure to replace the host name and port with the appropriate values for your environment. The default port for WLS_FORMS is 9001.

`http://hostName:9001/forms/html/runform.htm`

3. Using the default values, click the **Run form** button. This should load the Oracle Forms test form. If displayed, it will indicate that Oracle Forms has been successfully installed. Press the **Exit** button to exit the test form.

   It is also possible to run the test form directly, without using the Web Form Tester page, by using the URL: `http://hostName:9001/forms/frmservlet?form=test`

The possible reasons for the Web Form Tester page or the test form to not display:

- Incorrect host name and/or port number used.
- WLS managed server (e.g. WLS_FORMS) is not running.
- Incorrect network configuration on server host and/or client.
- Installation and/or post installation Configuration not successfully completed.

Diagnose FRM-XXXXX Errors

Use the Oracle Forms Applet tool to diagnose and resolve FRM-XXXXX errors.

The Oracle Forms Applet

The brief message about the FRM error should help in identifying the basic cause of the problem. Often, everything required to identify the cause an FRM error is contained in the error reported by the Forms applet. When a FRM error is raised, the error dialog will have a **Details** button. Click the **Details** button will show the current Java stack. The exact stack is tied to the root cause and the version of Oracle Forms. This is due to the differing package structure used for the applet class files in the different releases.

Diagnosing Server Crashes with Stack Traces

If the Forms web runtime terminates unexpectedly, then it writes a stack trace to the directory `$DOMAIN_HOME/system_components/FORMS/forms1/trace`.

The filename will have the format `<forms_runtime_process>_dump_<process id>`.The dump file contains a stack trace of the running process, and shows the last successful operation performed by Forms. This core file users can use to assemble a stack trace with symbol names using GNU Debugger, dbx or similar debugging tool on the machine where the dump occurred.

The following topics are included:

- Stack Traces
- Configuring and Using Stack Traces

Stack Traces

A stack trace is useful for the following two reasons:

- The information in the stack is used to identify a known issue. It is not 100% reliable, but an identical stack trace is a good indicator of a matching problem. Even if it is not the same, there may be a workaround or patch for an existing bug that can be tested.
- If the problem is not a known bug, then the stack may provide valuable information to assist development efforts to pinpoint the cause.

Configuring and Using Stack Traces

To configure and use Stack Traces you have to verify the environment, and understand UNIX and Windows Stack Traces.

Verifying the Environment

To test stack tracing on UNIX or Windows you can set the environment variable `FORMS_DELIBERATECRASH`. As the name suggests, setting this will cause the forms runtime process to crash. Oracle Forms currently recognizes two settings: `1` and `2`. If `FORMS_DELIBERATECRASH` is set to `1` then forms will crash at runtime whenever the BELL Built-in is executed. If it is set to `2` then forms will crash at runtime whenever a when-button-pressed trigger is fired. This environment variable can be set in the environment (for example, `default.env`) file.

Understand UNIX Stack Traces

In a UNIX stack trace, the top two functions `siehjmpterm()` and `sigacthandler()` are the signal handling code - these functions will often be present in the stack trace. To see the function the program was in when the error occurred you need to read further down the stack.

If you set `FORMS_CATCHTERM=0` the two functions do not show up in the dump file. The stack trace is displayed without the crash handling symbols.

> **Note:**
>
> `FORMS_CATCHTERM` is deprecated and should no longer be used. It has been replaced with `FORMS_ABTERM_CLEANUP`.

Understand Windows Stack Traces

Stack tracing works differently on UNIX and on Windows. The symbol information is contained inside the executable files and shared libraries on Unix. On Windows this information is stripped out at link time and is in the form of binary `.sym` files. There should be one `.sym` file for every Oracle Forms executable or DLL. The `.sym` files are installed by default. On Windows the files are located in the `ORACLE_HOME\bin` directory. The mechanism on Windows platforms is such that in the event of a crash the Forms runtime process reads all the .sym files that correspond to the forms executable files loaded into memory. It then uses the information in the .sym files to lookup the symbol name.

Diagnosing Client Crashes

Information is provided about diagnosing client crashes and diagnosing hanging applications.

Client Crashes Diagnosis

If the Forms applet disappears unexpectedly, accompanied by a dialog indicating a fatal error, then the Forms applet has crashed. On Windows, a crash will result in the operating system raising an 'illegal operation' dialog, or may cause the "Not responding" flag in Task Manager. To verify the crash, check for a stack trace file on the client. If the client has crashed then a file with the .rpt extension will be created in the same directory as the executable. The root of the filename will be the name of the executable.

Sometimes the applet may appear to have crashed, but no corresponding .rpt file can be found. In this case it is likely that the Oracle Forms has unexpectedly disconnected from the client. The applet will still be running, but it has shutdown all the Forms windows, giving the appearance of a client crash.

Hanging Applications Diagnosis

If the client appears to hang then it is important to verify that the server process is still alive. If the server process has not crashed, but the client no longer appears to respond to user interaction then the application is said to be hanging.

In such cases a thread dump can point to the deadlock. A thread dump can be obtained by pressing t in the Java console. This displays a list of all the threads running in the client JVM.

The information contained in the dump file is extremely useful to Oracle development, and should be included in any bug filed to report the problem.

Causes of Hanging Applications

One cause could be a mismatch between the Java class files and the Oracle Forms version. Communication between the applet and the Forms runtime process is based on message ID. If these message ID's are out of sync, then the applet may not understand an instruction from the server, and vice versa. If you are using JAR files, then try with the <ARCHIVE> tag removed. If the problem persists then pull the correct class files off the installation/patch CD by hand.

One possible cause of a hanging or crashed appearance is that the Forms Runtime process may have died. Check if the Forms Runtime process on the server is still alive. Check that the FORMS_TIMEOUT parameter is set. It defines how long the server should wait for a ping from the Oracle Forms client, only cleaning up the runtime process when there has been no activity from the Forms client for the specified time. The client sends out a HEARTBEAT every two minutes by default. If FORMS_TIMEOUT is set to two minutes or longer, the server will stay up as long as it hears a HEARTBEAT from the client. Set to shorter than the HEARTBEAT interval, it will shut down after the interval specified in FORMS_TIMEOUT. You can set the interval by setting the HEARTBEAT applet parameter in formsweb.cfg, see Configuring Asynchronous Communication. Although this is primarily intended to prevent orphaned server processes, it can also prevent the unwanted premature cleanup of server processes.

Forms Trace and Servlet Logging Tools

Forms Trace and Servlet Logging are two more tools to use in troubleshooting your Oracle Forms Environment.

To configure and use Forms Trace, see Forms Trace and Taking Advantage of Oracle Diagnostics and Logging Tools.

Resolving Memory Problems

To resolve memory problems you have to learn how Java applet uses memory, setting the initial Java heap, and memory leaks.

The following topics are include:

- How Java Uses Memory
- Setting the Initial Java Heap
- Memory Leaks

- Improve Performance with Caching

How Java Uses Memory

Like all software programs, a Java applet uses memory. For Java, the language specification requires a 'garbage collector', which is in an internal memory manager for the Java Virtual Machine (JVM). When a Java program needs memory, it requests this memory from the JVM. If there is no memory left, then the JVM will attempt to free some memory by using the garbage collector. The garbage collector will try to release memory that is no longer required to run the program back to the JVM. If there is still insufficient memory to perform the required task then the JVM will attempt to get more memory from the operating system. If that memory allocation fails, then the Java program will be unable to continue.

Setting the Initial Java Heap

You can specify the initial Java Heap (the memory used by the JVM) for your application through Fusion Middleware Control. For the client, you can change the setting in the Java control panel after you've installed the Oracle Java Plug-in.

> **Note:**
>
> The JVM will only use the memory it is told it is allowed to use. Even if you have memory available with the operating system, the JVM will not use it if told not to.

Memory Leaks

A *memory leak* is an error in a program's dynamic-store allocation logic that causes it to fail to reclaim discarded memory, leading to eventual collapse due to memory exhaustion.

For example, when a program runs it may need to allocate some memory to perform a particular task. If the program has finished with that memory and no longer has any use for it, but fails to make that memory available to other programs running on the computer, then it is said to have leaked the memory.

A typical method used to spot memory leaks is to repeat a series of steps, and observe the memory in use by the application - if the memory usage continues to rise with each iteration, then the assumption is often that the program has a memory leak.

However, some complex applications may choose to retain control of memory it has previously allocated so that it can reuse it at a later point - memory allocation can be an expensive operation, and if the program expects that it will need more memory later it may be more efficient to keep the unused memory available for reuse.

Memory Leaks in Java

The Java language specification demands that the JVM has a garbage collector. In Java, the programmer allocates memory by creating a new object. There is no way to deallocate that memory. Periodically the garbage collector sweeps through the memory allocated to the program, and determines which objects it can safely destroy, therefore releasing the memory. To determine which objects it can safely destroy, the garbage collector uses a 'mark and sweep' algorithm. The garbage collector scans the dynamically allocated memory for objects, marking those which still have active references to them.

After all possible paths to objects have been investigated, unmarked objects that are known to be no longer needed can be garbage collected. A common myth with Java programming is that the presence of a garbage collector means that there can be no memory leaks. This is not true because the garbage collector simply marks those objects, which have active references, and destroys those that do not. It is possible to have an active reference to an object that is no longer needed. This is a memory leak in Java. The solution to the leak is to destroy the references to the object once it is no longer needed so that the garbage collector can identify it as safe to destroy. If a memory leak exists in a Java program, then calling the garbage collector more frequently will not help.

To complicate matters further, the JVM may choose not to release unused memory back to the operating system. In the real world this seldom matters, as most programs will typically require more memory at some point in the near future and can reuse the free memory in the JVM. However, it is worth bearing in mind that not all the memory allocated to the JVM will be in use by the program running in the JVM.

Identifying Memory Leaks

Typically, if a growth in memory usage is observed each time a particular series of operations is performed, then it is a memory leak. The ideal proof is to:

1. Get the form into an initial base state, and record the memory usage,
2. Perform a series of steps to illustrate the problem,
3. Return to the initial base state, and record the memory usage.

By repeating steps 2 and 3, it is possible to determine whether there is a steady memory leak or not. If the growth in memory is small over a large number of iterations, then it may not be a leak at all; it could be that the JVM is retaining unused memory, or the garbage collector is not activating as frequently as expected.

Improve Performance with Caching

When any Java program runs, the Java Virtual Machine needs to load class files. When running over the Internet, the time taken to download a class file each time the program runs can lead to performance problems. To solve this download problem, the JDK supports Java Archive (JAR) files. A JAR file is simply a collection of class files bundled into one compressed file. Typically, the size of the JAR file will be much smaller than the combined size of the class files it contains.

When the JVM first references a class, it checks the local computer to see if any of the previously cached JAR files contain this class. If the class does exist in one of the pre-cached JAR files, then the JVM checks to see if there is a newer version of this JAR file on the application server. If there is a newer JAR file available then the new copy of the JAR file is downloaded to the client cache. If the cached JAR file is up to date, then the class file is loaded from the cached JAR file rather than from over the network.

Caching is important because if the application JAR files do not change, then after the application has run once, and all the JAR files required have been cached on the client, then subsequent invocations of the application will always load the classes from the local cached copies. This can lead to significant performance improvements in the startup time for the application. If new classes are needed to run a specific part of the application, these will be downloaded as required.

Troubleshooting Tips

The troubleshooting list in this section will help you deal with complex issues, but it is not a definitive guide to problem solving or a guaranteed set of solutions to your Oracle Forms environment.

Be methodical

Do not immediately leap to the area you believe to be the cause based on a hunch, or a guess - make sure you eliminate the other possibilities first. An easy trap to fall into is that of spending long periods of time trying to find evidence to support your theory, rather than concentrating on what the evidence shows. Do not overlook the trivial or the obvious.

Divide the problem into sections

- Chop the problem into manageable sections - this helps eliminate whole areas from investigation. As you investigate an area and satisfy yourself that the problem does not lie there, you can proceed to the next section. An approach to diagnosing a problem that is often successful is to reduce it to its essential parts. This will be important if you need to discuss the problem with Oracle Support Services to obtain a solution.
- Define what happens, when it happens, how often it happens. Of equal importance is, understanding what does not happen, when it does not happen etc. For example, if a group of users in the same building all get the problem, and it always happens between 9 and 10am, it is just as important to know that it never reproduces in another building, or after 10pm. Perhaps the users only use a particular Form between 9 and 10, or the load on the system is highest between 9 and 10am.

Read the error messages

It sounds obvious, but often the solution information is within the error text. This document will help you understand the error messages, and help identify what action to take.

Make sure you can reproduce the problem, if possible

If you can reproduce the problem yourself, you may notice some behavior that the end user never spotted - perhaps it had always happened, so they simply assumed it was meant to happen. If you can reproduce the problem then you have already started the first step to resolve it.

Make sure you understand the tools you are trying to use

If you decide to use a diagnostic tool, make sure you know how to use it, and how to interpret the data it produces. Time spent in investigating the usage of a tool before the problem happens is time well invested. Make time to learn the tool as well.

Need More Help?

If this *Troubleshooting Oracle Forms Services* appendix does not solve the problem you encountered, try looking for a solution on My Oracle Support, see https://support.oracle.com/ (formerly Oracle *MetaLink*). You can also raise a service request, if you are unable to find a solution for your problem. See Release Notes for Oracle Forms and Reports.

Part VIII
Forms References

This part details reference information such as client configurations, configuration files, error messages, configuration parameters, and environment variables.

Specifically, this part contains the following chapters:

- Configuration Files
- Environment Variables
- Web Configuration Parameters
- Font, Image, and Style Mapping
- Error Messages
- Forms Java EE Application Deployment Descriptors

Configuration Files

The table in this section lists the default locations of Forms configuration files on UNIX. The location of these files in Windows is similar. Samples and usage of specific Forms configuration files is also provided.

> **Note:**
>
> Where indicated, the location of files are given relative to the `DOMAIN_HOME` directory. Forward slashes should be replaced by back slashes on Windows. For information about terminology used such as Middleware home, Oracle home, Oracle instance, and so on, see Starting and Stopping Oracle Fusion Middleware in *Administering Oracle Fusion Middleware*.

Table 63 List of Files and their Locations in Release

File Name	Location in this Release	Description
`formsweb.cfg`	`$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config`	`formsweb.cfg` is used to set applet and servlet parameters used at runtime. Applet parameters in this configuration file are defined in the Forms template files located in `$FORMS_INSTANCE/server`. To edit the files, see Configuring Forms Services.
`default.env`	`$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config`	`default.env` is used to set environment variables used at runtime. To edit the files, see Managing Environment Variables.

Table 63 (Cont.) List of Files and their Locations in Release

File Name	Location in this Release	Description
base.htm	$FORMS_INSTANCE/server	When you generate the HTML page that is used to start an Oracle Forms application, base.htm is used as a template by the Forms servlet. To change the baseHTML file, create your own version and reference it from the formsweb.cfg file by changing the appropriate settings.
basejpi.htm	$FORMS_INSTANCE/server	When you generate the HTML page that is used to start an Oracle Forms application, basejpi.htm is used as a template by the Forms servlet. To change the baseHTML file, create your own version and reference it from the formsweb.cfg file by changing the appropriate settings.
webutilbase.htm	$FORMS_INSTANCE/server	This is the default baseHTML file for running a WebUtil enabled form using a generic APPLET tag.
webutiljpi.htm	$FORMS_INSTANCE/server	This is the default baseHTML file for running a WebUtil enabled form using the Java Plugin. For example, this file can be used when running a WebUtil enabled form with Firefox on UNIX.
ftrace.cfg	$FORMS_INSTANCE/server	ftrace.cfg file is used to configure the various configure parameters used for diagnostics, see Enable and Configure Forms Trace. It is located at $FORMS_INSTANCE/server/.
web.xml	$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_14.1.2/*<random_string>*/war/WEB-INF	For more information, see web.xml.
weblogic.xml	$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_14.1.2/*<random_string>*/war/WEB-INF	weblogic.xml is the web application deployment descriptor file.
forms.conf	$DOMAIN_HOME/fmwconfig/components/OHS/instances/*<OHS INSTANCE NAME>*/moduleconf	forms.conf is the Oracle HTTP listener configuration file for Oracle Forms Services. It includes Forms Services related directives, like Forms WebLogic Managed Server handler mappings. To configure Oracle HTTP Server for use with Oracle Forms, see: • Enabling Oracle HTTP Server with Oracle Forms Services • About Editing forms.conf • Configuring OHS

Table 63   (Cont.) List of Files and their Locations in Release

File Name	Location in this Release	Description
jvmcontroller.cfg	$FORMS_INSTANCE/tools/jvm/	A Forms application can be configured to use a specific JVM controller using the jvmcontroller parameter. This parameter is specified in formsweb.cfg. The parameters that are used by the JVM controller are specified in the JVM controller's configuration file, jvmcontrollers.cfg, see Managing JVM Pooling from Fusion Middleware Control.
webutil.cfg	$FORMS_INSTANCE/server	webutil.cfg file is one of the files used to configure WebUtil at run time. This file provides all of the configuration settings for WebUtil, including: • Logging Options • Installation Options • File Upload and Download Options • Server Side Logging Options for logging errors and log messages
Registry.dat	$DOMAIN_HOME/config/fmwconfig/servers/WLS_FORMS/applications/formsapp_14.1.2/config/oracle/forms/registry	Registry.dat enables you to change the default font, font mappings, and icons that Forms Services uses. For more information, see Deploying Fonts, Icons, and Images.
base.jnlp	$FORMS_INSTANCE/server	When you generate the JNLP that is used to start an Oracle Forms application with Java Web Start, base.jnlp is used as a template by the Forms servlet. To change the template file, create your own version and reference it from the formsweb.cfg file by changing the appropriate settings.
basejpi_jnlp.htm	$FORMS_INSTANCE/server	When you generate the HTML page that is used to start an Oracle Forms application, basejpi_jnlp.htm is used as a template by the Forms servlet. To change the template file, create your own version and reference it from the formsweb.cfg file by changing the appropriate settings.
basesaa.txt	$FORMS_INSTANCE/server	When you generate the text page that is used to start an Oracle Forms application running with Forms Standalone Launcher, basesaa.txt is used as a template by the Forms servlet. To change the template file, create your own version and reference it from the formsweb.cfg file by changing the appropriate settings.

Table 63 (Cont.) List of Files and their Locations in Release

File Name	Location in this Release	Description
webutil.jnlp	$FORMS_INSTANCE/server	This is the default basejnlp file for running a WebUtil enabled form with Java Web Start or Embedded JNLP.
webutilsaa.txt	$FORMS_INSTANCE/server	This is the default basesaa file for running a WebUtil enabled form with the Forms Standalone Launcher (FSAL).
extensions.jnlp	$ORACLE_HOME/forms/java	extensions.jnlp is used when configured to run applications using Java Web Start. When using Java Web Start, any Java JAR files other than frmall.jar or frmwebutil.jar must be listed in this file. Example syntax for adding entries is included within the file. This file is not managed by Fusion Middleware Control and therefore must be manually edited in a text editor. Changes to this file will require the restarting of the Forms Managed Server.

web.xml

The web.xml file is the web application deployment descriptor file for forms Java EE application. Advanced users might want to edit this file to run Oracle Forms using static HTML pages (rather than the Forms servlet).

When Oracle Forms applications are run using a method other than the Forms servlet (for example, static HTML pages or JSPs), parameter settings in the formsweb.cfg file are not used. You may therefore need to define servlet parameters for the Listener Servlet, such as envFile (specifying the current FORMS_PATH for the Forms runtime processes and other application specific environment settings to be used).

Such changes can be inserted into web.xml or plan.xml. See Modifying of Forms J2EE Application Deployment Descriptors.

The following table describes two servlet mappings.

Table 64 web.xml Servlet Mappings

URL Path	Type	Maps to	Purpose
/forms/frmservlet	Servlet mount point	Forms servlet	Generate HTML page to run a form
/forms/lservlet	Servlet mount point	Forms Listener servlet	Handles message traffic from the Forms applet

Environment Variables

This chapter describes Forms environment variables associated with configuration.

These notes apply to the following table:

- Required: An environment variable is required if Forms requires a non-null value in order to function correctly.
- Valid values: In a few cases, these are listed explicitly, but in most cases they can be inferred from the description.
  If the description implies that the value is Boolean (for example, it states Specifies whether), the valid values are TRUE, True, true, T, t, YES, Yes, yes, Y, y, 1, or FALSE, False, false, F, f, NO, No, no N, n, 0, (case-sensitive), unless indicated otherwise. Numeric values should be integers specified as decimal numbers, optionally followed by K, M, G, or T [powers of 1024, not 1000], unless indicated otherwise. The elements in lists of files or directories should be separated by ':' [Unix] or ';' [Windows] unless indicated otherwise.
- Default: If specified, this is the non-null value that produces the same behavior as not specifying a value. If omitted, the implication is that the default value is null.

Table 65 Environment Variables

Environment Variable	Applies to	Description
`CA_GPREFS`	Builder	Specifies an alternative path for the Form Builder global preferences file. Applies to Microsoft Windows only.
`CA_UPREFS`	Builder	Specifies an alternative path for the Form Builder user preferences file. Applies to Microsoft Windows only.
`CLASSPATH`	Runtime, Builder	The Java class path: a list of directories and/or JAR files. Required (for the Forms debugger and for forms using imported Java).
`COMPONENT_CONFIG_PATH`	Runtime	Used for Reports integration. Value should point to Reports Tools component path, like this: `<DOMAIN_HOME>\config\fmwconfig\components\ReportsToolsComponent\<REPORTS TOOLS INSTANCE NAME>`
`DE_PREFS_TABSIZE`	Builder	Specifies the number of spaces to be used when tab is pressed while in the Form Builder code editor. Default: 2
`DE_VARIABLESHARE`	Runtime	Specifies whether Forms should free variables shared between Program Units. Default: 0
`FORMS_ABTERM_CLEANUP`	Runtime	Specifies whether to enable normal Forms cleanup after an operating system exception in the runtime process. This consists of freeing various resources (for example, database connections) which the Forms runtime process had allocated. Setting this environment variable to `FALSE` bypasses the normal cleanup. Normally, this variable should not be set unless requested by Oracle Support. It might be appropriate to set it to `FALSE` if the Forms cleanup logic appears to be looping. Setting this environment variable will have no effect if the `FORMS_CATCHTERM` environment variable is set to `FALSE`. Default: `TRUE`
`FORMS_ALLOW_DB_CONNECT_STRING`	Runtime, Builder	Specifies whether the database to be logged into can be specified as a full connect string (the right-hand side of an entry in `tnsnames.ora`). If `FALSE` (the default) only a short tns name (the left-hand side of an entry in `tnsnames.ora`) can be specified. The connect string must not contain a value of `BEQ` for `PROTOCOL` or `PRO`, or any value for `COMMAND` or `CMD`. Default: `FALSE`

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_ALLOW_JAVASCRIPT_EVENTS	Runtime	Specifies whether Java Script events should be processed or ignored. If the applications run in this environment do not knowingly use Java Script integration, this setting should be set to FALSE. Default: TRUE
FORMS_API_COMPILE_ALL	JDAPI	Forces the full PL/SQL compilation of modules before generating the executable (FMX, MMX, PLX). Default: FALSE
FORMS_API_TK_BYPASS	JDAPI	Specifies whether to bypass the Forms internal Toolkit (TK) when initializing the JDAPI. If using JDAPI on a Unix/Linux platform and DISPLAY is not set or accessible, set this variable to TRUE. Default: FALSE
FORMS_AQ_PORT_RANGE	Runtime	Specifies a port, a set of ports, or a range of ports needed for Forms to integrate with DB Advanced Queuing. The value can be a single port number, a comma-separated list of ports, a range of ports (for example, 2000-2100), or a combination of ports and ranges. If no value is provided, an available port is assigned at random.
FORMS_ASSUME_MULTI_MONITOR	Builder	Specifies whether or not the Form Builder should retain window position information upon exiting. When set to TRUE, a window positioned on a secondary monitory when exiting will have its position on that monitor retained. If the secondary monitor is expected to not be available in the future, setting this variable to FALSE will cause window positions to not be retained and restarting the Builder will occur on the main display. Default: TRUE
FORMS_AVOID_PU_WARNING	JDAPI	Prevents the displaying of an error dialog that suggests there are duplicate program units (PU) in a module when actually the duplication is the result of having a PU spec and body of the same name.
FORMS_BI_SERVER	Runtime	Specifies the location of the BI Publisher Service Location in the format <protocol>://<server>:<port>. If the Service Location specified in a report object is a relative URL, the value of FORMS_BI_SERVER will be prepended to it.
FORMS_BLOCK_HOST_CMD	Runtime	Specifies whether to block application calls to the HOST built-in. Valid values: • NONE (or leaving blank or unset): Indicates that the HOST command should not be blocked. • SILENT: Indicates that the HOST command should be ignored. All code that follows will continue to execute. • CATCH: Indicates that the HOST command should be blocked and raise FORM_TRIGGER_FAILURE to stop all processing.
FORMS_BUILDER_CLASSPATH	Builder	Specifies the Java CLASSPATH used by the Builder. This is used in conjunction with the native Java CLASSPATH thereby allowing the number of JAR and class file references to be greatly extended. The maximum length of CLASSPATH and FORMS_BUILDER_CLASSPATH may vary based on the platform.

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_BUILDER_FSAL_LOGGING	Builder	Specifies whether to enable logging and at what level when using the Forms Standalone Launcher to run forms modules directly from the Builder. When set to a valid integer, log files are created in the operating system's configured TEMP directory. The file name for the created log file is prefixed with "fsal_". Any invalid value will assume 0. When this variable is no set (NULL), logging is disabled. This functionality is based on the showDetails argument available when running the Forms Standalone Launcher tool on the command line. Valid values: • 0: Indicates that only minimal information should be logged • 1: Indicates that location details from where resources like JAR files are loaded should be logged • 2: Indicates that details related to SSL/TLS certificates should be logged. This detail is not available when the FSAL TrustStore is not used. • 99: Indicates that all available details should be logged Other values are reserved for future use.
FORMS_BUILDER_MUTE_WARNINGS	Builder	Specifies whether to mute Builder warnings so they are not presented to the user in a dialog. Such warnings may occur, for example, while attempting to compile or generate or when a property value is validated. Errors are not affected by this flag and will continue to be presented. Warnings are still reported in .err files, which are created when generating executable modules. Default: 0
FORMS_BUILDER_SILENT_CONVERT	Builder	Specifies whether to suppress the XML Converter **Success/Failure** dialog when the Builder preference setting, **Create XML After Saving**, is enabled and the user attempts to save a module. Default: FALSE
FORMS_C_STACK_TRACE_TO_FILE	Runtime	Specifies where a stack trace is written in the event of an operating system exception in the runtime process. Setting this environment variable to TRUE (or leaving it unset) causes the stack trace to be written to a file (identified by FRM-93658 message in the Oracle Diagnostic Log). Setting this environment variable to FALSE causes the stack trace to be written to the Oracle Diagnostic Log. Setting this environment variable will have no effect if the FORMS_CATCHTERM environment variable is set to FALSE. Default: TRUE

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_CATCHTERM	Runtime	Specifies whether to enable normal Forms handling of an operating system exception in the runtime process. This consists of two parts: 1. Writing a stack trace to a file or to the Oracle Diagnostic Log. (See the FORMS_C_STACK_TRACE_TO_FILE environment variable). 2. Freeing various resources (for example, database connections) which the Forms runtime process had allocated, unless the FORMS_ABTERM_CLEANUP environment variable was set to FALSE. Setting this environment variable to FALSE bypasses the normal Forms exception handling. Normally, this variable should not be set unless requested by Oracle Support. **Note:** The use of FORMS_CATCHTERM is not recommended. It has been retained for backward compatibility only. Refer to FORMS_ABTERM_CLEANUP instead. Default: TRUE
FORMS_CHANGE_PASSWORD_HINT	Runtime	Specifies the text that should be displayed in the default logon dialog box. The total number of characters cannot exceed 255.
FORMS_CHECK_WEB_LISTENER	Builder	Specifies whether the Builder should verify that the server referenced in the **Application Server URL** Preference setting can be reached and is responding, before attempting to run the current form when the **RunForm** button is pressed. Default: TRUE
FORMS_COMPUTED_RGFS_DIVIDEND	Runtime	If the Record Group Fetch Size property for a record group is set 0, the effective value is given by the environment variables FORMS_COMPUTED_RGFS_DIVIDEND, FORMS_MIN_COMPUTED_RGFS, and FORMS_MAX_COMPUTED_RGFS: FORMS_COMPUTED_RGFS_DIVIDEND divided by the record group record size, but no less than FORMS_MIN_COMPUTED_RGFS and no greater than FORMS_MAX_COMPUTED_RGFS. Default: 512K

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_DATETIME_LOCAL_TZ	Runtime	Specifies the end-user local time zone and defaults to the time zone of the Java client. Oracle Forms uses time zone data to support the ADJUST_TZ built-in and to adjust the date/time displayed in DATETIME items. Values loaded from the database or set by application logic are interpreted as being in the *server* time zone, whereas values displayed in DATETIME items are converted from the server time zone to the end-user *local* time zone. The default values for FORMS_DATETIME_SERVER_TZ and FORMS_DATETIME_LOCAL_TZ are usually different. **Note:** When upgrading from Forms 11*g* (11.1.2) or older releases, add FORMS_DATETIME_LOCAL_TZ=GMT to the environment variable configuration file (such as, default.env) to ensure that the end-user local time zone is identical to the server time zone.
FORMS_DATETIME_SERVER_TZ	Runtime	Specifies the server time zone and defaults to GMT. Oracle Forms uses time zone data to support the ADJUST_TZ built-in and to adjust the date/time displayed in DATETIME items. Values loaded from the database or set by application logic are interpreted as being in the *server* time zone, whereas values displayed in DATETIME items are converted from the server time zone to the end-user *local* time zone. The default values for FORMS_DATETIME_SERVER_TZ and FORMS_DATETIME_LOCAL_TZ are usually different.
FORMS_DB_IDLE_TIME	Runtime	Specifies how long, in seconds the connection to the database can be idle/unused before raising the SYSTEM_DB_IDLE System Event.
FORMS_DBALIAS_SPECIAL_CHARS	Runtime	Specifies whether the database to be logged into can be specified as a TNS name containing ":" (used for EZCONNECT). Default: 1
FORMS_DECIMAL_PREFIX	Runtime	
FORMS_DEFAULTFONT	Builder	Specifies the default font that should be used by the Builder Layout Editor when creating new objects. The syntax for this setting is: *<font name>.<size>.<style>.<weight>*. Example value: arial.10.plain.bold
FORMS_DETECT_DBJAVA	Builder	Specifies whether the Builder should check for the presence of JAVA in DB. Setting to 0 (FALSE) causes the Builder to not perform the check. Setting to 1 (TRUE) causes the Builder to determine if Java exists in the DB. Default: 1
FORMS_DISABLE_UNPAD_LOV	Runtime	
FORMS_DISCONNECT_AWARE	Builder	Specifies whether or not the Form Builder should attempt to detect if the database connection has been lost. Default: TRUE

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_EDITOR`	Builder	Specifies the desired editor to use when the **Use System Editor** Builder Preference is enabled and the user attempts to open a multi-line property value. If no value is provided, the system default, such as Notepad on Microsoft Windows, is used. **Note:** Supports only single document editors. Editors displaying more than one document at a time cannot be used.
`FORMS_EXTENDED_STRING`	Runtime	Specifies whether columns referenced in a Record Group may contain more than 4000 characters. This is only possible if the database has this setting: `MAX_STRING_SIZE=EXTENDED`. Default: `TRUE`
`FORMS_FORCE_ENCRYPT_HANDSHAKE`	Runtime	Specifies whether to force an encryption *handshake* round trip to occur during application startup. If set to `FALSE`, the handshake is suppressed if message obfuscation is suppressed (which will happen if `FORMS_MESSAGE_ENCRYPTION` is set to `FALSE` or if it's unset and SSL/TLS [https] is being used). Specifying a value of `FALSE` reduces network traffic. Specifying (or defaulting to) a value of `TRUE` provides compatibility with the behavior of prior releases. This may be necessary for certain testing tools or third-party products that use Oracle Forms. Default: `TRUE` See also `FORMS_MESSAGE_ENCRYPTION`.
`FORMS_FORCE_MENU_MNEMONICS`	Runtime, Compiler, JDAPI	Specifies whether to automatically use the first character of labels on a custom menu for the access key if no other has been specified. If `FALSE`, the first letter is not used as the access key if no other access key is provided. If creating menu labels at design-time, this variable must be set in the compiler/generator environment before creating a the executable (MMX). If labels are set dynamically at runtime, then this must be set in the runtime environment configuration. Default: `TRUE`
`FORMS_FORCE_ROWBANDING`	Runtime	Specifies whether or not row banding is applied when the underlying canvas has a background color set by the application developer. If this variable is set to `FALSE`, row banding is not applied and the background color shows through. Row banding is enabled with the design-time Block and Item level property, `Row Banding Frequency`. Default: `FALSE`

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_FRMBLD_STARTS_JVM	Builder	Specifies whether the Builder starts a Java Virtual Machine (JVM) when it starts. A JVM is required to use the Builder Help, Java Importer, and Debugger. If set to FALSE, Builder startup performance is improved and the memory footprint is reduced. The JVM will be started on demand, for example, when attempting to open the Builder Help. Once the JVM starts, it remains running through the life of the Builder session. When set to FALSE, the Debugger is not available. Default: TRUE
FORMS_FSAL_OBR_JAVA	Builder	Specifies an alternate Java to be used by the Forms Standalone Launcher (FSAL) when started from the Builder. By default, when FSAL is launched from the Builder it uses Java installed in the Oracle Home. Setting this variable to an alternate Java Home allows FSAL to use a different Java version than the one used by the Builder. **Note:** Be sure to check the Fusion Middleware Certification Matrix for certified Java versions that can be used with FSAL.
FORMS_HIDE_LOGIN_RESET	Runtime	Specifies whether to hide the **Change Password** dialog when the database password has expired. If set to TRUE and an application attempts to connect to a database using an expired password, the application throws an error and then exits. If set to FALSE, the user is presented with the **Change Password** dialog and must provide updated credentials. Setting to FALSE is typically only necessary in cases where the user is unaware that database credentials are needed to run the application, for example when using single sign-on (SSO). Default: FALSE
FORMS_HIDE_OBR_PARAMS	Builder	Specifies whether to hide URL parameters when using "OneButtonRun" (the **RunForm** button) to launch a form from the Builder. Default: FALSE
FORMS_HIDE_PASSWORD	Runtime	Specifies whether the Forms built-in GET_APPLICATION_PROPERTY can retrieve the current user's password. Default: FALSE

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_HTTP_DIAGNOSTIC_LEVEL	Runtime	Specifies the initial value of the HTTP_DIAGNOSTIC_LEVEL application property. Indicates how much diagnostic information related to REST service HTTP requests and responses should be written to the Oracle Diagnostic Log (ODL). The HTTP_DIAGNOSTIC_LEVEL can be overridden programmatically for the current session/application, using the SET_APPLICATION_PROPERTY built-in within the application code. However, doing this is not recommended for production applications. Valid values (0 - 5): • 0: If a request to the REST service returns an unexpected HTTP status (not listed in the 'accept' parameter passed to FHTTP.ISSUE_REQUEST, and not 200 or any other status that is always considered acceptable), then the HTTP response is written to ODL. (The response is presumed to be an error message from the REST service.) No other diagnostic information about the HTTP requests or responses are logged. • 1: Like 0, except that in addition, FJSON.ANALYZE_EXCEPTION (see Form Builder Help) may log coarse-grained information about the scalar value obtained from the HTTP response that appears to have triggered the exception—whether it was null, and if not, whether it was a VARCHAR2, NUMBER, or BOOLEAN. For a VARCHAR2, the number of characters and bytes is logged. For a NUMBER, a range is logged: < -2G, >= -2G, < 0, : 0, > 0, < 2G, or >= 2G. • 2: Like 1, except that FJSON.ANALYZE_EXCEPTION or POPULATE_GROUP_FROM_JSON (see Form Builder Help) may log the actual value that appears to have triggered the exception. • 3: Like 2, except that in addition, the entire HTTP request is written to a file in the event of an error—for example a syntax error or unexpected content. The name of the file is written to ODL. • 4: Like 3, except that in addition, the entire HTTP response is also written to a file in the event of an error. The name of the file is written to ODL. • 5: Like 2, except that in addition, all HTTP requests and responses to and from REST services are written to files (whose names are written to ODL). This value should not be specified for production systems, but may occasionally be useful during development of a Forms application. SET_APPLICATION_PROPERTY(HTTP_DIAGNOSTIC_LEVEL, 5) and a subsequent SET_APPLICATION_PROPERTY(HTTP_DIAGNOSTIC_LEVEL, previous_level) could be used to restrict the files written for requests and responses to those that are relevant to the problem being debugged. Default: 1
FORMS_HTTP_PROXY_HOST	Runtime, Builder	Specifies the HTTP-enabled proxy host (if any) used for calls to Oracle BI-Publisher, or for calls to a remote server [a server other than the one where Forms is running] from server-side Java (Imported Java) in a form. You may also need to set FORMS_HTTP_PROXY_PORT. If you are using the JVM Controller, see Network Proxies and Java Calls Using JVM Controller and the Java equivalent http.proxyHost.
FORMS_HTTP_PROXY_PORT	Runtime, Builder	Used in conjunction with FORMS_HTTP_PROXY_HOST.

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_HTTPS_HOST_VERIFICATION	Builder	Specifies whether to perform host name verification when making SSL/TLS requests. If disabled (0), host name verification is not performed. **Note:** Disabling host name verification is generally not recommended.
FORMS_HTTPS_PROXY_HOST	Runtime, Builder	Specifies the HTTPS-enabled proxy host (if any) used for calls to Oracle BI-Publisher, or for calls to a remote server [a server other than the one where Forms is running] from server-side Java (Imported Java) in a form. You may also need to set FORMS_HTTPS_PROXY_PORT. If you are using the JVM Controller, see Network Proxies and Java Calls Using JVM Controller and the Java equivalent https.proxyHost.
FORMS_HTTPS_PROXY_PORT	Runtime, Builder	Used in conjunction with FORMS_HTTPS_PROXY_HOST.
FORMS_HTTPS_TRUSTSTORE	Builder	Specifies the path and file name of the Java trust store to use when making SSL/TLS requests. For example, this could be used by the REST Package Designer (RPD). When not set, the default Java trust store is used.
FORMS_HTTPS_TRUSTSTORE_PWD	Builder	Specifies the Java trust store password. When not set, the default Java trust store password is used.
FORMS_INSTANCE	Runtime	The directory that contains all configuration files, repositories, log files, deployed applications, and temporary files. Required.
FORMS_JSON_NEW_MEMORY_LIMIT	Runtime	Specifies the maximum amount of memory (in bytes) allowed for JSON parse trees produced by built-ins other than FHTTP.ISSUE_REQUEST (see Form Builder Help). If the limit is exceeded, an FJSON.GLOBAL_ALLOCATION (see Form Builder Help) exception is raised. There will typically be a partially built parse tree that is in an inconsistent state, and further requests to create new parse trees will usually fail. So if the application chooses to handle this exception, the only realistic thing it can do is to issue FJSON.FREE_ALL (see Form Builder Help). Any end user updates which were recorded in JSON parse trees will be lost. Valid values: Must be >= 1M Default: 4G
FORMS_JSON_PARSE_MEMORY_LIMIT	Runtime	Specifies the maximum amount of memory (in bytes) allowed for JSON parse trees produced by a single call to FHTTP.ISSUE_REQUEST (see Form Builder Help). If the limit is exceeded, an FJSON.PARSE_ALLOCATION (see Form Builder Help) exception is raised and the memory allocated for the request is released. Valid values: Must be >= 1M Default: 4G

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_LIMIT_EXTERNAL_MESSAGES	Runtime	Unix only. If set to '1', "external" messages [messages from components other than Oracle Forms] written to stdout or stderr are suppressed. This may occasionally be necessary to prevent disk file systems from filling up. However, external messages are sometimes useful in diagnosing problems, for example, with loading executables. Leaving this environment variable unset causes external messages to be written to the Oracle Diagnostic Log, in the body of message FRM-93546.
FORMS_LOG_MODULE_OPEN	Runtime	Specifies whether to log opened modules to the Oracle Diagnostic Log. If set to TRUE, information such as path and module name is logged for any FMX, MMX, and PLL/PLX modules opened by the application. Opened module information is logged at the notification level using message FRM-91940. Default: False
FORMS_LOGIN_DIALOG_BLANK	Runtime, Builder	Specifies whether the user name and database name appears in the logon dialog when presented after a failed logon attempt. If set to 1 (TRUE) when used with the Builder, the database connection name does not appear in the Builder message bar. Default: FALSE
FORMS_LOGON_HINT	Runtime	Specifies the text that should be displayed in the default logon dialog box. The total number of characters cannot exceed 255.
FORMS_LOV_INITIAL	Runtime	Used when the Interaction Mode form property is set to Non-Blocking. Specifies the time, in milliseconds, before the Cancel Query window appears after a query is initiated. Default: 1000 Related topics: • FORMS_LOV_MINIMUM • FORMS_LOV_WEIGHT • FORMS_NONBLOCKING_SLEEP
FORMS_LOV_MINIMUM	Runtime	Used when the Interaction Mode form property is set to Non-Blocking. In conjunction with FORMS_LOV_WEIGHT, specifies the time, in milliseconds, between pollings of the Cancel Query window, to determine whether to cancel a query. Default: 500 Related topics: • FORMS_LOV_INITIAL • FORMS_LOV_WEIGHT • FORMS_NONBLOCKING_SLEEP
FORMS_LOV_WEIGHT	Runtime	Used when the Interaction Mode form property is set to Non-Blocking, in conjunction with FORMS_LOV_MINIMUM, to specify how to prolong the time between Cancel Query pollings. FORMS_LOV_WEIGHT is a ratio which has an effect in proportion to the average speed of a round trip between the Java client and Forms server. Therefore, the slower the round trips are, the longer a particular value of FORMS_LOV_WEIGHT will delay making another time-consuming round trip to the Cancel Query window. Default: 0 Related topics: • FORMS_LOV_INITIAL • FORMS_LOV_MINIMUM • FORMS_NONBLOCKING_SLEEP

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_MAP_GIF_IMAGE_TO_NATIVE	Runtime	Specifies if image items containing GIF images should bypass the Forms Multimedia processing layer. Bypassing the Multimedia processing layer will deliver uncompressed images to the user interface and will typically exhibit superior image quality when Native (a new Builder property for image items) is specified. However, image loading performance may be degraded. If the behavior of Native image items is desired for image items that specify an Image Format of GIF, this can be achieved without modifying or recompiling fmb files. Set the value of this environment variable to TRUE (or 1) to enable this feature. Set to FALSE (or 0) to disable. Default: FALSE
FORMS_MAP_JFIF_IMAGE_TO_NATIVE	Runtime	Specifies if image items containing JFIF/JPEG/JPG images should bypass the Forms Multimedia processing layer. Bypassing the Multimedia processing layer will deliver uncompressed images to the user interface and will typically exhibit superior image quality when Native (a new Builder property for image items) is specified. However, image loading performance may be degraded. If the behavior of Native image items is desired for image items that specify an Image Format of JFIF/JPEG/JPG, this can be achieved without modifying or recompiling fmb files. Set the value of this environment variable to TRUE (or 1) to enable this feature. Set to FALSE (or 0) to disable. Default: FALSE
FORMS_MAX_COMPUTED_RGFS	Runtime	If the Record Group Fetch Size property for a record group is set 0, the effective value is given by the environment variables FORMS_COMPUTED_RGFS_DIVIDEND, FORMS_MIN_COMPUTED_RGFS, and FORMS_MAX_COMPUTED_RGFS: FORMS_COMPUTED_RGFS_DIVIDEND divided by the record group record size, but no less than FORMS_MIN_COMPUTED_RGFS and no greater than FORMS_MAX_COMPUTED_RGFS. Default: 100
FORMS_MAX_JVM_STACKSIZE	Runtime	The thread stack size passed to the JVM via the -Xss flag, if the runtime process needs spawn a JVM, for example, for certain Webutil operations, or for processing an expired password during logon when single sign-on is in effect. Specifying a large value may be necessary when a user exit allocates a large amount of memory. Valid values: A decimal number optionally followed by k, K, m, or M.
FORMS_MESSAGE_ENCRYPTION	Runtime	Specifies whether to apply proprietary obfuscation to Forms messages between the client and the server. Default: FALSE when using SSL or TLS [https]; TRUE otherwise [http] See also FORMS_FORCE_ENCRYPT_HANDSHAKE.
FORMS_MIN_COMPUTED_RGFS	Runtime	If the Record Group Fetch Size property for a record group is set 0, the effective value is given by the environment variables FORMS_COMPUTED_RGFS_DIVIDEND, FORMS_MIN_COMPUTED_RGFS, and FORMS_MAX_COMPUTED_RGFS: FORMS_COMPUTED_RGFS_DIVIDEND divided by the record group record size, but no less than FORMS_MIN_COMPUTED_RGFS and no greater than FORMS_MAX_COMPUTED_RGFS. Default: 20

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_MMAP	Runtime	Specifies whether to use memory mapped file I/O when reading forms or menus (fmx or mmx files). A value of TRUE reduces CPU and I/O time, and reduces the amount of memory required on a system when more than one user is running the same form. However, a value of TRUE has the side effect of preventing an fmx or mmx file from being rewritten while the form or menu is running, which may be undesirable in a development environment. Default: TRUE
FORMS_MODULE_PATH	Runtime	Setting this environment variable to a non-empty value restricts the directories from which Forms applications may be launched. The significant effects are as follows: If the initial form specifies a path, it must appear either in the value of FORMS_PATH, ORACLE_PATH, FORMS_MODULE_PATH, or in a subdirectory (without any references to the parent directory) of a path in the value of FORMS_MODULE_PATH. If no such match is found, an error message FRM-40010: Cannot read form is displayed. If a form, menu, or PL/SQL library (.fmx, .mmx, .plx, or .pll file) is specified without a path (either as the initial form or in a CALL_FORM, NEW_FORM, or OPEN_FORM statement), the current working directory is not searched. The only directories searched are those specified by the FORMS_PATH and ORACLE_PATH environment variables. Value specified in the default.env file that is shipped with the Forms product: $FORMS_PATH (Unix) or %FORMS_PATH% (Windows).
FORMS_MULTIPLE_FILE_SELECT	Builder	Specifies whether to allow the user to select multiple files in the **File Open** dialog when opening modules. Default: TRUE
FORMS_NONBLOCKING_SLEEP	Runtime	Specifies a sleep interval, in milliseconds. Used when the Interaction Mode form property is set to Non-Blocking, and a query is initiated for a datablock, a record group, or a non-longlist LOV, or when such a query is resumed after displaying or polling the Cancel Query window. If FORMS_NONBLOCKING_SLEEP were set to 0, Oracle Forms would continuously check to see if the query was still executing, until it completed or until the time specified by FORMS_LOV_INITIAL or FORMS_LOV_MINIMUM an FORMS_LOV_WEIGHT had elapsed. However, this would consume excessive CPU time. To reduce the CPU time, Oracle Forms will sleep for the interval specified by FORMS_NONBLOCKING_SLEEP between performing checks to see if the query is still executing. The larger the value specified for FORMS_NONBLOCKING_SLEEP, the bigger the reduction in CPU time. However, if the value significantly exceeds the typical elapsed time required to fetch a set of records, queries will take noticeably longer to complete. Also, the value should not exceed to value of FORMS_LOV_INITIAL or FORMS_LOV_MINIMUM. Default: 100 Related topics: • FORMS_LOV_INITIAL • FORMS_LOV_MINIMUM • FORMS_LOV_WEIGHT

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_OBFUSCATE_CONCEALED_DATA	Runtime	Specifies whether to hide a show password button in the Forms logon dialog. Enabling this setting hides the show password button in the password field. Default: 0
FORMS_OBR_REMOVE_PATH	Builder	Specifies whether or not the full file system path is included in the URL when using the Builder's One Button Run feature. This is helpful when the Application Server URL is set to a remote node. Default: FALSE
FORMS_ON_MODAL_DETECT_IDLE	Runtime	When set to TRUE, the CLIENT_IDLE System Event executes immediately after the blocked condition is released and any previously executing triggers complete processing. This does not apply to custom Java Beans that expose modal Java dialogs or any other blocked condition not originating from native Forms. Default: FALSE
FORMS_ORDER_PKG_SPEC_BODY	Compiler	
FORMS_OVERRIDE_ENV	Runtime	
FORMS_PATH	Runtime, Builder, Compiler	FORMS_PATH and ORACLE_PATH specify the directories that Oracle Forms searches when looking for a form (fmx file), menu (mmx file) to run, or for a PL/SQL library (pll or plx file) to attach, when the file does not specify a path. The directories specified by FORMS_PATH are searched before the directories specified by ORACLE_PATH. Required.
FORMS_PLSQL_BHVR_COMMON_SQL	Builder, Compiler, JDAPI	Specifies whether Forms PL/SQL uses the common SQL parser in the RDBMS SQL engine for compiling SQL code, rather than the separate one built into PL/SQL that is used for compiling static SQL. Default: FALSE
FORMS_PLSQLV1_NAME_RESOLUTION	Runtime	
FORMS_POPUPMENU_SYSVARS	Runtime	Specifies if the system sets values for mouse-event system variables when running the popup menu. If the variable is set to TRUE, the system sets values for these variables. If it is set to FALSE (or not set at all), the system only sets values for these variables on mouse-triggers. The following Forms mouse related SYSTEM variables are populated as a result of enabling this environment variable: • SYSTEM.MOUSE_ITEM • SYSTEM.MOUSE_CANVAS • SYSTEM.MOUSE_RECORD • SYSTEM.MOUSE_OFFSET The following values are set to 0/null: • SYSTEM.MOUSE_X_POS • SYSTEM.MOUSE_Y_POSE • SYSTEM.MOUSE_BUTTON_PRESSED • SYSTEM.MOUSE_BUTTON_SHIFT_STATE Default: FALSE

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_PROXY_BYPASS	Runtime, Builder	Specifies a list of hosts that should not be accessed using a proxy. Hosts in the list are separated by the **|** character. An individual host can include the wildcard character *****. See FORMS_HTTP_PROXY_HOST. If you are using the JVM Controller, see Network Proxies and Java Calls Using JVM Controller and the Java equivalent http.nonProxyHosts.
FORMS_RECMGR_ACTIVE_RECS_DELTA	Runtime	This environment variable is intended for users with critical performance requirements. Most users can simply use the default value. FORMS_RECMGR_ACTIVE_RECS_RATIO and FORMS_RECMGR_ACTIVE_RECS_DELTA together specify the default number of records to be kept active in each data block (and potentially in each record group). See FORMS_RECMGR_ACTIVE_RECS_RATIO for details. Valid values: 3-1T (3 to 1,099,511,627,776 records) Default: 3

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_RECMGR_ACTIVE_RECS_RATIO	Runtime	This environment variable is intended for users with critical performance requirements. Most users can simply use the default value. FORMS_RECMGR_ACTIVE_RECS_RATIO and FORMS_RECMGR_ACTIVE_RECS_DELTA together specify the default number of records to be kept active within each data block (and potentially within each record group). *Active* records are in a format which allows direct access to each item instance or cell. The remaining records in a data block or record group are *archived*. These are in a packed format which requires less memory. Within each data block or record group, the active records are generally the most recently used records. Increasing the number of active records may improve performance (by reducing the frequency of conversions between active and archived format), but will increase memory usage. Conversely, decreasing the number of active records will decrease memory usage but may degrade performance. When FORMS_RECMGR_ARCHIVE is MAP_SWAP or MAP_TMPFILE, memory usage for archived versus active records in a data block (or record group) can be summarized as follows: • In both cases, virtual memory will be required for the actual data in all item instances in a data block (or cells in a record group). • In an archived record, approximately 4 bytes of control information will be required for each item instance (or cell). In an active record, approximately 24 bytes of control information and padding will be required for each item instance (or cell), plus 64 bytes per record. • Record data in archived records will typically exhibit much less fragmentation and much greater locality of reference than record data in active records, although this advantage will largely disappear after calling the SORT_BLOCK built-in procedure. Specifying WRITE_TMPFILE for FORMS_RECMGR_ARCHIVE will increase memory and CPU usage, as noted under FORMS_RECMGR_ARCHIVE. The default number of active records in a data block is (Records_Displayed * FORMS_RECMGR_ACTIVE_RECS_RATIO) + FORMS_RECMGR_ACTIVE_RECS_DELTA (truncated to an integer). If FORMS_RECMGR_ARCHIVE is MAP_SWAP or MAP_TMPFILE, and FORMS_RECMGR_USED_FOR_RECGRPS is TRUE, the number of active records in a record group is FORMS_RECMGR_ACTIVE_RECS_RATIO + FORMS_RECMGR_ACTIVE_RECS_DELTA (truncated to an integer). Otherwise, all records in a record group are in effect active (although their internal format differs from active records managed by the record manager). The default number of active records in a data block can be overridden by setting the Number of Records Buffered property for a data block in the builder. The number of active records can be overridden globally at runtime by setting the buffer_records runtime property to yes. This will set the number of active records in each data block to Records_Displayed + 3. This option is not recommended, but is supported for backward compatibility only. Specifying a very large value (for example 1T) for FORMS_RECMGR_ACTIVE_RECS_DELTA causes all records in data blocks or record groups to be active (except where the value is overridden by the Number of Records Buffered property for a data block), regardless of the value of FORMS_RECMGR_ACTIVE_RECS_RATIO).

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
		Note: When a data block or record group is being populated after being cleared, the number of active records may temporarily exceed the "target" number specified by `FORMS_RECMGR_ACTIVE_RECS_RATIO` and `FORMS_RECMGR_ACTIVE_RECS_DELTA` (or by the `Number of Records Buffered` property), if the records are relatively small. The creation of archived records is deferred until the number of records reaches a certain limit (based on the record size). If and when that limit is reached, the number of active records will be reduced to the target number. Valid values: 1.0-5.0. Fractional values are allowed. Default: `3.0`

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_RECMGR_ARCHIVE`	Runtime	Specifies where and how the Forms record manager stores archived records. For each data block and record group, Oracle Forms maintains a small number of *active* records in virtual memory. An active record is in a format which allows direct access to data values and control information for each item instance or cell in the record. The remaining records are *archived*. An archived record is in a packed format which does not allow direct access to item instances or cells. Archived records reside in the *archive store*. Valid values: • `MAP_SWAP` specifies that archived records are stored in virtual memory backed by swap space. • `MAP_TMPFILE` specifies that archived records are stored in virtual memory backed by temporary files (typically in the `/tmp` or `/var/tmp` directory). Each open data block or record group has its own temporary file. Specifying `MAP_SWAP` will usually provide superior performance, but specifying `MAP_TMPFILE` may be appropriate when an installation's real memory plus swap space is insufficient for archived records. Restriction: This option is not available on Windows systems. • `WRITE_TMPFILE` specifies that archived records are written to the backing file: a single temporary file (typically in the `/tmp` or `/var/tmp` directory on Unix systems, and in the `%TMP%` or `%TEMP%` directory on Windows). Specifying `MAP_SWAP` or `MAP_TMPFILE` will provide superior performance, and will usually reduce memory usage significantly (by roughly the ratio of maximum to average item instance data value size, in most cases, although typically less when a data block contains image items). Specifying `WRITE_TMPFILE` is appropriate on a Windows system when the installation's real memory plus swap space is insufficient for archived records. Restriction: This option does not support record groups. [It supports only data blocks.] If this option is specified, record groups are maintained using pre-release-12 technology, as occurs when `FORMS_RECMGR_USED_FOR_RECGRPS` is set to `FALSE`. If a form contains a record group whose effective record size (excluding LONG columns) exceeds 64Kb, a fatal error is reported when the form is loaded. Default: `MAP_SWAP` Related topics: There are other environment variables which can affect the behavior of the record manager. • When `FORMS_RECMGR_ARCHIVE` is set to `MAP_SWAP` or `MAP_TMPFILE`, see: – `FORMS_RECMGR_ARCHIVE_THRESHOLD` – `FORMS_RECMGR_RESERVE_SPACE` • When `FORMS_RECMGR_ARCHIVE` is set to `WRITE_TMPFILE`, refer to `FORMS_RECMGR_MAX_TMPFILE_SIZE`. In addition, there are other environment variables which can affect the performance of the record manager. These are intended for users with critical performance requirements. Most users can simply use the default values. • When `FORMS_RECMGR_ARCHIVE` is set to `MAP_SWAP` or `MAP_TMPFILE`, refer to:

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
		– FORMS_RECMGR_ACTIVE_RECS_RATIO – FORMS_RECMGR_ACTIVE_RECS_DELTA – FORMS_RECMGR_BLOCK_SIZE – FORMS_RECMGR_REPORT_RECGRP_MEM – FORMS_RECMGR_SLACK_MAX – FORMS_RECMGR_SLACK_RATIO – FORMS_RECMGR_USED_FOR_RECGRPS • When FORMS_RECMGR_ARCHIVE is set to MAP_TMPFILE, refer to FORMS_RECMGR_TMPPOOL_ENTRIES • When FORMS_RECMGR_ARCHIVE is set to WRITE_TMPFILE, the following are relevant: – FORMS_RECMGR_ACTIVE_RECS_DELTA – FORMS_RECMGR_ACTIVE_RECS_RATIO – FORMS_RECMGR_RETRY_INTERVAL – FORMS_RECMGR_SEGMENT_SIZE – FORMS_RECMGR_SEGPOOL_ENTRIES – FORMS_RECMGR_TMPFILE_DENSITY
FORMS_RECMGR_ARCHIVE_THRESHOLD	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to MAP_SWAP or MAP_TMPFILE. It specifies a threshold on the number of bytes in all O/S pages containing archived record segments. When the number of bytes exceeds the threshold, creation of records (for insert, query, or backup) will fail, at least until the next CLEAR_BLOCK operation (at which point creation of records will be allowed again if the number of bytes has fallen below the threshold). Smaller values will reduce the likelihood of the system encountering an out-of-memory condition, and will also reduce the likelihood of a Forms application being selected as the process to be terminated when such a condition does occur. The memory usage reported by message FRM-91992 may help to determine a reasonable value for this environment variable. It shouldn't be set to a value that would cause many Forms applications to exceed the threshold. [But bear in mind that some applications might be using more than their fair share, due for example, to omitting a WHERE clause from a large query.] Note that FRM-91992 will appear only if the log level (appended to the serverURL configuration parameter) is set to /sessionperf, /perf, or /debug. Valid values: Integers in the range 64K-2T. The specified value will be truncated to a multiple of the O/S page size. Default: 4G

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_RECMGR_BLOCK_SIZE	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVEis set to MAP_SWAP or MAP_TMPFILE. It's intended for users with critical performance requirements. Most users can simply use the default value. When FORMS_RECMGR_ARCHIVE is set to MAP_SWAP or MAP_TMPFILE, the record manager allocates virtual memory for a data block or record group in large blocks, from which space for archived records is suballocated. This environment variable specifies the size of such blocks. Larger values reduce CPU usage (because there are fewer system calls to allocate or free large blocks of memory) but consume more memory, at least in the case where FORMS_RECMGR_RESERVE_SPACE is set to TRUE. When FORMS_RECMGR_RESERVE_SPACE is set to FALSE, the additional memory consumed by larger values is very small, because real memory won't be allocated until data is written to a page, and disk space won't actually be reserved until a page is flushed to disk. Very large values might require some additional real memory for second or third-level page tables. Very large values also increase the likelihood of hitting per-process limits on the size of the data segment. Valid values: Integers in the range 32K-512K if FORMS_RECMGR_RESERVE_SPACE is set to TRUE, or 128K-2G if it's set to FALSE. The specified value will be truncated to a multiple of the O/S page size. Default: 64K if FORMS_RECMGR_RESERVE_SPACE is set to TRUE, or 2M if it's set to FALSE
FORMS_RECMGR_MAX_TMPFILE_SIZE	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to WRITE_TMPFILE. It specifies the maximum size, in bytes, of the record manager backing file. If creation of a record (for insert, query, or backup) would cause this limit be exceeded, the operation will fail. Other operations that would cause this limit be exceeded will succeed, but the archived record (or portions of it) will be kept in main memory until disk space becomes available. The memory usage reported by message FRM-91992 may help to determine a reasonable value for this environment variable. It shouldn't be set to a value that would cause many Forms applications to exceed the maximum size. [But bear in mind that some applications might be using more than their fair share, due for example, to omitting a WHERE clause from a large query.] Note that FRM-91992 will appear only if the log level (appended to the serverURL configuration parameter) is set to /sessionperf, /perf, or /debug. Valid values: Must be at least the segment size specified by FORMS_RECMGR_SEGMENT_SIZE and must not exceed 2T (or 1T if the temporary file system page size is 512). The specified value will be truncated to a multiple of the temporary file system page size. Default: 4G

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_RECMGR_REPORT_RECGRP_MEM	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to MAP_SWAP or MAP_TMPFILE, and FORMS_RECMGR_USED_FOR_RECGRPS is set to TRUE, and log level (appended to the serverURL configuration parameter) is /sessionperf, /perf, or /debug. It's intended for users with critical performance requirements. Most users won't need to set this environment variable. It specifies whether to report record group memory usage, with and without the record manager. A value of TRUE (if applicable) specifies that at the end of the Forms session, a diagnostic message (FRM-91994) will display record group memory usage statistics, as of the time of the overall maximum archived memory usage. These statistics will show the memory used for archived records and the approximate memory required for active records and additional control information, both the actual usage (using the record manager), and what memory usage would have been if FORMS_RECMGR_USED_FOR_RECGRPS were set to FALSE. Note that the additional CPU time required to produce these statistics may be non-negligible. Default: FALSE

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_RECMGR_RESERVE_SPACE`	Runtime	This environment variable is applicable only when `FORMS_RECMGR_ARCHIVE` is set to `MAP_SWAP` or `MAP_TMPFILE`. If applicable, it specifies whether space should be reserved when the record manager allocates a large block of memory. • A value of `TRUE` specifies that when the record manager allocates a large block of virtual memory for a data block or record group (from which space for archived records will be suballocated), swap space or temporary file system space will be reserved for the entire large block. Specifying `TRUE` reduces the likelihood of a Forms application being killed without warning if the system runs out of memory. When swap space or temporary file system space is inadequate, memory allocations will fail, but the Forms application will respond gracefully: record creation (for insert, query, or backup) will be disallowed (at least until the next time a data block or record group is cleared), and the end user will have the opportunity to commit changes to the database. The downside of specifying `TRUE` is that disk space will be wasted: the Forms application may wind up never using all the reserved disk space (and even if it does, it won't use most of it right away). So subsequent memory allocations (requested by the Forms application or by other processes) may fail needlessly. This downside may be mitigated to a certain extent by specifying a small value for `FORMS_RECMGR_BLOCK_SIZE`, but smaller values increase CPU usage. • A value of `FALSE` specifies that space or temporary file system space typically won't be reserved until a page within a large block needs to be flushed to disk. Specifying `FALSE` makes better use of system resources, and memory allocations will almost never fail. However, a Forms application could potentially get killed without warning if the system runs out of memory. This should typically be rare, especially if a reasonable value is specified for `FORMS_RECMGR_ARCHIVE_THRESHOLD`. The `FORMS_RECMGR_RESERVE_SPACE` environment variable is ignored when `FORMS_RECMGR_ARCHIVE` is set to `WRITE_TMPFILE`. In that case, most file systems will reserve disk space when the record manager writes an archived record (even though the file system may defer flushing the page(s) that contain the record to disk). Default: `FALSE` (if supported: see restrictions that follow) Restrictions: • When `FORMS_RECMGR_ARCHIVE` is set to `MAP_SWAP`: – On Windows systems, only `TRUE` is supported. – On Linux systems, only `FALSE` is supported, unless `/proc/sys/vm/overcommit_memory` is set to `2`, in which case both `FALSE` and `TRUE` are supported. – On AIX systems, whether memory is reserved for each memory allocation request is specified on a per-process basis by the `PSALLOC` environment variable (which defaults to *late*). A value of *early* requests that swap space be reserved when memory is allocated. In this case, the only supported value for `FORMS_RECMGR_RESERVE_SPACE` is `TRUE`. Otherwise, the only supported value is `FALSE`. • When `FORMS_RECMGR_ARCHIVE` is set to `MAP_TMPFILE`, all file systems support `FALSE`, but many file systems do not support `TRUE`,

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
		or support it only very inefficiently (by writing zeros to each reserved page). Modern extent-based file systems, such as ext4, btrfs, and xfs on Linux, support efficient preallocation of disk space. A value of `TRUE` is not recommended for any other file system. **Note:** Future releases of Forms may choose to disallow a value of `TRUE` for certain file systems which currently allow it, but which support it only very inefficiently.
`FORMS_RECMGR_RETRY_INTERVAL`	Runtime	This environment variable is applicable only when `FORMS_RECMGR_ARCHIVE` is set to `WRITE_TMPFILE`. It's intended for users with critical performance requirements. Most users can simply use the default value. It specifies the time interval, in milliseconds, during which Forms will not attempt writes to the record manager backing file, beyond the highest disk address to which a write has been attempted, after having encountered an *out of disk space* error on a write. [The assumption is that disk space has probably not been freed by other processes during this interval.] After this interval has expired, Forms will resume attempting writes to all disk addresses, unless the *out of disk space* error was **hard**, that is it's guaranteed to recur on all subsequent write attempts at or beyond the address that encountered the error. [An example on Unix systems is a write that would exceed the process file size limit.] A value of 0 indicates that Forms will never suppress write attempts after a **soft** *out of disk space* error. Note that on Windows, the available timer may have a resolution considerably coarser than a millisecond (for example, 10-16 milliseconds), so the value specified for this environment variable is only approximate. Valid values: A non-negative integer < 4G. Default: `2000`
`FORMS_RECMGR_SEGMENT_SIZE`	Runtime	This environment variable is applicable only when `FORMS_RECMGR_ARCHIVE` is set to `WRITE_TMPFILE`. It's intended for users with critical performance requirements. Most users can simply use the default value. It specifies the maximum *segment* size, in bytes. The record manager will split flattened images that exceed this size into multiple segments. A value larger than the default will typically reduce the number of I/O events in an application that manipulates images, at the cost of increased temporary file system usage. Valid values: Powers of 2 no smaller than the temporary file system page size (or 16K, if the temporary file system page size exceeds that), and no larger than 1M. Default: `64K`

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_RECMGR_SEGPOOL_ENTRIES`	Runtime	This environment variable is applicable only when `FORMS_RECMGR_ARCHIVE` is set to `WRITE_TMPFILE`. It's intended for users with critical performance requirements. Most users can simply use the default value. It specifies the total number of entries in the free segment pool. This is a pool of 4-byte disk addresses (expressed as page numbers) that are available for writing maximum-sized segments. A value of `1` is recommended for an application that contains no image items and no data blocks whose record size exceeds the maximum segment size (as specified by `FORMS_RECMGR_SEGMENT_SIZE`). For an application that contains image items or data blocks with very large records, the number of entries in the pool should be large enough to *smooth out* most of the peaks and valleys in the number of maximum-sized segments that are made available and subsequently reused during record navigation within a data block, because reusing these segments improves performance. A good rule of thumb is to analyze each data block that contains image items and/or very large records in order to determine an adequate pool size for that data block, and then take the maximum over all data blocks. To determine the adequate pool size for a particular data block: 1. Determine an approximate floor and ceiling on the number of maximum-sized segments required for a record. [It's acceptable for 5%-10% of the records to fall outside these bounds.] 2. Subtract the floor from the ceiling, and multiply by the maximum number of active records for the data block, as specified by the `FORMS_RECMGR_ACTIVE_RECS_RATIO` and `FORMS_RECMGR_ACTIVE_RECS_DELTA` environment variables, or by the data block's Number of Records Buffered property. Valid values: 1-1024. Default: `256`
`FORMS_RECMGR_SLACK_MAX`	Runtime	This environment variable is applicable only when `FORMS_RECMGR_ARCHIVE` is set to `MAP_SWAP` or `MAP_TMPFILE`. It's intended for users with critical performance requirements. Most users can simply use the default value. `FORMS_RECMGR_SLACK_RATIO` and `FORMS_RECMGR_SLACK_MAX` together specify the maximum number of *slack* bytes in each data block (or record group). See `FORMS_RECMGR_SLACK_RATIO` for details. Valid values: Integers in the range 0-64K. Default: `8K`

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_RECMGR_SLACK_RATIO	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to MAP_SWAP or MAP_TMPFILE. It's intended for users with critical performance requirements. Most users can simply use the default value. While a data block (or record group) is being populated (for example, by a query), archived records are laid out sequentially in large blocks of memory, with no gaps (free regions). If the Forms application subsequently chooses to navigate through the records in the data block, the record manager will typically maintain a varying amount of *slack* (free regions interspersed between archived records), as part of its strategy to minimize fragmentation and improve locality of reference. [If the application navigates forward sequentially, there will typically be a single free region that moves forward through the large blocks of memory, varying in size when the records vary in size.] FORMS_RECMGR_SLACK_RATIO and FORMS_RECMGR_SLACK_MAX together specify the maximum number of slack bytes in each data block (or record group). A data block's maximum slack is FORMS_RECMGR_SLACK_RATIO times the average size of the data block's archived records, not exceeding FORMS_RECMGR_SLACK_MAX. The default values for these environment variables will usually produce near-optimal behavior. When a Forms application contains data blocks which exhibit extreme variations in record size, specifying larger values for these environment variables might reduce fragmentation and improve locality of reference. Larger values should not be specified if they cause a significant increase in memory used for archived records, as reported in the FRM-91992 message written at the end of the Forms session if the log level (appended to the serverURL configuration parameter) is /sessionperf, /perf, or /debug. Conversely, when the data blocks in a Forms application all exhibit small variations in size, specifying smaller values for these environment variables might reduce memory usage without significantly increasing fragmentation or reducing locality of reference. Smaller values should not be specified unless they cause a significant decrease in memory used for archived records. Valid values: 0-32.0. Fractional values are allowed. Default: 8.0

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_RECMGR_TMPFILE_DENSITY	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to WRITE_TMPFILE. It's intended for users with critical performance requirements. Most users can simply use the default value. It specifies the approximate density of the backing file. The value indicates the maximum allowable size of the file at any given time, relative to the minimum possible size (roughly the size of all pages currently in use). The valid values (0-7) have the following meanings: • 0: File can grow to twice as large as the minimum • 1: File can grow 1/2 larger than the minimum • 2: File can grow 1/4 larger than the minimum • 3: File can grow 1/8 larger than the minimum • 4: File can grow 1/16 larger than the minimum • 5: File can grow 1/32 larger than the minimum • 6: File can grow 1/64 larger than the minimum • 7: File can grow 1/128 larger than the minimum Values smaller than the default may reduce CPU time, at the cost of increased temporary file system usage. Conversely, larger values may reduce temporary file system usage, at the cost of increased CPU time. Default: 4
FORMS_RECMGR_TMPPOOL_ENTRIES	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to MAP_TMPFILE. It's intended for users with critical performance requirements. Most users can simply use the default value. When FORMS_RECMGR_ARCHIVE is set to MAP_TMPFILE, the record manager needs an open temporary file for each data block or record group that currently contains archived records. It maintains a pool of open temporary files, whose size is specified by this environment variable. When the record manager needs a temporary file, it will take one from the pool unless the pool is empty, in which case it will create a temporary file. When a data block or record group is cleared, the temporary file will be truncated to zero length and added to the pool unless the pool is full, in which case the temporary file will be deleted. Larger values reduce CPU usage (because there are fewer system calls to create, open, close, and delete temporary files) but will typically increase the peak number of concurrently open files. Valid values: 0-256. Default: 16

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_RECMGR_USED_FOR_RECGRPS	Runtime	This environment variable is applicable only when FORMS_RECMGR_ARCHIVE is set to MAP_SWAP or MAP_TMPFILE. It's intended for users with critical performance requirements. Most users can simply use the default value. If applicable, it specifies whether the record manager will be used for all record groups. A value of FALSE specifies that the record manager will not be used for record groups whose effective record size (excluding LONG columns) is less than 64Kb; instead, the pre-release-12 technology will be used. Specifying FALSE will reduce CPU usage, and may reduce memory usage when the number of records in a record group is small. Note that a value of FALSE is likely to increase memory usage significantly when the number of records in a record group is large (by roughly the ratio of maximum to average cell data value size, although typically less when a record group contains a LONG column). You can compare record group memory usage with and without the record manager. See FORMS_RECMGR_REPORT_RECGRP_MEM for details. When FORMS_RECMGR_ARCHIVE is set to MAP_SWAP or MAP_TMPFILE, the record manager is always used for record groups whose effective record size exceeds 64Kb. This environment variable is ignored when FORMS_RECMGR_ARCHIVE is set to WRITE_TMPFILE. In that case, the record manager is never used for record groups. If a form contains a record group whose effective record size exceeds 64Kb, a fatal error is reported when the form is loaded. Default: TRUE
FORMS_RECORD_GROUP_MAX	Runtime	The maximum number of records to be fetched from any record group (including those used by LOVs). Reaching this limit is treated essentially like end of query, although it may affect completion messages displayed in, for example, LOVs. A value of 0 indicates no limit. Default: 0 (no limit)
FORMS_REDIRECT_DATA_PROPS	Runtime	
FORMS_REJECT_GO_DISABLED_ITEM	Runtime	
FORMS_REMOTE_DEBUGGER_PORT_RANGE	Runtime	Specifies a range of ports (min and max port numbers, separated by a hyphen), or a single port, via which Oracle Forms will attempt to connect to a remote Forms debugger. A value of 0 (or an invalid value) will cause a port to be automatically allocated. Specifying a non-zero value may be appropriate when the remote debugger is sitting behind a firewall and only a limited number of ports should be opened. Default: 0
FORMS_REPORTSSRV_USE_SQLNET	Runtime	

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_RESTRICT_ENTER_QUERY`	Runtime	Specifies whether arbitrary SQL expressions are disallowed in enter-query mode. When a value of `TRUE` is specified, Oracle Forms limits the types of query criteria that can be entered when in Enter-Query mode. In general, it disallows the use of: • Conjunctions (AND, OR) keywords which modify parts of the SELECT statement outside of the WHERE clause (ORDER BY) • All functions, including SQL functions (LENGTH, TO_CHAR, LPAD, SUBSTR) • Query/Where window Specify a value of `FALSE` (or remove the specification in `default.env`) for end-users who need access to the query-where functionality which potentially allows them to enter arbitrary SQL statements when in enter-query mode. Note, however, that this may be deemed a security exposure. Value specified in the `default.env` file that is shipped with the Forms product: `TRUE`. Default if unspecified: `FALSE`.
`FORMS_RESTRICT_MDIWIN_POS`	Runtime	Specifies whether or not to restrict the position of the MDI window to the main display. Application developers can use the design-time built-in, `SET_WINDOW_PROPERTY()`, to set the position of the MDI window. If `FORMS_RESTRICT_MDIWIN_POS` is set to `TRUE`, the system restricts the X position (left-right) of the MDI window to the main display when repositioning it. If set to `FALSE`, the built-in can move the MDI window beyond the main display based on the given X coordinate. This setting allows application developers to position the MDI window on a second display. **Note:** If set to `FALSE`, the system won't attempt to validate the MDI window's position. Application developers should use care when coding their applications to avoid moving the MDI window completely off the display so that is it unreachable. Default: `TRUE`.
`FORMS_ROWID_IS_NAVIGATION_ITEM`	Runtime	
`FORMS_ROWLOCK_OPTIMIZATION`	Runtime	
`FORMS_RUEI_SEND_FORM_NAME`	Runtime	Specifies whether the Forms server will send the name of the form module to the client for each window that is created in the form. Specifying a value of `TRUE` will help to enhance the Oracle Real User Experience Insight (RUEI) user experience, by enabling RUEI to easily associate a particular window and its contents with a particular form. Default: `FALSE`

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_SCROLL_ALL_BUT_ONE`	Runtime	
`FORMS_SELECT_ON_CLICK`	Runtime	Specifies whether or not the text in a field should be selected when using the mouse to enter the field. Default: `FALSE`
`FORMS_SEPARATE_DEBUGGER`	Builder	
`FORMS_SHOW_ERROR_IN_JCONSOLE`	Runtime	Specifies whether to display Forms error messages within the running form and also in the Java Console. This applies to most Forms generated messages/errors, such as FRM-12345. When set to `TRUE`, Forms generated messages/errors appear within the form *and* in the Java Console. Custom messages created by the application developer continue to appear within the application, but do not appear in the Java Console, regardless of this setting. Default: `FALSE`
`FORMS_SHOW_REP_DIALOG`	Builder	Specifies whether or not to display the Create Report dialog when creating a new report object in the Object Navigator. Setting to `TRUE` (`1`) will cause the dialog to be displayed. Setting to `FALSE` (`0`) or unset will prevent the dialog from appearing. Default: `FALSE`
`FORMS_STDIN_PASSWORD`	Builder	
`FORMS_SUPPRESS_LOGON`	`Runtime`	Specifies whether to suppress the initial Forms logon dialog. It may be desirable to suppress the logon dialog in situations like these: • When using SSO and the administrator does not want the user to interact with the logon in the event connecting to the database fails • Where the administrator or application sets the user credentials in the configuration or in the application code and the user is not permitted to enter their own database credentials **Note:** This setting does not affect the functionality provided with the Forms `LOGON` built-in command. Default: `0`
`FORMS_SUPPRESS_RECURSIVE_DUMPS`		Specifies whether to suppress a core dump (or a stack trace) if an operating system exception occurs while Forms is already processing a previous operating system exception in the runtime process. Normally, this environment variable should be set to `FALSE` (the default), because the core dump or stack trace may provide useful diagnostic information, but setting it to `TRUE` may be necessary if an excessive number of core dump files are being produced. Default: `FALSE`

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_SWITCH_JAVA_EVENTS	Runtime	Specifies whether a Pluggable Java Component (PJC) or Java Bean event occurs immediately or is queued for processing when other, possibly more important, events are being processed. When enabled (1), PJC or Bean events are placed in the Forms queue and are only processed when Forms is not busy. In most cases, this setting should not be needed. **Caution:** Do not use this setting unless you have carefully tested the results of enabling this setting. Default: 0
FORMS_SYSADM_MESSAGE_LEVEL	Builder	When enabled, exposes additional diagnostic information in the event of a failure while working with the Builder REST Package Designer (RPD). On Unix/Linux the output will be shown in the shell used to start the tool or an operating system dialog if running on MS Windows. A value of 0 will enable this additional output and clearing or setting to any other value disables the additional output. Although a value of 0 used to enable this may seem unintuitive, this behavior is necessary as this is an extension of ODL, which uses a value of 0 to enable debug mode in Forms.
FORMS_SYSTEM_EVENT_NAVIGATION	Runtime	Specifies whether another form can service a system event when the current form doesn't have a subscription to the event. If set to TRUE, the behavior depends on whether a form has been designated as the master system event form (via SET_APPLICATION_PROPERTY (MASTER_SYSTEM_EVENT_FORM, form_id.id)). If so, it will service the event if it has a subscription to it. If not, the first opened form that has a subscription to the event (if there is such a form) will service the event. Default: FALSE

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_TIMEOUT	Runtime	Specifies the elapsed time, in minutes, before the runtime process is terminated if there has been no communication from the client. **Note:** The value of FORMS_TIMEOUT should always be greater that the value of the applet parameter, heartBeat. Setting it to less than the heartBeat value will cause an improper termination of the user's running session. Terminating the running form in this manner can result in the user session becoming unresponsive. See heartBeat in Web Configuration Parameters. Valid values: Integers in the range 3-1440 (up to 1 day). Values of 1 and 2 are treated like 3, and values > 1440 are treated like 1440. Default: 15

Table 65   (Cont.) Environment Variables

Environment Variable	Applies to	Description
`FORMS_TRACE_DIR`	Runtime	Specifies the location (directory) of dump files produced as the result of a crash of any of the Forms runtime executables. The dump files contain diagnostic information about events at the time the process crashed. Also specifies the directory which will contain files produced when tracing is enabled. Default for Unix: • For dump files, and trace files produced when record=forms is specified as a runform parameter: `$DOMAIN_HOME/system_component/FORMS/$FORMS_INSTANCE_NAME/trace`. • For other sorts of trace files: `$DOMAIN_HOME/system_component/FORMS/$FORMS_INSTANCE_NAME/log`. **Note:** The value of `$DOMAIN_HOME` is typically a subdirectory of `ORACLE_HOME`, for example `$ORACLE_HOME/user_projects/domains/base_domain`. The value of `$FORMS_INSTANCE_NAME` is typically something like `forms1`. Default for Windows: • For dump files, and trace files produced when record=forms is specified as a runform parameter: `%DOMAIN_HOME%/system_component/FORMS/%FORMS_INSTANCE_NAME%/trace`. • For other sorts of trace files: `%DOMAIN_HOME%/system_component/FORMS/%FORMS_INSTANCE_NAME%/log`. **Note:** The value of `%DOMAIN_HOME%` is typically a subdirectory of `%ORACLE_HOME%\user_projects\domains\base_domain`. The value of `%FORMS_INSTANCE_NAME%` is typically something like `forms1`.
`FORMS_UPPERCASE_WHERE`	Runtime	
`FORMS_USE_CBO`	Runtime	Specifies whether a `USE_CONCAT` optimizer hint should be added to queries into data blocks whose WHERE clause contains a reference to an item whose Case Insensitive Query property is `TRUE`. Default: `FALSE`
`FORMS_USE_NATIVE_MALLOC`	Runtime	
`FORMS_USER_CALENDAR`	Runtime	
`FORMS_USER_DATE_FORMAT`	Runtime	
`FORMS_USER_DATETIME_FORMAT`	Runtime	

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
FORMS_USEREXITS	Runtime	The full paths of shared object files (Unix) or DLL files (Windows) containing foreign functions that can be invoked from a user exit. Example (Windows): `FORMS_USEREXITS=C:/mathlib/add.dll;C:/mathlib/mult.dll`
FORMS_USERNAME_CASESENSITIVE	Runtime	Specifies if the logon user name should be case sensitive or not. Default: `0`
FORMS_WJSI_OVERRIDE_TIMEOUT	Runtime	Specifies the elapsed time, in minutes, before the runtime process is terminated if there has been no communication from the client while a JavaScript function or expression is being evaluated through Jetty websocket. Valid values: Integers in the range 0-1440 (up to 1 day). Values less than or equal to the value of `FORMS_TIMEOUT` are treated like the value of `FORMS_TIMEOUT`, and values > 1440 are treated like 1440. Specifying a value larger than the value of `FORMS_TIMEOUT` may be necessary if Jetty websocket is being used to evaluate a JavaScript function or expression, because heartbeats will be blocked during the evaluation. Default: `0`
LD_LIBRARY_PATH	Runtime, Builder	Unix system environment variable. Specifies the directories which will be searched for shared objects dynamically linked by Oracle Forms and Reports. Required on Unix platforms.
LD_PRELOAD	Runtime	Unix system environment variable. Specifies shared objects which will be *preloaded* (loaded before all other shared objects). Oracle Forms and Reports require that libjsig.so be preloaded, to support the signal-chaining facility offered by JVM 1.5 and above. Required on Unix platforms. **Note:** If there are multiple environment files, ensure that LD_PRELOAD has the same settings as in `default.env`.
MM_USE_TIF_G4	Runtime	
ORA_CLIENTTRACE_DIR		Specifies where the middle-tier interface to the Oracle RDBMS will write trace files.
ORACLE_HOME	Runtime, Builder, Compiler, JDAPI	The base installation directory for Oracle products. Required.
ORACLE_PATH	Runtime, Builder, Compiler, JDAPI	`FORMS_PATH` and `ORACLE_PATH` specify the directories that Oracle Forms searches when looking for a form (`fmx` file), menu (`mmx` file) to run, or for a PL/SQL library (`pll` or `plx` file) to attach, when the file does not specify a path. The directories specified by `FORMS_PATH` are searched before the directories specified by `ORACLE_PATH`. `ORACLE_PATH` is also used by other Oracle products in addition to Oracle Forms.
PATH	Runtime, Builder	System environment variable. Specifies the directories that contain the binary executables for Oracle products.

Table 65 (Cont.) Environment Variables

Environment Variable	Applies to	Description
`REPORTS_SERVERMAP`	`Runtime`	Specifies the name of the Reports Cluster and Server, for example: `REPORTS_SERVERMAP=`*`<rep_cluster>`*`:`*`<rep_server>`* This should also be set on each node, instead setting *<rep_server>* to the name of the server on that host, for example: `myrep_cluster:myrep_server2` The appropriate setting must also be set in the Reports `rwservlet`.properties file of each host. An example entry might look like this: *`<reports_servermap>`*`myrep_cluster:myrep_server1`*`</reports_servermap>`*
`TK2_DELAY_WIDTH_CALC`	`Builder`	Specifies whether to bypass the preloading of font metrics during Builder startup. Enabling this setting can help to improve Builder startup performance in some cases. **Note:** Applies to Microsoft Windows only. Default: `0`
`TK2_HIDE_EDITOR_LINE_NUMBER`	Builder	Specifies whether to show line numbers in the PL/SQL Editor in the Builder when on Microsoft Windows.
`TNS_ADMIN`	Runtime, Builder, Compiler, JDAPI	The directory containing TNS files such as `tnsnames.ora` and `sqlnet.ora`. Required.
`UI_ICON`	Builder	Specifies the physical path of the location where icons and images are stored. Only one directory path can be specified. This is used by the Builder only for the purpose of displaying icons in the Builder Layout Editor. It has no impact on runtime. Images must reside on the file system and cannot be contained in a zip or jar file. Some image formats are not supported for display in the Builder. Unsupported formats may appear black or not displayed at all.
`UI_ICON_EXTENSION`	Builder	Specifies the default image file extension to use if one is not provided in the Builder. If no value is provided, the platform default will be used, which is `ICO` for Microsoft Windows and `XPM` for Unix/Linux.
`WEBUTIL_CONFIG`	Runtime	The full path of the WebUtil default configuration file.

Web Configuration Parameters

This chapter describes configuration parameters and applet parameters.

These notes apply to the following table.

- Required: A parameter is required if Forms requires a non-null value (from `formsweb.cfg` or, where allowed, from the URL) in order to function correctly.
- Valid values: In a few cases, these are listed explicitly, but in most cases they can be inferred from the description. If the description implies that the value is Boolean (for example, "Specifies whether"), the valid values are TRUE or FALSE (case-insensitive),

except for runform parameters (see following table), in which case the valid values are YES, Y, ON, or NO, N, OFF (case-insensitive). Numeric values should be integers specified as decimal numbers, unless indicated otherwise. The elements in lists of files or directories should be separated by ':' [Unix] or ';' [Windows] unless indicated otherwise.

- Default: For required parameters, the parameter description lists the default value from the default section of the `formsweb.cfg` that is shipped with the product (or at least indicates that it specifies an appropriate value). For optional parameters, the parameter description may show a non-null default value from the default section of the `formsweb.cfg` that is shipped with the product. In addition, the parameter description may show the default value that is assumed if no value is specified. (This is the non-null value that produces the same behavior as an unspecified or null value). When the description for an optional parameter simply shows an unqualified default value, the implication is that this value is both the default value from the default section of the `formsweb.cfg` that is shipped with the product, and also the default value that is assumed if no value is specified. When the description for an optional parameter does not explicitly specify a default value, the implication is that the default value is null.
- Application system parameters: The descriptions for some configuration parameters indicate that they are application system parameters. Such a parameter applies to an application that has been launched via a URL. Its value is taken from the default config section or from the config section specified by the `config` parameter in the URL. A value specified for an application system parameter in the URL is ignored.
- Global system parameters: The descriptions for some configuration parameters indicate that they are global system parameters. Such a parameter applies to the installation as a whole. Its value is taken from specific config section(s); the exact semantics are spelled out in the description of the parameter.
- Application user parameters: If a configuration parameter is not described as an application system parameter or a global system parameter, the implication is that it is an application user parameter. Such parameters are similar to application system parameters, but a value taken from a config section may be overridden by a value specified for the parameter in the URL.
- Runform parameters: The descriptions for some parameters indicate that they are runform parameters. They are passed to the `frmweb` process using the `serverArgs` config or applet parameter. For such a parameter, when specifying a value that contains special characters, follow the syntax rules listed in Specifying Special Characters in Values of Runform Parameters.
- Sub-arguments for `otherparams`: The descriptions for some parameters indicate that they are sub-arguments for `otherparams`. That means that in order for the parameter to take effect (when specified in `formsweb.cfg` or the URL), it must appear in the form `name=%name%` within the value of the `otherparams` parameter. So, for example, if you are adding the `array` parameter (with a value of`no`) to a configuration section, you must also add `array=%array%` to the value of the `otherparams` parameter. Notice that these parameters are all runform parameters (because the `otherparams` parameter is itself a runform parameter), and so when specifying a value that contains special characters adhere to the syntax rules described in Specifying Special Characters in Values of Runform Parameters. These parameters are treated as application system parameters if the `otherparams` parameter is specified in the `restrictedURLparams` parameter.

The value specified for *Category* is one of 11 possible values.

- The value for a preconfigured configuration parameter is one of the 8 values in the **Show** list in the Enterprise Manager Web Configuration page. These are `basic`, `sso`, `trace`, `plugin`, `html`, `applet`, `advanced`, and `all`.

- A value of `add-on` indicates a configuration parameter that's not preconfigured. Such a parameter can be added to a configuration section as described in Managing Parameters. Once added, the Enterprise Manager Web Configuration page will show the parameter as `advanced`.
- Values of `applet add-on` and `sso add-on` are similar to `add-on`, but the Enterprise Manager Web Configuration page will show the parameter as `applet` or `sso` if it's added to a configuration section.
- A value of `add-on applet param` indicates a parameter which can be added to applet definition template files that you have created by modifying the templates provided by Oracle, as described in Creating Your Own Applet Definition Template Files. For added flexibility (for example, if you wish to allow end users to specify the value in the URL), the applet parameter can specify that its value is the value of a configuration parameter (typically of the same name). Such a parameter can be added to a configuration section as described in Managing Parameters. When added, the Enterprise Manager Web Configuration page will show the parameter as `advanced`.

Table 66 Web Configuration Parameters

Parameter	Category	Description
`allowAlertClipboard`	`advanced`	Specifies whether an alert dialog should be presented if the system clipboard is inaccessible. Specifying FALSE will cause failed access to the clipboard to be reported on the Java Console. Default: `TRUE`
`allowLservletDebug`	`add-on`	Specifies whether to allow debug messages (and other messages with log level less than NOTIFICATION) to be written from the Listener Servlet (`/forms/lservlet`) to the Oracle Diagnostic Log, in the (normal) case where the Listener Servlet was launched from the Forms Servlet. **Note:** In a case where the Listener Servlet was not launched from the Forms Servlet (for example, by submitting static HTML or entering a `/forms/lservlet` URL in a browser), the `allowLservletDebug` lservlet initialization parameter specifies whether to allow debug messages to be written from the Listener Servlet to the Oracle Diagnostic Log. This parameter is an application system parameter. Default: `FALSE`

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
allowNewConnections	advanced	Specifies whether new Forms sessions are allowed. This parameter is a global system parameter. It must be specified in the default config section of formsweb.cfg. The value is read from formsweb.cfg on every request to the Forms servlet. Default: TRUE
allow_debug	advanced	A value of TRUE or TESTMODE allows debug messages (and other messages with log level less than NOTIFICATION) to be written from the Forms Servlet (/forms/frmservlet) to the Oracle Diagnostic Log, and also allows certain test commands, such as info, admin, setcookie, setcookiesess, checkcookie, and proctest to be executed. You can execute these commands by specifying them in the frmservlet URL, for example /forms/frmservlet/info or /forms/frmservlet?ifcmd=info. A value of TESTMODE also causes the page displayed by the admin command to contain the Listener Servlet version number and links to various other test commands. A value of FALSE suppresses diagnostic messages with log level less than NOTIFICATION and disallows test commands. **Note:** Even when TRUE or TESTMODE is specified, certain debug messages produced before the Forms Servlet has read the allow_debug parameter will be suppressed. To see these messages (in addition to other debug messages), specify a value of TRUE or YES for the testMode frmservlet initialization parameter. This will override any value that was specified (or defaulted) for the allow_debug configuration parameter. This parameter is an application system parameter. Default: FALSE

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
alwaysOnTop	applet add-on	Specifies whether the Forms separate frame will remain on top of all other open windows. When used with separateFrame=true, the Forms separate frame will remain on top of all other open windows. If separateFrame=false, setting this parameter will have no effect on the frame. This is not supported with Web Start. Default: FALSE
applet_name	advanced	Configuration for JavaScript integration. This is name of the Forms applet that users can use to refer to it from a JavaScript code.
archive	plugin	Comma-separated list of archive files that are used or downloaded to the client. For each file, include the file name if the file is in the codebase directory, or include the virtual path and file name. Required. Default: frmall.jar
array	add-on	Set this parameter to NO to suppress array processing. This causes Forms to send only a single row at a time to the database for an INSERT, UPDATE, or DELETE, and it causes the database to return only a single row of query results at a time. This usually results in the first retrieved record displaying faster, but the total time to display all rows in the query result is longer. This parameter is a sub-argument for otherparams. Default: YES
background	applet	Specifies the image file that should appear in the background of the main applet window. Specify NO for no background. Leave empty to use the default background. Supported image formats are: gif, png, and jpg.
baseHTML	html	Specifies the applet definition template file to be used for an application that's not Web Start or Standalone, if the client browser is not on Microsoft Windows or does not support the <OBJECT> tag, or if the baseHTMLjpi parameter is unspecified or empty. The file can be specified as a full path, or as the name of a file within $DOMAIN_HOME/config/fmwconfig/components/FORMS/instances/<Forms Instance Name>/server. This parameter is an application system parameter. Required. Default: base.htm

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`baseHTMLjpi`	`html`	Specifies the applet definition template file to be used for an application that's not Web Start or Standalone, if the client browser is on Microsoft Windows and supports the `<OBJECT>` tag. The file can be specified as a full path, or as the name of a file within `$DOMAIN_HOME/config/fmwconfig/components/FORMS/instances/<Forms Instance Name>/server.` This parameter is an application system parameter. Required. Default: `basejpi.htm`
`basejnlp`	`html`	Specifies the applet definition template file to be used for a Web Start application or an embedded JNLP file. The file can be specified as a full path, or as the name of a file within `$DOMAIN_HOME/config/fmwconfig/components/FORMS/instances/<Forms Instance Name>/server`. This parameter is an application system parameter. Required. Default: `base.jnlp`
`baseSAAfile`	`html`	Specifies the applet definition template file to be used for a Standalone application. The file can be specified as a full path, or as the name of a file within `$DOMAIN_HOME/config/fmwconfig/components/FORMS/instances/<Forms Instance Name>/server`. This parameter is an application system parameter. Required. Default: `basesaa.txt`
`centerOnStartup`	`applet add-on`	Specifies whether the Forms separate frame will start centered on the screen. When used with `separateFrame=true`, the Forms separate frame will start centered on the screen. If `separateFrame=false`, setting this parameter will have no effect on the frame. Default: `FALSE`
`clientDPI`	`add-on`	Dots per inch (DPI). Overrides the DPI setting returned by the JVM, allowing you to manage varying DPI settings per platform. Oracle recommends that you use an integer between 50 and 200.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`clientDPIRatio`	`advanced`	Specifies the percentage of the usable display that the application's MDI window should scale to. The window is scaled based on the display size but retains the `HEIGHT` to `WIDTH` ratio set in the Forms Web Configuration. This ensures that neither dimension becomes larger than the display size and that the application window does not become distorted. This is only supported when configured to run with Java Web Start (JWS), FSAL, or embedded with `separateFrame=true`. **Note:** If `clientDPI` is set, `clientDPIRatio` is ignored. Values: Integers from `10` to `100`. Invalid values are ignored.
`codebase`	`plugin`	The virtual directory you define to point to the physical directory `ORACLE_HOME/forms/java`, where, by default, the applet JAR files are downloaded from. Required. Default: `/forms/java`
`colorScheme`	`applet`	Specifies the application's color scheme. Valid values [case-insensitive]: `Teal`, `Titanium`, `Red`, `Khaki`, `Blue`, `BLAF`, `SWAN`, `Olive`, `Purple`. **Note:** `colorScheme` is ignored if `LookAndFeel` is set to Generic. Default: `SWAN`
`connectionDisallowedURL`	`add-on`	The URL to which Forms Servlet requests are redirected when connections to new Forms sessions are disallowed. This parameter is an application system parameter.
`consoleUseRegistryFont`	`add-on applet param`	Specifies whether the application's console (or message bar) uses the default font attributes specified in **Font and Icon Mapping** (`Registry.dat`).
`cursorBlinkRate`	`add-on applet param`	The duration (in milliseconds) that the cursor is visible during a blink cycle. This is also the duration that the cursor is invisible. A value of `0` disables blinking and the cursor remains visible all the time. Default: `600`. The cursor completes one full blink every 1.2 seconds

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
customColorScheme	advanced	The name of the custom colorscheme created in Registry.dat. Setting this will override the colorscheme parameter.
darkLook	add-on	Specifies whether Forms should attempt to adjust the color of graphic objects and boiler text if the background and foreground colors are such that the foreground is not visible—for example, when the background and foreground are both black. In a scenario where there is black (dark) text on a dark background, the text is automatically changed to white if this parameter is set to TRUE. This setting is ignored if lookandfeel=generic. **Note:** Although "dark" in Forms is any color with a luminance value less than 139, there are some cases where Forms may not be able to make an appropriate correction of the foreground color. The default value of 139 luminance can be overridden by setting the Registry.dat parameter default.luminance.dark to the desired value. See Font, Image, and Style Mapping. Default: FALSE
darkMode	add-on	Specifies whether Forms should try to determine if the canvas background color is so dark that graphic text (boiler plate) on it cannot be easily read—for example, if the text is black and the canvas is dark brown. If so, and this setting is TRUE, Forms makes the dark text white. Default: FALSE
debug	trace	Allows running in debug mode. This parameter is a runform parameter. Default: NO
debug_messages	add-on	Set this parameter to YES to cause Forms to display ongoing messages about trigger execution while the form runs. This parameter is a sub-argument for otherparams. Default: NO

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`defaultcharset`	`add-on`	The character set to be used in servlet requests and responses. Defaults to ISO-8859-1 (also known as Latin-1). Ignored if the servlet request specifies a character set (for example, in the content-type header of a POST). The values of this parameter may be specified either as an IANA character set name (for example, SHIFT_JIS) or as an Oracle character set name (for example, JA16SJIS). It should match the character set specified in the NLS_LANG environment variable, and it should also be a character set that the browser can display. Also, if the browser allows multibyte characters to be entered directly into a URL, for example, using the IME, as opposed to URL escape sequences, and if end users should be allowed to do this, then the value of this parameter should match the character set that the browser uses to convert the entered characters into byte sequences. **Note:** If your configuration file contains configuration sections with names that contain characters other than 7-bit ASCII characters, then the following rules apply. If a config parameter is specified in a URL or in the body of a POST request with no specified character set, and the value contains non-7-bit ASCII characters, then the value is interpreted using a character set named in the `defaultcharset` parameter. However, only the language-dependent default section and the language-independent default section of the configuration file is searched for the `defaultcharset` parameter. No other configuration section is searched because the name is not yet known. This parameter is an application system parameter.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
digitSubstitution	advanced	Specifies the shape of decimal digits in BIDI environments. Valid values [case-insensitive]: • None, which indicates European digits, regardless of the field direction. • National, which indicates Arabic-Indic digits (U+066x or Hindi format), regardless of the field direction. • Context, which indicates European digits in a left-to-right field, and Arabic-Indic digits in a right-to-left field. Default: Context
disableMDIScrollbars	add-on	Specifies whether to disable horizontal and vertical scroll bars in the Forms main applet window. Default: FALSE
disableValidateClipboard	advanced	Specifies whether to disable validation of the clipboard when focus moves. This may improve performance. However, the PASTE menu bar icon will be enabled whenever the item that now has focus is editable, even if it an attempt to paste into it would fail. Default: FALSE
dynamicLayout	add-on applet param	Specifies whether the Forms applet should redraw the parent window (and notify the server) while resizing the window. Enabling causes an increase of network traffic while resizing is in progress. Default: TRUE
enableJavaScriptEvent	advanced	Specifies whether JavaScript events can be raised on the client, and whether the WEB.JAVASCRIPT_EVAL_EXPR procedure or WEB.JAVASCRIPT_EVAL_FUNC function can be executed on the server. Default: TRUE
EndUserMonitoringEnabled	trace	Specifies whether End User Monitoring integration is enabled. Default: FALSE
EndUserMonitoringURL	trace	Specifies where to record End User Monitoring data.
envFile	basic	The name of the environment configuration file. This parameter is an application system parameter. Required. Default: default.env

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`escapeparams`	`advanced`	If set to FALSE, runform will treat special characters in runform parameters as it did in releases earlier than 9.0.4. This parameter has no effect unless it is the first sub-argument of the `serverArgs` config or applet parameter, in which case it specifies whether to escape certain special characters in values extracted from the URL for subsequent runform parameters in the `serverArgs` parameter. This parameter is an application system parameter. Default: `TRUE` Default if unspecified: `FALSE`
`form`	`basic`	The name of the top-level (initial) Forms module (fmx file) to run. This parameter is a runform parameter. Required. Default: `test.fmx`
`formsMessageListener`	`advanced`	The class that the Forms client uses to enable recording of Forms messages for Tool Vendor Interface (TVI)/Intercept Server.
`fsalcheck`	`advanced`	Specifies whether to perform checksum comparison of Forms standalone app launcher. When enabled, it triggers a comparison to be made on server. The checksum of FSAL on the client machine will be compared with the checksum of FSAL archived on the server. Setting this parameter to `FALSE` also disables automatic updating for FSAL when needed. To disable automatic updating while retaining `fsalCheck` functionality, set `fsalEnableAutoUpdate` to `FALSE`. Setting `fsalCheck` to FALSE is not recommended. This parameter is an application system parameter. Default: `TRUE`
`fsalEnableAutoUpdate`	`add-on`	Specifies whether to present the FSAL update dialog if the user's detected FSAL version is not compatible with the server. Default: `TRUE`

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
fsalJavaVersion	add-on	Specifies which Java version is to be used on the end-user environment. Example values: • 1.8.0_351 or 17.0.5: Runs this version only. • 1.8* or 1.8.0*: Runs any version in the Java 8 family, but no family earlier or later than Java 8. • 1.8.0_351*: Runs only the specified version or later in the Java 8 family. • 17* or 17.0*: Runs any version in the Java 17 family, but no family earlier or later than Java 17. • 17.0.5*: Runs only the specified version or later in the Java 17 family. • 1.8+ or 1.8.0+: Runs any version in the Java 8 family or any family newer. • 1.8.0_351+: Runs the specified version or later in the Java 8 family or any version in a later family. • 17+ or 17.0+: Runs any version in the Java 17 or later family. • 17.0.5+: Runs the specified version or later in the Java 17 family or any version in a later family.
fsalUpdateDialogText	add-on	Specifies the text that is presented in the Forms Standalone Launcher Updater dialog. If no value is specified, the default text provided by Oracle Forms is displayed. If a value is set, its maximum length cannot exceed 128 characters. Any text beyond 128 characters is truncated.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
guiMode	applet	Specifies the visibility of the default Windows menu bar and the Windows title bar. **Note:** This parameter is applicable for a menu bar only when no menu is specified for a form in the Form Builder; if there is any menu associated with the form, then this parameter is not applicable. In case of window-bars, this parameter is applicable even if there is a menu specified for that form in the Form Builder. This parameter does not affect title bars in alert windows or pop-up windows. Valid values are: • 0. Indicates that the Windows title bar and the default Windows menu bar are both visible. • 1. Indicates that the Windows title bar is visible, but the default Windows menu bar is not visible (unless it has children). • 2. Indicates that the Windows title bar is not visible, but the default Windows menu bar is visible. • 3. Indicates that the Windows title bar is not visible, and the default Windows menu bar is also not visible (unless it has children). Default: 0
gzipCompressApplet	add-on	Specifying TRUE will cause the Forms Servlet to attempt compressing the html/jnlp content sent to the client. This assumes that the request coming from the client indicates support for compression in its HTTP Accept-Encoding header. Enabling this parameter can help to reduce application start up time. This setting may interfere with some testing tools. If such a problem occurs, simply disable this parameter while testing is being performed. This parameter is an application system parameter. Default: FALSE

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`heartBeat`	`add-on`	The interval, in minutes, at which a client sends a packet to the server to indicate that it is still running. Fractional values are allowed, for example 0.5 for 30 seconds. **Caution:** Setting this value too low will cause a significant increase in network traffic and therefore should be used cautiously. Keep in mind: • This parameter is similar to `maxeventwait`: the lower interval takes precedence. • If the `heartBeat` is less than the value specified by the `FORMS_TIMEOUT` environment variable, the user's session is kept active, even if they are not actively using the form. **Note:** It is not recommended to set the value of `heartBeat` greater than the value of `FORMS_TIMEOUT` because this will result in the termination of the user's existing session. Terminating the running form in this manner will result in the user session becoming unresponsive. • If `heartBeat` is higher than the parameter, `session-timeout`, then the value of `session-timeout` takes precedence over `heartBeat`. To increase the value of `heartBeat`, the value of `session-timeout` must be greater than `heartBeat`. For information about this parameter, see Session Timeout in *Developing Web Applications, Servlets, and JSPs for Oracle WebLogic Server*. This parameter is an application system parameter. Default: `2 minutes`

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
height	basic	The height of the form applet, in pixels. You can also specify the value of the height as a percentage. This value is relative to the size of the content area of the browser. The value should not exceed 100% or be less than 1%. To use a percentage value, the numeric value must be followed by a percentage (%) sign, like this: HEIGHT=75%. This example specifies that the applet height is 75% of the size of the browser's content area. Required. Default: 600
hideActivityBar	add-on applet param	Specifies whether the Forms activity bar (also known as progress bar) should be hidden. Default: FALSE
hideAppletVersion	add-on applet param	Specifies whether to display the Forms applet version in the client side console. This parameter is an application system parameter. Default: FALSE
hideClientExceptions	add-on applet param	Specifies whether Java exceptions should not be displayed to the end-user in both the Java console and error message dialog. Default: FALSE
hideWindowMenuItem	add-on	Specifies whether to hide a "Window" menu item in the Forms menu bar. When set to TRUE, the Window menu item does not appear if the form does not have a custom or default menu. **Note:** When set to TRUE, the menu bar logo still appears unless it is disabled or hidden by setting the logo parameter to FALSE and/or if the guiMode parameter is set to 1 or 3. Default: FALSE
highContrast	advanced	When highContrast is set to TRUE, frame labels are black if foreground and background colors are not specified. Default: FALSE
honorLineWidth	add-on	Specifies whether to use the line width size specified at design-time for graphic objects like lines, rectangles, frames, and so on. Set to TRUE to enable this functionality. Default: FALSE

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
host	trace	Specifies the host for the debugging session. This parameter is necessary for debugging purposes only. It identifies the host on which the forms engine process is started. This parameter is a runform parameter.
HTMLafterForm	html	HTML content to be added to the page below the area where the Forms application is displayed.
HTMLbeforeForm	html	HTML content to be added to the page above the area where the Forms application is displayed.
HTMLbodyAttrs	html	Attributes for the <BODY> tag of the HTML page.
HTMLdelimiter	advanced	The delimiter for variable parameters in the base HTML files. This parameter is an application system parameter. Default: %
idleTimeout	advanced	Specifies, in seconds, how much idle time can elapse before a SYSTEM_CLIENT_IDLE event will fire. This parameter is an application system parameter.
ignoreMissingSaaArchives	advanced	Specifies whether the application should continue initializing and/or running even if a needed resource, such as a JAR file, cannot be found. This parameter is intended for troubleshooting purposes only. Even though an application may continue to run with a missing resource, errors may still be presented. Default: FALSE
ignoreSaaCache	add-on applet param	Specifies whether the Oracle Forms Standalone Application Launcher should ignore files specified by the archive configuration parameter (for example, frmall.jar) that were originally downloaded from the Forms server but have been cached on the java client machine from which the Oracle Forms Standalone Application Launcher is being run. If FALSE (the default) is specified, then a cached file will be used if its path corresponds (modulo some hashing) to the full URL of the file on the Forms server and the timestamp of the cached file matches the timestamp of the Forms server file. If TRUE is specified, then cached files are ignored: all files specified by the archive configuration parameter are downloaded from the Forms server. Default: FALSE

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`imageBase`	`plugin`	Specifies where icon or image files are stored. Valid values: - `codeBase`, which indicates that the icon search path is relative to the directory that contains the Java classes. Use this value if you store your icons or images in a JAR file (recommended). - `documentBase`, which indicates that the icon search path is relative to the Forms webapp's directory. The Forms webapp's directory is located at `$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_12.2.1/<random string>/war`. Default: `codeBase` Default if unspecified: `documentBase`
`isResizable`	`applet add-on`	Specifies whether the Forms separate frame resizing ability is enabled. When used with `separateFrame=true`, the Forms separate frame resizing ability can be enabled or disabled. If `separateFrame=false`, setting this parameter will have no effect on the frame. Default: `TRUE`
`JavaScriptBlocksHeartBeat`	`advanced`	Specifies whether HeartBeat is blocked when a JavaScript call is a blocking call. Default: `FALSE`
`java_version`	`plugin`	Specifies which Java version is to be used on the end-user environment. This parameter is ignored when using the Forms Standalone Launcher (FSAL). For FSAL, use `fsalJavaVersion` instead. Refer to the entry for `fsalJavaVersion` in this table for details. Required. Default: `1.7+`
`jnlpMatchIP`	`add-on`	Specifies whether a JNLP file is rejected if launched from an IP address that differs from the IP address where the JNLP file was generated. This can help ensure the JNLP file is not moved from one machine to another. **Note:** This parameter may not work correctly when used behind some proxy servers, such as a load balancer or Oracle HTTP Server. In such a case, disable this setting. Default: `FALSE`

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
`jnlpTimeout`	`add-on`	Specifies the duration (in seconds) a Java Web Start (JWS) jnlp file should be valid. After the timeout period, the jnlp file is no longer valid and must be refreshed in order to run the application. Valid values: Whole numbers (`0` or greater). `0` or an invalid value disables the timeout. Default: `600` seconds when using JWS with SSO and `0` when using JWS without SSO.
`jnlp2FA`	`add-on`	Specifies whether to enforce the use of a verification code in order to start the application when launched using Java Web Start (JWS/jnlp). When enabled, a verification code will be presented in the browser if the application was requested through the Forms Servlet (dynamically). This code must be entered in the application when prompted. This behavior is similar to two-factor authentication except that the code is provided on the same device from where the initial request was made. This ensures the jnlp file is launched on the same machine from where it was requested and it has not been shared. Default: `FALSE`
`jpi_classid`	`plugin`	The Oracle Java Plug-in class ID. `formsweb.cfg` specifies an appropriate value. This parameter is required. **Note:** This parameter applies to the Java Plugin and embedded applets. The feature is desupported but retained for backward compatibility only. It is recommended that Java Web Start or the Forms Standalone Launcher be used to run applications. Default: `clsid:8AD9C840-044E-11D1-B3E9-00805F499D93`
`jpi_codebase`	`plugin`	The location from which the Java Plug-in (not the applet) can be downloaded if necessary. Currently unused because Java itself no longer provides a default location that would work with this value.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
jpi_download_page	plugin	The Oracle Java Plug-in download page. formsweb.cfg specifies an appropriate value. **Note:** This parameter applies to the Java Plugin and embedded applets. The feature is desupported but retained for backward compatibility only. It is recommended that Java Web Start or the Forms Standalone Launcher be used to run applications. Default: https://www.oracle.com/java/technologies/downloads
jpi_mimetype	plugin	The value of the TYPE applet parameter. Required. Default: application/x-java-applet
legacy_lifecycle	advanced	Applet parameter for Oracle Java Plug-in. A value of TRUE causes a running applet to be reused when requested. This parameter also affects the contents of the initial page that is generated as the response from the Forms servlet, to ensure the reusability of the applet when legacy_lifecycle is set to TRUE. When set to TRUE, JavaScript must be enabled on the Java client. **Note:** This parameter applies to the Java Plugin and embedded applets. The feature is desupported but retained for backward compatibility only. It is recommended that Java Web Start or the Forms Standalone Launcher be used to run applications. Default: FALSE
log	trace	Supports tracing and logging. The value of this parameter, if set, is the file name of the trace log file. This parameter is a sub-argument for otherparams. This parameter is treated as an application system parameter even if the otherparams parameter is not specified in the restrictedURLparams parameter.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`logo`	`applet`	Specifies the image file that should appear at the Forms menu bar. Specify `NO` for no logo. Leave empty to use the default Oracle logo.
`logoClickURL`	`add-on`	Specifies a URL for the menu bar logo icon hyperlink. When a value is provided, users see a mouse hand icon when hovering over the logo. Clicking the logo opens the specified URL in a web browser.
`logoutTargetURLParamName`	`advanced`	The name of the URL parameter which specifies the URL to which Oracle Access Manager will redirect after performing a logout. This parameter is an application system parameter. Default: `end_url`
`lookAndFeel`	`applet`	Specifies the application's look-and-feel. Valid values [case-insensitive]: `Oracle` or `Generic` (Windows look-and-feel) Default: `Oracle`
`maxeventwait`	`add-on`	The interval, in milliseconds, at which a client sends a packet to the server to indicate that it is still running and check for new events that might have occurred. This parameter is similar to `heartbeat`: the lower interval takes precedence. This parameter is an application system parameter. **Caution:** Setting this value too low will cause a significant increase in network traffic and therefore should be use cautiously.
`maxRuntimeProcesses`	`add-on`	The maximum allowable number of concurrent Forms run-time processes. It should be set a value that reflects the customer's hardware configuration (and the portion that can be used by Forms applications). A value of `0` (the default) indicates that there is no explicit limit. This default is not recommended, because it leaves the system vulnerable to Denial of Service attacks. This parameter is a global system parameter. It must be specified in the default config section of `formsweb.cfg`. The value is read from `formsweb.cfg` on every request to the Forms servlet. Default: `0`
`networkRetries`	`advanced`	Number of times the client should retry if a network failure occurs. Default: `0`

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`networkStats`	`add-on applet param`	Specifies whether to enable the display of the aggregate statistics in the status bar and the display of round-trip statistics in the Java console. Default: `FALSE`
`oam_redirect_root`	`servlet`	Used in cases where mixed protocols are used between the end-user, OHS, Forms, and OAM. Valid values include `HTTP`, `HTTPS`, or a fully qualified URL to the server hosting Forms—for example,*`<protocol>`*`://`*`<client-visible-host-name.domain>`*`:`*`<client-visible-port>`*.
`OAuth2AuthorizationTimeout`	`advanced`	Specifies the amount of time, in seconds, to wait for OAuth2 authorization to be completed and control to return to the Forms application. Default: `300`
`obr`	`advanced`	For internal use only. This parameter is a sub-argument for `otherparams`. Default: `NO`
`otherparams`	`advanced`	Command line parameters to pass to the Forms run-time process in addition to form and userid. This parameter is a runform parameter. **Note:** Special syntax rules apply to this parameter when it is specified in a URL: a + may be used to separate multiple name=value pairs. See Specifying Special Characters in Values of Runform Parameters. For production environments, include the `otherparams` parameter in the value of the `restrictedURLparams` parameter, in order to provide better control over which runform parameters end users can specify in a URL. Default: `obr=%obr% record=%record% tracegroup=%tracegroup% log=%log% term=%term% ssoProxyConnect=%ssoProxyConnect%`
`pageTitle`	`html`	HTML page title. Default: `Oracle Fusion Middleware Forms Services`

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
`pingStats`	`add-on applet param`	Specifies whether to enable the pinging of the managed server by the Java applet when Forms is being rendered. The ping result is, then, displayed on the Java console. Default: `FALSE`
`pingWait`	`add-on applet param`	The maximum amount of time (in milliseconds) that pinging of the managed server (specified by the pingStats parameter) must wait to receive a response from the server. Default: `300`
`port`	`trace`	Port to use for debugging. The value of this parameter identifies the port on which the forms engine process is listening and is used for debugging purposes only. This parameter is ignored if `serverURL` has been specified. This parameter is a runform parameter. Default: `9000`
`preserveHttpSessionId`	`add-on`	Specifies whether the HTTP session ID should be preserved after being made visible. This can happen during SSO authentication in two cases: • For a Web Start application, the HTTP session ID will be visible in the generated JNLP file. • For a Standalone application, the HTTP session ID will be visible in the browser's URL address bar in the dynamic resource creation page, if URL rewriting is being used for session management [cookies-enabled specified as false in `weblogic.xml` or `weblogic-application.xml`]. A value of `FALSE` indicates that the HTTP session ID should be changed after being made visible. **Caution:** Changing the HTTP session ID might break certain load balancer configurations. This parameter is an application system parameter. Default: `FALSE`
`prestartIncrement`	`add-on`	The number of run-time processes to be created when the number of prestarted run-time processes is less than `prestartMin`. This parameter is a global system parameter. For each config section, the value specified or inherited by that config section applies to any prestarting for the config section. The value is read from `formsweb.cfg` whenever a prestarted runform process is taken from the pool for the config section. Default: `1`

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
prestartInit	add-on	The number of the run-time processes that should be spawned initially. This parameter is a global system parameter. For each config section, the value specified or inherited by that config section applies to any prestarting for the config section. The value is read from formsweb.cfg only during frmservlet initialization. Default: 1
prestartMin	add-on	The minimum number of run-time processes to exist in the pool. This parameter is a global system parameter. For each config section, the value specified or inherited by that config section applies to any prestarting for the config section. The value is read from formsweb.cfg whenever a prestarted runform process is taken from the pool for the config section. Default: 0
prestartRuntimes	add-on	Specifies whether run-time prestarting or pooling is enabled. This parameter is a global system parameter. For each config section, the value specified or inherited by that config section applies to any prestarting for the config section. The value is read from formsweb.cfg only during frmservlet initialization. Default: FALSE
prestartTimeout	add-on	Time in minutes after which all the prestarted processes of this pool (configuration section) is stopped. A run-time process is removed from the prestart pool after the client connection is made and thus is not stopped. This parameter is a global system parameter. For each config section, the value specified or inherited by that config section applies to any prestarting for the config section. The value is read from formsweb.cfg only during frmservlet initialization. Default: 0
query_only	add-on	Set this parameter to YES to prevent the end user from inserting, updating, or deleting records. This parameter is a sub-argument for otherparams. Default: NO
quiet	add-on	Set this parameter to YES to prevent messages from producing an audible beep. This parameter is a sub-argument for otherparams. Default: NO
record	trace	Supports tracing and logging. Valid values: • forms, used for standard tracing with a tracegroup. • names, used by external testing tools. This parameter is a sub-argument for otherparams.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
recordFileName	advanced	The name of the file (for example, d:\temp\out.log) that stores the recorded Forms messages.
removeCommentLinesFromApplet	add-on	Specifies whether to remove comment lines from html/jnlp sent to client. This can improve start up performance and reduce the amount of content being sent to user. This parameter is an application system parameter. Default: FALSE
restrictedURLchars	add-on	Comma-separated list of characters which are restricted from use in the request URL's query string. This parameter is an application system parameter.
restrictedURLparams	advanced	Comma-separated list of parameters which are rejected if specified in a URL. This parameter is an application system parameter. Default: pageTitle,HTMLbodyAttrs,HTMLbeforeForm,HTMLafterForm,log
scaleLogo	add-on applet param	Specifies the scaling behavior when clientDPI is increased from the default. If set to TRUE, all objects and fonts in a running form, including the logo on the right side of the menu bar, are properly magnified across all objects of the application. If set to FALSE, all objects and fonts, except for the logo, are sized accordingly. The logo remains unchanged. Default: TRUE
sendHeartBeatBean	add-on applet param	Set this parameter to TRUE to prevent Java Beans or PJCs that display modal dialogs/windows from blocking the Forms heartbeat. Default: FALSE
separateFrame	applet	Specifies whether the applet appears within a separate window. **Note:** This parameter applies to the Java Plugin and embedded applets. The feature is desupported but retained for backward compatibility only. It is recommended that Java Web Start or the Forms Standalone Launcher be used to run applications. Default: FALSE

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`separate_jvm`	`advanced`	A Boolean parameter specifying that the applet should run in its own JVM instance on the user environment. This is useful for applets which can not tolerate any interference from other applets running in the same JVM and potentially consuming heap space or other resources. This setting only applies to running applications embedded in a browser like Microsoft Internet Explorer. **Note:** This parameter applies to the Java Plugin and embedded applets. The feature is desupported but retained for backward compatibility only. It is recommended that Java Web Start or the Forms Standalone Launcher be used to run applications. Default: `FALSE`
`serverApp`	`advanced`	Specifies the name of the application-specific Registry.dat entries to be used. The value of `serverApp` must correspond to the attribute `NAME` in the Registry.dat. For example, Registry.dat may include two or more default Fontname settings: • `default.fontMap.defaultFontname=Dialog` (A default Oracle-provided setting); and • `sales.fontMap.defaultFontname=Dialog` (A custom setting for Sales). To use the Sales custom settings, set `serverApp` to `sales`. Otherwise, leave `serverApp` set to the default (`default`). **Note:** Only entries that begin with `default` in `Registry.dat` can be customized using the `serverApp` parameter. See Managing Registry.dat with Fusion Middleware Control. Default: `default`

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
serverURL	advanced	The URL path to Forms Listener Servlet. This parameter is an application system parameter. Required. Default: /forms/lservlet
sessionCookieName	add-on	The cookie-name (if any) specified in weblogic.xml or weblogic-application.xml. Required for SSO processing if a cookie-name other than JSESSIONID is specified in weblogic.xml or weblogic-application.xml. This parameter is a global system parameter. It must be specified in the default config section of formsweb.cfg. The value is read from formsweb.cfg only during frmservlet initialization. Default: JSESSIONID
sessionCookiePath	add-on	The cookie-path (if any) specified in weblogic.xml or weblogic-application.xml. This is required for SSO processing if a cookie-path other than / is specified. This parameter is a global system parameter. It must be specified in the default config section of formsweb.cfg. The value is read from formsweb.cfg only during frmservlet initialization. Default: /
showMDITitleBar	add-on	Specifies whether the Forms MDI (parent) window title bar should be displayed. This setting is most often used in conjunction with the alwaysOnTop=true web configuration setting (see here) along with adding SET_WINDOW_PROPERTY (FORMS_MDI_WINDOW, WINDOW_STATE, MAXIMIZE) to the startup module's WHEN-NEW-FORM-INSTANCE trigger in order to achieve kiosk-like functionality. Default: TRUE
smartBarHeight	add-on applet param	Specifies the desired Smartbar menu size to be used at runtime. Smartbar icons larger than the button size will be clipped. Smartbar icons smaller than the button size will scale up to fill the button, unless the smartbarIconScaling parameter is set to FALSE. The default (for example, unset) or a setting to any value other than those listed here will result in buttons fixed to contain icons that are 16x16. These buttons will not scale if the value of clientDPI is changed. Valid values [case-insensitive]: medium (32x32), large (48x48), dynamic (adapts based on value of clientDPI).
smartbarIconScaling	add-on applet param	Specifies whether Smartbar icons smaller than the button size will scale up to fill the button. Default: TRUE

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
`smoothScalingMaxZoom`	`add-on applet param`	Smooth scaling often produces a significant improvement in image quality when the scaling factor is significantly lower than 100%. This improvement decreases as scale increases, and seems unnoticeable when the scaling factor is significantly greater than 100%. The downside of smooth scaling is that a large scaling factor may produce an `OutOfMemoryError` [which will leave the image item blank]; the probability increases as the scaling factor increases. Smooth scaling is by default not enabled. This parameter specifies the tradeoff between image quality and the probability of an `OutOfMemoryError`. It specifies the maximum zoom percent for which smooth scaling will be used. Fractional values are allowed. For example, set `smoothScalingMaxZoom=200`. This would specify that smooth scaling should be used for all downscaling, and for upscaling up to a factor of 2 (200%). Note that `smoothScalingMaxZoom` can be specified in a URL. An end user might want to lower it or disable it altogether [by specifying `smoothScalingMaxZoom=0`] if an `OutOfMemoryError` occurred. [Even if the Java console is not visible, a blank image item would suggest that an `OutOfMemoryError` had occurred.]
`splashScreen`	`applet`	Specifies the image file that should appear before the applet appears. Specify `NO` for no splash. Leave empty to use the default splash image. Supported image formats are: `gif`, `png`, and `jpg`. To set the parameter include the file name (for example, `myfile.gif`) or the virtual path and file name (for example, `images/myfile.gif`).
`ssoCancelUrl`	`sso`	The URL to redirect to if the user presses Cancel in the dynamic resource creation page. This parameter is an application system parameter.
`ssoDynamicResourceCreate`	`sso`	Specifies whether dynamic resource creation is enabled if the resource is not yet created in OID (Oracle Internet Directory) or OPSS (Oracle Platform Security Services). If set to TRUE, users are presented with a web page where they are prompted to provide the necessary database credentials needed to run the application. This parameter is an application system parameter. Default: `TRUE`
`ssoErrorUrl`	`sso`	The URL to redirect to if `ssoDynamicResourceCreate` is set to `FALSE`. This parameter is an application system parameter.
`ssoLogout`	`sso`	Specifies whether the SSO session should be logged out while exiting the Forms application. This parameter is an application system parameter. Default: `FALSE`
`ssoLogoutRedirect`	`sso`	The URL which the browser should be redirected to after SSO logout. This parameter is an application system parameter.

Table 66   (Cont.) Web Configuration Parameters

Parameter	Category	Description
ssoMode	sso	Specifies whether to enable single sign-on (SSO). This parameter is an application system parameter. Default: FALSE
ssoProxyConnect	sso	Specifies whether session should operate in proxy user support. Set ssoProxyConnect to YES to enable for particular application. This parameter is a sub-argument for otherparams. Default: NO
ssoSaaBrowserLaunchTimeout	add-on	Specifies, in seconds, how long the Forms servlet will wait for the initial request from the browser that was spawned by the Standalone launcher for SSO authentication. If the interval expires, the fatal error FRM-93249 is reported. This parameter is an application system parameter. Valid values: Integers in the range 1-300 Default: 15
ssoSaaBrowserPageTimeout	add-on	Specifies, in seconds, how long the Forms servlet will wait for the user to enter data into a browser page during SSO authentication for a Standalone application. If the interval expires, the fatal error FRM-93382 or FRM-93383 is reported. This parameter is an application system parameter. Valid values: An integer >= 15, or 0 (which indicates that the Forms servlet will wait indefinitely) Default: 300
ssoSaaWaitInterval	add-on	Specifies the interval, in seconds, at which the Standalone launcher reissues requests directly to the Forms servlet while SSO authentication is proceeding in the launched browser window. Larger values reduce network traffic but increase the chances of an intermediate agent timing out (thereby producing the fatal error FRM-93248). This parameter is an application system parameter. Valid values: An integer >= 5, or 0 (which indicates that the Standalone launcher should not reissue requests) Default: 25
ssoSuccessLogonUrl	sso add-on	The URL to redirect to if SSO authentication completes successfully for a Standalone application. This parameter is an application system parameter.
tabstop	add-on applet param	Specifies the tab stop size for all multi-line text fields in the application. Valid values: A positive integer no greater than the total character width of any field in which tabs are expected to be used. Default: 4
term	advanced	The full path of a custom key binding file (to be used instead of the standard fmrweb or fmrweb_utf8 files). This parameter is a sub-argument for otherparams.

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`tracegroup`	`trace`	Supports tracing and logging. This parameter is a sub-argument for `otherparams`.
`treeIconHeight`	`add-on`	Specifies an alternative hierarchical tree icon size. If not specified, h-trees will only display 16×16 pixel icons when added to a tree node. Icons larger than the set size will be clipped. Icons smaller than the set size will scale up to fill the available space, unless the `treeIconScaling` parameter is set to `FALSE`. The default (not set) or any setting to a value other than those listed here will result in icons that are 16×16. These will not scale if the value of clientDPI is changed. **Note:** The space between the tree nodes is determined by the font size used in the tree. Adjusting the font size may be necessary in order to avoid having nodes overlap or become excessively spaced apart when using various icon sizes. Valid values [case-insensitive]: medium (32×32), large (48×48), dynamic (adapts based on value of clientDPI).
`treeIconScaling`	`add-on`	Specifies whether hierarchical tree (h-tree) icons that are smaller than the tree node size scale up to fill the available space. Default: `TRUE`
`useAntiAlias`	`add-on`	Specifies whether to use anti-aliasing for rendering Forms graphic objects like lines, rectangles, ellipse, polyline, and so on. Setting this to `TRUE` reduces the jagged edges that can appear with drawing objects. Although setting this to `FALSE` appears to sharpen the drawing objects, it does increase the jagged edges. Default: `TRUE`
`userid`	`basic`	Login string. Example: `user1/user1@ORADB`. This parameter is a runform parameter.
`websocketJSILogging`	`add-on applet param`	Specifies whether to enable WJSI logging. Output will be sent to the Java Console. Default: `FALSE`
`websocketJSIServerTimeout`	`add-on applet param`	The time, in minutes, that the WJSI server should continue to run while idle. After this time, the value specified by the `FORMS_TIMEOUT` environment variable will be honored. Default: `0`

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`websocketJSISessionTimeout`	`add-on applet param`	The time, in minutes, that the WJSI session should be permitted to remain idle before being terminated. Default: `5`
`webstart`	`plugin`	Specifies whether Web Start should be enabled. This parameter is an application system parameter.
`webstart_codebase`	`add-on`	The URL to the application Codebase. This may be required when using Java Web Start and the Forms managed server (WLS) is behind a proxy server. Example: `webstart_codebase=http://proxy_server:port/forms/java` **Note:** The use of this parameter is often not necessary when the WebLogic setting, `WebLogic Plug-in`, is enabled for either all associated Managed Servers (such as `WLS_FORMS`) or the responsible Cluster (such as `cluster_forms`). This parameter is an application system parameter.
`WebUtilArchive`	`add-on`	A comma-separated list of archives (JARs) needed to support a WebUtil-enabled application. The Oracle provided `frmwebutil.jar` file must be included in this list. Additional files are optional, based on the application design. **Note:** The contents of this list are appended to the content of the `ARCHIVE` parameter. Therefore, use caution to avoid duplicating entries.
`WebUtilDispatchMonitorInterval`	`add-on`	Used with the `WebUtil_Session` package to control how often the monitor thread checks to see if the Forms session is still alive. The value is measured in seconds. Refer to the WebUtil Guide in the Builder Help. Default: `5`

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
WebUtilErrorMode	add-on	Specifies where errors are displayed. Errors will always be displayed on the Java console but they can also be displayed elsewhere. Valid values are: console, server, alert, and all **Note:** Refer to the WebUtil Guide in the Builder Help. Default: CONSOLE
WebUtilLogging	add-on	Specifies the logging level for WebUtil. Valid values include: off, console, server, and all **Note:** Refer to the WebUtil Guide in the Builder Help. Default: OFF
WebUtilLoggingDetail	add-on	Specifies the level of detail WebUtil logging should record. Valid values include: normal and detailed **Note:** Refer to the WebUtil Guide in the Builder Help. Default: NORMAL

Table 66 (Cont.) Web Configuration Parameters

Parameter	Category	Description
`WebUtilMaxTransferSize`	`add-on`	Specifies the maximum segment size (in bytes) to break up a file when transferring it with WebUtil. Larger values can improve performance, but may cause failures on low bandwidth or unstable networks. A valid value must be a whole number from 1 - 24573. A setting of 24573 is the maximum and recommended value for most cases, as it provides the best transfer performance. **Note:** Refer to the WebUtil Guide in the Builder Help. Default: `24573`
`WebUtilNextGenHost`	`add-on`	Specifies whether to use the new Java security model needed to make external operating system commands when using `CLIENT_HOST`. In most cases, this setting should not be changed from its default. Default: `TRUE`
`WebUtilTrustInternal`	`add-on`	Specifies whether to trust computers within an intranet. When set to `TRUE` (default), this indicates that all the intranet computers are trusted. When set to `FALSE`, users always need to type the domain suffixes to match the domain filter in `TrustedDomains.txt`. **Note:** Refer to the WebUtil Guide in the Builder Help. Default: `TRUE`
`WebUtilVersion`	`add-on applet param`	Specifies whether to display the WebUtil version in the client side console. Default: `FALSE`
`width`	`basic`	The width of the form applet, in pixels. You can also specify the value of the width in percentage. This value is relative to the size of the content area of the browser. The value should not exceed 100% or be less than 1%. To use a percentage value, the numeric value must be followed by a percentage (%) sign, like this: `WIDTH=75%`. This example specifies that the applet width is 75% of the size of the browser's content area. Required. Default: `750`

Font, Image, and Style Mapping

This chapter describes font, image, and style mapping parameters.

Table 67   Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`app.ui.lovButtons`	Specifies whether to show a button within text fields that have associated lists of values (LOVs)	The presented button uses a small part of the horizontal space assigned to the entire field. As a result, the total width of the item is slightly reduced in order to accommodate this button. The button is only displayed when the item has focus. The button cannot be customized. This lovButton should not be used in conjunction with the item property, `Conceal Data Button`. When enabled, the design-time item property, `LOV Button`, is not honored at runtime. Default: `FALSE`
`app.ui.requiredFieldVA`	Specifies whether a required field should be displayed with an alternative background color	Valid values: `TRUE`, `FALSE` Default: `FALSE`
`app.ui.requiredFieldVABGColor`	Specifies the background color for required fields	Enabled by setting `app.ui.requiredFieldVA` to `TRUE`. Enabling this overrides the design-time item property, `Background Color`. The color value must be entered using the RGB format (such as `255,0,100`). Default: `255,0,0`
`app.ui.requiredFieldVABorderColor`	Specifies the border color of items that are identified as "Required" at design time or runtime.	When set to a valid RGB color (such as `255,0,0`) and the item's `Data Required` property is set to `YES`, the item's border displays the set color for that item in the current record. Upon navigating away from the current record, the border returns to its previous appearance. This border setting takes precedence over other border settings. The required border color appears on all four sides of the item, regardless of the selected border sides.

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`colorScheme.<NAME>.dark`	Specifies the color to be used for the Forms virtual color, "dark"	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.darker`	Specifies the color to be used for the Forms virtual color, "darker"	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.darkest`	Specifies the color to be used for the Forms virtual color, "darkest"	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.description`	Description for this custom color scheme	The description is for documentation purposes only and cannot be accessed within the running application.

Table 67   (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`colorScheme.<NAME>.desktop`	Specifies the color to be used for the Forms default menu bar, message/status bar, and applet container background	This is only visible when run embedded in a browser. The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.light`	Specifies the color to be used for the Forms virtual color, "light"	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.lighter`	Specifies the color to be used for the Forms virtual color "lighter"	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`colorScheme.<NAME>.lightest`	Specifies the color to be used for the Forms virtual color, "lightest"	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.pinstripe1`	Specifies the color to be used for the Forms virtual color, `OLAF3`. "pinstripe1" refers to the first stripe color of the two used when row banding is enabled.	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.pinstripe2`	Specifies the color to be used for the Forms virtual color, `OLAF4`. "pinstripe2" refers to the second stripe color of the two used when row banding is enabled.	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.

Table 67   (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`colorScheme.<NAME>.selection`	Specifies the color to be used for text selection	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`colorScheme.<NAME>.smartBar`	Specifies the color to be used for the Forms smartbar	The Forms applet parameter, `customColorScheme=<NAME>` is required to enable this palette. All colorScheme entries must have valid values. If any values are invalid or null, `customColorScheme` is ignored. This parameter is not supported with the "Generic" look-and-feel. The color value must be entered using either RGB (for example `255,0,100`) or hex (for example `0xFF0064`) format.
`default.border.bottom`	Specifies whether to show the bottom border of the object	The item property. `Show Border Sides`, must be set to `Yes` to enable this feature. When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: `<serverApp value>.border.bottom` Valid values: `TRUE`, `FALSE` Default: `TRUE`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
default.border.color	Specifies the object border color	The color value must be entered using the RGB format (such as 255,0,100). The Text item property, Show Border Color, must be set to Yes to enable this feature. When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: *<serverApp value>*.border.color Default: 255,0,0
default.border.highlightColor	Specifies the alternative object border color that can be used to raise immediate visual awareness to the item.	This parameter is intended to bring attention to an item, for example, when a user enters an invalid password. The color value must be entered using the RGB format (such as 255,0,100). When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: *<serverApp value>*.border.highlightColor Default: 255,0,0
default.border.left	Specifies whether to show the left border of the object	The item property. Show Border Sides, must be set to Yes to enable this feature. When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: *<serverApp value>*.border.left Valid values: TRUE, FALSE Default: TRUE

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.border.right`	Specifies whether to show the right border of the object	The item property. `Show Border Sides`, must be set to `Yes` to enable this feature. When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: *`<serverApp value>`*`.border.right` Valid values: `TRUE`, `FALSE` Default: `TRUE`
`default.border.rolloverColor`	Specifies the object border color to present when the mouse passes over the object	The color value must be entered using the RGB format (such as `255,0,100`). The Text item property, `Rollover Border Color`, must be set to `Yes` to enable this feature. When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: *`<serverApp value>`*`.border.rolloverColor` Default: `255,0,0`
`default.border.top`	Specifies whether to show the top border of the object	The item property. `Show Border Sides`, must be set to `Yes` to enable this feature. When running with the Oracle look-and-feel, this parameter applies to edit/text items, display items, poplists, combo-boxes, tlists, spinlists, and image items. This parameter is not supported with the "Generic" look-and-feel. Syntax: *`<serverApp value>`*`.border.top` Valid values: `TRUE`, `FALSE` Default: `TRUE`

Table 67   (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.concealedData.character`	Specified the character to be used for concealed data fields	Value can be one of most typeable characters such as `x`. Some non-typeable special characters can be used. To do so, enter `\u` followed by the Unicode identifier for the character. For example, use `\u25cf` to display a large dot. Syntax: *`<serverApp value>`*`.concealedData.character`
`default.fontMap.appFontnames`	Specifies a comma delimited list of application font names used for mapping to Java equivalent fonts	Represents the font names that most closely match those used in the application at design-time. The number of entries in the `appFontnames` list must match the number in the `javaFontnames` list. The elements of the list are comma separated and all characters are taken literally. Leading and trailing spaces are stripped from face names. Syntax: *`<serverApp value>`*`.fontMap.appFontnames`
`default.fontMap.defaultFontname`	Specifies the default Java font name that should be used for objects that do not have a specific real font name defined at design-time or runtime	The value for `defaultFontname` must be a font name that appears in the list of Java font names (from `javaFontnames`). Syntax: *`<serverApp value>`*`.fontMap.defaultFontname`

Table 67   (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.fontMap.defaultMapping`	Specifies whether to use the built-in Java font mapping or to map to the font set in the application	Valid values: - `FULL`: The built-in Java font mapping is used. If a specified font is not found in the `appFontnames` list, then the default font is used. - `PARTIAL`: If an application's specified font is not found in the `appFontnames` list, the application's requested font is passed directly to Java Java attempts to use the specified font if found on the user's computer. - `NONE`: The built-in Java font mapping is disabled, and the requested font is passed directly to Java Java attempts to use the specified font if found on the user's computer. Syntax: *`<serverApp value>`*`.fontMap.defaultMapping` Default: `FULL`
`default.fontMap.defaultSize`	Specifies the default font size that should be used for objects that do not have a specific font size defined	The value for font size is the desired point size multiplied by 100. For example, 9 pt should be entered as 900. Syntax: *`<serverApp value>`*`.fontMap.defaultSize`
`default.fontMap.defaultStyle`	Specifies the default font style that should be used for objects that do not have a specific font style defined	Syntax: *`<serverApp value>`*`.fontMap.defaultStyle` Valid values: `PLAIN`, `ITALIC`
`default.fontMap.defaultWeight`	Specifies the default font weight that should be used for objects that do not have a specific font weight defined	Syntax: *`<serverApp value>`*`.fontMap.defaultWeight` Valid values: `PLAIN`, `BOLD`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.fontMap.javaFontnames`	Specifies a comma delimited list of Java font names	The number of entries in the `javaFontnames` list must match the number in the `appFontnames` list. The elements of the list are comma separated and all characters are taken literally. Leading and trailing spaces are stripped from face names. There is a one-to-one positional relationship between the `appFontnames` and `javaFontnames`. Therefore, it is possible that some Java logical font names may be used more than once in the list. For more information about Java logical fonts, see Physical and Logical Fonts in the Java documentation. Syntax: `<serverApp value>.fontMap.javaFontnames`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.gradient.blueDelta`	Specifies the blue gradient color range beginning from the source color for the canvas background	Uses a range of `-255` to `255`. Any value that results in the color value going beyond either `0` or `255` is automatically limited to `0` or `255`. Using a positive value, such as `200`, causes the color range to extend toward white. A negative value, such as `-200`, causes the color range to extend in the black direction. **Note:** The Canvas property, `GradientStartSide`, must be set to a value other than `None` to enable this feature. Syntax: *`<serverApp value>`*`.gradient.blueDelta` Default: `200`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.gradient.greenDelta`	Specifies the green gradient color range beginning from the source color for the canvas background	Uses a range of `-255` to `255`. Any value that results in the color value going beyond either `0` or `255` is automatically limited to `0` or `255`. Using a positive value, such as `200`, causes the color range to extend toward white. A negative value, such as `-200`, causes the color range to extend in the black direction. **Note:** The Canvas property, `GradientStartSide`, must be set to a value other than `None` to enable this feature. Syntax: *`<serverApp value>`*`.gradient.greenDelta` Default: `200`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.gradient.redDelta`	Specifies the red gradient color range beginning from the source color for the canvas background	Uses a range of `-255` to `255`. Any value that results in the color value going beyond either `0` or `255` is automatically limited to `0` or `255`. Using a positive value, such as `200`, causes the color range to extend toward white. A negative value, such as `-200`, causes the color range to extend in the black direction. **Note:** The Canvas property, `GradientStartSide`, must be set to a value other than `None` to enable this feature. Syntax: *`<serverApp value>`*`.gradient.redDelta` Default: `200`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.icons.iconextension`	Specifies the default icon file extension to use if one is not provided at design-time or runtime	Valid values: `GIF`, `JPG`, `PNG` Syntax: `<`*`serverApp value`*`>.icons.iconextension` Default: `gif`
`default.icons.iconpath`	Specifies the virtual path where icons can be found	The application level icon files are located relative to the `DOCUMENTBASE`. Either a relative or fully-qualified path can be used for the value. Syntax: `<`*`serverApp value`*`>.icons.iconpath`
`default.icons.style`	Specifies whether to use legacy icons for the Forms built-in smartbar and alerts	Setting to `LEGACY` results in the use of icons from versions prior to 12.2.1.*x*. Leaving unset results in the use of higher resolution and updated icons. Syntax: `<`*`serverApp value`*`>.icons.style` Valid values: `LEGACY` Default: `NULL`
`default.lovRowLine.color`	Specifies the horizontal line color between entries in all list of values (LOVs)	The color value must be entered using the RGB format (such as `255,0,100`). If no value is provided or an invalid value is provided, lines will not appear in the LOV. Each line is presented as a row of small dots. Solid lines are not supported. Syntax: `<`*`serverApp value`*`>.lovRowLine.color`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.luminance.dark`	Specifies the luminance value to be used as a threshold to identify a "dark" color.	This parameter is used by Forms to determine whether the background color for some item types is light or dark. Forms uses this value to choose appropriately between two colors for the foreground in order to ensure readability: using a light color when the background is "dark" and a dark color when the background is "light". The luminance value is on a scale of `0` to `255` with darker colors having lower values. If this variable is not set, Forms uses `139` as the default threshold for that determination. Valid values: `0-255` Default: `139`
`default.placeholder.color`	Specifies the color to be used for placeholder and persistent placeholder text.	The color value must be entered using the RGB format (such as `255,0,100`). If no value is provided, the current foreground color is used, although muted. Syntax: *`<serverApp value>`*`.placeholder.color`
`default.tabBar.transparent`	Specifies whether the tabbed canvas tab bar background should be transparent. When set to `TRUE`, any object behind the bar is visible in areas where there are no tabs (typically at the end of the bar).	This setting does not change the behavior of the tab labels or the tabbed canvas. This parameter is not supported with the "Generic" look-and-feel. Syntax: *`<serverApp value>`*`.tabBar.transparent` Default: `FALSE`
`default.tabLabel.selectedBgColor`	Specifies the tab label background color to use when the tab is selected	The color value must be entered using the RGB format (such as `255,0,100`). This parameter is not supported with the "Generic" look-and-feel. Syntax: *`<serverApp value>`*`.tabLabel.selectedBgColor`

Table 67 (Cont.) Font, Image, and Style Mapping Parameters

Parameter	Description	Usage Notes
`default.tabLabel.selectedFgColor`	Specifies the tab label foreground color to use when the tab is selected	The color value must be entered using the RGB format (such as `255,0,100`). This parameter is not supported with the "Generic" look-and-feel. Syntax: `<`*`serverApp value`*`>.tabLabel.selectedFgColor`
`default.tabLabel.unselectedBgColor`	Specifies the tab label background color to use when the tab is selected	The color value must be entered using the RGB format (such as `255,0,100`). This parameter is not supported with the "Generic" look-and-feel. Syntax: `<`*`serverApp value`*`>.tabLabel.unselectedBgColor`
`default.tabLabel.unselectedFgColor`	Specifies the tab label foreground color to use when the tab is not selected	The color value must be entered using the RGB format (such as `255,0,100`). This parameter is not supported with the "Generic" look-and-feel. Syntax: `<`*`serverApp value`*`>.tabLabel.unselectedFgColor`

Error Messages

This chapter gives you general information and helpful tips about error messages.

FRM-10200: Illegal function in this context.
Cause: You pressed a key that is not valid in this context.

Action: Press [Show Keys] to view a list of valid function keys.
Level: 25
Trigger: None

FRM-10201: No parameters needed.
Cause: You pressed [Enter Application Parameters] or [Enter Menu Parameters], but none are required in this context.

Action: No action required.
Level: 25
Trigger: None

FRM-10202: Menus are nested too deeply.
Cause: You tried to select an item that would nest menus more than 10 deep.

Action: Press [Main Menu] to return to the main menu, then navigate to the menu of your choice.
Level: 25
Trigger: None

FRM-10203: Selected item is not in this menu.
Cause: In a full-screen menu, you entered a number that exceeds the maximum number of menu items.

Action: Choose an item that is on this menu.
Level: 25
Trigger: None

FRM-10204: No command defined for the selected background item.
Cause: You pressed [Background Menu n], where n was greater than the maximum number on the background menu.

Action: No action required. Press [Show Background Menu] to see the valid background menu items.
Level: 25
Trigger: None

FRM-10205: Menu %s not found.
Cause: In the choice field of a full-screen menu, you entered a menu name that does not exist in this application or is not found in the library.

Action: No action is required if the menu does not exist in the application. If it does, recompile the library.
Level: 25
Trigger: None

FRM-10206: memory allocation failure
Cause: A memory allocation failed when Forms Runtime attempted a menu operation.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 25
Trigger: None

FRM-10207: No background menu present.
Cause: You pressed [Show Background Menu], but no background menu exists.

Action: No action required.
Level: 25
Trigger: None

FRM-10208: Parameter %s not found.
Cause: A menu item referenced an undefined parameter.

Action: Contact your DBA.
Level: 25
Trigger: None

FRM-10209: No next menu from background in this context.
Cause: The application attempted to navigate to a named menu from the background menu.

Action: No action required.

Level: 25
Trigger: None

FRM-10210: Response required.
Cause: You did not enter a required parameter, or you left the choice field blank in a full-screen menu.

Action: Make an entry.
Level: 25
Trigger: None

FRM-10211: Field must be filled completely.
Cause: You partially entered a parameter that must be entered completely.

Action: Enter enough data to completely fill the field.
Level: 25
Trigger: None

FRM-10212: Login failed for this username and password.
Cause: You specified an illegal username and password.

Action: Check the username and password and try again.
Level: 25
Trigger: None

FRM-10213: Login procedure terminated.
Cause: You failed to logon to ORACLE three times in a row.

Action: No action required. If you are a valid user, check your user name and password.
Level: 25
Trigger: None

FRM-10214: No authorization to run any application.
Cause: You are not a valid user of any module in Oracle Forms.

Action: No action required. If you think that you should be a valid user, ask your DBA to grant you access to the module you wish to run.
Level: 25
Trigger: None

FRM-10215: No help available.
Cause: You pressed [Help], but none is available for this item.

Action: No action required.
Level: 25
Trigger: None

FRM-10216: Failed to spawn a command to the operating system.
Cause: The operating system could not spawn a sub-process.

Action: Refer to the error message that the operating system issued.
Level: 25
Trigger: None

FRM-10217: No authorization for any item in selected menu.
Cause: You tried to move to a menu that has no items you can access.

Action: Check the menu name you entered and try again.
Level: 25
Trigger: None

FRM-10218: Error for menu %s.
Cause: Oracle Forms could not read the library information for this menu, or an invalid menu name was specified.

Action: Recompile the library or correct the menu name.
Level: 25
Trigger: None

FRM-10219: Item number is invalid.
Cause: In a full-screen menu, you entered an invalid number in the choice field.

Action: Check the item number and re-enter it.
Level: 25
Trigger: None

FRM-10220: No detailed help available for this item.
Cause: You pressed [Help], but none is available for this menu item.

Action: No action required.
Level: 25
Trigger: None

FRM-10221: Cannot read file %s.
Cause: Either file privileges are set incorrectly, or the library you tried to open is invalid.

Action: Recompile the application library and try again.
Level: 25
Trigger: None

FRM-10222: Menu %s was created by an old version of the Form Compiler.
Cause: You are using a newer version of Oracle Forms than the one that created this menu module.

Action: Recompile the menu module and re-execute the command.
Level: 25
Trigger: None

FRM-10223: Application parameter module does not exist.
Cause: The parameter information could not be located in the library. This may be due to a library file that is invalid, or one that contains a different application.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25
Trigger: None

FRM-10224: Application bind variable module does not exist.
Cause: The bind variable information could not be located in the library. This may be due to an invalid library file.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25

Trigger: None

FRM-10225: Could not read parameter data.
Cause: The application library is invalid.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25
Trigger: None

FRM-10226: Could not read bind variable data.
Cause: The application library is invalid.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25
Trigger: None

FRM-10227: Too many menu parameters.
Cause: The application contains more menu parameters than can be used on your operating system.

Action: Revise and recompile the application, or contact your DBA.
Level: 25
Trigger: None

FRM-10228: Could not read help text.
Cause: The application library is invalid.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25
Trigger: None

FRM-10229: Could not close file %s.
Cause: Operating system error or internal error.

Action: Contact your DBA.
Level: 25
Trigger: None

FRM-10230: Application procedure module does not exist.
Cause: The procedure information could not be located in the library. This may be due to an invalid library file.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25
Trigger: None

FRM-10231: Could not read procedure data.
Cause: The application library is invalid.

Action: Recompile the application library and try again. If this is unsuccessful, contact your DBA.
Level: 25
Trigger: None

FRM-10233: Navigational procedures/macros not valid in current menu style.
Cause: You tried to use the full-screen, navigational packaged procedures, or macros in the pull-down or menu bar display style.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10234: Semicolon missing in macro statement.
Cause: The command line specified for this item has a syntax error.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10235: Macro %s not found.
Cause: The menu designer specified an undefined macro to be executed.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10236: No procedure/macro specified.
Cause: The menu designer has specified a blank command.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10237: Argument(s) not allowed for this procedure/macro.
Cause: The menu designer specified an argument to a command that does not take arguments.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10238: Error executing %s. Check argument(s).
Cause: The menu designer specified an argument to a command that does not take arguments.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10239: Cannot read form by that name.
Cause: Oracle Forms tried to read a form that does not exist in the current directory.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10240: Form name not specified.
Cause: Forms Runtime command did not give the name of a form to execute.

Action: Notify your DBA.
Level: 25

Trigger: None

FRM-10241: Illegal operation when the Form Builder is active.
Cause: The menu designer specified a Built-in or macro that cannot be used when the Form Builder calls Forms Runtime.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10242: Cannot call linked-in Forms from Oracle Forms.
Cause: The menu designer specified a call to linked-in Forms from within Oracle Forms.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10243: Error occurred during invocation of Oracle Forms.
Cause: A call to Forms Runtime failed.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10244: Application %s does not exist.
Cause: The application name you specified does not exist in the database, or you do not have access privileges to it.

Action: Check the application name and try again, or contact your DBA.
Level: 25
Trigger: None

FRM-10245: Already on first item.
Cause: You pressed [Previous Item] from the first item in the parameter form.

Action: No action required. You cannot go to an item prior to the first item in a parameter form.
Level: 25
Trigger: None

FRM-10246: Error executing packaged procedure - inactive form.
Cause: The menu designer specified a built-in that cannot be executed in the current context.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10247: No active items in root menu of application.
Cause: You tried to open an application but its root menu has no items that you can access. The root menu is either the application's main menu or another menu specified when Oracle Forms called Forms Designer.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10248: No direct menu selection allowed when using a root menu.
Cause: The root menu is not the module's main menu, because Oracle Forms specified another root menu when calling Forms Designer.

Action: No action required. You can only use direct menu selection when the module's main menu is the root menu.
Level: 25
Trigger: None

FRM-10249: No authorization to run application %s.
Cause: You are not a valid user of the application you tried to run.

Action: No action required. If you think you should be a valid user, ask your DBA to grant you access privileges to the application.
Level: 25
Trigger: None

FRM-10250: Error initializing Forms Runtime application.
Cause: You did not name the module properly.

Action: Check the module name and enter it correctly.
Level: 25
Trigger: None

FRM-10251: Unsupported command type 4 switch used (-e,-i,-r,-w).
Cause: This menu option attempted to run a form, but specified a command line argument for Forms Runtime which is invalid when running a form from a menu.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10252: Unknown command type 4 switch used.
Cause: This menu option attempted to run a form, but specified an unknown command line argument for Forms Runtime.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10253: File name must be entered.
Cause: You have not entered a name (or you have deleted a name) for the file.

Action: You must enter a file name.
Level: 25
Trigger: None

FRM-10254: Cannot open file for screen shot.
Cause: The operating system could not open a file (e.g. permission problems, lack of disk space).

Action: Resolve the operating system condition that caused the error.
Level: 25
Trigger: None

FRM-10255: Error occurred during printing of screen shot.
Cause: The operating system had trouble with a file.

Action: Resolve the operating system condition that caused the error.
Level: 25
Trigger: None

FRM-10256: User is not authorized to run Oracle Forms Menu.
Cause: You are not enrolled in Oracle Forms. You do not have SELECT permission on the Oracle Forms base tables.

Action: Notify your DBA.
Level: 25
Trigger: None

FRM-10257: User is not authorized to select specified option.
Cause: You tried to select a menu item to which you do not have access.

Action: Choose another item or notify your DBA.
Level: 25
Trigger: None

FRM-10258: Specified menu is already active.
Cause: You tried to navigate to the current menu.

Action: No action required.
Level: 25
Trigger: None

FRM-10259: Invalid null argument to packaged procedure or function.
Cause: You did not specify an argument to a built-in, or the argument is invalid.

Action: Check the online Help for the proper built-in syntax.
Level: 25
Trigger: None

FRM-10260: No active items in selected menu.
Cause: You do not have access privileges for any items in this menu.

Action: Contact your DBA for access privileges if you think you should have access to the items on this menu.
Level: 25
Trigger: None

FRM-10261: Menu %s was created by a new version of the Form Compiler.
Cause: You are using an old version of Forms Runtime with a new version of the Form Compiler.

Action: Upgrade to new version of Forms Runtime.
Level: 25
Trigger: None

FRM-10262: Cannot put radio items, check boxes, or separators in menu bar.
Cause: You attempted to put radio items, check boxes, or separators in menu bar.

Action: Put these items in a submenu.
Level: 25
Trigger: None

FRM-10263: Cannot find icon file for iconic menu item.
Cause: No directory name for this icon.

Action: Contact the person who created the menu application.
Level: 25
Trigger: None

FRM-10264: Specified menu item does not exist.
Cause: You specified a menu item that does not exist in the form.

Action: Try retyping the name or choose another item name.
Level: 25
Trigger: None

FRM-10265: Library was created by an old version of the Form Compiler.
Cause: Oracle Forms cannot use library.

Action: Recompile the library with the current version of the Form Compiler.
Level: 25
Trigger: None

FRM-10266: Library was created by a new version of the Form Compiler.
Cause: Oracle Forms cannot use library.

Action: Recompile the library with current version of the Form Compiler.
Level: 25
Trigger: None

FRM-10267: Help type magic menu item must be placed on top-level menu.
Cause: You placed a help magic menu item on a submenu.

Action: Move the help magic menu item to the top-level menu (main menu).
Level: 25
Trigger: None

FRM-10268: Error: Program unit %s in library %s is uncompiled.
Cause: You called an uncompiled program unit from a library.

Action: Follow the PL/SQL program error.
Level: 25
Trigger: None

FRM-10269: Warning! Program unit %s in library %s is uncompiled.
Cause: In debug Forms Runtime, you called an uncompiled program unit in a library.

Action: This is just a warning. Forms Runtime will attempt to compile and run the program unit.
Level: 25
Trigger: None

FRM-10270: Cannot attach library %s while opening menu %s.
Cause: The specified library file is attached to the given menu, but cannot be located in the search path for PL/SQL libraries.

Action: Make sure the library file can be located before attempting to run with the specified menu again. For example, have it in the working directory.
Level: 25
Trigger: None

FRM-21011: PL/SQL unhandled exception %s.
Cause: An unhandled exception occurred while executing a menu trigger.

Action: Examine the text of the exception in this message. If this indicates a cause, correct it. If the problem persists, contact Oracle Support Services.
Level: 25
Trigger: None

FRM-40007: Invalid user identifier or password. Re-enter.
Cause: You entered an incorrect ORACLE username or password.

Action: Retype your username and password properly.
Level: 99
Trigger: ON-ERROR

FRM-40010: Cannot read form %s.
Cause: One of the following:
1. You entered a nonexistent form name.
2. You typed an incomplete path.
3. You do not have the proper privileges to run the form.
4. You do not have a compiled copy of the form.

Action: Retype the form name correctly, provide the proper path name, contact your system administrator, or compile the form.
Level: 5
Trigger: ON-ERROR

FRM-40011: Form was created by an old version of Oracle Forms.
Cause: The .FMB file was created with an old and incompatible version of the Form Compiler.

Action: Recompile the form or relink Generate.
Level: 99
Trigger: ON-ERROR

FRM-40012: Form was created by a new version of Oracle Forms.
Cause: The .FMB file was created by a new and incompatible version of the Form Compiler.

Action: Recompile the form.
Level: 99
Trigger: ON-ERROR

FRM-40013: Program Error: error occurred while reading form.
Cause: An internal error occurred while Oracle Forms was trying to read the .FMB file.

Action: Recompile the form.
Level: 99
Trigger: None

FRM-40014: Not enough memory to load the form.
Cause: Internal error. Your computer does not have enough memory to run the form.

Action: The designer might be able to modify the form so that it will run. If that is not feasible, your installation must make more memory available, either by modifying the operating system parameters or by adding more memory to the computer.
Level: 99
Trigger: ON-ERROR

FRM-40015: Unexpected end of file reading form.
Cause: The form was fragmented or incomplete.

Action: Recompile the form.
Level: 99
Trigger: None

FRM-40019: Unknown screen number to display.
Cause: An internal error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-40020: Page %d too small for this form.
Cause: Application design error. An item is positioned off the page.

Action: Ensure that all items that are associated with the given page fit completely on that page. You can reposition the items or resize the page.
Level: 99
Trigger: None

FRM-40021: Item in form file is too large.
Cause: An internal error occurred while Oracle Forms was reading the form.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-40023: Error creating record manager context.
Cause: Oracle Forms could not initialize its internal record manager.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40024: Out of memory.
Cause: Internal error. Your computer does not have enough memory to run the form.

Action: The designer might be able to modify the form so that it will run. If that is not feasible, your installation must make more memory available, either by modifying the operating system parameters or by adding more memory to the computer.
Level: 99
Trigger: ON-ERROR

FRM-40025: Cannot suppress screen output without file input.
Cause: You tried to run a form on the command line using incompatible preferences. The output_file preference works only in conjunction with the keyin preference.

Action: Retype the command to include both the output_file and keyin preferences.
Level: 99
Trigger: None

FRM-40026: Error opening key script file.
Cause: Oracle Forms cannot open the file you specified with the keyin preference.

Action: Make sure the file exists and the file protections are set properly. Or create a file with the keyin preference.
Level: 99
Trigger: None

FRM-40027: Error opening display spool file.
Cause: Operating system error. Oracle Forms cannot open a file specified with the output_file preference because there is insufficient disk space or because you have specified an incorrect filename.

Action: Contact your system administrator.
Level: 99
Trigger: None

FRM-40028: Error opening message file %s%s%s.MSB.
Cause: Oracle Forms cannot find the message file.

Action: Make sure the message file exists and the appropriate path is set.
Level: 99
Trigger: None

FRM-40029: Already logged on. Must logout before changing connections.
Cause: The Login() Built-in was issued while already logged on.

Action: Use the Logout() Built-in first.
Level: 25
Trigger: ON-ERROR

FRM-40030: File %s is not a Forms file.
Cause: The file specified on the command line was not a valid Oracle Forms file.

Action: Re-enter Forms Runtime startup command with the name of a valid file.
Level: 99
Trigger: ON-ERROR

FRM-40031: File %s is not a Forms Runtime file.
Cause: The file specified on the command line is not a Forms Runtime (.FMX) file.

Action: Re-enter a valid Forms Runtime (.FMX) file.
Level: 99
Trigger: ON-ERROR

FRM-40032: Internal Error: file %s contains an improper chunk size.
Cause: Internal error. File was compiled incorrectly or is corrupted.

Action: Recompile your file.
Level: 99
Trigger: ON-ERROR

FRM-40033: Internal Error: file %s contains a bad chunk table.
Cause: Internal error. File was compiled incorrectly or is corrupted.

Action: Recompile your file.
Level: 99
Trigger: ON-ERROR

FRM-40034: Cannot attach the library file.
Cause: Oracle Forms was unable to find the specified library file.

Action: Exit Forms Runtime and try again.
Level: 99
Trigger: ON-ERROR

FRM-40036: Library was created by a new version of Oracle Forms.
Cause: Oracle Forms unable to use library.

Action: Recompile the library with current version of Oracle Forms.
Level: 99
Trigger: ON-ERROR

FRM-40037: Library was created by an old version of Oracle Forms.
Cause: Oracle Forms unable to use library.

Action: Recompile the library with current version of Oracle Forms.
Level: 99
Trigger: ON-ERROR

FRM-40039: Cannot attach library %s while opening form %s.
Cause: The given library is attached to the form but cannot be located in the search path for PL/SQL libraries.

Action: Make sure that the given library can be found and that it has read permissions set.
Level: 99
Trigger: ON-ERROR

FRM-40040: Cannot perform proxy connection.
Cause: Database privileges for proxying user may not be configured on the database side or database account for SSO user not created.

Action: Make sure database is appropriately configured for making proxy connection.
Level: 99
Trigger: ON-ERROR

FRM-40041: Form %s requires a UTF8 character set.
Cause: An attempt was made to execute the specified form, but it contained one or more items whose datatype was NCHAR, and the NLS_LANG environment variable did not specify the UTF8 or AL32UTF8 character set.

Action: Set the NLS_LANG environment variable to a value which specifies the UTF8 or AL32UTF8 character set, and restart the application.
Level: 25
Trigger: ON-ERROR

FRM-40042: Unable to update the database password in the RAD repository.
Cause: An error occurred while trying to update the database password in the RAD repository.

Action: Contact your system administrator and get the password updated in the RAD repository. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40100: At first record.
Cause: You pressed [Previous Record] when the cursor was at the first record.

Action: No action is necessary.
Level: 5
Trigger: ON-ERROR

FRM-40101: Cannot position to a key item. None are navigable.
Cause: You pressed [Next Primary Key Item], but there are no enterable primary key items in this block.

Action: Use [Next Item] for navigation rather than [Next Primary Key Item].
Level: 10
Trigger: ON-ERROR

FRM-40102: Record must be entered or deleted first.
Cause: You pressed [Next Record] or [Down] in a context where it is meaningless. Either:
1. The last record in a block is the current record.
2. The block is empty.
3. You are in a new record in the middle of the block created by pressing [Insert Record].

Action: No action is necessary.
Level: 5
Trigger: ON-ERROR

FRM-40103: Cannot position to a key item. None are queryable.
Cause: You tried to use [Next Primary Key Item], but none of the primary key items in the block allow you to enter query criteria.

Action: No action is necessary.
Level: 10
Trigger: ON-ERROR

FRM-40104: No such block: %s.
Cause: Runtime error. A GO_BLOCK statement references a nonexistent block.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-40105: Unable to resolve reference to item %s.
Cause: Runtime error. A GO_ITEM statement references a nonexistent item.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-40106: No navigable items in destination block.
Cause: Runtime error. A GO_BLOCK statement references a block with no enterable items.

Action: Remove the statement or make at least one item in the block enterable.
Level: 99
Trigger: ON-ERROR

FRM-40107: Cannot navigate to non-displayed item %s.
Cause: Runtime error. A GO_ITEM statement references a non-displayed item.

Action: Remove the statement or turn on the Displayed Property for the indicated item.
Level: 20
Trigger: ON-ERROR

FRM-40108: No such form: %s.
Cause: You attempted to get/set properties of a nonexistent or unloaded form.

Action: Use a valid form name.
Level: 99
Trigger: ON-ERROR

FRM-40109: Cannot navigate out of current block in enter-query mode.
Cause: You attempted to navigate out of the current block during enter-query mode.

Action: No action is necessary. You cannot navigate out of the current block or record during enter-query mode.
Level: 99
Trigger: None

FRM-40110: At first block.
Cause: You attempted [Previous Block] when at the first block.

Action: None. Consider an alternative method of navigation.
Level: 5
Trigger: ON-ERROR

FRM-40111: At last block.
Cause: You attempted [Next Block] when at the last block.

Action: None. Consider an alternative method of navigation.
Level: 5
Trigger: ON-ERROR

FRM-40112: Attempted go_item to non enabled item %s:%s.
Cause: You attempted to issue a go_item to a non enabled item.

Action: None. Consider an alternative method of navigation.
Level: 99
Trigger: ON-ERROR

FRM-40200: Field is protected against update.
Cause: You tried to update a field that does not allow updates.

Action: No action is necessary. You cannot update this field in this form.
Level: 15
Trigger: ON-ERROR

FRM-40201: Field is full. Can't insert character.
Cause: Oracle Forms is in insert mode, and the current field is full.

Action: Delete a character to make room for the new character or press [Insert/Replace] to activate replace mode.
Level: 99
Trigger: None

FRM-40202: Field must be entered.
Cause: You have not entered a value (or you have deleted a value) in a field that requires data input.

Action: You must enter a value in this field.
Level: 15

Trigger: ON-ERROR

FRM-40203: Field must be entered completely.
Cause: You have not entered a complete value (or you have deleted part of a value) in a field that has a fixed length requirement.

Action: Enter a complete value (one that extends to the end of the field).
Level: 15
Trigger: ON-ERROR

FRM-40204: Cursor is at beginning of field value.
Cause: You tried to delete a character before the first character position of the field.

Action: Use [Delete Character] to delete the character that the cursor is on.
Level: 10
Trigger: ON-ERROR

FRM-40205: Cursor is beyond the current field value.
Cause: On a block mode terminal, you positioned the cursor out of a field.

Action: Move the cursor into the field and try the entry again.
Level: 99
Trigger: None

FRM-40206: Previous character is currently hidden.
Cause: You tried to delete a character that is off the screen.

Action: Scroll the character you want to delete into view using the arrow keys or [Scroll Left] and [Scroll Right].
Level: 10
Trigger: ON-ERROR

FRM-40207: Must be in range %.30s to %.30s.
Cause: You entered a value not in the valid item range.

Action: Enter a value in the range shown.
Level: 99
Trigger: None

FRM-40208: Form running in query-only mode. Cannot change database fields.
Cause: You entered a value on a query-only form.

Action: Do not enter values on this form. You can execute queries and view data, but you cannot alter existing data or enter new data.
Level: 15
Trigger: ON-ERROR

FRM-40209: Field must be of form %s.
Cause: The value that you entered did not match the format mask on the field.

Action: Retry with a field value that matches the format mask.
Level: 99
Trigger: ON-ERROR

FRM-40210: Search string not found.
Cause: Search string does not exist in the module.

Action: Check your search string to make sure it is accurate or try another search string.
Level: 99
Trigger: ON-MESSAGE

FRM-40211: Warning! Newlines may be stripped from this field.
Cause: You attempted to assign data with newlines to a single-line text field.

Action: Assign to a multi-line text field if you need the newlines.
Level: 5
Trigger: ON-MESSAGE

FRM-40212: Invalid value for field %s.
Cause: Caused by one of the following:
1. The value is not of the proper data type.
2. The value does not match any of the list of acceptable values.
3. For a text field, the value does not match the specified range.

Action: Retry with another value.
Level: 20
Trigger: ON-ERROR

FRM-40213: Cannot Copy_Region/Cut_Region; region not selected.
Cause: Region not selected.

Action: Select a region and try again.
Level: 5
Trigger: ON-ERROR

FRM-40214: Cannot open the clipboard for the copy/cut.
Cause: Clipboard unavailable.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40215: Cannot write to the clipboard.
Cause: Clipboard unavailable.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40216: Cannot open the clipboard for the paste.
Cause: Clipboard unavailable.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40217: Cannot get the data size from the clipboard.
Cause: Invalid data.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40218: Cannot read from the clipboard.
Cause: Invalid data.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40219: Cannot format the data read from the clipboard.
Cause: Invalid data.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40220: Cannot paste from the clipboard; value too long.
Cause: Invalid data.

Action: Platform specific.
Level: 99
Trigger: ON-ERROR

FRM-40221: Cannot Paste_Region; region not selected.
Cause: Paste_Region was invoked while no portion of the image item was selected.

Action: Select a region of the image item prior to calling Paste_Region.
Level: 99
Trigger: ON-ERROR

FRM-40222: Disabled item '%s.%s' failed validation.
Cause: Probable application design error. Forms determined that the item contains an invalid value, but it cannot give focus to the item because it is disabled. This could happen because either:
The application programmatically assigned an invalid valid to the item.
The application programmatically disabled the item after the end user entered an invalid valid in the item (and before the item was validated).

Action: Correct the application logic.
Level: 25
Trigger: ON-ERROR

FRM-40223: Field contains an invalid string for security purposes.
Cause: Field contains a string that could be a potential security violation.

Action: Remove the offending string.
Level: 15
Trigger: ON-ERROR

FRM-40301: Query caused no records to be retrieved. Re-enter.
Cause: No records matched the query criteria. Still in Enter Query mode.

Action: Either adjust the query criteria or press [Exit/Cancel] to leave Enter Query mode.
Level: 99
Trigger: ON-MESSAGE

FRM-40302: Cannot enter a query. No fields are queryable.
Cause: You pressed [Enter Query] while the cursor was in a block with no queryable fields.

Action: No action is necessary.
Level: 15
Trigger: ON-ERROR

FRM-40303: No base table fields in the block.
Cause: One of the blocks in the current module has no base table fields.

Action: No action is necessary.
Level: 99
Trigger: ON-ERROR

FRM-40350: Query caused no records to be retrieved.
Cause: The current query fetched no records from the table. The table is empty, or it contains no records that meet the query's search criteria.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-40352: Last record of query retrieved.
Cause: You pressed [Down], [Next Record], [Next Set of Records], or [Scroll Down] after all records had been retrieved.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-40353: Query cancelled.
Cause: You pressed [Exit/Cancel] in Enter Query mode, or you pressed CTRL-C (or its equivalent) while Oracle Forms was fetching rows from the database.

Action: No action is necessary.
Level: 5 when the query was canceled by CTRL-C; 10 otherwise
Trigger: ON-MESSAGE

FRM-40355: Query will retrieve 1 record.
Cause: You pressed [Count Query Hits]. If you now press [Execute Query], the number of records will be retrieved.

Action: No action is necessary.
Level: 25
Trigger: ON-MESSAGE

FRM-40356: Invalid number in example record. Query not issued.
Cause: In Enter Query mode, you entered an invalid number in the example record.

Action: Correct the entry and retry the query.
Level: 99
Trigger: ON-ERROR

FRM-40357: Invalid string in example record. Query not issued.
Cause: In query mode, you entered an invalid ALPHA or CHAR value in the example record.

Action: Correct the entry and retry the query.
Level: 99
Trigger: ON-ERROR

FRM-40358: Invalid date in example record. Query not issued.
Cause: In Enter Query mode, you entered an invalid DATE in the example record.

Action: Correct the entry and retry the query.
Level: 99
Trigger: ON-ERROR

FRM-40359: Invalid date or time in example record. Query not issued.
Cause: In Enter Query mode, you entered an invalid JDATE, EDATE, or TIME value in the example record.

Action: Correct the entry and retry the query.
Level: 99
Trigger: ON-ERROR

FRM-40360: Cannot query records here.
Cause: You attempted to query a block that does not allow queries.

Action: Do not attempt to query this block.
Level: 10
Trigger: ON-ERROR

FRM-40361: Query operation not support for TIME data type.
Cause: You made a query with a % "like" operator in a time field, which is not supported.

Action: Try to restate your query without a time data type.
Level: 10
Trigger: ON-ERROR

FRM-40364: The data type of item '%s' does not match the corresponding column in the stored procedure.
Cause: The data type of the item is different from the data type of the corresponding column in the stored procedure.

Action: Make the data type of the item in the block and the column in the stored procedure the same.
Level: 20
Trigger: ON-ERROR

FRM-40367: Invalid criteria in field %s in example record.
Cause: Only simple clauses are allowed in restricted enter query mode.

Action: Re-enter the criteria.
Level: 99
Trigger: ON-ERROR

FRM-40400: Transaction complete: %d records applied and saved.
Cause: Save complete.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-40401: No changes to save.
Cause: No records were added or modified since the last apply or save. Caution: Unapplied database changes that were made through explicit sql (DML) are still applied, even when this message is displayed.

Action: No action is necessary.
Level: 5
Trigger: ON-ERROR

FRM-40402: Save cancelled.
Cause: You pressed CTRL-C (or the equivalent) while waiting for a lock.

Action: No action is necessary.
Level: 10
Trigger: ON-MESSAGE

FRM-40403: A calling form has unapplied changes. Save not allowed.
Cause: A calling form has unapplied changes.

Action: Apply the changes or return to the calling form and retry the save.
Level: 15
Trigger: ON-ERROR

FRM-40404: Database apply complete: %d records applied.
Cause: Apply complete.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-40405: No changes to apply.
Cause: No records were added or modified since the last apply or save.

Action: No action is necessary.
Level: 5
Trigger: ON-ERROR

FRM-40406: Transaction complete: %d records applied; all records saved.
Cause: You finished an apply that recorded your changes and saved previously applied changes.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-40407: Transaction complete: applied records saved.
Cause: You finished a save that saved previously applied changes.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-40408: database commit failure.
Cause: A database commit failed.

Action: Examine integrity constraints on the database tables that were updated. If any were violated, redo the updates without violating the constraints. If necessary, do the updates and the commit in sqlplus, and see if it issues an ORA-nnnnn message that will identify the constraint that was violated.
Level: 99
Trigger: ON-ERROR

FRM-40501: ORACLE error: unable to reserve record for update or delete.
Cause: A fatal error occurred while trying to select the record for update.

Action: Pressing [Display Error] provides more information, if it is available. You can also try to update or delete this record later. If necessary, contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40502: ORACLE error: unable to read list of values.
Cause: A fatal error occurred while trying to read a list of values.

Action: Contact your system administrator. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40504: ORACLE error: unable to execute a %s trigger.
Cause: A fatal error occurred while trying to execute a trigger.

Action: Contact your system administrator. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40505: ORACLE error: unable to perform query.
Cause: Processing error encountered. The table associated with the current block of the form might not exist, or your username might not have authority to perform the specified action on the table.

Action: Pressing [Display Error] provides more information, if it is available. You can also try to update or delete this record later. If necessary, contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40506: ORACLE error: unable to check for record uniqueness.
Cause: Processing error encountered while checking a record's primary key items for uniqueness. The table associated with the current block of the form does not exist, or you do not have authority to access the table.

Action: Contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40507: ORACLE error: unable to fetch next query record.
Cause: One of the following:
1. Oracle Forms was unable to allocate a record buffer (as reported in a previous FRM-40900).
2. If you are connected to an non-Oracle datasource through ODBC, the cursor loses its position in the result set after a commit.
3. A fatal error occurred while trying to fetch the next query record.

Action: For 1, refer to FRM-40900. For 2, requery. For 3, contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40508: ORACLE error: unable to INSERT record.
Cause: A fatal error occurred while trying to insert a record. The table associated with the current block of the form might not exist, your username might not have authority to perform the specified action on the table, or some other reason might have caused the fatal error.

Action: Contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40509: ORACLE error: unable to UPDATE record.
Cause: A fatal error occurred while trying to update a record. The table associated with the current block of the form might not exist, your username might not have authority to perform the specified action on the table, or some other reason might have caused the fatal error.

Action: Contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40510: ORACLE error: unable to DELETE record.
Cause: A fatal error occurred while trying to delete a record. The table associated with the current block of the form might not exist, your username might not have authority to perform the specified action on the table, or some other reason might have caused the fatal error.

Action: Contact your DBA.
Level: 99
Trigger: ON-ERROR

FRM-40511: ORACLE error occurred while executing a %s trigger.
Cause: A fatal error occurred while trying to execute a trigger. The table associated with the current block of the form might not exist, your username might not have authority to perform the specified action on the table, or some other reason might have caused the fatal error.

Action: Contact your DBA
Level: 99
Trigger: None

FRM-40512: ORACLE error: unable to issue SAVEPOINT command.
Cause: While attempting to call a new form or to commit, the issued SAVEPOINT command failed. This generally means that the module has run out of savepoints.

Action: Press [Display Error] to display the specific ORACLE error. You might be able to increase the maximum number of savepoints in the INIT.ORA file.
Level: 99
Trigger: ON-ERROR

FRM-40513: ORACLE error: unable to get date/time from database.
Cause: An error occurred while trying to resolve a database date/time initial value.

Action: Connect if you have not already done so. Verify database status.
Level: 10
Trigger: ON-ERROR

FRM-40514: Operation requires a database connection.
Cause: You tried to perform an database operation without connecting to the database.

Action: Connect to the database and retry.
Level: 20

Trigger: ON-ERROR

FRM-40515: ORACLE error: unable to open cursor.
Cause: You reached the limit in the number of cursors you can open.

Action: Check the number of cursors you have open.
Level: 99
Trigger: ON-ERROR

FRM-40600: Record has already been inserted.
Cause: You attempted to insert or update a record, but uniqueness is enforced on the block's primary key items. The record, as inserted or updated, is not unique.

Action: Change the values in one or more primary key fields of the current record, making them unique. If the requirement of unique primary key fields creates difficulties, consider eliminating the constraint.
Level: 25
Trigger: ON-ERROR

FRM-40602: Cannot insert into or update data in a view.
Cause: You tried to modify the contents of a view in a manner that is not permitted.

Action: No action is necessary; you cannot perform the operation you have attempted.
Level: 20
Trigger: ON-ERROR

FRM-40603: Records no longer reserved for update. Re-query to make changes.
Cause: You committed your modifications in a block where you had previously entered an ENTER_QUERY or EXECUTE_QUERY packaged procedure with the FOR_UPDATE parameter. This action released all locks on the records in this block.

Action: If you want to modify the block, you will need to re-query.
Level: 99
Trigger: ON-MESSAGE

FRM-40652: Cannot lock table in shared update mode.
Cause: Caused by one of the following:
1. You do not have access to this table.
2. Oracle Forms cannot lock the table in shared update mode.

Action: Contact your DBA.
Level: 15
Trigger: ON-ERROR

FRM-40653: Record not reserved for update or delete. Try again later.
Cause: You pressed CTRL-C (or the equivalent) to cancel. The operation that was attempting to update or delete the record was terminated.

Action: No action is necessary.
Level: 20
Trigger: ON-MESSAGE

FRM-40654: Record has been updated by another user. Re-query to see change.
Cause: Another user has updated this record since you performed a query and has changed at least one field in the record. Your actions have not changed the record in memory.

Action: You can update or delete this record now only if another user has restored the field values back to the way they were when you performed the query. Otherwise, you must re-query to fetch and display the new record into the form before you can update or delete it.
Level: 20
Trigger: ON-ERROR

FRM-40655: SQL error forced rollback: clear form and re-enter transaction.
Cause: A deadlock or some other error has caused the current transaction to fail. Your changes were rolled back.

Action: Clear the form (or exit and re-enter the form) and re-enter the transaction. You might have to modify the form's design to prevent the error from recurring.
Level: 25
Trigger: ON-ERROR

FRM-40657: Record changed or deleted by another user.
Cause: Another user has deleted the record since the query was executed, or database access control does not allow the operation.

Action: You can clear this record from your screen, but you cannot update or delete it since it no longer exists in the database, or database access control does not allow the operation. Check database access control policy.
Level: 20
Trigger: ON-MESSAGE

FRM-40659: Last row of query retrieved. Re-query to see remaining records.
Cause: A FOR_UPDATE query has been closed by executing a commit. Because the query was open prior to the commit, there may be more records to retrieve.

Action: Re-query to see remaining records.
Level: 5
Trigger: ON-MESSAGE

FRM-40700: No such trigger: %s.
Cause: Application design error. The form attempted to execute a trigger that doesn't exist, causing a fatal error.

Action: Correct the reference to the trigger.
Level: 20
Trigger: ON-ERROR

FRM-40702: Cannot call form with changes to save
Cause: You attempted to call another form with unsaved changes in the current form and savepoint mode off.

Action: Commit/post changes and then retry.
Level: 15
Trigger: ON-ERROR

FRM-40703: Fetched field cannot be changed in query mode.
Cause: You attempted to modify a fetched item in query mode.

Action: None required.
Level: 20
Trigger: ON-ERROR

FRM-40704: Illegal SQL statement in query-only mode
Cause: Application design error. The form tried to execute a function that is illegal in a query-only form.

Action: You might need to redesign the form.
Level: 20
Trigger: ON-ERROR

FRM-40705: Illegal SQL statement in non-commit-time trigger.
Cause: Application design error. The current trigger contains a SQL statement that is illegal for the trigger type.

Action: Rewrite the trigger text or use a different type of trigger.
Level: 20
Trigger: ON-ERROR

FRM-40714: Function illegal in this context.
Cause: Application design error. The current trigger contains an illegal function code.

Action: Rewrite the trigger text or use a different type of trigger.
Level: 20
Trigger: ON-ERROR

FRM-40724: Missing selector in CASE statement.
Cause: Application design error. The selector portion is missing in a CASE statement.

Action: Correct the statement.
Level: 99
Trigger: None

FRM-40730: Invalid message suppress level--unchanged from %d.
Cause: Application design error. A trigger attempted to set the system message level to an invalid number.

Action: Reset the SYSTEM.MESSAGE_LEVEL system variable to a valid number.
Level: 99
Trigger: ON-ERROR

FRM-40732: Target of GOTO does not exist in this macro.
Cause: Application design error. The label referenced in PL/SQL does not exist.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-40733: PL/SQL Built-in %s failed.
Cause: A fatal error occurred in Oracle Forms or in PL/SQL during trigger execution.

Action: Examine application logic to see if the Built-in is invoked incorrectly. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40734: Internal Error: PL/SQL error occurred.
Cause: An internal error occurred in PL/SQL during trigger execution.

Action: If the problem persists, contact Oracle Support Services.

Level: 99
Trigger: ON-ERROR

FRM-40735: %s trigger raised unhandled exception %s.
Cause: Application design error. The current trigger raised an exception (other than FORM_TRIGGER_FAILURE), but it did not handle the exception.

Action: Rewrite the trigger text to handle the exception.
Level: 99
Trigger: ON-ERROR

FRM-40736: Cannot initialize PL/SQL.
Cause: An internal error occurred while initializing PL/SQL.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40737: Illegal restricted procedure %s in %s trigger.
Cause: Application design error. A trigger tried to execute a restricted packaged procedure.

Action: Remove the packaged procedure from the trigger text.
Level: 99
Trigger: ON-ERROR

FRM-40738: Argument %d to builtin %s cannot be null.
Cause: Application design error. No arguments were provided to the Built-in.

Action: Refer to the online Help for the correct usage of this Built-in.
Level: 99
Trigger: ON-ERROR

FRM-40739: Full rollback not allowed in post-only form.
Cause: Application design error. A trigger tried to issue a CLEAR_FORM packaged procedure with the FULL_ROLLBACK parameter in a post-only, called form.

Action: Remove the FULL_ROLLBACK parameter or ensure that the calling form does not have unposted changes when the call occurs.
Level: 99
Trigger: ON-ERROR

FRM-40740: Procedure %s only allowed in an on-%s trigger.
Cause: Application design error. A non-transactional trigger attempted to invoke a Built-in procedure that is restricted to a given trigger.

Action: Refer to the online Help for the correct usage of this procedure.
Level: 99
Trigger: ON-ERROR

FRM-40741: Unable to locate record %d on block %s.
Cause: You attempted to get or set record properties for an invalid record number for the given block.

Action: Verify your Get/Set record property parameters.
Level: 20
Trigger: ON-ERROR

FRM-40742: Illegal status conversion on record %d: %s to %s.
Cause: Application design error. A call to SET_RECORD_PROPERTY attempted an illegal conversion between record statuses.

Action: Refer to SET_RECORD_PROPERTY in online Help for correct transitions.
Level: 99
Trigger: ON-ERROR

FRM-40743: This operation with no base table requires the %s trigger.
Cause: Application design error. Attempted a database operation (query, insert, update, etc.) on a non-base table block without the appropriate transactional trigger.

Action: Refer to online Help for the appropriate transactional trigger and then create the correct trigger.
Level: 20
Trigger: ON-ERROR

FRM-40744: Truncation of input value will occur if editor accepted.
Cause: The editor's buffer is too small to accept the input.

Action: Change editors or enlarge the buffer.
Level: 99
Trigger: None

FRM-40745: Output value of Built-in %s was truncated.
Cause: Output variable is too small.

Action: Increase the size of the PL/SQL output variable.
Level: 15
Trigger: ON-ERROR

FRM-40746: Cannot call Built-in %s from startup debugger window.
Cause: Built-in is not accessible from the startup debugger window.

Action: Refer to the Oracle Forms Developer's Guide for a list of Built-ins that are not accessible from the startup debugger window.
Level: 99
Trigger: None

FRM-40747: Cannot call Built-in %s from a debug trigger.
Cause: Built-in is not accessible from the debug trigger.

Action: Refer to the Oracle Forms Developer's Guide for a list of Built-ins that are accessible from a debug trigger .
Level: 99
Trigger: None

FRM-40748: Trigger %s terminated by reset command.
Cause: You issued the reset command or you pressed the reset button in the debugger.

Action: If you want to navigate downward, go to the call stack.
Level: 99
Trigger: None

FRM-40749: Invalid record status specified for record %d.
Cause: Application design error. An attempt was made to set the Status Property of a record to an invalid value.

Action: The record's Status Property should be set to NEW_STATUS, QUERY_STATUS, INSERT_STATUS, or CHANGED_STATUS.
Level: 99
Trigger: ON-ERROR

FRM-40750: Record %d: Can't set status to QUERY or CHANGED in a control block.
Cause: Application design error. An attempt was made to set the Status Property of a record in a control block to QUERY_STATUS or CHANGED_STATUS.

Action: The record's Status Property should be set to NEW_STATUS or INSERT_STATUS.
Level: 99
Trigger: ON-ERROR

FRM-40800: User exit %s does not exist.
Cause: The form tried to invoke a user exit that does not exist. This could be caused by one of the following:
1. You are using the wrong version of Forms Runtime.
2. There could be an error in the form.
3. The FORMS_USEREXITS environment variable is not set correctly.

Action: Make sure that the dynamic library which defines the user exit symbol is listed in FORMS_USEREXITS list.
Level: 20
Trigger: ON-ERROR

FRM-40801: memory allocation failure
Cause: A memory allocation failed when Forms Runtime attempted to create an internal macro.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 25
Trigger: ON-ERROR

FRM-40808: Cannot execute HOST command. Error code = %s.
Cause: Cannot execute a HOST statement because of an operating system error.

Action: Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-40809: HOST command had error code = %s.
Cause: The operating system command resulted in the above error code.

Action: Verify that you entered the command properly.
Level: 99
Trigger: ON-ERROR

FRM-40811: Shell command had error.
Cause: The operating system command resulted in the above error code.

Action: Verify that you entered the command properly.
Level: 99
Trigger: ON-ERROR

FRM-40815: Variable GLOBAL.%s does not exist.
Cause: Application design error. A trigger references a global variable that does not exist.

Action: Create the global variable or remove the reference.
Level: 20
Trigger: ON-ERROR

FRM-40816: Could not allocate memory for new symbol.
Cause: A memory allocation failed when Forms Runtime attempted to access a new global variable.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-40817: Could not allocate memory for new value.
Cause: A memory allocation failed when Forms Runtime attempted to access a new global variable.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-40818: System variable name not defined.
Cause: You have tried to access a system variable that does not exist.

Action: Check the system variable name.
Level: 99
Trigger: ON-ERROR

FRM-40819: System variable is not modifiable.
Cause: You have tried to modify a system variable.

Action: You cannot modify system variables.
Level: 99
Trigger: ON-ERROR

FRM-40828: CALL or CALLQRY with invalid variable reference.
Cause: Application design error. A CALL or CALLQRY function code contains an invalid variable reference.

Action: Correct the statement.
Level: 99
Trigger: None

FRM-40831: Truncation occurred: value too long for field %s.
Cause: Application design error. A trigger, query, or user exit read a value into an item that is not long enough to hold the entire value. The item truncated the value.

Action: Increase the target item's item length to avoid truncation.
Level: 15
Trigger: ON-ERROR

FRM-40832: FORMS_USEREXITS environment variable not set. User exit %s did not execute.
Cause: FORMS_USEREXITS must be set for forms USER_EXIT Built-in to work.

Action: Set FORMS_USEREXITS to dynamic libraries, which define the user exit symbols. Searching is done in the order libraries are listed.
Level: 20
Trigger: ON-ERROR

FRM-40833: Could not completely load the dynamic user exit libraries. User exit %s did not execute.
Cause: User exit symbol was not found and there were failures in opening some of the dynamic user exit libraries.

Action: Make sure that all the user exit libraries listed in the FORMS_USEREXITS environment variable are correct and available.
Level: 20
Trigger: ON-ERROR

FRM-40834: Value from item %s.%s is too long for result set column (actual: %d bytes, maximum: %d bytes).
Cause: In a block based on a stored procedure, the application attempted to insert or update a value with too many bytes into a result set column with BYTE length semantics. The value was obtained from an instance of an item in the block. The insert or update attempt was suppressed.

Action: Ensure that the Maximum Length and Data Length Semantics in character-datatype items in the block match the definitions of the result-set columns in the stored procedure.
Level: 99
Trigger: ON-ERROR

FRM-40835: Value from item %s.%s is too long for result set column (actual: %d characters, maximum: %d characters).
Cause: In a block based on a stored procedure, the application attempted to insert or update a value with too many characters into a result set column with CHAR length semantics. The value was obtained from an instance of an item in the block. The insert or update attempt was suppressed.

Action: Ensure that the Maximum Length and Data Length Semantics in character-datatype items in the block match the definitions of the result-set columns in the stored procedure.
Level: 99
Trigger: ON-ERROR

FRM-40900: Unable to allocate record buffer for insert record, update record, or fetch. Operation aborted.
Cause: An insert record, update record, or fetch required main memory or space in the temporary record buffer file, but the main memory or disk space was unavailable. Further information is available from a preceding message: FRM-41839 (I/O error on the temporary record buffer file), FRM-41840 insufficient main memory), FRM-41847 (temporary record buffer file size limit exceeded), or FRM-41850 (archived record memory threshold exceeded).

Action: Refer to the preceding FRM-41839, FRM-41840, FRM-41847, or FRM-41850.
Level: 99
Trigger: ON-MESSAGE

FRM-40901: Note: not enough memory to remember all or part of this query.
Cause: A memory allocation failed when Forms Runtime attempted to save a query.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99

Trigger: ON-ERROR

FRM-40902: SQL statement too large.
Cause: Application design error. The form's design includes a SQL command that is more than 2048 characters long.

Action: Shorten the SQL command.
Level: 99
Trigger: ON-ERROR

FRM-40903: Cannot create output file.
Cause: You pressed [Print Screen], but screen contents could not be written to a file because of one of the following:
1. You have entered an illegal file name.
2. The operating system does not give you authority to create files.
3. The necessary disk or directory space is not available.

Action: Check the file name you have entered and correct it if necessary. If you need additional help, contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-40904: Program error: unknown operation to be performed on record.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40905: Unable to buffer more records on disk.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-40906: FATAL ERROR: cannot write a buffered record to disk.
Cause: Internal error while trying to write a buffered record to the disk.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40907: FATAL ERROR: cannot read a buffered record from disk.
Cause: Internal error while trying to read a buffered record from the disk.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40908: RAM Internal Error: %s
Cause: An internal error occurred within the form's internal record manager.

Action: If the problem persists, contact Oracle Support Services.
Level: 20
Trigger: ON-ERROR

FRM-40909: Internal Error: unknown error %d.
Cause: Internal error. Forms Runtime attempted to issue an unknown error message.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-40911: Record not created due to sequence number generation error.
Cause: Internal error. Either the sequence number object does not exist, or the designer does not have privileges for the sequence number object, or some other fatal database error occurred.

Action: Contact your DBA. If your DBA cannot correct the problem, and the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40912: WHEN-NEW-RECORD trigger failed. Record not created.
Cause: A runtime error occurred in a When-New-Record trigger that caused the trigger to fail. No new record was created.

Action: Contact your DBA. If your DBA cannot correct the problem, and the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40913: List of Values maximum exceeded. Some values are not displayed.
Cause: Application design error. Unable to return all the records in the current list of values; the number exceeds the maximum limit.

Action: Specify no more than 32,767 records to be returned in a list of values.
Level: 25
Trigger: ON-ERROR

FRM-40914: Memory allocation error: unable to complete transaction.
Cause: A memory allocation failed while Forms Runtime attempted to complete a transaction.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40915: Memory allocation error: unable to execute trigger %s.
Cause: A memory allocation failed while Forms Runtime attempted to execute a trigger.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40916: Memory allocation error: unable to execute query.
Cause: A memory allocation failed while Forms Runtime attempted to execute a query.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40917: Memory allocation error: unable to lock record.
Cause: A memory allocation failed while Forms Runtime attempted to lock a record.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40919: Internal SQL statement execution error: %d.
Cause: Error in the SQL statement Oracle Forms has tried to execute.

Action: Check the last SQL statement.
Level: 25
Trigger: ON-ERROR

FRM-40920: Unable to create view: low on system resources.
Cause: Something in your environment or application has prevented view creation.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40921: Could not create item: %s.
Cause: Something in your environment or application has prevented item creation.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-40922: An OLE error occurred: 0x%x.
Cause: A Built-in called an OLE or OLE-related function which failed.

Action: You need to lookup the error number in an OLE manual for further details.
Level: 99
Trigger: ON-ERROR

FRM-40923: OLE is not supported on this platform.
Cause: A Built-in that required OLE support was called, and your platform does not support OLE.

Action: Don't call OLE-related Built-ins on platforms that don't support OLE.
Level: 99
Trigger: ON-ERROR

FRM-40924: Invalid argument index specified.
Cause: An attempt was made to retrieve a value from the OLE-argument stack whose index was out of bounds for the size of the current OLE-argument stack

Action: Argument indices range from 1 to the number specified in the last call to FORMS_OLE.InitArgs()
Level: 99
Trigger: ON-ERROR

FRM-40925: No space initialized in OleArg for argument.
Cause: An attempt was made to store too many arguments into the initialized OLE-argument stack.

Action: Make sure you specify enough space in your call to FORMS_OLE.InitArgs().
Level: 99
Trigger: ON-ERROR

FRM-40926: OLE Object is NULL.
Cause: You cannot operate on a NULL OLE object.

Action: Do not attempt to call FORMS_OLE Built-ins with NULL OLE-objects.
Level: 99
Trigger: ON-ERROR

FRM-40927: Variant is not an array.
Cause: An attempt was made to access a variant as if it contained an array, and it did not.

Action: You can call FORMS_OLE.Get_Dims() to ensure that you have a variant with an array. For arrays, the return value for the function is greater than or equal to 1.
Level: 99
Trigger: ON-ERROR

FRM-40928: Too many array indices specified.
Cause: An attempt was made to access a variant that holds an array, but too many array indices were specified.

Action: You must use the correct number of array indices, the same number as returned by FORMS_OLE.GET_Dims().
Level: 99
Trigger: ON-ERROR

FRM-40929: Too few array indices specified.
Cause: An attempt was made to access a variant that holds an array, but too few array indices were specified.

Action: You must use the correct number of array indices, the same number as returned by FORMS_OLE.GET_Dims().
Level: 99
Trigger: ON-ERROR

FRM-40930: Array index was non-numeric.
Cause: An attempt was made to access a variant that holds an array, but the supplied array indices were non-numeric and not either ROW or COLUMN.

Action: The only valid array indices are numbers, which should be separated by columns. If you're fetching into a table from a variant, ROW and COLUMN can be used as placeholders for the row and column iterators during table construction.
Level: 99
Trigger: ON-ERROR

FRM-40931: Cannot populate table because datatype is unsupported.
Cause: An attempt to populate a table failed because one of its columns used an unsupported datatype.

Action: Restrict your column types to integers, numbers, strings, and dates.
Level: 99

Trigger: ON-ERROR

FRM-40932: Cannot populate variant because table's datatype is unsupported.
Cause: An attempt to populate a variant failed because one of the source table's columns used an unsupported datatype.

Action: Restrict your column types to integers, numbers, strings, and dates.
Level: 99
Trigger: ON-ERROR

FRM-40933: Cannot populate table because datatype is incorrect.
Cause: An attempt to populate a table failed because the block's datatype did not match the table's datatype.

Action: The table datatype must match the datatypes of the block's columns that are being retrieved.
Level: 99
Trigger: ON-ERROR

FRM-40934: Cannot populate table because records are out of bounds.
Cause: An attempt to populate a table failed because an illegal start or end record was specified.

Action: start_rec and end_rec parameters must fall between 1 and the number of retrievable records. end_rec may also be ALL_RECORDS.
Level: 99
Trigger: ON-ERROR

FRM-40935: Object does not exist locally.
Cause: An attempt to release an object failed because the 'kill_persistent' parameter was set to FALSE, and no local object existed to release.

Action: Never release objects you don't own.
Level: 10
Trigger: ON-ERROR

FRM-41000: This function is not currently available.
Cause: You pressed an undefined function key.

Action: Press [Show Keys] to determine which function key you should have pressed.
Level: 5
Trigger: ON-ERROR

FRM-41001: This function is not allowed on this device.
Cause: You tried to execute the Insert/Replace function.

Action: No action is necessary.
Level: 5
Trigger: ON-ERROR

FRM-41002: Please make a valid selection.
Cause: You entered an invalid selection number on the block menu; that block does not exist in this form.

Action: Select an existing block.
Level: 10
Trigger: ON-ERROR

FRM-41003: This function cannot be performed here.
Cause: You tried to perform a function that references a table, but current block does not correspond to any table.

Action: No action is necessary. You cannot perform the requested function on this block.
Level: 10
Trigger: ON-ERROR

FRM-41004: This function is not allowed in this mode.
Cause: You pressed a function key that does not work in this mode.

Action: No action is necessary.
Level: 10
Trigger: ON-ERROR

FRM-41005: Internal Error: function key not implemented.
Cause: You pressed a disabled function key.

Action: No action is necessary. You cannot use the function key in the current context unless the form's definition is modified.
Level: 25
Trigger: ON-ERROR

FRM-41007: Cursor not in a valid item. Function key was ignored.
Cause: You were not in a valid item when you pressed the function key.

Action: Position the cursor inside the item and press the function key again.
Level: 10
Trigger: ON-ERROR

FRM-41008: Undefined function key. Press %s for list of valid keys.
Cause: You pressed an undefined function key.

Action: Press [Show Keys] to determine which function key you should have pressed.
Level: 99
Trigger: ON-ERROR

FRM-41009: Function key not allowed. Press %s for list of valid keys.
Cause: You pressed a function key that is not allowed in this environment.

Action: Press [Show Keys] to determine which function key you should have pressed.
Level: 99
Trigger: ON-ERROR

FRM-41010: Cannot set attribute of the current item.
Cause: Application design error. A SET_ITEM statement tried to turn off the Input Allowed Property for the current item.

Action: Eliminate the statement or rewrite the trigger.
Level: 99
Trigger: ON-ERROR

FRM-41011: Undefined visual attribute.
Cause: Application design error. A Built-in tried to set an undefined visual attribute.

Action: Correct the statement.
Level: 99

Trigger: ON-ERROR

FRM-41012: Undefined item or variable reference.
Cause: Application design error. A NAME_IN statement tried to reference a nonexistent item or variable.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-41013: Undefined property specified for item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in specified an undefined property.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-41014: Cannot set property of null canvas item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in tried to change some property of a NULL canvas item.

Action: Specify a canvas for the item, or remove the statement.
Level: 99
Trigger: ON-ERROR

FRM-41015: Cannot set ENTERABLE Property of the current item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in tried to change the Enterable Property of the current item.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-41016: Cannot set DISPLAYED Property of the current item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in tried to change the Displayed Property of the current item.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-41017: Cannot set UPDATE ALLOWED Property of non-enabled item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in tried to turn on the Update Allowed Property of a non-enterable item.

Action: To turn on the Update Allowed Property of an item you must also turn on the Input Allowed Property of the item.
Level: 99
Trigger: ON-ERROR

FRM-41018: Cannot set UPDATE_NULL Property of non-enabled item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in tried to turn on the Update If Null Property of a non-enterable item.

Action: To turn on the Update If Null Property of an item you must also turn on the Input Allowed Property of the item.
Level: 99
Trigger: ON-ERROR

FRM-41019: Cannot set REQUIRED Property of non-enabled item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in tried to turn on the Required Property of a non-enterable item.

Action: To turn on the Required Property of an item you must also turn on the Input Allowed Property of the item.
Level: 99
Trigger: ON-ERROR

FRM-41020: Cannot set ENTERABLE Property of non-displayed item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in tried to turn on the Enterable Property of a non-displayed item.

Action: To turn on the Input Allowed Property of an item you must also turn on the Displayed Property of the item.
Level: 99
Trigger: ON-ERROR

FRM-41021: Cannot set QUERYABLE Property of non-displayed item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in tried to turn on the Query Allowed Property of a non-displayed item.

Action: To turn on the Query Allowed Property of an item you must also turn on the Displayed Property of the item.
Level: 99
Trigger: ON-ERROR

FRM-41022: Cannot set REQUIRED Property of non-updateable item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in tried to turn on the Required Property of a non-updateable item.

Action: To turn on the Required Property of an item you must also turn on either the Update Allowed Property or the Update If Null Property of the item.
Level: 99
Trigger: ON-ERROR

FRM-41023: Cannot set UPDATE ALLOWED Property of secure item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in tried to change the Update Allowed Property of a database item which the user does not have permission to update.

Action: Either correct the SET_ITEM statement or grant update permission on the column to the user.
Level: 99
Trigger: ON-ERROR

FRM-41024: Cannot set UPDATE_NULL Property of secure item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in tried to change the Update If Null Property of a database item which the user does not have permission to update.

Action: Either correct the SET_ITEM statement or grant update permission on the column to the user.
Level: 99
Trigger: ON-ERROR

FRM-41025: Page number %d does not exist.
Cause: Application design error. Attempted an operation on a non-existent page.

Action: Check arguments to page related Built-ins.
Level: 99
Trigger: ON-ERROR

FRM-41026: Field does not understand operation.
Cause: You attempted to perform an operation that is invalid for the given item type.

Action: Do not attempt to perform the operation on an item to which the operation cannot be applied.
Level: 99
Trigger: ON-ERROR

FRM-41027: Primary key must be defined for this block.
Cause: There are no primary key items in the block and one of the following has happened:
1. You attempted to set Key_Mode of primary key option on for the block.
2. The Key_Mode is set to Automatic and the datasource to which your are connected does not support UNIQUE key mode.

Action: Specify one or more primary key items on the block.
Level: 99
Trigger: ON-ERROR

FRM-41028: Invalid property.
Cause: You passed an invalid property constant to a Get or Set property Built-in.

Action: Verify arguments.
Level: 99
Trigger: ON-ERROR

FRM-41029: Invalid parameter.
Cause: You attempted to set a form, block, item, or record property to an invalid value.

Action: Verify arguments to SET_FORM_PROPERTY, SET_BLOCK_PROPERTY, SET_ITEM_PROPERTY, or SET_RECORD_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41030: Cannot reset ITEM_LENGTH of item %s.%s.
Cause: Application design error. Attempted to change the length of a fixed length item.

Action: Statically declare the item to be of the maximum necessary length or change item type.
Level: 99
Trigger: ON-ERROR

FRM-41031: Cannot reset ITEM_LENGTH greater than the allocated buffer.
Cause: Application design error. Tried to reset ITEM_LENGTH greater than the allocated buffer.

Action: Increase item length in the form definition.
Level: 99
Trigger: ON-ERROR

FRM-41032: Cannot set ENABLED Property of current item %s.%s.
Cause: A call to SET_ITEM_PROPERTY attempted to set the Enabled Property of the current item.

Action: Either correct the call to SET_ITEM_PROPERTY or navigate to another item before setting the Enabled Property.
Level: 99
Trigger: ON-ERROR

FRM-41033: Cannot set ENABLED Property of non-displayed item %s.%s.
Cause: A call to SET_ITEM_PROPERTY attempted to set the Enabled Property of a non-displayed item.

Action: First navigate to the item, then set the Enabled Property with a call to SET_ITEM_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41034: Cannot set NAVIGABLE Property of non-displayed item %s.%s.
Cause: A call to SET_ITEM_PROPERTY attempted to set the Navigable Property of a non-displayed item.

Action: First navigate to the item, then set the Navigable Property with a call to SET_ITEM_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41035: Cannot set NAVIGABLE Property of non-enabled item %s.%s.
Cause: A call to SET_ITEM_PROPERTY attempted to set the Navigable Property of a non-enabled item.

Action: First set the Enabled Property of the item with a call to SET_ITEM_PROPERTY. Then set the Navigable Property of the item with another call to SET_ITEM_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41036: Cannot modify a checkbox that does not allow querying.
Cause: You attempted to modify a check box that does not allow querying.

Action: First set the Query Allowed Property to True, then the end user may shift and click the check box to enable or disable the item.
Level: 99
Trigger: ON-ERROR

FRM-41037: Cannot modify a radio group that does not allow querying.
Cause: You attempted to modify a radio group that does not allow querying.

Action: First set the Enabled Property to True with a call to SET_RADIO_BUTTON_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41038: Item %s is not a checkbox.
Cause: A call to CHECKBOX_CHECKED was made to an item which was not a check box.

Action: Correct the call to CHECKBOX_CHECKED.
Level: 20
Trigger: ON-ERROR

FRM-41039: Invalid Alert ID %d.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_ALERT will be performed.
Level: 99
Trigger: ON-ERROR

FRM-41040: Cannot find radio button: %s.
Cause: An invalid ID or name was passed to a Built-in subprogram.

Action: Check the name or ID that you entered and try again.
Level: 99
Trigger: ON-ERROR

FRM-41041: Cannot find form module: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_FORM will be performed.
Level: 99
Trigger: ON-ERROR

FRM-41042: No such property for Set_Item_Property.
Cause: You attempted to set an invalid item property.

Action: Check the documentation for setting item properties and try again.
Level: 99
Trigger: ON-ERROR

FRM-41043: Cannot find timer: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_TIMER will be performed.
Level: 99
Trigger: ON-ERROR

FRM-41044: Error deleting timer %s
Cause: An error occurred while executing a DELETE_TIMER Built-in.

Action: Verify that your timer has been created correctly.
Level: 99
Trigger: ON-ERROR

FRM-41045: Cannot find item: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_ALERT will be performed.
Level: 99
Trigger: ON-ERROR

FRM-41046: Invalid parameter used for Set_Item_Property.
Cause: An invalid parameter was passed to SET_ITEM_PROPERTY.

Action: Verify the valid parameters for SET_ITEM_PROPERTY and try again.
Level: 99
Trigger: ON-ERROR

FRM-41047: Cannot navigate out of current block in enter-query mode.
Cause: An illegal attempt to navigate out of the current block when in Enter Query mode.

Action: Perform operation before entering query mode.
Level: 99
Trigger: ON-ERROR

FRM-41048: Procedure %s is not valid in a %s trigger.
Cause: The indicated procedure is not valid when called from the indicated trigger. The procedure may be a restricted procedure, which cannot be called from any trigger that fires during navigation.

Action: Correct the invalid trigger.
Level: 20
Trigger: ON-ERROR

FRM-41049: You cannot delete this record.
Cause: You attempted to delete a record on a block that does not allow deletes.

Action: Do not attempt to delete records in this block until you have set the Delete Allowed Property to True.
Level: 10
Trigger: ON-ERROR

FRM-41050: You cannot update this record.
Cause: You attempted to update a record on a block that does not allow updates.

Action: Do not attempt to update records in this block until you have set the Update Allowed Property to True.
Level: 10
Trigger: ON-ERROR

FRM-41051: You cannot create records here.
Cause: You attempted to create records on a block that does not allow inserts.

Action: Do not attempt to create and insert new records into this block until you have set the Insert Allowed Property to True.
Level: 10
Trigger: ON-ERROR

FRM-41052: Cannot find Window: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_WINDOW will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41053: Cannot find Canvas: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_CANVAS will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41054: No such property for Get_Record_Property.
Cause: You attempted to get a non-existent record property.

Action: Verify call to GET_RECORD_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41055: No such property for Set_Record_Property.
Cause: You attempted to set a non-existent record property.

Action: Verify call to SET_RECORD_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41056: Cannot find Block: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_BLOCK will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41057: No such property for Set_View_Property.
Cause: You attempted to set a non-existent view property.

Action: Verify call to SET_VIEW_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41058: No such property for Get_Item_Property.
Cause: You attempted to get a non-existent item property.

Action: Verify call to GET_ITEM_PROPERTY for valid property.
Level: 20
Trigger: ON-ERROR

FRM-41059: No such property for Set_Canvas_Property.
Cause: You attempted to set a non-existent canvas property.

Action: Verify call to SET_CANVAS_PROPERTY for valid property.
Level: 20
Trigger: ON-ERROR

FRM-41060: Cannot disable Primary Key Property of only key item.
Cause: You attempted to turn off the Primary Key Property on the last primary key item on a block with one of the following:
1. The Primary Key Property.
2. Key mode of the primary key.
3. The database to which you are connected does not support UNIQUE key mode.

Action: Disable the block properties first.
Level: 20
Trigger: ON-ERROR

FRM-41061: No such property for Get_Window_Property.
Cause: You attempted to get a non-existent window property.

Action: Verify call to GET_WINDOW_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41062: Cannot find Editor: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_EDITOR will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41063: Cannot create Editor.
Cause: Not enough memory available for Forms Runtime.

Action: Try calling SHOW_EDITOR again after closing some of your windows.
Level: 20
Trigger: ON-ERROR

FRM-41064: Cannot create Timer %s: illegal identifier name.
Cause: Illegal identifier name.

Action: Check legal syntax for naming timers.
Level: 99
Trigger: ON-ERROR

FRM-41065: Cannot find Menu: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_MENU will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41066: No such property for Get_Form_Property.
Cause: You attempted to get a non-existent form property.

Action: Verify call to GET_FORM_PROPERTY for valid property.
Level: 20
Trigger: ON-ERROR

FRM-41067: Cannot find Menu Item: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_MENU_ITEM will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41068: Error in Set_Menu_Item_Property.
Cause: Invalid call to SET_MENU_ITEM_PROPERTY.

Action: Verify valid parameters and try again.
Level: 20
Trigger: ON-ERROR

FRM-41069: Error in Get_Menu_Item_Property.
Cause: Invalid call to GET_MENU_ITEM_PROPERTY.

Action: Verify valid parameters and try again.
Level: 20
Trigger: ON-ERROR

FRM-41070: Unknown property for Set_Menu_Item_Property.
Cause: You attempted to set a non-existent menu item property.

Action: Verify call to SET_MENU_ITEM_PROPERTY for valid property.
Level: 20
Trigger: ON-ERROR

FRM-41071: Unknown property for Get_Menu_Item_Property.
Cause: You attempted to get a non-existent menu item property.

Action: Verify call to GET_MENU_ITEM_PROPERTY for valid property.
Level: 20
Trigger: ON-ERROR

FRM-41072: Cannot create Group %s
Cause: Caused by one of the following
1. Duplicate column names in SQL statement.
2. Invalid record group name.
3. Query is invalid.

Action: Check the group name and/or correct the SQL statement.
Level: 20
Trigger: ON-ERROR

FRM-41073: Cannot find Group: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_GROUP will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41074: Cannot find Group or Column: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_GROUP or FIND_COLUMN will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41075: Error deleting Group.
Cause: The record group name or ID specified in the call to DELETE_GROUP is invalid, or the record group was not dynamically created.

Action: Check the record group name or ID, and make sure that the specified record group was dynamically created.
Level: 20
Trigger: ON-ERROR

FRM-41076: Error populating Group.
Cause: Caused by one of the following
1. An invalid group name or ID was specified.

2. An nonempty static record group was specified.
3. Query failed due to an invalid column or table name, or the query and group column structure do not match.

Action: Correct the application logic.
Level: 20
Trigger: ON-ERROR

FRM-41077: Error deleting Group Row(s).
Cause: DELETE_GROUP_ROW cannot be used to delete records from a static record group, or you specified an invalid row number.

Action: Correct the call to DELETE_GROUP_ROW.
Level: 20
Trigger: ON-ERROR

FRM-41078: Error resetting Group selection.
Cause: Record group name or ID specified is invalid.

Action: Check the record group name or ID and try again.
Level: 20
Trigger: ON-ERROR

FRM-41079: Error adding Group column.
Cause: Caused by one of the following:
1. You cannot add columns to a group that already has rows.
2. The width of CHAR columns cannot be less than the width of the corresponding database column.
3. You entered the name of a nonexistent or invalid record group.
4. You entered the name of a nonexistent or invalid column.
5. You entered a column type other than CHAR, NUMBER, or DATE.
6. Adding the column would cause the total memory required for non-LONG columns in a record to exceed 65535 bytes, and the record manager cannot be used for the record group because the FORMS_RECMGR_ARCHIVE environment variable has been set to WRITE_TMPFILE.

Action: Do one of the following:
1. If the group already has rows, delete the rows with DELETE_GROUP_ROW, then add the column.
2. Specify valid record group names, column names, and column types.
3. Set the FORMS_RECMGR_ARCHIVE environment variable to MAP_SWAP or MAP_TMPFILE, or leave it unset. (It will default to MAP_SWAP).
Level: 20
Trigger: ON-ERROR

FRM-41080: Error adding Group row.
Cause: Caused by one of the following:
1. You attempted to add rows to a group that is nonexistent or has no columns.
2. You entered the name of a nonexistent record group.
3. You provided a row number that is out of range or invalid.

Action: Create the group and add columns first. Check the call to ADD_GROUP_ROW to make sure that the record group name and row number are valid.
Level: 20
Trigger: ON-ERROR

FRM-41081: Cannot move Item: invalid position.
Cause: You attempted to move the item to an invalid position on the canvas.

Action: Make sure the coordinates you chose in your call to SET_ITEM_PROPERTY are valid.
Level: 99
Trigger: ON-ERROR

FRM-41082: Cannot resize item: position of item places it off of canvas.
Cause: The height and/or width you specified in your call to SET_ITEM_PROPERTY is invalid, or the height and/or width you specified causes the item to extend off of the canvas.

Action: Correct the call to SET_ITEM_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41083: No such property for Set_Form_Property
Cause: You attempted to set a nonexistent form property.

Action: Verify call to SET_FORM_PROPERTY for valid property.
Level: 20
Trigger: ON-ERROR

FRM-41084: Error getting Group Cell.
Cause: Invalid call to GET_GROUP_CHAR_CELL, GET_GROUP_DATE_CELL, OR GET_GROUP_NUMBER_CELL.

Action: Make sure the column type is of CHAR, DATE, or NUMBER, respectively. Check the validity of the row number and column name specified.
Level: 20
Trigger: ON-ERROR

FRM-41085: Error getting Group Row count.
Cause: Invalid call to GET_GROUP_ROW_COUNT.

Action: Check the record group name and try again.
Level: 20
Trigger: ON-ERROR

FRM-41086: Error getting Group selection count.
Cause: You specified an invalid record group name. Invalid call to GET_GROUP_SELECTION_COUNT.

Action: Correct the call to GET_GROUP_SELECTION.
Level: 20
Trigger: ON-ERROR

FRM-41087: Error getting Group selection.
Cause: You specified an invalid record group name or selection number. Invalid call to GET_GROUP_SELECTION.

Action: Correct the call to GET_GROUP_SELECTION.
Level: 20
Trigger: ON-ERROR

FRM-41088: Cannot set Group selection.
Cause: You specified an invalid record group name, ID, or row number.

Action: Correct the call to SET_GROUP_SELECTION.
Level: 20
Trigger: ON-ERROR

FRM-41089: Cannot move View: invalid position.
Cause: The x, y pair specified in the call to SET_VIEW_PROPERTY is invalid.

Action: Correct the call to SET_VIEW_PROPERTY by making sure that the position specified by your coordinates is on the canvas.
Level: 99
Trigger: ON-ERROR

FRM-41090: Invalid item type for go_item: %s.
Cause: You cannot navigate to the item.

Action: Check to make sure the item is a navigable item.
Level: 20
Trigger: ON-ERROR

FRM-41091: Cannot find LOV: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_LOV will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41092: No records in block %s.
Cause: You attempted to place a value into an item on a block that has no records.

Action: Put records in the block first.
Level: 20
Trigger: ON-ERROR

FRM-41093: Error setting item property: %s.
Cause: You specified Lock Record and the item was not a text item, or you specify Case Insensitive Query and the data type was not ALPHA or CHAR.

Action: In the case of Lock Record, make sure that the item is a text item. When specifying Case Insensitive Query, make sure that the data type is ALPHA or CHAR.
Level: 20
Trigger: ON-ERROR

FRM-41094: No such property for Get_View_Property.
Cause: You attempted to get a non-existent view property.

Action: Verify call to GET_VIEW_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41095: No such property for Get_Canvas_Property.
Cause: You attempted to get a non-existent canvas property.

Action: Verify call to GET_CANVAS_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41096: Cannot resize View: invalid size.
Cause: The x, y coordinates place the view off the canvas.

Action: Choose another x, y pair.
Level: 99
Trigger: ON-ERROR

FRM-41097: Cannot resize Canvas: invalid size.
Cause: The x, y coordinates place the view off the window.

Action: Choose another x, y pair.
Level: 99
Trigger: ON-ERROR

FRM-41098: Cannot modify Display Position of a content view.
Cause: The Display Position Property applies to a stacked canvas-view only.

Action: Correct the call to SET_VIEW_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41099: Cannot modify Size of a content view.
Cause: The size of a content view is dependent on window size. Only stacked view sizes may be modified using SET_VIEW_PROPERTY.

Action: Correct the call to SET_VIEW_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41100: Cannot find relation %s.
Cause: You attempted to get, set, or find using an invalid relation.

Action: Check call to Built-in for correct arguments.
Level: 99
Trigger: ON-ERROR

FRM-41101: No such property for Get_Relation_Property.
Cause: You attempted to get a non-existent relation property.

Action: Verify call to GET_RELATION_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41102: No such property for Set_Relation_Property.
Cause: You attempted to set a non-existent relation property.

Action: Verify call to SET_RELATION_PROPERTY for valid property.
Level: 99
Trigger: ON-ERROR

FRM-41103: No such property value for Set_Relation_Property.
Cause: Application design error. Improper relation property value passed to SET_RELATION_PROPERTY Built-in.

Action: Correct call to SET_RELATION_PROPERTY Built-in and retry.
Level: 99
Trigger: ON-ERROR

FRM-41104: Cannot find Relation: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_RELATION will be performed.
Level: 20
Trigger: ON-ERROR

FRM-41105: You cannot query records without a saved parent record.
Cause: You attempted to query detail records without first creating a master record.

Action: Create a master record, and then query the detail records.
Level: 10
Trigger: ON-ERROR

FRM-41106: You cannot create records without a parent record.
Cause: You attempted to create new detail records without first creating a master record.

Action: Create a master record, and then add the detail records.
Level: 10
Trigger: ON-ERROR

FRM-41107: Master delete option for the relation is invalid.
Cause: An invalid query data source type or an invalid DML data target type is specified for the detail block.

Action: Verify that the detail block's query data source and the DML data targets are of type table.
Level: 99
Trigger: ON-ERROR

FRM-41200: Integration error: invalid product.
Cause: Invalid product name specified during integration.

Action: Check the integration parameters.
Level: 99
Trigger: ON-ERROR

FRM-41201: Integration error: communication mode must be SYNCHRONOUS or ASYNCHRONOUS.
Cause: Invalid communication mode specified in RUN_REPORT_OBJECT.

Action: Check the RUN_REPORT_OBJECT parameters and try again.
Level: 20
Trigger: ON-ERROR

FRM-41202: Integration error: parameter list %s has no parameters.
Cause: Parameter list has no arguments.

Action: Check the specified parameter list for parameters.
Level: 20
Trigger: ON-ERROR

FRM-41203: Integration error: invalid parameter list ID.
Cause: An invalid parameter list ID was passed.

Action: Check the parameter list ID name and try again.
Level: 20

Trigger: ON-ERROR

FRM-41204: Integration error: memory allocation error.
Cause: An internal error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 20
Trigger: ON-ERROR

FRM-41208: Integration error: execution mode must be BATCH or RUNTIME.
Cause: Invalid execution mode specified in RUN_REPORT_OBJECT.

Action: Specify either BATCH or RUNTIME for the execmode parameter.
Level: 20
Trigger: ON-ERROR

FRM-41209: Integration error: document location must be FILESYSTEM or DB.
Cause: Invalid document location specified while integrating with another product.

Action: Specify either FILESYSTEM or DB for the location parameter.
Level: 20
Trigger: ON-ERROR

FRM-41211: Integration error: SSL failure running another product.
Cause: There is a problem detected when launching another product.

Action: Check the RUN_REPORT_OBJECT Built-in.
Level: 99
Trigger: ON-ERROR

FRM-41212: Integration error: invalid communication mode for data exchange
Cause: User specified an asynchronous RUN_REPORT_OBJECT.

Action: Change to a synchronous RUN_REPORT_OBJECT communication mode.
Level: 20
Trigger: ON-ERROR

FRM-41213: Unable to connect to the Report server %s.
Cause: There is a problem connecting to the specified Report server.

Action: Check the Report server and make sure it is up and running.
Level: 99
Trigger: ON-ERROR

FRM-41214: Unable to run report.
Cause: The report server was unable to run the specified report.

Action: Check the Report server and make sure it is up and running.
Level: 99
Trigger: ON-ERROR

FRM-41215: Invalid server name or jobid.
Cause: There is a problem decoding the return value from the Built-in run_report.

Action: The return value from the Built-in run_report should not be modified before being passed to another report Built-in.
Level: 99

Trigger: ON-ERROR

FRM-41216: Unable to cancel job.
Cause: There is a problem cancelling a report job.

Action: Check the Report server and make sure that the specified job exists.
Level: 99
Trigger: ON-ERROR

FRM-41217: Unable to get report job status.
Cause: There is a problem getting report status for a given report job.

Action: Check the Report server and make sure that the specified job exists.
Level: 99
Trigger: ON-ERROR

FRM-41218: Unable to copy report output.
Cause: There is a problem copying report output for a given report job.

Action: Check the Report server and make sure that the specified output file exists.
Level: 99
Trigger: ON-ERROR

FRM-41219: Cannot find report: invalid ID.
Cause: The user has specified an invalid report object name.

Action: Check the form and make sure that the report object exists.
Level: 99
Trigger: ON-ERROR

FRM-41220: Failed to authenticate user.
Cause: There was a failure in displaying the web report.

Action: Check if the user credentials are valid against identity store in use.
Level: 99
Trigger: ON-ERROR

FRM-41221: Failed to connect to identity store.
Cause: There was a failure in connecting to authentication service of identity store.

Action: Check if the authentication service is running.
Level: 99
Trigger: ON-ERROR

FRM-41222: Invalid property for this report object type.
Cause: There was a reports property mismatch.

Action: Make sure properties being handled belong to correct report object type. It is a property mismatch if a property belonging to reports object type 'ORAREPORTS' is being operated for reports object of type 'ORABIP' and vice versa.
Level: 99
Trigger: ON-ERROR

FRM-41223: BI Publisher integration error.
Cause: There was an error while invoking BI Publisher.

Action: Contact your system administrator.

Level: 99
Trigger: ON-ERROR

FRM-41224: Invalid BI Publisher service location.
Cause: You specified an invalid value for service location.

Action: Make sure value of service location property is complete.
Level: 99
Trigger: ON-ERROR

FRM-41225: SSL mandatory but service location non-SSL.
Cause: A non-SSL service location was specified when SSL Connection property set to mandatory.

Action: Make sure an SSL service location is specified when SSL Connection configured. Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41226: Access denied on BI Publisher server.
Cause: You do not have authority to access BI Publisher server.

Action: Make sure correct BI Publisher user name and password specified. Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41227: Invalid parameter reported by BI Publisher.
Cause: BI Publisher call caused InvalidParametersException.

Action: Make sure all of the report properties are valid.
Level: 99
Trigger: ON-ERROR

FRM-41228: Operation failed at BI Publisher server.
Cause: There was an error on BI Publisher server when executing the operation.

Action: Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41229: Output file name missing.
Cause: No file name was specified when calling operation to copy report output.

Action: Make sure a valid file name is specified for the operatin to copy report output.
Level: 99
Trigger: ON-ERROR

FRM-41230: File Not Found Exception.
Cause: FileNotFoundException thrown when trying to copy report output.

Action: Make sure you have file creation priviledge for the file name provided. Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41231: Failed to write report output.
Cause: IOException thrown when trying to write report output.

Action: Check your write priviledge and space on the disk. Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41232: Failed to initialize JVM.
Cause: Unable to initialize JVM required to invoke BI Publisher call.

Action: Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41300: Invalid parameter used for Set_Radio_Button_Property.
Cause: You specified a parameter that does not exist.

Action: Check the list of legal parameters.
Level: 99
Trigger: ON-ERROR

FRM-41301: Invalid parameter used for Set_View_Property.
Cause: You specified a parameter that does not exist.

Action: Check the list of legal parameters.
Level: 99
Trigger: ON-ERROR

FRM-41302: Invalid parameter used for Set_Canvas_Property.
Cause: You specified a parameter that does not exist.

Action: Check the list of legal parameters.
Level: 99
Trigger: ON-ERROR

FRM-41303: No such property for Set_Window_Property.
Cause: You specified a property that does not exist.

Action: Check the list of legal properties.
Level: 99
Trigger: ON-ERROR

FRM-41304: No such property for Set_Block_Property.
Cause: You specified a property that does not exist.

Action: Check the list of legal properties.
Level: 99
Trigger: ON-ERROR

FRM-41305: No such property for Get_Block_Property.
Cause: You specified a property that does not exist.

Action: Check the list of legal properties.
Level: 99
Trigger: ON-ERROR

FRM-41306: Invalid parameter used for Set_Window_Property.
Cause: You specified a parameter that is not valid.

Action: Check the list of valid parameters.
Level: 99
Trigger: ON-ERROR

FRM-41307: Invalid parameter used for Set_Block_Property.
Cause: You specified a parameter that is not valid.

Action: Check the list of valid parameters.
Level: 99
Trigger: ON-ERROR

FRM-41308: Error unsetting Group selection.
Cause: You tried to deselect a record or a subset of records that was not selected or is not in the record group.

Action: Check the records that are expected in the group.
Level: 20
Trigger: ON-ERROR

FRM-41309: No such property for Get_Radio_Button_Property.
Cause: You specified a property that is invalid.

Action: Check the list of valid properties.
Level: 99
Trigger: ON-ERROR

FRM-41310: No such property for Set_Radio_Button_Property.
Cause: You specified a property that is invalid.

Action: Check the list of valid properties.
Level: 99
Trigger: ON-ERROR

FRM-41311: Invalid argument or argument ordering for %s.
Cause: You supplied an incorrect argument list.

Action: Check the list of valid arguments.
Level: 99
Trigger: ON-ERROR

FRM-41312: Must have at least one writable item in block.
Cause: A block with the Insert Allowed Property or Update Allowed Property set to True must have at least one writable item. You attempted to make the only remaining base table item in the block not writable by setting either the Derived Column Property or Query Only Property to True.

Action: Set Insert Allowed or Update Allowed to False for the block, rather than setting Derived Column or Query Allowed to False for each item.
Level: 10
Trigger: ON-ERROR

FRM-41313: No such property for Set_Alert_Property.
Cause: An invalid property has been specified for SET_ALERT_PROPERTY.

Action: Enter a valid alert property.
Level: 99
Trigger: ON-ERROR

FRM-41314: Cannot set Insert Allowed Property of current item %s.%s
Cause: You attempted to set the Insert Allowed Property for a current item.

Action: The Insert Allowed Property is only valid on non-current items. Make sure the item is not current.
Level: 99
Trigger: ON-ERROR

FRM-41315: Cannot set Insert Allowed Property of non-displayed item %s.%s
Cause: You tried to set Insert Allowed Property for a non-displayed item.

Action: The Insert Allowed Property is only valid on displayed items. Make sure the item is displayed.
Level: 99
Trigger: ON-ERROR

FRM-41316: Cannot set Insert Allowed Property of disabled item %s.%s
Cause: You tried to set Insert Allowed Property for a disabled item.

Action: The Insert Allowed Property is only valid on enabled items. Make sure the item is enabled.
Level: 99
Trigger: ON-ERROR

FRM-41317: Item is not a radio button %s
Cause: You tried to use a radio button Built-in with an item that is not a radio button.

Action: Make sure the item is a radio button.
Level: 99
Trigger: ON-ERROR

FRM-41318: Item %s is not a VBX item.
Cause: You tried to use a VBX Built-in with an item that is not a VBX item.

Action: Make sure the item is a VBX item.
Level: 99
Trigger: ON-ERROR

FRM-41319: Invalid property %s specified for VBX item %s.
Cause: You tried to get or set an invalid property for the specified VBX item.

Action: Make sure the property is valid for the specified VBX item.
Level: 99
Trigger: ON-ERROR

FRM-41320: Unable to get property %s for VBX item %s.
Cause: Could not get the valid property for the VBX item.

Action: Check the list of legal properties.
Level: 99
Trigger: ON-ERROR

FRM-41321: Unable to set property %s for VBX item %s.
Cause: Could not set the valid property for the VBX item.

Action: Check the list of legal properties.
Level: 99
Trigger: ON-ERROR

FRM-41322: Invalid event %s for VBX item %s.
Cause: You tried to get or set an invalid event for the specified VBX item.

Action: Make sure the event is valid for the specified VBX item.
Level: 99
Trigger: ON-ERROR

FRM-41323: Too many parameters for event %s for VBX item %s.
Cause: You specified too many parameters for the event name for the VBX item.

Action: Make sure there is a valid number of parameters for the event.
Level: 99
Trigger: ON-ERROR

FRM-41324: Too few parameters for event %s for VBX item %s.
Cause: You specified too few parameters for the event name for the VBX item.

Action: Make sure there is a valid number of parameters for the event.
Level: 99
Trigger: ON-ERROR

FRM-41325: VBX event parameter must be a string.
Cause: The VBX event parameter is not a string.

Action: Make sure the VBX event parameter is a string.
Level: 99
Trigger: ON-ERROR

FRM-41326: Failed to deliver event %s to VBX item %s.
Cause: The VBX event failed.

Action: Make sure the event is valid for the specified VBX item.
Level: 99
Trigger: ON-ERROR

FRM-41327: Failed to get default property for VBX item %s.
Cause: The VBX.GET_VALUE_PROPERTY Built-in failed.

Action: Make sure an initial value is assigned to the VB Control Value property.
Level: 99
Trigger: ON-ERROR

FRM-41328: Failed to set default property for VBX item %s.
Cause: The VBX.SET_VALUE_PROPERTY Built-in failed.

Action: Make sure you are setting a valid value property.
Level: 99
Trigger: ON-ERROR

FRM-41329: Item %s is not a List item.
Cause: You tried to add a list element to an item that is not a list.

Action: Make sure the item is a List item.
Level: 99
Trigger: ON-ERROR

FRM-41330: Could not insert list element into %s.
Cause: You tried to insert an other values element when the block contained either queried or changed records.

Action: For more information, refer to help for restrictions on <a href="../builta_c/addliste.html">ADD_LIST_ELEMENT Built-in</a>.
Level: 99
Trigger: ON-ERROR

FRM-41331: Could not delete element from %s.
Cause: Caused by one of the following:
You tried to delete the other values element when the block contained either queried or changed records.
You tried to delete an element from a list that does not contain an other values element when the block contained either queried or changed records.

Action: For more information, refer to help for restrictions on <a href="../builta_c/clearlis.html">CLEAR_LIST</a> and <a href="../builtd_f/dellsele.html">DELETE_LIST_ELEMENT</a>.
Level: 99
Trigger: ON-ERROR

FRM-41332: List element index out of range.
Cause: An invalid index (e.g. a negative number) was specified to the Add_List_Element Built-in.

Action: Correct the index in the call to Add_List_Element.
Level: 99
Trigger: ON-ERROR

FRM-41333: Cannot convert list element value.
Cause: Could not resolve list element value to a string.

Action: Make sure list element is a string.
Level: 99
Trigger: ON-ERROR

FRM-41334: Invalid record group for list population.
Cause: You tried to populate a list from a record group that does not exist.

Action: Make sure the record group exists.
Level: 99
Trigger: ON-ERROR

FRM-41335: Populate_List: invalid column type for column 1.
Cause: The record group does not have a column of the same type.

Action: Make sure record group has a column of the same type.
Level: 99
Trigger: ON-ERROR

FRM-41336: Populate_List: invalid column type for column 2.
Cause: The record group does not have a column of the same type.

Action: Make sure record group has a column of the same type.
Level: 99
Trigger: ON-ERROR

FRM-41337: Cannot populate the list from record group.
Cause: The record group is invalid or the list item does not satisfy the requirements for deleting and adding elements.

Action: Make sure the record group is valid. For more information about deleting and adding list elements, refer to help for restrictions on <a href="../builtd_f/dellsele.html">DELETE_LIST_ELEMENT</a> and <a href="../builta_c/addliste.html">ADD_LIST_ELEMENT</a>.
Level: 99
Trigger: ON-ERROR

FRM-41338: Cannot retrieve the list into record group.
Cause: The record group is invalid.

Action: Make sure the record group is valid.
Level: 99
Trigger: ON-ERROR

FRM-41339: Cannot clear the list.
Cause: A memory allocation failed when Forms Runtime attempted to clear a list.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-41340: No such property or value for Set_Application_Property.
Cause: You specified an invalid property and/or an invalid value for a property.

Action: Specify a valid property and/or a valid value.
Level: 99
Trigger: ON-ERROR

FRM-41341: Invalid cursor shape %s specified.
Cause: There is a predefined set of cursor types, and an invalid cursor type was specified.

Action: Specify a valid cursor type.
Level: 99
Trigger: ON-ERROR

FRM-41342: Invalid parameter %s specified for VBX event %s.
Cause: You specified an invalid parameter for a VBX event.

Action: Check the parameter type.
Level: 99
Trigger: ON-ERROR

FRM-41343: Item %s is not an OLE object.
Cause: Invalid item passed to OLE Built-in.

Action: Specify a valid OLE item.
Level: 99
Trigger: ON-ERROR

FRM-41344: OLE object not defined for %s in the current record.
Cause: An empty OLE container is defined.

Action: Define an OLE object to reside in the OLE container.
Level: 20
Trigger: ON-ERROR

FRM-41345: Cannot find the verb %s for this server.
Cause: You specified an invalid OLE verb.

Action: Specify a valid OLE verb.
Level: 99
Trigger: ON-ERROR

FRM-41346: Cannot determine the verb count for OLE object %s.
Cause: Could not communicate with OLE server.

Action: Re-install the OLE server.
Level: 99
Trigger: ON-ERROR

FRM-41347: Invalid verb index for OLE object %s.
Cause: You provided an index that is greater than the verb count.

Action: Check the index value.
Level: 99
Trigger: ON-ERROR

FRM-41348: OLE server error: %s.
Cause: OLE server detects an error.

Action: Try to resolve the error based on the message from the OLE server.
Level: 99
Trigger: ON-ERROR

FRM-41349: OLE object %s cannot execute verb; verb id %d
Cause: OLE object does not recognize the verb.

Action: Try to execute another verb.
Level: 99
Trigger: ON-ERROR

FRM-41350: OLE object is currently not displayed.
Cause: You tried to close a server that is not running.

Action: Ask if the server is active in a record that is not currently active.
Level: 99
Trigger: ON-ERROR

FRM-41351: Cannot navigate out of current form.
Cause: You cannot navigate to an inactive form.

Action: Check to make sure the form you are navigating to is active.

Level: 99
Trigger: ON-ERROR

FRM-41352: Failed to create a new session.
Cause: You attempted to open a new form with a new session.

Action: Check the database server.
Level: 99
Trigger: ON-ERROR

FRM-41353: Cannot start another call form.
Cause: You went to a peer form and performed a call form.

Action: Make sure you are not at a peer form when calling the form.
Level: 99
Trigger: ON-ERROR

FRM-41354: Cannot close form %s.
Cause: Unsuccessful attempt to close a form.

Action: Make sure the form is open.
Level: 99
Trigger: ON-ERROR

FRM-41355: Cannot navigate to form %s.
Cause: You cannot navigate to an inactive form.

Action: Check to make sure you are navigating to an active form.
Level: 99
Trigger: ON-ERROR

FRM-41356: Invalid method %s for VBX item %s.
Cause: You specified an invalid method name for the VBX item.

Action: Specify a valid method name for the VBX item.
Level: 99
Trigger: ON-ERROR

FRM-41357: Incorrect number of arguments to method %s for VBX item %s.
Cause: You specified an incorrect number of arguments to the method for the VBX item.

Action: Make sure the number of arguments is what the VBX item expects.
Level: 99
Trigger: ON-ERROR

FRM-41358: Method %s failed for VBX item %s.
Cause: You specified an invalid method name for the VBX item.

Action: Specify a valid method name for the VBX item.
Level: 99
Trigger: ON-ERROR

FRM-41359: The Open_Form session feature is not enabled. Cannot create new session.
Cause: You do not have the multiple sessioning feature enabled on the database.

Action: The Open_Form session feature is only available for use against a database with multiple sessioning enabled.
Level: 99
Trigger: ON-ERROR

FRM-41360: Invalid value used in Set_Window_Property for window %s.
Cause: You are using an invalid value when attempting to set a window property.

Action: Specify a valid window property value.
Level: 99
Trigger: ON-ERROR

FRM-41361: Cannot navigate out of current form in Enter-Query mode.
Cause: You are in Enter-Query mode and trying to navigate to another form when using Open Form.

Action: Exit Enter-Query mode and try again.
Level: 10
Trigger: ON-ERROR

FRM-41362: No such property for Set_Alert_Button_Property.
Cause: You specified an invalid property for Set_Alert_Button_Property.

Action: Specify a valid property for Set_Alert_Button_Property.
Level: 99
Trigger: ON-ERROR

FRM-41363: No such property for Set_LOV_Column_Property.
Cause: You specified an invalid property for Set_LOV_Column_Property.

Action: Specify a valid property for Set_LOV_Column_Property.
Level: 99
Trigger: ON-ERROR

FRM-41364: Invalid column number specified for LOV %s.
Cause: You specified an invalid column number for the LOV.

Action: Specify a valid column number for the LOV.
Level: 99
Trigger: ON-ERROR

FRM-41365: No such property for Set_TabPage_Property.
Cause: You specified an invalid property for Set_TabPage_Property.

Action: Specify a valid property for Set_TabPage_Property.
Level: 99
Trigger: ON-ERROR

FRM-41366: No such property for Get_TabPage_Property.
Cause: You specified an invalid property parameter.

Action: Check the list of valid properties.
Level: 99
Trigger: ON-ERROR

FRM-41367: Cannot find TabPage: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to FIND_TABPAGE will be performed.
Level: 99
Trigger: ON-ERROR

FRM-41368: Invalid parameter used for Set_TabPage_Property.
Cause: You specified a parameter that is not valid.

Action: Check the list of valid parameters.
Level: 99
Trigger: ON-ERROR

FRM-41369: Cannot insert a second record into a single-record block.
Cause: You (or the application) have attempted to insert a second record into a block whose Single Record Property is TRUE.

Action: Don't attempt to insert a record into such a block.
Level: 99
Trigger: ON-ERROR

FRM-41370: Cannot modify calculated item %s.%s.
Cause: Application design error. The application attempted to assign a value to a calculated item.

Action: If the calculated item is a formula item, then its formula determines its value at all times. It may be appropriate to modify the formula. Or it may be appropriate to change the calculated item to a non-calculated control item whose value is set in various triggers.
Level: 99
Trigger: ON-ERROR

FRM-41371: Cannot set INSERT_ALLOWED Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in attempted to set a calculated item's INSERT_ALLOWED Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41372: Cannot set ITEM_IS_VALID Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in attempted to set a calculated item's ITEM_IS_VALID Property to FALSE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41373: Cannot set LOCK_RECORD Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in attempted to set a calculated item's LOCK_RECORD Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41374: Cannot set PRIMARY_KEY Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in attempted to set a calculated item's PRIMARY_KEY Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41375: Cannot set QUERYABLE Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in attempted to set a calculated item's QUERYABLE Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41376: Cannot set REQUIRED Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in attempted to set a calculated item's REQUIRED Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41377: Cannot set UPDATEABLE Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in attempted to set a calculated item's UPDATEABLE Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41378: Cannot set UPDATE_NULL Property of calculated item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY Built-in attempted to set a calculated item's UPDATE_NULL Property to TRUE.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41379: Cannot recalculate non-formula item %s.%s.
Cause: Application design error. A RECALCULATE Built-in specified an item which is not a formula item.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41380: Cannot set the blocks query data source.
Cause: The user attempt to change the block's data source dynamically has failed.

Action: Check the form and make sure that the specified block is not a control block and the block status is new.
Level: 99
Trigger: ON-ERROR

FRM-41381: Cannot set the blocks DML data target source.
Cause: The user attempt to change the block's DML data target dynamically has failed.

Action: Check the form and make sure that the specified block is not a control block and the block status is new.
Level: 99
Trigger: ON-ERROR

FRM-41382: No such property for Get_Item_Instance_Property.
Cause: Application design error. A GET_ITEM_INSTANCE_PROPERTY Built-in specified an invalid property.

Action: Change the property to one that is documented as supported by the built-in, or else remove the call to the built-in.
Level: 99
Trigger: ON-ERROR

FRM-41383: No such property for Set_Item_Instance_Property.
Cause: Application design error. A SET_ITEM_INSTANCE_PROPERTY Built-in specified an invalid property.

Action: Change the property to one that is documented as supported by the built-in, or else remove the call to the built-in.
Level: 99
Trigger: ON-ERROR

FRM-41384: Invalid parameter used for Set_Item_Instance_Property.
Cause: Application design error. A SET_ITEM_INSTANCE_PROPERTY Built-in specified an invalid value for a property.

Action: Modify or remove the call to the built-in.
Level: 99
Trigger: ON-ERROR

FRM-41385: Maximum number of queried records exceeded.
Cause: The user specified maximum number of records for a given block is reached.

Action: Check the forms block and form level properties.
Level: 99
Trigger: ON-ERROR

FRM-41386: Cannot set VISIBLE Property of tab page containing current item.
Cause: You tried to set the Visible Property for the tab page which contains the current item.

Action: The Visible Property is only valid for tab pages which don't contain the current item. Navigate to an item on a different tab page or different canvas first.
Level: 99
Trigger: ON-ERROR

FRM-41387: Cannot set VISIBLE Property of last enterable tab page.
Cause: You tried to set the Visible Property for the only enterable tab page on the canvas.

Action: Make sure there is at least one other enterable tab page on the canvas before trying to set the Visible Property.
Level: 99
Trigger: ON-ERROR

FRM-41388: Cannot set ENABLED Property of tab page containing current item.
Cause: You tried to set the Enabled Property for the tab page which contains the current item.

Action: The property is only valid for tab pages which don't contain the current item. Navigate to an item on a different tab page or different canvas first.
Level: 99
Trigger: ON-ERROR

FRM-41389: Cannot set ENABLED Property of last enterable tab page.
Cause: You tried to set the Enabled Property for the only enterable tab page on the canvas.

Action: Make sure there is at least one other enterable tab page on the canvas before trying to set the Enabled Property.
Level: 99
Trigger: ON-ERROR

FRM-41390: Cannot set REQUIRED Property of subordinate mirror item %s.%s.
Cause: Application design error. A SET_ITEM_PROPERTY or SET_ITEM_INSTANCE_PROPERTY Built-in attempted to set the Required Property of a subordinate mirror item. The Required Property will be obtained from the master mirror item (the item specified by the Synchronize With Item Property).

Action: Set the Required Property of the master mirror item.
Level: 25
Trigger: ON-ERROR

FRM-41391: Cannot find visual attribute: invalid ID.
Cause: An invalid ID was passed to a Built-in subprogram.

Action: Verify that a proper call to Find_VA will be performed.
Level: 99
Trigger: ON-ERROR

FRM-41392: No such property for Get_VA_Property.
Cause: You attempted to get a non-existent visual attribute property.

Action: Verify call to Get_VA_Property for a valid property.
Level: 99
Trigger: ON-ERROR

FRM-41393: No such property for Set_VA_Property.
Cause: You attempted to set an invalid visual attribute property.

Action: Check the documentation for setting visual attribute properties and try again.
Level: 99
Trigger: ON-ERROR

FRM-41394: Invalid parameter value used for Set_VA_Property.
Cause: You attempted to set an invalid value for a visual attribute property.

Action: Check the documentation for setting visual attribute properties and try again.
Level: 99
Trigger: ON-ERROR

FRM-41395: Invalid parameter used for Set_Report_Object_Property.
Cause: You specified a parameter that does not exist.

Action: Check the list of legal parameters.
Level: 99
Trigger: ON-ERROR

FRM-41396: No such property for Get/Set_Report_Object_Property.
Cause: You specified a property that does not exist.

Action: Check the list of legal properties.
Level: 99
Trigger: ON-ERROR

FRM-41397: Invalid parameter used for MESSAGE.
Cause: Application design error. A MESSAGE Built-in specified an invalid option. The option must be one of ACKNOWLEDGE, NO_ACKNOWLEDGE, RUEI_BEGIN, RUEI_END, ODL_DEBUG, ODL_NOTIFICATION, ODL_WARNING, or ODL_ERROR.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-41402: Invalid type of visual attribute passed to Set_<object>_Property.
Cause: You attempted to set an object's visual attribute to a VA of the wrong type.

Action: Verify VA types in the Builder and specify a valid VA for this object.
Level: 99
Trigger: ON-ERROR

FRM-41403: Cannot set DEFAULT_WHERE: invalid value.
Cause: The user attempted to set the DEFAULT_WHERE to an invalid value.

Action: Check the value you chose in your call to SET_BLOCK_PROPERTY is valid.
Level: 99
Trigger: ON-ERROR

FRM-41411: SELECTED_RADIO_BUTTON property allowed only on a radio group.
Cause: The user attempted to obtain the SELECTED_RADIO_BUTTON property for an item which is not a radio group.

Action: Check the value that was specified for the item.
Level: 99
Trigger: ON-ERROR

FRM-41412: Cannot set scrollbar position for specified block.
Cause: The user attempted to set a scrollbar position property for a block which has no scrollbar.

Action: Check the value that was specified for the block.
Level: 99
Trigger: ON-ERROR

FRM-41413: Cannot get scrollbar position for specified block.
Cause: The user attempted to get a scrollbar position property for a block which has no scrollbar.

Action: Check the value that was specified for the block.
Level: 99
Trigger: ON-ERROR

FRM-41414: Combo box item element %s is longer than Maximum Length.
Cause: The label for combo box item element %s is longer than Maximum Length.

Action: Reduce the number of characters in the element's label.
Level: 99
Trigger: ON-ERROR

FRM-41415: Cannot set record distance: position of item places it off of canvas.
Cause: The record distance you specified in your call to SET_ITEM_PROPERTY is invalid, or the record distance you specified causes the item to extend off of the canvas.

Action: Correct the call to SET_ITEM_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-41416: No credentials provided. Application running in suppress logon mode.
Cause: Forms is exiting as no credentials are provided and the application is running in suppress logon mode.

Action: Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-41417: Block QRCN support not available - QRCN turned off.
Cause: The underlying block query failed the QRCN subscription registration.

Action: Check the underlying query table for the cause. Too many columns in the query, unsupported column type, or GUARANTEED mode could not be supported for the QRCN Subscription Mode (try Best Effort if possible).
Level: 99
Trigger: ON-ERROR

FRM-41418: invalid record group column type
Cause: A record group column of an invalid type was passed to a record group builtin, or was encountered in a record group passed to a record group builtin. In the case where the builtin is POPULATE_GROUP_FROM_JSON, no column in the passed-in record group may be of type REF, and if the passed-in JSON collection is a JSON object, the first column in the record group must be of type CHAR, LONG, or NUM.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41419: attempted update to nonempty static record group
Cause: An attempt was made to programmatically update a nonempty static record group at run time.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41420: CURRENT_RECORD not valid when setting ROW_VISUAL_ATTRIBUTE.
Cause: The CURRENT_RECORD constant is not valid to be used in Set_Item_Instance_Property when setting the ROW_VISUAL_ATTIBUTE.

Action: Correct the application logic by explicitly specifying the desired row number or by using :SYSTEM.CURSOR_ROW.
Level: 99
Trigger: ON-ERROR

FRM-41421: Invalid row %d for item %s.%s.
Cause: You attempted to set the row visual attribute for an invalid row number for the given item.

Action: Verify your Set_Item_Instance_Property parameters.
Level: 20
Trigger: ON-ERROR

FRM-41422: Unsupported sort column datatype for item %s.%s.
Cause: An attempt was made to sort a block item with a datatype which is not supported.

Action: Sort column can only be of type char, number, or date.
Level: 99
Trigger: ON-ERROR

FRM-41423: Inadequate memory to sort column.
Cause: An attempt to sort column data failed because of inadequate system memory.

Action: Check system memory and insure it's adequate for running the application.
Level: 99
Trigger: ON-ERROR

FRM-41424: Unsupported Filter Summary column datatype for item %s.%s.
Cause: Attempt to perform a Filter Summary on a block item with a datatype which is not supported.

Action: Filter Summary column can only be of type CHAR, NUMBER,or DATE.
Level: 99
Trigger: None

FRM-41425: Inadequate memory for Filter Summary column.
Cause: Attempt to Filter Summary column data failed because of inadequate system memory.

Action: Check system memory and insure it's adequate for running the application.
Level: 99
Trigger: None

FRM-41426: Unable to add column for Filter Summary creation to design-time record group %s.
Cause: Filter Summary failed due to a design-time record group being specified.

Action: Make sure record group for the Filter Summary is updateable at runtime.
Level: 99
Trigger: None

FRM-41427: Record group %s column mismatch for Filter Summary item.
Cause: The Filter Summary record group columns do not match the required types.

Action: Make sure the record group has the correct types matching the Filter Summary item and a number count column.
Level: 99
Trigger: None

FRM-41428: Inadequate number of columns in record group %s for the Filter Summary item.
Cause: The Filter Summary record group contains less than two columns.

Action: Make sure the record group has the correct number of coumns.
Level: 99
Trigger: None

FRM-41429: Warning - Possible truncation when adding Filter Summary results to record group %s.
Cause: Attempt to add a row from Filter Summary result to the record group may result in truncation.

Action: Make sure the record group columns are large enough to house the Filter Summary result values.
Level: 99
Trigger: None

FRM-41430: Filter Summary record group %s Count column is of wrong type.
Cause: Invalid column type for the Filter Summary's record group Count column.

Action: Record group Count column can only be of type NUMBER.
Level: 99
Trigger: None

FRM-41431: Filter Summary results failed to be added to record group %s.
Cause: Attempt to add a row from the Filter Summary result to the record group failed.

Action: Check that the record group can hold all the column values.
Level: 99
Trigger: None

FRM-41472: scratchpad memory allocation failed
Cause: An FSCRATCHPAD operation was unable to allocate adequate memory. An FSCRATCHPAD.ALLOCATION exception was raised.

Action: Modify application logic so as to limit the amount of data written to the scratchpad.
Level: 99
Trigger: ON-ERROR

FRM-41480: invalid 'uri_template' parameter passed to FSCRATCHPAD.EXPAND_TEMPLATE
Cause: The 'uri_template' parameter passed to FSCRATCHPAD.EXPAND_TEMPLATE must specify a syntactically correct URI template [RFC 6570, level 4]. A more specific FRM-41480 message is written to ODL.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41484: invalid 'parameters' parameter passed to FSCRATCHPAD.EXPAND_TEMPLATE
Cause: (a) The 'parameters' parameter passed to FSCRATCHPAD.EXPAND_TEMPLATE must specify null or a JSON object element; (b) the object's child values must be strings, arrays of strings, or objects whose child values are strings or arrays of strings; (c) where the URI template allows strings to contain %-encodings, such encodings must be valid. For violations of (b) and (c), a more specific FRM-41484, FRM-41486, or FRM-41487 message is written to ODL.

Action: Correct the application logic.
Level: 99

Trigger: ON-ERROR

FRM-41490: invalid charset specified for scratchpad file operation
Cause: The 'charset' parameter passed to FSCRATCHPAD.APPEND_FILE or FSCRATCHPAD.TO_FILE was invalid. It must be a valid Oracle Database character set name or else 'NLS_LANG', which indicates the Oracle Database character set name that was specified in the NLS_LANG environment variable.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41491: conflicting parameters: charset and base64 both specified for scratchpad file operation
Cause: FSCRATCHPAD.APPEND_FILE or FSCRATCHPAD.TO_FILE was passed both a non-null 'charset' parameter and a 'base64_encode' or 'base64_decode' parameter with a value of true. Charset conversion is appropriate only for character files and base64 encoding or decoding is appropriate only for binary files.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41492: scratchpad contents not a valid base64 encoding
Cause: 'base64_decode := true' was specified for FSCRATCHPAD.TO_FILE, but the scratchpad contents (up to the string specified by the 'stop_at' parameter, if specified) did not constitute a valid base64 encoding. A valid base64 encoding must contain only white space [blank, tab, carriage return, or line feed] or base64 alphabet characters ['A'-'Z', 'a'-'z', '0'-'9', '+', '/', or '=']. '=' characters are not required, but if present, they must occur only at the end, and they must ensure that the total number of base64 alphabet characters is a multiple of 4. Information that may help diagnose the issue was written to ODL.

Action: Correct application logic that populates the scratchpad.
Level: 99
Trigger: ON-ERROR

FRM-41493: Scratchpad contents could not be written.
Cause: FSCRATCHPAD.TO_FILE was unable to successfully write the scratchpad contents to the location specified by the 'file_name' parameter. The FRM-41493 message written to ODL specifies the full name of the file.

Action: Verify that the file name is spelled correctly and that the application has permission to write a file to the specified directory, and that the specified directory has adequate space.
Level: 99
Trigger: ON-ERROR

FRM-41494: File could not be read into scratchpad.
Cause: The 'file_name' parameter passed to FSCRATCHPAD.APPEND_FILE specified a file that could not be found or could not be read. The FRM-41494 or FRM-41495 message written to ODL specifies the full name of the file.

Action: Verify that the file name is spelled correctly and that the application has permission to read the file.
Level: 99
Trigger: ON-ERROR

FRM-41500: invalid FJSON.ELEMENT_T value
Cause: Application design error. A value that was passed to a function or procedure in the FJSON, FHTTP, or FSCRATCHPAD package, that was expecting an FJSON.ELEMENT_T value, either was not in fact an FJSON.ELEMENT_T value or else specified an element that did not currently exist in a JSON parse tree. An FJSON.BAD_ARGUMENT exception was raised.

Action: Ensure that the value is of type FJSON.ELEMENT_T and that FJSON.FREE_ALL was not called after the element was created.
Level: 99
Trigger: ON-ERROR

FRM-41501: Target element of specified JSON operation cannot be null.
Cause: Application design error. A null FJSON.ELEMENT_T value was passed as the first parameter to a function or procedure in the FJSON or FHTTP package which requires a target element. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41502: Element value for a JSON array cannot be null.
Cause: Application design error. A null FJSON.ELEMENT_T value was passed as the last parameter to the FJSON.ADD or FJSON.SET procedure. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41503: BOOLEAN value used to construct a JSON boolean element cannot be null.
Cause: Application design error. A null BOOLEAN value was passed as the last (or only) parameter to the FJSON.NEW_BOOLEAN function, or to the FJSON.ADD_BOOLEAN, FJSON.SET_BOOLEAN, or FJSON.PUT_BOOLEAN procedure. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41504: NUMBER value used to construct a JSON number element cannot be null.
Cause: Application design error. A null NUMBER value was passed as the last (or only) parameter to the FJSON.NEW_NUMBER function, or to the FJSON.ADD_NUMBER, FJSON.SET_NUMBER, or FJSON.PUT_NUMBER procedure. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41505: NUMBER value used to construct a JSON number element cannot be infinity.
Cause: Application design error. A NUMBER value of infinity was passed as the last (or only) parameter to the FJSON.NEW_NUMBER function, or to the FJSON.ADD_NUMBER, FJSON.SET_NUMBER, or FJSON.PUT_NUMBER procedure. RFC 8259 does not allow a JSON number that represents infinity. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41506: 'expected_num_elements' parameter to FJSON.NEW_OBJECT function cannot be null.
Cause: Application design error. A null expected_num_elements parameter was passed to the FJSON.NEW_OBJECT function. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41507: 'expected_num_elements' parameter to FJSON.NEW_OBJECT function cannot be negative.
Cause: Application design error. A negative expected_num_elements parameter was passed to the FJSON.NEW_OBJECT function. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41508: invalid sort option(s) passed to FJSON.SORT_NUMBERS or FJSON.SORT_STRINGS
Cause: Application design error. The sort option(s) passed to the FJSON.SORT_NUMBERS and FJSON.SORT_STRINGS functions must be ASCENDING, DESCENDING, NULLS_LAST, NULLS_FIRST, CASE_SENSITIVE, CASE_INSENSITIVE, BINARY_ORDER, or LINGUISTIC_ORDER. Duplicate or contradictory options are not allowed. The last 4 are not allowed for FJSON.SORT_NUMBERS. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41510: JSON array position cannot be null.
Cause: Application design error. A null array position was passed as the second parameter to the FJSON.FIND function, or to the FJSON.ADD, FJSON.SET, or FJSON.REMOVE procedure. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41511: invalid JSON array position
Cause: Application design error. The array position passed as the second parameter to the FJSON.FIND function, or to the FJSON.ADD, FJSON.SET, or FJSON.REMOVE procedure, was negative, zero, or too large. An FJSON.BAD_ARGUMENT exception was raised. For FJSON.SET and FJSON.REMOVE, the array position must be no greater than the number of elements in the array. For FJSON.ADD, the array position may be equal to one more than the number of elements in the array, but no greater. For FJSON.FIND, any positive value is acceptable. If the specified element does not exist, FJSON.FIND will quietly return null.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41512: Element value for a JSON array resides in the same array.
Cause: Application design error. An FJSON.ELEMENT_T value that was passed as the third parameter to the FJSON.ADD or FJSON.SET procedure was already an element of the JSON array that was passed as the first parameter. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic. If you want to move an element of an array from one position to another (thereby reordering the elements of the array), it is recommended that you:
1. Use FJSON.FIND to obtain the element to be moved.
2. Use FJSON.REMOVE_FROM_PARENT to remove it from the array. For best performance, this should if possible be done immediately after step 1 [or at any rate before doing any subsequent FJSON.FIND on the same array].
3. Use FJSON.ADD to insert it at the desired new position.
Level: 99
Trigger: ON-ERROR

FRM-41513: too many elements in JSON array or object
Cause: Application design error. An invocation of the FJSON.ADD or FJSON.PUT procedure would have caused the number of elements in a JSON array or object to equal 2G. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41514: invalid JSON object key
Cause: Application design or configuration error. The object key passed as the second parameter to the FJSON.PUT or FJSON.REMOVE_PROPERTY procedure was too large. An FJSON.BAD_ARGUMENT exception was raised. By default, the AL32UTF8 representation of an object key must not exceed 32767 bytes. Note that an object key whose NLS_LANG charset representation does not exceed 32767 bytes could exceed 32767 bytes when converted to AL32UTF8.

Action: If there is a requirement to create REST endpoint request bodies that contain object keys larger than 32767 bytes, the FORMS_JSON_KEY_LENGTH_BITS environment variable [which defaults to 15] can be set to a value N in the range 16..30. This will allow keys of any size less than 2 raised to the power of N. The tradeoff is that the total number of bytes required for all strings, numbers, and hash arrays for all elements in all JSON parse trees must not exceed 2 raised to the power of 64-N. Also bear in mind that keys larger than 32767 bytes [in the NLS_LANG charset] cannot be used in PL/SQL VARCHAR2 variables.
Level: 99
Trigger: ON-ERROR

FRM-41515: syntax error in application-created JSON text
Cause: A parse of the JSON text that was passed to the errpos-omitted flavor of FJSON.PARSE encountered a violation of RFC 7159. The parse was aborted, and the memory that had already been allocated for the parse tree was made available for subsequent JSON parse tree operations. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41516: loop in JSON parse tree
Cause: Application design error. An FJSON.CLONE, FJSON.GENERATE, or FHTTP.ISSUE_REQUEST detected a loop in a JSON parse tree [so it wasn't really a tree at

all]: the target element passed to FJSON.CLONE or FJSON.GENERATE, or the request_body element passed to FHTTP.ISSUE_REQUEST, was encountered while traversing its descendent elements. The FJSON.CLONE, FJSON.GENERATE, or FHTTP.ISSUE_REQUEST was aborted and the memory it consumed was released. An FJSON.BAD_ARGUMENT or FHTTP.BAD_ARGUMENT exception was raised. This condition was caused by an earlier call to FJSON.ADD, FJSON.SET, or FJSON.PUT that added a child element C to an array or object A, where C was equal to A or was an ancestor of A.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41520: application-created element of incorrect type specified as target of JSON operation
Cause: Application design error. An FJSON.ELEMENT_T value passed as the first parameter to a function or procedure in the FJSON package, or as the second parameter to the POPULATE_GROUP_FROM_JSON procedure, specified an element whose element type is not supported by the specified function or procedure, and the element was programmatically created by the application. An FJSON.BAD_ARGUMENT exception was raised (except for POPULATE_GROUP_FROM_JSON).

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41521: application-created element of unexpected type or value found in JSON array or object
Cause: Application design error. One of the following occurred:
1. An FJSON.FIND_x or FJSON.GET_x function produced an element of an unexpected type that was programmatically created by the application.
2. The JSON array passed as the first parameter to the FJSON.JOIN function contained an element that was not a JSON string, and the element was programmatically created by the application.
3. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained an initial element that was a JSON object, but also contained an element that was not a JSON object, and that element was programmatically created by the application.
4. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained an initial element that was a JSON array or scalar element, but also contained an element that was a JSON object, and that element was programmatically created by the application.
5. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained a scalar element, or a JSON array or JSON object that in turn contained a scalar element, and the type of the scalar element did not match the corresponding record group column. [DATE columns are a special case: if a null epoch_unit was passed to POPULATE_GROUP_FROM_JSON, the corresponding scalar element must be a JSON string (representing an ISO 8601 date/time string); otherwise, it must be a JSON number (representing "epoch": seconds since 1970-01-01).] The scalar element was programmatically created by the application.
6. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained a scalar element, or a JSON array or JSON object that in turn contained a scalar element, and the corresponding record group column was of type DATE. Either a null epoch_unit was passed to POPULATE_GROUP_FROM_JSON and the scalar element was a JSON string which was not in valid ISO 8601 format, or a non-null epoch_unit was specified and the scalar element was a

JSON number which was not a valid epoch. The scalar element was programmatically created by the application.
7. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained a JSON string element, or a JSON array or JSON object that in turn contained a JSON string element, and the value of the string element exceeded 32767 bytes. The string element was programmatically created by the application.
8. A JSON object was passed as the second parameter to the POPULATE_GROUP_FROM_JSON function. It contained an element whose object key exceeded 32767 bytes or, in the case where the first record group column was of type NUMBER, could not be converted to a number. The element was programmatically created by the application.
In cases 1 and 2, an FJSON.BAD_ARGUMENT exception was raised.
In cases 2-8, additional information that may help diagnose the issue was written to ODL (starting with an FRM-41533, FRM-41534, FRM-41535, or FRM-41536 message for cases 2-5, 6, 7, and 8, respectively).

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41522: expected element not found in application-created JSON array or object
Cause: Application design error. An FJSON.FIND_ARRAY, FJSON.FIND_OBJECT, FJSON.GET_ARRAY, or FJSON.GET_OBJECT function produced no element, and the FJSON.ELEMENT_T value passed as the first parameter to the function was programmatically created by the application. An FJSON.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41530: retrieved element of incorrect type specified as target of JSON operation
Cause: Application design error. An FJSON.ELEMENT_T value passed as the first parameter to a function or procedure in the FJSON package, or as the second parameter to the POPULATE_GROUP_FROM_JSON procedure, specified an element whose element type is not supported by the specified function or procedure, and the element was retrieved from a REST endpoint. An FJSON.CONTENT exception was raised (except for POPULATE_GROUP_FROM_JSON). Additional information that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41530 message).

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41531: retrieved element of unexpected type or value found in JSON array or object
Cause: Application design error. One of the following occurred:
1. An FJSON.FIND_x or FJSON.GET_x function produced an element of an unexpected type that was retrieved from a REST endpoint.
2. The JSON array passed as the first parameter to the FJSON.JOIN function contained an element that was not a JSON string, and the element was retrieved from a REST endpoint.
3. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained an initial element that was a JSON object, but also contained an element that was not a JSON object, and that element was retrieved from a REST endpoint.

4. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained an initial element that was a JSON array or scalar element, but also contained an element that was a JSON object, and that element was retrieved from a REST endpoint.
5. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained a scalar element, or a JSON array or JSON object that in turn contained a scalar element, and the type of the scalar element did not match the corresponding record group column. [DATE columns are a special case: if a null epoch_unit was passed to POPULATE_GROUP_FROM_JSON, the corresponding scalar element must be a JSON string (representing an ISO 8601 date/time string); otherwise, it must be a JSON number (representing "epoch": seconds since 1970-01-01).] The scalar element was retrieved from a REST endpoint.
6. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained a scalar element, or a JSON array or JSON object that in turn contained a scalar element, and the corresponding record group column was of type DATE. Either a null epoch_unit was passed to POPULATE_GROUP_FROM_JSON and the scalar element was a JSON string which was not in valid ISO 8601 format, or a non-null epoch_unit was specified and the scalar element was a JSON number which was not a valid epoch. The scalar element was retrieved from a REST endpoint.
7. The JSON array or JSON object passed as the second parameter to the POPULATE_GROUP_FROM_JSON function contained a JSON string element, or a JSON array or JSON object that in turn contained a JSON string element, and the value of the string element exceeded 32767 bytes. The string element was retrieved from a REST endpoint.
8. A JSON object was passed as the second parameter to the POPULATE_GROUP_FROM_JSON function. It contained an element whose object key exceeded 32767 bytes or, in the case where the first record group column was of type NUMBER, could not be converted to a number. The element was retrieved from a REST endpoint.
In cases 1 and 2, an FJSON.CONTENT exception was raised.
In all cases, additional information that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41531 message for case 1, and an FRM-41533, FRM-41534, FRM-41535, or FRM-41536 message for cases 2-5, 6, 7, and 8, respectively).

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41532: expected element not found in retrieved JSON array or object
Cause: Application design error. An FJSON.FIND_ARRAY, FJSON.FIND_OBJECT, FJSON.GET_ARRAY, or FJSON.GET_OBJECT function produced no element, and the FJSON.ELEMENT_T value passed as the first parameter to the function was retrieved from a REST endpoint. An FJSON.CONTENT exception was raised. Additional information that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41532 message).

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41540: unexpected content detected in JSON document
Cause: The application called the FJSON.UNEXPECTED_CONTENT procedure, presumably because its logic detected unexpected content of some sort. The message passed to that procedure was incorporated into an FRM-41540 or FRM-41541 message written to ODL. An FJSON.CONTENT exception was raised.

Action: Try to determine what is producing the unexpected JSON content, e.g. by contacting the owner of the REST service from which the JSON document was retrieved. Correct application logic if necessary.
Level: 99
Trigger: ON-ERROR

FRM-41549: unexpected scalar conversion exception encountered while processing JSON document
Cause: The application called the FJSON.ANALYZE_EXCEPTION procedure within a PL/SQL exception block, and it tentatively determined that the exception could plausibly have been produced by a PL/SQL assignment from a scalar derived from a REST endpoint response, to a PL/SQL variable (or an assignment from that variable to another, using a conversion function such as TO_DATE). An FJSON.CONTENT exception was raised. Additional information that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41549 message).

Action: Try to determine what is causing the scalar conversion exception. It could be due to an application design error unrelated to the REST service; if so, correct the application logic. Otherwise, try to determine what is producing the unexpected JSON content, e.g. by contacting the owner of the REST service from which the JSON document was retrieved. Correct application logic if necessary.
Level: 99
Trigger: ON-ERROR

FRM-41560: syntax error in retrieved JSON text
Cause: A parse of the JSON text response from a REST endpoint encountered a violation of RFC 7159. The parse was aborted, and the memory that had already been allocated for the parse tree was made available for subsequent JSON parse tree operations. An FJSON.PARSE_FAILURE exception was raised. Additional information that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41560 message).

Action: Try to determine why the erroneous JSON text is being returned, e.g. by contacting the owner of the REST service.
Level: 99
Trigger: ON-ERROR

FRM-41561: oversize object key in retrieved JSON text
Cause: A parse of the JSON text response from a REST endpoint encountered an oversize object key in an object member definition. By default, Oracle Forms does not support object keys whose AL32UTF8 representation exceeds 32767 bytes. [Larger keys cannot be used in PL/SQL VARCHAR2 variables.] The parse was aborted, and the memory that had already been allocated for the parse tree was made available for subsequent JSON parse tree operations. An FJSON.PARSE_FAILURE exception was raised. Additional information that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41561 message).

Action:
Level: 99
Trigger: ON-ERROR

FRM-41562: oversize array or object in retrieved JSON text
Cause: A parse of the JSON text response from a REST endpoint encountered an array or object which contained 2G elements or more. The parse was aborted, and the memory that had already been allocated for the parse tree was made available for subsequent JSON parse tree operations. An FJSON.PARSE_FAILURE exception was raised. Additional information

that may help diagnose the issue was written to ODL (starting with a more detailed FRM-41562 message).

Action: Try to determine why the oversize array or object is being returned, e.g. by contacting the owner of the REST service.
Level: 99
Trigger: ON-ERROR

FRM-41570: JSON parse memory limit exceeded
Cause: A parse of the JSON text response from a REST endpoint required more memory than the JSON parse memory limit (4Gb or the limit specified by the FORMS_JSON_PARSE_MEMORY_LIMIT environment variable). The parse was aborted, and the memory that had already been allocated for the parse tree was made available for subsequent JSON parse tree operations. An FJSON.PARSE_ALLOCATION exception was raised.

Action: Modify the selection criteria specified in the request to the REST endpoint, so as to limit the size of the response, or increase the value of the FORMS_JSON_PARSE_MEMORY_LIMIT environment variable.
Level: 99
Trigger: ON-ERROR

FRM-41571: JSON new memory limit exceeded
Cause: An attempt to programmatically create a JSON element (using e.g. FJSON.CLONE or an FJSON.NEW_x function) would have caused the memory consumed by all programmatically created JSON elements to exceed the new memory limit (4Gb or the limit specified by the FORMS_JSON_NEW_MEMORY_LIMIT environment variable). An FJSON.GLOBAL_ALLOCATION exception was raised.

Action: Revise application logic so as to programmatically create fewer JSON elements, or increase the value of the FORMS_JSON_NEW_MEMORY_LIMIT environment variable.
Level: 99
Trigger: ON-ERROR

FRM-41572: JSON memory allocation failed
Cause: An operation to create, update, or generate a JSON parse tree was unable to allocate adequate memory. An FJSON.GLOBAL_ALLOCATION exception was raised.

Action: Modify the selection criteria specified in the request to the REST endpoint, so as to limit the size of the response, or revise application logic so as to programmatically create fewer JSON elements.
Level: 99
Trigger: ON-ERROR

FRM-41573: JSON parse trees require excessive memory for element values.
Cause: This error arises only if both the following occur:
1. The FORMS_JSON_KEY_LENGTH_BITS environment variable [which defaults to 15, limiting object keys to 32767 bytes] was set to a value N in the range 16..30 [the maximum allowed].
2. The total number of bytes required for all strings, numbers, and hash arrays for all elements in all JSON parse trees exceeded 2 raised to the power of 64-N.
If this error occurred, then an FJSON.GLOBAL_ALLOCATION exception was raised.

Action: Modify the selection criteria specified in the request to the REST endpoint, so as to limit the size of the response, or revise application logic so as to programmatically create fewer JSON elements.

Level: 99
Trigger: ON-ERROR

FRM-41574: too many elements in JSON parse trees
Cause: An operation to create or update a JSON parse tree would have caused the total number of elements in all JSON parse trees to equal or exceed 4G. An FJSON.GLOBAL_ALLOCATION exception was raised.

Action: Modify the selection criteria specified in the request to the REST endpoint, so as to limit the size of the response, or revise application logic so as to programmatically create fewer JSON elements.
Level: 99
Trigger: ON-ERROR

FRM-41580: invalid $ref string
Cause: The element passed to FJSON.RESOLVE$REF specified a $ref string which was not a syntactically valid URI. ODL will contain a more detailed message.

Action: Contact the provider of the service which produced the document containing the invalid JSON reference.
Level: 99
Trigger: ON-ERROR

FRM-41581: $ref recursion excessive or produced loop
Cause: The element passed to FJSON.RESOLVE$REF specified a $ref string which resolved to another $ref element, which in turn resolved to yet another $ref element, and so forth. ODL will contain a more detailed message.

Action: Contact the provider of the service which produced the document containing the invalid JSON reference.
Level: 99
Trigger: ON-ERROR

FRM-41584: invalid JSON pointer
Cause: Either (1) an invalid JSON pointer was passed to FJSON.EVALUATE_POINTER, or (2) the element passed to an invocation of FJSON.RESOLVE$REF specified a $ref string whose fragment [the portion after the '#'] was not a valid JSON pointer. The JSON pointer did not begin with '/' or contained an invalid escape sequence ['~' not followed by '0' or '1']. ODL will contain a more detailed message.

Action: In case (1), correct the application logic. In case (2), contact the provider of the service which produced the document containing the invalid JSON reference.
Level: 99
Trigger: ON-ERROR

FRM-41590: invalid charset specified for JSON file operation
Cause: The 'charset' parameter passed to FJSON.NEW_STRING_FROM_FILE or FJSON.STRING_VALUE_TO_FILE was invalid. It must be a valid Oracle Database character set name or else 'NLS_LANG', which indicates the Oracle Database character set name that was specified in the NLS_LANG environment variable.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41591: conflicting parameters: charset and base64 both specified for JSON file operation
Cause: FJSON.NEW_STRING_FROM_FILE or FJSON.STRING_VALUE_TO_FILE was passed both a non-null 'charset' parameter and a 'base64_encode' or 'base64_decode' parameter with a value of true. Charset conversion is appropriate only for character files and base64 encoding or decoding is appropriate only for binary files.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41592: JSON string not a valid base64 encoding
Cause: 'base64_decode := true' was specified for FJSON.STRING_VALUE_TO_FILE, but the value of the JSON string specified by the 'target' parameter was not a valid base64 encoding. A valid base64 encoding must contain only white space [blank, tab, carriage return, or line feed] or base64 alphabet characters ['A'-'Z', 'a'-'z', '0'-'9', '+', '/', or '=']. '=' characters are not required, but if present, they must occur only at the end, and they must ensure that the total number of base64 alphabet characters is a multiple of 4. Information that may help diagnose the issue was written to ODL.

Action: If the string was retrieved from a REST endpoint, contact the owner of the REST service. If the string was created by the application, correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41593: JSON string could not be written.
Cause: FJSON.STRING_VALUE_TO_FILE was unable to successfully write the string value to the location specified by the 'file_name' parameter. The FRM-41493 message written to ODL specifies the full name of the file.

Action: Verify that the file name is spelled correctly and that the application has permission to write a file to the specified directory, and that the specified directory has adequate space.
Level: 99
Trigger: ON-ERROR

FRM-41594: File could not be read into JSON string.
Cause: The 'file_name' parameter passed to FJSON.NEW_STRING_FROM_FILE specified a file that could not be found or could not be read. The FRM-41494 or FRM-41495 message written to ODL specifies the full name of the file.

Action: Verify that the file name is spelled correctly and that the application has permission to read the file.
Level: 99
Trigger: ON-ERROR

FRM-41600: invalid '%s' parameter passed to FHTTP.ISSUE_REQUEST
Cause: Application design error. The specified parameter specified a value that violated one of the following constraints:
1. The 'method' parameter value must (a) be GET, HEAD, POST, PUT, DELETE, OPTIONS, TRACE, or PATCH and (b) be acceptable for configured security settings and the specific URL that was generated. For violations of (b), a more specific FRM-41610 message is written to ODL.
2. The 'server_url_or_id' parameter value must, if specified, either be a well-formed http, https, ws, or wss URL or else contain only alphanumeric characters and '_', '.', '-'. In the second case, the '='-separated concatentaion of the 'config' parameter specified in the frmservlet URL and this 'server_url_or_id' parameter are used as a key to a generic credential. The generic

credential must specify a well-formed http, https, ws, or wss URL (which will be prepended to the URL generated from the 'uri_template' and 'url_parameters' parameters). A more specific FRM-41620, FRM-41621, FRM-41622, or FRM-41623 message is written to ODL.
3. The 'uri_template' parameter value must specify a syntactically correct URI template [RFC 6570, level 4], and it must, in conjunction with the URL derived from the 'server_url_or_id' parameter and the 'url_parameters' parameter, generate a well-formed http, https, ws, or wss URL. A more specific FRM-41630 or FRM-41633 message is written to ODL.
4. (a) The 'url_parameters' parameter value must specify null or a JSON object element; (b) the object's child values must be strings, arrays of strings, or objects whose child values are strings or arrays of strings; (c) where the URI template allows strings to contain %-encodings, such encodings must be valid. For violations of (b) and (c), a more specific FRM-41640, FRM-41642, or FRM-41643 message is written to ODL.
5. (a) The 'request_headers' parameter value must specify null or a JSON object element; (b) the object's child values must be strings; (c) the child keys and values must be valid HTTP header names and values [RFC 7230]. For violations of (b) and (c), a more specific FRM-41650, FRM-41652, or FRM-41653 message is written to ODL.
6. The 'accept' and 'parse' parameter value must be null or a comma-separated list of (a) 3-digit HTTP response status codes in the range 100-599 and/or (b) ranges of such status codes, specified as hyphen-separated low and high values.
7. The 'request_body' parameter must not be specified if the 'method' parameter is GET, HEAD, DELETE, OPTIONS, or TRACE.
8. The 'authorization_info' parameter value must consist of tokens separated by unit separator characters [u001f]. The tokens must be syntactically valid and must specify a supported authorization mode. An authorization that requires a logon dialog must be last. A more specific FRM-41712, FRM-41713, FRM-41714, or FRM-41715 message is written to ODL.
In all cases, an FHTTP.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41660: HTTP request encountered error communicating with an external service
Cause: The request issued by FHTTP.ISSUE_REQUEST encountered an I/O error while communicating with the external service. An FHTTP.BAD_RESPONSE exception was raised. ODL will contain a more detailed message.

Action: If the problem persists, contact the owner of the external service.
Level: 99
Trigger: ON-ERROR

FRM-41661: HTTP request received unexpected response status code %d.
Cause: The request issued by FHTTP.ISSUE_REQUEST received a response status code that was not specified in the 'accept' parameter and was not one of the status codes that are always accepted, namely 200, and [for all methods except TRACE] 204, and [for PUT and POST] 201 and 205. An FHTTP.BAD_RESPONSE exception was raised. ODL will contain a more detailed message.

Action: If the status code is a legitimate response from the accessed external service, include it in the 'accept' parameter, and add appropriate logic to handle it. Otherwise, inspect the diagnostic information in the response identified by message FRM-41691 in ODL.
Level: 99
Trigger: ON-ERROR

FRM-41670: HTTP URL and request headers require excessive memory.
Cause: The URL generated from the 'uri_template' and 'url_parameters' parameters passed to FHTTP.ISSUE_REQUEST, together with the headers generated from the 'request_headers'

parameter, required nearly 2Gb of memory. An FHTTP.BAD_ARGUMENT exception was raised.

Action: Verify that the 'uri_template', 'url_parameters', and 'request_headers' parameters are reasonable.
Level: 99
Trigger: ON-ERROR

FRM-41671: memory allocation failure encountered while processing an HTTP request
Cause: FHTTP.ISSUE_REQUEST was unable to allocate adequate memory in the runtime process. An FHTTP.ALLOCATION exception was raised. ODL will contain a more detailed message.

Action: Specify fewer or smaller request headers or a smaller request body, or modify the selection criteria specified in the request to the REST endpoint, so as to limit the size of the response.
Level: 99
Trigger: ON-ERROR

FRM-41672: error in FHTTP.ISSUE_REQUEST: WebLogic was unable to create a thread to process a request to an external service.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-41673: OutOfMemoryError encountered while processing an HTTP request
Cause: FHTTP.ISSUE_REQUEST was unable to allocate adequate memory in the JVM that's issuing the request. An FHTTP.ALLOCATION exception was raised. ODL will contain a more detailed message.

Action: Specify fewer or smaller request headers or a smaller request body, or modify the selection criteria specified in the request to the REST endpoint, so as to limit the size of the response.
Level: 99
Trigger: ON-ERROR

FRM-41674: error in Dynamic Monitoring Service (DMS): WebLogic was unable to create a thread to remove unused DMS objects.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-41675: error in Forms prestart processing: WebLogic was unable to create a thread to fill the pool of prestarted Forms runtime processes.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert

Level: 99
Trigger: None

FRM-41698: invalid '%s' parameter passed to FHTTP.SET_SERVER_OPTION
Cause: Application design error. The specified parameter specified a value that violated one of the following constraints:
1. The 'server' parameter value must be a well-formed URL wthout a query or fragment whose protocol is http, https, ws, or wss, or an initial portion of such a URL. It will typically specify a protocol and a domain.
2. The 'server_option' parameter value must be one that is supported by FHTTP.SET_SERVER_OPTION.
In all cases, an FHTTP.BAD_ARGUMENT exception was raised.

Action: Correct the application logic.
Level: 99
Trigger: ON-ERROR

FRM-41699: FHTTP PL/SQL package supported only in Forms Servlet mode
Cause: A function or procedure in the FHTTP PL/SQL package was invoked in Forms Listener mode, which is no longer supported. A Form_Fatal_Error exception was raised.

Action: Use Forms Servlet mode.
Level: 99
Trigger: ON-ERROR

FRM-41700: HTTP request authorization failed
Cause: A request issued by FHTTP.ISSUE_REQUEST required authorization and the 'authorization_info' parameter value was syntactically valid, but authorization was unsuccessful. ODL will contain more detailed message(s).

Action: Examine the message(s) in ODL.
Level: 99
Trigger: ON-ERROR

FRM-41701: OAuth2 authorization timed out
Cause: A request issued by FHTTP.ISSUE_REQUEST required an OAuth2 authorization code grant. This in turn required the OAuth2 authorization server to interact with the end user. The end user did not either approve or deny the requsted privilege within the interval specified by the ssoSaaBrowserPageTimeout configuration parameter, or the OAuth2 authorization server encountered an error.

Action: Increase the value of the ssoSaaBrowserPageTimeout configuration parameter, if desired.
Level: 99
Trigger: ON-ERROR

FRM-41702: username and password for REST service not both specified
Cause: A request issued by FHTTP.ISSUE_REQUEST required basic authentication or an OAuth2 password credentials grant. A logon dialog was presented, but an empty username and/or an empty password were entered.

Action: Enter a non-empty username and a non-empty password.
Level: 99
Trigger: ON-ERROR

FRM-41703: colon specified in username for basic authentication
Cause: A request issued by FHTTP.ISSUE_REQUEST required basic authentication. A logon dialog was presented, but a username containing a colon was entered. This is not allowed for basic authentication.

Action: Enter a corrected username.
Level: 99
Trigger: ON-ERROR

FRM-41704: specified username and password specified rejected by REST service
Cause: A request issued by FHTTP.ISSUE_REQUEST required basic authentication or an OAuth2 password credentials grant. A logon dialog was presented, and a username and a password were entered, but the REST service rejected the HTTP request as unauthorized.

Action: Determine the correct username/password and enter them.
Level: 99
Trigger: ON-ERROR

FRM-41705: REST logon dialog terminated after 3 unsuccessful attempts
Cause:

Action: Determine the correct username/password and restart the Forms application.
Level: 99
Trigger: ON-ERROR

FRM-41800: List of Values not available for this field.
Cause: You pressed [List], but the form does not provide a list of values for this field.

Action: No action is necessary.
Level: 10
Trigger: ON-MESSAGE

FRM-41801: Last value retrieved.
Cause: You pressed [List] and then pressed [Next Item] after the last value in the list was displayed.

Action: Enter an item value or press [List] again to display the list of possible values.
Level: 10
Trigger: ON-MESSAGE

FRM-41802: Duplicate record function allowed on new records only.
Cause: You pressed [Duplicate Record], but the current record is the one that has been fetched from the database.

Action: No action is necessary. You can use [Duplicate Record] only when creating a new record.
Level: 10
Trigger: ON-ERROR

FRM-41803: No previous record to copy value from.
Cause: You pressed [Duplicate Item] or [Duplicate Record], but the current record is the first record in the block.

Action: No action is necessary. [Duplicate Item] and [Duplicate Record] are meaningless in this context.
Level: 10
Trigger: ON-ERROR

FRM-41804: Variable was not entered: %.30s.
Cause: Your response to the Query Where alert contained a placeholder not used in any of the query items.

Action: Correct the placeholder in your response, or define it in one of the query items. Then re-execute the query.
Level: 99
Trigger: ON-ERROR

FRM-41805: Ambiguous item name: %s.
Cause: Application design error. A call to a Built-in specified an ambiguous item name. (No block was specified, and more than one block contains an item of the specified name).

Action: Specify a block name (block.item).
Level: 99
Trigger: ON-ERROR

FRM-41806: Too many variables used.
Cause: You used more than 25 substitution variables in your query.

Action: Reduce the number of substitution variables and re-query.
Level: 99
Trigger: ON-ERROR

FRM-41809: Error initializing Menu.
Cause: You tried to use the menu component from within Oracle Forms, and an internal error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-41810: Error creating menu.
Cause: You tried to use the menu component from within Oracle Forms, and an internal error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-41811: Error removing menu.
Cause: You tried to use Menus from within Oracle Forms, and an internal Menu error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-41812: Error resetting Menu.
Cause: You tried to use the menu component from within Oracle Forms, and an internal error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-41813: Form exited by debug mode.
Cause: You selected the Exit Oracle Forms Runtime option on the Break Processing menu.

Action: No action is necessary.
Level: 99
Trigger: None

FRM-41814: Invalid page position.
Cause: Application design error. A trigger tried to move or resize a view to a page that would cause all or part of the view to display off of the screen.

Action: Correct the statement.
Level: 99
Trigger: ON-ERROR

FRM-41815: No such property for Get_LOV_Property.
Cause: You attempted to get a nonexistent LOV property.

Action: Verify the valid LOV properties and try again.
Level: 20
Trigger: ON-ERROR

FRM-41816: Attempt to create existing timer: %s.
Cause: Attempted to create a timer that already exists.

Action: Delete or alter the existing timer before re-creating a new one.
Level: 99
Trigger: ON-ERROR

FRM-41817: No such timer: %s.
Cause: You attempted to alter or delete a non-existent timer.

Action: Check the Built-in for proper arguments.
Level: 99
Trigger: ON-ERROR

FRM-41818: Toolkit failed to create timer %s :may be out of memory.
Cause: An internal error occurred while attempting to create a timer, possibly as a result of memory constraints.

Action: Check and adjust memory quotas as necessary.
Level: 99
Trigger: ON-ERROR

FRM-41819: Timers are not supported on this platform.
Cause: Illegal attempt to create a timer on a platform where timers are not supported.

Action: None. A timer option is unavailable on your platform.
Level: 99
Trigger: ON-ERROR

FRM-41820: Toolkit failed to delete timer: %s.
Cause: Internal error caused by timer failure.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-41821: Timer name too long: %s...
Cause: You attempted to create a timer with a name longer than 30 bytes.

Action: Retry with a shorter name.
Level: 99
Trigger: ON-ERROR

FRM-41822: Timer name may not be null string.
Cause: You attempted to create a timer with a null name.

Action: Retry with a non-null name.
Level: 99
Trigger: ON-ERROR

FRM-41823: Illegal timer interval for timer %s.
Cause: You attempted to create a timer with an interval less than 1 millisecond.

Action: Recreate your timer with an interval of at least 1 millisecond.
Level: 99
Trigger: ON-ERROR

FRM-41824: Date/time operation failed for %s.
Cause: An internal error occurred while attempting to resolve a date/time initial value for an item.

Action: If the problem persists, contact Oracle Support Services.
Level: 10
Trigger: ON-ERROR

FRM-41825: No such property for Set_LOV_Property.
Cause: You attempted to set a nonexistent LOV property.

Action: Verify the valid LOV properties and try again.
Level: 20
Trigger: ON-ERROR

FRM-41826: Cannot replace group; columns don't match LOV.
Cause: Cannot replace the list of values' current record group with a record group that is incompatible with the LOV column structure.

Action: Do not attempt to assign this record group to this LOV.
Level: 20
Trigger: ON-ERROR

FRM-41827: Group does not exist.
Cause: The group name or ID specified is invalid.

Action: Check the name or ID entered and try again.
Level: 20
Trigger: ON-ERROR

FRM-41828: LOV does not exist.
Cause: LOV name or ID specified is invalid.

Action: Check the name or ID entered and try again.
Level: 20
Trigger: ON-ERROR

FRM-41829: Record not created.
Cause: Application design error. The record failed to get its initial value.

Action: Contact the application designer.
Level: 20
Trigger: ON-ERROR

FRM-41830: List of Values contains no entries.
Cause: The record group underlying the LOV contains no records.

Action: Check to be sure that any criteria used to reduce a long list LOV did not eliminate all matches.
Level: 5
Trigger: ON-ERROR

FRM-41832: Error: program unit %s in library %s is uncompiled.
Cause: You called an uncompiled program unit from a library.

Action: Follow the PL/SQL program error.
Level: 99
Trigger: ON-ERROR

FRM-41833: Warning! Program unit %s in library %s is uncompiled.
Cause: You called an uncompiled program unit in a library when debug mode was specified.

Action: This is just a warning. Forms Runtime will attempt to compile and run the program unit.
Level: 99
Trigger: None

FRM-41835: Canvas %s is not a tab canvas.
Cause: You tried to perform a tab canvas specific operation on a canvas which is not a tab canvas.

Action: Make sure the canvas specified is a tab canvas.
Level: 99
Trigger: ON-ERROR

FRM-41836: No tab page %s in canvas %s.
Cause: You tried to perform an operation on a tab page which does not exist in the specified canvas.

Action: Make sure you specify a tab page which exists in the specified tab canvas.
Level: 99
Trigger: ON-ERROR

FRM-41837: Error raising tab page %s.
Cause: The specified tab page could not be brought to the top (made the current page of the tab canvas).

Action: Make sure the specified page is enabled, and not hidden.
Level: 99
Trigger: ON-ERROR

FRM-41838: Unable to open temporary record buffer file
Cause: Unable to open file used as temporary record buffer.

Action: Verify that the file system or directory in which the file resides exists and that you have permissions to read and write to it.
Level: 99
Trigger: ON-ERROR

FRM-41839: Disk I/O error on temporary record buffer file
Cause: An I/O error occurred on attempting to read or write a record to the temporary record buffer file.

Action: Verify that the file system or directory in which the file resides exists, and that you have permissions to read and write to it, and that it contains sufficient space. If the file system is full, and you are executing a large query, retry it with additional query criteria (in order to reduce the size of the result set). Alternatively, commit changes and, if necessary, clear one or more blocks to free up disk space.
Level: 99
Trigger: ON-ERROR

FRM-41840: Insufficient main memory for record buffers
Cause: Unable to allocate memory for a record being created in a block or record group.

Action: If you are executing a large query, retry it with additional query criteria (in order to reduce the size of the result set). Alternatively, commit changes and, if necessary, clear one or more blocks to free up memory, or try restarting the application when fewer programs are running concurrently, or on a machine with more memory.
Level: 99
Trigger: ON-ERROR

FRM-41841: Use the debugger-enabled executable if specifying DEBUG=YES.
Cause: You tried to use the debugger from an executable which doesn't include it.

Action: Run the other executable (name will vary with operating system), which includes the debugger.
Level: 99
Trigger: None

FRM-41843: Invalid time zone region %s for ADJUST_TZ.
Cause: A call to the ADJUST_TZ procedure specified an invalid 'from' or 'to' time zone region name.

Action: Specify a valid name. If the name is valid, you may need to ask your system administrator to install a larger time zone file.
Level: 25
Trigger: ON-ERROR

FRM-41844: ADJUST_TZ could not convert date.
Cause: A call to the ADJUST_TZ procedure specified valid 'from' and 'to' time zone region names, but nevertheless failed. This probably indicates that the date was too close to the boundary dates of Jan 1, 4712 BC or Dec 31, 9999 AD.

Action: Specify a valid date.
Level: 25
Trigger: ON-ERROR

FRM-41845: Javascript events have been disabled.
Cause: Either the environment variable FORMS_ALLOW_JAVASCRIPT_EVENTS or the applet parameter enableJavaScriptEvent has been set to FALSE.

Action: Set client applet's parameter enableJavaScriptEvent and the server's environment variable FORMS_ALLOW_JAVASCRIPT_EVENTS to true. The default value of both these variables is true.
Level: 15
Trigger: ON-ERROR

FRM-41847: Temporary record buffer file size limit exceeded
Cause: An insert record, update record, or fetch required space in the temporary record buffer file, but it had reached its size limit (4GB or the value specified by the FORMS_RECMGR_MAX_TMPFILE_SIZE environment variable).

Action: If you are executing a large query, retry it with additional query criteria (in order to reduce the size of the result set). Alternatively, commit changes and, if necessary, clear one or more blocks to free up disk space.
Level: 99
Trigger: ON-ERROR

FRM-41848: JavaScript execution is disabled during webstart session.
Cause: JavaScript integration is not supported when running webstart.

Action: Use a browser if you want to run JavaScript integration with Oracle Forms.
Level: 15
Trigger: ON-ERROR

FRM-41849: JavaScript execution is disabled for Stand-alone App.
Cause: JavaScript integration is not supported with Stand-alone App.

Action: Use a browser if you want to run JavaScript integration with Oracle Forms.
Level: 15
Trigger: ON-ERROR

FRM-41850: Archived record memory threshold exceeded
Cause: An insert record, update record, or fetch required virtual memory, but the virtual memory consumed by all archived records had exceeded the archive threshold (4GB or the value specified by the FORMS_RECMGR_ARCHIVE_THRESHOLD environment variable).

Action: If you are executing a large query, retry it with additional query criteria (in order to reduce the size of the result set). Alternatively, commit changes and, if necessary, clear one or more blocks to free up memory.
Level: 99
Trigger: ON-ERROR

FRM-41851: RECORDS_DISPLAYED property must be between 1 and MAX_RECORDS_DISPLAYED property
Cause: An attempt was made to set the number of records displayed to an invalid value.

Action: Make sure that the requested number of records displayed is a positive integer less than or equal to the maximum number of records displayed for the block.
Level: 20
Trigger: ON-ERROR

FRM-41852: RECORDS_DISPLAYED property can't be updated for autosize blocks
Cause: An attempt was made to set the number of records displayed in a block which is set to autosize.

Action: Either remove the call to set the number of records displayed, or change the block to not be autosize.

Level: 20
Trigger: ON-ERROR

FRM-41853: Popup menu %s not found
Cause: The specified popup menu does not exist in the module.

Action: Either remove the call to set the popup menu, or change the menu name to specify one that exists.
Level: 99
Trigger: ON-ERROR

FRM-41854 Cannot assign new menu when initial menu not set
Cause: A canvas or item that does not have a popup menu specified at design-time cannot have one added at runtime.

Action: Either remove the call to set the popup menu, or ensure that a popup menu is associated with the object at design-time.
Level: 99
Trigger: ON-ERROR

FRM-41855 Canvas %s does not have a splitter associated with it
Cause: The splitter position cannot be set because the specified canvas does not have a spitter associated with it.

Action: Either remove the call to set the splitter position, or ensure that the canvas has a splitter associated with it.
Level: 99
Trigger: ON-ERROR

FRM-41856 Invalid position specified for splitter
Cause: The splitter position is inappropriate for the specified canvas.

Action: Ensure that the splitter position is non-negative, and is less than or equal to the height of the view (for canvases with a vertical splitter orientation) or the width of the view (for canvases with a horizontal splitter orientation).
Level: 99
Trigger: ON-ERROR

FRM-41857 Minimum UI Value must be less than Maximum UI Value
Cause: The minimum and maximum UI values conflict.

Action: Ensure that the minimum UI value property is smaller than the maximium UI value property.
Level: 99
Trigger: ON-ERROR

FRM-41858 Minimum UI Value must be divisible by UI Increment Value
Cause: The minimum UI value must be evenly divisible by the UI Increment Value.

Action: Ensure that the minimum UI value property is divisible by the UI Increment value property.
Level: 99
Trigger: ON-ERROR

FRM-41859 Maximum UI Value must be divisible by UI Increment Value
Cause: The maximum UI value must be evenly divisible by the UI Increment Value.

Action: Ensure that the maximum UI value property is divisible by the UI Increment value property.
Level: 99
Trigger: ON-ERROR

FRM-41900: Run aborted by fatal error.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-41901: Error: %d cursors were not closed.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-41902: Total cursors used %d.
Cause: This message appears when you run a form with the Statistics preference set to True.

Action: No action is necessary.
Level: 20
Trigger: ON-MESSAGE

FRM-41903: Run aborted by end of input file.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: None

FRM-42017: Module name must be specified.
Cause: You did not specify a module name.

Action: Specify a module name.
Level: 99
Trigger: None

FRM-42100: No errors encountered recently.
Cause: You pressed [Display Error], but no error has occurred recently.

Action: No action is necessary.
Level: 5
Trigger: ON-MESSAGE

FRM-42400: Performing event trigger %s.
Cause: This message is displayed during a trigger when debug mode is specified.

Action: No action is necessary.
Level: 99
Trigger: ON-MESSAGE

FRM-42401: Performing program trigger %s.
Cause: This message is displayed during a trigger when debug mode is specified.

Action: No action is necessary.
Level: 99
Trigger: ON-MESSAGE

FRM-42423: Cannot execute trigger %s: no compiled state
Cause: This message is displayed when running a form in debug mode, if the compiled state of a trigger has been destroyed. This can happen if you apply a change to the trigger text and that change results in compilation errors.

Action: You can recompile the trigger in the debugger or exit the form and start it up again.
Level: 99
Trigger: ON-ERROR

FRM-42431: Unable to initialize debugger.
Cause: An error occured while attempting to initialize debugger. This could be caused by one of the following:
1. The JVM failed to startup.
2. The Classpath does not include the debugger dependencies.

Action: Restart the JVM or modify the Classpath.
Level: 99
Trigger: ON-ERROR

FRM-42435: Remote Debugger: Specified port(s) are not available.
Cause: Other processes are using the specified port(s).

Action: Try some other port.
Level: 99
Trigger: None

FRM-47000: Cannot create Parameter List %s : internal error.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47001: Cannot create Parameter List %s : list with this name exists.
Cause: The name you specified for the parameter list is already in use.

Action: Specify another name for the parameter list.
Level: 99
Trigger: ON-ERROR

FRM-47002: Cannot create Parameter List : name must not be null.
Cause: The parameter list name cannot be null.

Action: Specify a name for the parameter list.
Level: 99
Trigger: ON-ERROR

FRM-47003: Cannot delete Parameter List : internal error.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47004: Cannot delete Parameter List : invalid ID.
Cause: Attempted to pass an invalid parameter list ID.

Action: Check the list ID name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47005: Cannot validate parameter %s : internal error.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47006: Cannot create Parameter List '%s' : illegal identifier name.
Cause: Illegal identifier name.

Action: Check valid syntax for the identifier.
Level: 99
Trigger: ON-ERROR

FRM-47007: Cannot get parameter %s attributes from Parameter List : invalid list ID.
Cause: Specified an invalid parameter list ID.

Action: Check the parameter list ID name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47008: Cannot add parameter %s to Parameter List %s : internal error.
Cause: An internal error occurred.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47009: Cannot add parameter %s to Parameter List : invalid list ID.
Cause: You specified an invalid parameter list ID.

Action: Check the parameter list ID name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47010: Cannot add parameter to Parameter List %s : null key specified.
Cause: No name specified for the parameter.

Action: Specify a parameter name key in your call to ADD_PARAMETER.
Level: 99
Trigger: ON-ERROR

FRM-47011: Cannot add parameter %s to Parameter List %s : incorrect type specified.
Cause: You specified an invalid parameter type.

Action: Specify either TEXT_PARAMETER or DATA_PARAMETER.
Level: 99
Trigger: ON-ERROR

FRM-47012: Cannot add parameter %s to Parameter List %s : group %s does not exist.
Cause: Record group name does not exist.

Action: Check the record group name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47013: Cannot add parameter %s to Parameter List %s : parameter with this name exists.
Cause: Parameter with this name already exists.

Action: Specify another name for the parameter.
Level: 99
Trigger: ON-ERROR

FRM-47014: Cannot delete parameter %s from Parameter List %s : internal error.
Cause: Internal error.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47015: Cannot delete parameter %s from Parameter List : invalid list ID.
Cause: Specified an invalid parameter list ID.

Action: Check the list ID name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47016: Cannot delete parameter from Parameter List %s : null key specified.
Cause: You did not specify a name for the parameter.

Action: Correct the call to DELETE_PARAMETER by supplying a parameter name.
Level: 99
Trigger: ON-ERROR

FRM-47017: Cannot delete parameter %s from Parameter List %s : no such named parameter exists.
Cause: Caused by one of the following:
1. You specified an invalid parameter list ID.
2. You specified a parameter name that does not exist.

Action: Check the parameter name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47018: Cannot set parameter %s attributes in Parameter List %s : internal error.
Cause: Internal error: you specified an invalid parameter list ID.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47019: Cannot set parameter %s attributes in Parameter List : invalid list ID.
Cause: You specified an invalid parameter list ID.

Action: Check the parameter list ID name and try again.

Level: 99
Trigger: ON-ERROR

FRM-47020: Cannot set parameter %s attributes in Parameter List %s : no such named parameter exists.
Cause: Caused by one of the following:
1. You specified an invalid parameter list ID.
2. You specified a parameter name that does not exist.

Action: Check the parameter name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47021: No such parameter named %s exists in Parameter List %s.
Cause: Caused by one of the following:
1. You specified an invalid parameter list ID.
2. You specified a parameter name that does not exist.

Action: Check the name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47022: Cannot create Parameter List %s : name is a reserved word.
Cause: Caused by one of the following:
1. You specified an invalid parameter list ID.
2. You specified a name that is a reserved word.

Action: Specify another name for the parameter list.
Level: 99
Trigger: ON-ERROR

FRM-47023: No such parameter named %s exists in form %s.
Cause: Caused by one of the following:
1. You specified an invalid parameter list ID.
2. You specified a parameter name does not exist.

Action: Check the name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47024: Parameter %s type does not match definition in form %s.
Cause: You specified a parameter type that does not match the definition in the form.

Action: Specify a parameter type that matches the definition in the form.
Level: 99
Trigger: ON-ERROR

FRM-47025: Cannot get parameter %s attributes from Parameter List %s : internal error.
Cause: Internal error: you specified an invalid parameter list ID.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47026: Cannot get parameter %s attributes from Parameter List %s : no such named parameter exists.
Cause: Caused by one of the following:

1. You specified an invalid parameter list ID.
2. You specified a parameter name does not exist.

Action: Check the name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47027: Cannot add parameter %s to Parameter List %s : invalid key specified .
Cause: You specified an invalid parameter list ID.

Action: Check the name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47028: Cannot set parameter %s attribute in Parameter List %s : group %s does not exist.
Cause: You specified an invalid parameter list ID.

Action: Check the name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47029: Invalid parameter list ID in form %s.
Cause: Caused by one of the following:
1. You specified an invalid parameter list ID.
2. You specified a parameter name does not exist.

Action: Check the ID and try again.
Level: 99
Trigger: ON-ERROR

FRM-47030: Value of parameter %s is too long for definition in form %s.
Cause: You specified a parameter that is too long.

Action: Specify a parameter that is valid.
Level: 99
Trigger: ON-ERROR

FRM-47031: Cannot set value of parameter %s in DEFAULT parameter list: invalid value specified.
Cause: Application design error. A Built-in (such as SET_PARAMETER_ATTR) is attempting to set the value of a parameter which was defined when the form was designed, but the value specified is not legal for the parameter's datatype.

Action: The call to the Built-in must be modified or removed.
Level: 99
Trigger: ON-ERROR

FRM-47032: Cannot set value of parameter %s in DEFAULT parameter list: internal error.
Cause: An internal error while attempting to set the value of a parameter.

Action: If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47033: Cannot set value of read-only bind variable %s.
Cause: Application design error. The application attempted to assign a value to a bind variable which cannot be programmtically modified.

Action: The assignment must be removed.
Level: 25
Trigger: ON-ERROR

FRM-47100: Cannot read image file %s.
Cause: Caused by one of the following:
1. Oracle Forms was unable to find or open the file.
2. The data in the file is not in the specified format.

Action: Check the file name and file format and try again.
Level: 99
Trigger: ON-ERROR

FRM-47101: Cannot write image file %s.
Cause: Caused by one of the following:
1. Oracle Forms was unable to find or open the file.
2. The data in the file is not in the specified format.

Action: Check the file name, and make sure you have write privileges.
Level: 99
Trigger: ON-ERROR

FRM-47102: Cannot perform %s operation on images %s and %s.
Cause: This operation cannot be performed on color images.

Action: Check to see that both images are black and white.
Level: 99
Trigger: ON-ERROR

FRM-47103: Cannot zoom image %s.
Cause: Internal multimedia error caused by trying to scale a null image or invalid image data.

Action: Check the image name that you want to zoom and try again.
Level: 99
Trigger: ON-ERROR

FRM-47104: Invalid image type %s.
Cause: Data in the file name specified does not match the data type specified.

Action: Check the file name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47105: No image name specified.
Cause: You did not supply a name to the Built-in call.

Action: Refer to the documentation for the proper syntax for the Built-in in question.
Level: 99
Trigger: ON-ERROR

FRM-47106: No image type specified.
Cause: You did not specify an image type when calling READ_IMAGE_FILE or WRITE_IMAGE_FILE.

Action: Supply an image type as an argument in your call to READ_IMAGE_FILE or WRITE_IMAGE_FILE.
Level: 99
Trigger: ON-ERROR

FRM-47107: Invalid zoom factor %.10g for Image_Zoom.
Cause: You specified a zoom factor outside the range 100.0/4096.0 - 409600.0 for a zoom type of ZOOM_IN_FACTOR or ZOOM_OUT_FACTOR.

Action: Correct the call to IMAGE_ZOOM.
Level: 20
Trigger: ON-ERROR

FRM-47108: Item %s is not an image item.
Cause: You attempted to perform an image operation on an item that is not an image item.

Action: Check the item name and try again.
Level: 99
Trigger: ON-ERROR

FRM-47109: Cannot locate image file %s.
Cause: You specified a file that cannot be found or does not exist.

Action: Verify that the file exists and the pathname is correct.
Level: 20
Trigger: ON-ERROR

FRM-47110: No region was selected for Image_Zoom: %s
Cause: You tried to zoom an image without selecting an image region.

Action: Select an image region.
Level: 20
Trigger: ON-ERROR

FRM-47111: Cannot copy value to item: %s
Cause: You tried to use the COPY Built-in on an image item.

Action: You cannot use the COPY Built-in on an image item.
Level: 99
Trigger: ON-ERROR

FRM-47112: Invalid zoom type %d for Image_Zoom.
Cause: You specified a zoom type for Image_Zoom that was not one of ADJUST_TO_FIT, SELECTION_RECTANGLE, ZOOM_IN_FACTOR, ZOOM_OUT_FACTOR, or ZOOM_PERCENT.

Action: Correct the call to IMAGE_ZOOM.
Level: 20
Trigger: ON-ERROR

FRM-47113: Invalid zoom percent %.10g for Image_Zoom.
Cause: Caused by one of the following:
1. You specified an invalid zoom percent (outside the range 100.0/4096.0 - 409600.0) for a zoom type of ZOOM_PERCENT.

Action: Correct the call to IMAGE_ZOOM.
Level: 20

Trigger: ON-ERROR

FRM-47114: Cannot zoom or scroll an image item whose Sizing Style is Fill.
Cause: You attempted to call IMAGE_ZOOM or IMAGE_SCROLL on an image item whose Sizing Style is Fill.

Action: Correct or remove the call to IMAGE_ZOOM or IMAGE_SCROLL.
Level: 20
Trigger: ON-ERROR

FRM-47115: Internal error occured while reading unsupported image.
Cause: Image may be corrupt or may be non-standard format or compression.

Action: Verify image data, format, compression etc.
Level: 99
Trigger: ON-ERROR

FRM-47300: Item is not a hierarchical tree. (%s)
Cause: A hierarchical tree Built-in was invoked on a non-tree item.

Action: Check item type and name.
Level: 99
Trigger: ON-ERROR

FRM-47301: Cannot add data as sibling to the tree root.
Cause: ADD_TREE_DATA or ADD_TREE_NODE attempted to add data as a sibling of the root.

Action: Add data at a lower level in the tree.
Level: 99
Trigger: ON-ERROR

FRM-47302: Can only add data to tree as child or sibling.
Cause: ADD_TREE_DATA or ADD_TREE_NODE attempted to use an unknown offset_type value.

Action: Only PARENT_OFFSET and SIBLING_OFFSET are allowed for the offset_type parameter.
Level: 99
Trigger: ON-ERROR

FRM-47303: ADD_TREE_DATA only accepts data from a group or query.
Cause: ADD_TREE_DATA attempted to use an unknown datasource value.

Action: Only RECORD_GROUP and QUERY_TEXT are allowed for the datasource parameter.
Level: 99
Trigger: ON-ERROR

FRM-47304: Cannot delete the root node of a tree.
Cause: DELETE_TREE_NODE attempted to delete the root node.

Action: Check if a node is the root (use ID_NULL) before trying to delete it.
Level: 99
Trigger: ON-ERROR

FRM-47305: Can only search a tree looking for label or value.
Cause: FIND_TREE_NODE attempted to use an unknown search_by parameter value.

Action: Set the search_by parameter to NODE_LABEL or NODE_VALUE.
Level: 99
Trigger: ON-ERROR

FRM-47306: Search_type must be FIND_NEXT or FIND_NEXT_CHILD.
Cause: FIND_TREE_NODE attempted to use an invalid search_type parameter value.

Action: Set the search_type parameter to FIND_NEXT or FIND_NEXT_CHILD.
Level: 99
Trigger: ON-ERROR

FRM-47307: Cannot get the properties of the tree root node.
Cause: GET_TREE_NODE_PARENT or GET_TREE_NODE_PROPERTY attempted to obtain information from the root node.

Action: Only invoke the Built-ins against valid nodes.
Level: 99
Trigger: ON-ERROR

FRM-47308: Invalid property for GET or SET_TREE_NODE_PROPERTY.
Cause: You passed an invalid property constant to GET or SET_TREE_NODE_PROPERTY.

Action: Verify arguments.
Level: 99
Trigger: ON-ERROR

FRM-47309: Invalid property for GET or SET_TREE_PROPERTY.
Cause: You passed an invalid property constant to GET or SET_TREE_PROPERTY.

Action: Verify arguments.
Level: 99
Trigger: ON-ERROR

FRM-47310: Bad selection index for GET_TREE_SELECTION.
Cause: Selection index must be in the range 1 .. number of selected nodes.

Action: Ensure that the selection is within the required range.
Level: 99
Trigger: ON-ERROR

FRM-47311: Error populating record group.
Cause: Invalid record group specified for POPULATE_GROUP_FROM_TREE.

Action: Check existence of the record group, and that it isn't statically defined.
Level: 99
Trigger: ON-ERROR

FRM-47312: Internal error populating record group.
Cause: Internal error during POPULATE_GROUP_FROM_TREE. Note that some rows may already have been added.

Action: Unable to add row to the record group.
Level: 99
Trigger: ON-ERROR

FRM-47313: Invalid query for the hierarchical tree.
Cause: Unable to create valid tree data from the specified query text.

Action: Check the query text for a valid number of columns and valid data types.
Level: 99
Trigger: ON-ERROR

FRM-47314: Cannot set the properties of the tree root node.
Cause: SET_TREE_NODE_PROPERTY attempted to set a property of the root node.

Action: Only invoke the Built-in against valid nodes.
Level: 99
Trigger: ON-ERROR

FRM-47315: Invalid parameter value for SET_TREE_NODE_PROPERTY.
Cause: Check the parameter values for the tree node property being set.

Action: Correct the parameter value passed to SET_TREE_NODE_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-47316: Branch nodes with no children are not allowed.
Cause: Attempt to change the state of a node from a leaf node to a branch. This tree does not allow empty branch nodes.

Action: Either change the tree to allow empty branch nodes, or add children to the node.
Level: 99
Trigger: ON-ERROR

FRM-47317: Leaf nodes cannot have children.
Cause: Attempt to change the state of a node with children from a branch node to a leaf.

Action: Delete the children from the node before changing the node's state.
Level: 99
Trigger: ON-ERROR

FRM-47318: Invalid parameter value for SET_TREE_PROPERTY.
Cause: Check the parameter values for the tree property being set.

Action: Correct the parameter value passed to SET_TREE_PROPERTY.
Level: 99
Trigger: ON-ERROR

FRM-47319: Cannot select the tree root node.
Cause: SET_TREE_SELECTION attempted to select the root node.

Action: Only invoke the Built-in against valid nodes.
Level: 99
Trigger: ON-ERROR

FRM-47320: Bad selection type for SET_TREE_SELECTION.
Cause: SET_TREE_SELECTION attempted to use an invalid selection_type parameter value.

Action: Set the selection_type parameter to SELECT_ON, SELECT_OFF, or SELECT_TOGGLE.
Level: 99
Trigger: ON-ERROR

FRM-47321: Data used to populate tree is invalid.
Cause: ADD_TREE_DATA attempted to use data of wrong format.

Action: Check number and type of columns in group or query.
Level: 99
Trigger: ON-ERROR

FRM-47322: The specified tree data source is not a record group.
Cause: The data source for the tree was declared a record group, but isn't.

Action: Check that the id or name specified is of an existing record group.
Level: 99
Trigger: ON-ERROR

FRM-47323: No nodes are selected in the tree.
Cause: Attempt was made to obtain a selected node when none are currently selected.

Action: Check for number of selected nodes before trying to retrieve one of them.
Level: 99
Trigger: ON-ERROR

FRM-47324: Could not allocate memory for tree structures.
Cause: Unable to allocate memory for internal tree structures. Tree destroyed.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47325: Could not allocate memory for tree node.
Cause: Unable to allocate memory for a tree node.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47333: Could not set required state for tree node.
Cause: Unable to change state for a tree node.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47334: Could not allocate memory for tree node icon string.
Cause: Unable to allocate memory for the name of a tree node icon.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47335: Could not locate a tree node icon. (%s)
Cause: Unable to find the desired icon in standard locations.

Action: Check that your tree node icons are located in the proper directories.

Level: 99
Trigger: ON-ERROR

FRM-47336: Could not set tree node to the requested icon.
Cause: Unable to set the requested tree node icon.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47337: Tree node label cannot be null.
Cause: Attempt was made to set a tree node's label to a null value.

Action: Set the label to a non-null value.
Level: 99
Trigger: ON-ERROR

FRM-47338: Could not allocate memory for tree node label.
Cause: Unable to allocate memory for a tree node label.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47339: Could not set tree node to the requested label.
Cause: Unable to set the requested tree node label.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47340: Could not allocate memory for tree node value.
Cause: Unable to allocate memory for a tree node value.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47341: There are too many nodes for the tree.
Cause: Only MAX-SIGNED-4-BYTE nodes, both current and deleted, are permitted in a tree.

Action: Decrease the number of nodes placed in the tree. If constantly adding and removing nodes, you might need to clear and re-populate the tree.
Level: 99
Trigger: ON-ERROR

FRM-47342: Could not allocate memory for tree query text.
Cause: Unable to allocate memory for the tree query text.

Action: Try executing the application when the system is less heavily loaded. If the problem persists, contact Oracle Support Services.
Level: 99
Trigger: ON-ERROR

FRM-47343: Invalid node ID specified for hierarchical tree item %s
Cause: A hierarchical tree built-in was invoked on a hierarchical tree item, but the node ID passed to the built-in was not valid for the tree item.

Action: Specify a valid node ID.
Level: 99
Trigger: ON-ERROR

FRM-47344: RECORDS_DISPLAYED property can't be set for a tree item
Cause: An attempt was made to change the number of records displayed for a block that contains a hierarchical tree or the hierarchical tree itself but hierarchical trees must always be in single-record blocks.

Action: Specify a different block or item.
Level: 20
Trigger: ON-ERROR

FRM-47500: Failed to register database event %s
Cause: Attempt to register a database event failed

Action: Check the event attributes on the database side
Level: 5
Trigger: ON-ERROR

FRM-47501: Cannot find event: invalid ID.
Cause: The user has specified an invalid event object name.

Action: Check the form and make sure that the event object exists.
Level: 99
Trigger: ON-ERROR

FRM-47502: Invalid event property
Cause: Invalid event property

Action: Check the event event property
Level: 5
Trigger: ON-ERROR

FRM-47700: Failed to start the JVM.
Cause: An error occured while attempting to start the inprocess JVM.

Action: Make sure that jvm libraries can be located by the runtime process
Level: 99
Trigger: ON-ERROR

FRM-47800: Unable to communicate with the JVM Controller: %s.
Cause: Unable to communicate with the JVM Controller

Action: JVM Controller, to which runform is connected, might be down. Contact your system administrator.
Level: 99
Trigger: ON-ERROR

FRM-50000: Value is too long.
Cause: You entered a value which contains too many bytes or characters for the item.

Action: Enter a shorter value.

Level: 15
Trigger: ON-ERROR

FRM-50001: Acceptable characters are a-z, A-Z, and space.
Cause: You entered an unacceptable character into the item.

Action: Enter a character from a-z, A-Z, or a space.
Level: 15
Trigger: ON-ERROR

FRM-50002: Month must be between 1 and 12.
Cause: You entered an invalid month value in a date field.

Action: Enter a month value from 1 (for January) to 12 (for December).
Level: 15
Trigger: ON-ERROR

FRM-50003: Year must be in proper range.
Cause: You entered a year that is not valid for the applicable format mask year element.

Action: Enter a valid year. For most format mask year elements, a number between 0 and 9999 is acceptable. For signed format mask year elements, a number between -4712 and -1 may also be specified.
Level: 15
Trigger: ON-ERROR

FRM-50004: Day must be between 1 and last of month.
Cause: You entered an invalid day.

Action: Enter a valid day. For April, for example, enter a number between 1 and 30.
Level: 15
Trigger: ON-ERROR

FRM-50006: Legal characters are 0-9 + and -.
Cause: You entered an unacceptable character in a number item.

Action: Enter a valid number. A valid number has digits 0 through 9. A number may be preceded by a plus (+) or minus (-) sign. If the message allows it, a number may contain one decimal point at any location, except before the sign.
Level: 15
Trigger: ON-ERROR

FRM-50007: Too many digits after decimal point.
Cause: You entered a number with 3 or more decimal digits after the decimal point in an item with the MONEY or RMONEY data type.

Action: Re-enter a valid number.
Level: 15
Trigger: ON-ERROR

FRM-50009: Too many decimal points.
Cause: You entered a number that contains two or more decimal points, or you have entered a number that contains a decimal point in an item that requires a whole (non-decimal) number.

Action: Enter a number with no more than one decimal point. If you have used only one decimal, remove the decimal and the decimal part of the number.
Level: 15

Trigger: ON-ERROR

FRM-50010: Money format is [+-]9999999.99
Cause: You entered an invalid value in a MONEY or RMONEY item.

Action: Enter a valid value. This value should have zero or dollar digits, followed by a decimal and two cents digits. The entire number can be preceded by a plus (+) or a minus (-) sign.
Level: 15
Trigger: ON-ERROR

FRM-50011: Not a valid month name.
Cause: You entered an invalid month name in a date field.

Action: Enter a valid month name. Oracle Forms recognizes the first three characters of a month name. For example, JAN stands for January, JUN for June.
Level: 15
Trigger: ON-ERROR

FRM-50012: Date must be entered in a format like %s.
Cause: You entered an invalid or incorrectly formatted date.

Action: Re-enter the date in the requested format.
Level: 15
Trigger: ON-ERROR

FRM-50013: Plus or minus must be in first position.
Cause: You entered the plus or minus sign in the wrong position.

Action: Retype with the plus or minus sign in the first position.
Level: 15
Trigger: ON-ERROR

FRM-50014: Bad exponent.
Cause: You entered an exponent in an item that does not accept exponents.

Action: Enter a value without an exponent.
Level: 15
Trigger: ON-ERROR

FRM-50016: Legal characters are 0-9 - + E .
Cause: You entered an unacceptable character in a number item.

Action: Enter a valid number. A valid number has digits 0 through 9. A number may be preceded by a plus (+) or minus (-) sign. If the message allows it, a number may contain one decimal point at any location, except before the sign. You can use an E to specify scientific notation.
Level: 15
Trigger: ON-ERROR

FRM-50017: Hour must be between 0 and 23.
Cause: You entered an invalid hour.

Action: Enter a valid hour. Oracle Forms records time on a 24-hour basis.
Level: 15
Trigger: ON-ERROR

FRM-50018: Minutes must be between 00 and 59.
Cause: You entered an invalid minute value.

Action: Enter a valid minute value.
Level: 15
Trigger: ON-ERROR

FRM-50019: Seconds must be between 00 and 59.
Cause: You entered an invalid value.

Action: Enter a value between 00 and 59.
Level: 15
Trigger: ON-ERROR

FRM-50020: Missing exponent.
Cause: You failed to enter an exponent.

Action: Enter an exponent.
Level: 15
Trigger: ON-ERROR

FRM-50021: Date must be entered in a format like %s.
Cause: You entered an invalid or incorrectly formatted date.

Action: Re-enter the date in the requested format.
Level: 15
Trigger: ON-ERROR

FRM-50022: Time must be entered in a format like %s.
Cause: You entered an invalid or incorrectly formatted time.

Action: Re-enter the time in the requested format.
Level: 15
Trigger: ON-ERROR

FRM-50023: Date must be entered in a format like %s.
Cause: You entered an invalid or incorrectly formatted date.

Action: Re-enter the date in the requested format.
Level: 15
Trigger: ON-ERROR

FRM-50024: Space are allowed in leading positions only.
Cause: You entered spaces intermixed with data.

Action: Re-enter data with no spaces intermixed.
Level: 15
Trigger: ON-ERROR

FRM-50025: Date/time must be entered in a format like %s.
Cause: You entered an invalid or incorrectly formatted date and time.

Action: Re-enter the date and time in the requested format.
Level: 15
Trigger: ON-ERROR

FRM-50026: Date must be entered in a format like %s.
Cause: You entered an invalid or incorrectly formatted date.

Action: Re-enter the date in the requested format.
Level: 15
Trigger: ON-ERROR

FRM-50027: Invalid format mask for given datatype.
Cause: The format mask you assigned to a text item is incompatible with the data type of the text item.

Action: Assign a new format mask to the text item. For more information, refer to help on <a href="../../designing_forms/items/f500842.html">About Formatting Text Item Values with Format Masks</a>.
Level: 15
Trigger: ON-ERROR

FRM-50028: Format mask not allowed for this datatype.
Cause: The data types LONG and IMAGE do not support a format mask.

Action: Do not try to create a format mask for data types LONG or IMAGE.
Level: 15
Trigger: ON-ERROR

FRM-50029: Too many digits preceding decimal point for scientific notation.
Cause: You specified a number using scientific notation, but used more than one digit preceding the decimal point.

Action: Re-enter the number using scientific notation.
Level: 15
Trigger: ON-ERROR

FRM-50045: Seconds past midnight confilicts with hour.
Cause: You entered a time where the seconds past midnight component does not agree with the hour component.

Action: Make sure the hour and seconds past midnight agree, or use a format mask without seconds past midnight.
Level: 15
Trigger: ON-ERROR

FRM-50048: New passwords do not match. Please make them identical.
Cause: You entered different strings in 'New Password' and 'Retype New' fields.

Action: Re-enter the values in (New and Retype) fields such that they identical.
Level: 99
Trigger: None

FRM-91124: fatal error in runtime process: %s specified for FORMS_DECIMAL_PREFIX. Should be zero or the empty string
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91126: fatal error in runtime process: invalid value %s specified for environment variable %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91127: runtime process: invalid directory name specified for environment variable %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91129: runtime process: no value specified for required environment variable %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91132: fatal error in runtime process: invalid data in timezone file %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91135: fatal error in runtime process: message file %s is missing
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91137: runtime process: unable to open log file %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-91145: fatal error in runtime process: missing serverArgs parameter
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92000: internal error: cannot access Java class
Cause: The Forms server requested a Java class by specifying a numeric "handlerClassId", and the Forms Java client found an entry for the specified handlerClassId in the registry. However, the Java class that was specified by the registry entry could not be accessed.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause (e.g. a configuration problem), correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92020: invalid URL %s sent to browser with target %s. full details: %s
Cause: The Forms application executed the web.showDocument built-in, and the applet did not define the clientBrowser parameter, which caused Forms to attempt to resolve the specified URL relative to the applet's document base. However, this attempt encountered a MalformedURLException. This is not a fatal error.

Action: Correct the URL that is passed to the web.showDocument built-in.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92022: cannot launch URL %s with target %s. full details: %s
Cause: The Forms application executed the web.showDocument built-in, and encountered an Exception. This is not a fatal error.

Action: If the "full details" messages indicates the problem, correct it. Otherwise, if the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92030: internal error: no registry entry for handleClassId=%s
Cause: The Forms server requested a Java class by specifying a numeric "handlerClassId", but the Forms Java client could not find an entry for the specified handlerClassId in the registry.

Action: If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92040: internal error: cannot find Java class
Cause: The Forms server requested a Java class by specifying a numeric "handlerClassId", and the Forms Java client found an entry for the specified handlerClassId in the registry. However, the Java class that was specified by the registry entry could not be found.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause (e.g. a configuration problem), correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92050: fatal error: cannot connect to the server: %s:%s
Cause: An attempt was made to connect to the Forms server. The serverURL applet parameter was not specified, so Forms attempted to connect to the specified host machine, on the specified port. (These are derived from the serverHost and serverPort applet parameters, if specified). However, an unexpected Exception was encountered. This message appears when there is no message that gives a more specific reason for the connection failure.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92052: fatal error: cannot connect to the server at URL %s
Cause: An attempt was made to connect to the Forms server, at the specified URL. (This is derived from the serverURL applet parameter). However, an unexpected Exception was encountered. This message appears when there is no message that gives a more specific reason for the connection failure.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92054: error while terminating single sign-on session with URL %s
Cause: An attempt was made to connect to the single sign-on server using OHS redirect, at the specified URL. However, an unexpected Exception was encountered. This message appears when there is no message that gives a more specific reason for the connection failure.

Action: The system administrator should ensure that OHS redirects are working properly.
Appears: Java console
Level: 99
Trigger: None

FRM-92055: error reading cookies while terminating single sign-on session
Cause: An attempt was made to read the cookies while terminating single sign-on session. However, an unexpected Exception was encountered. This message appears when there is no message that gives a more specific reason for the failure.

Action: Make sure that Javascript is enabled for the browser.
Appears: Java console
Level: 99
Trigger: None

FRM-92056: error reading cookies while checking life of single sign-on session
Cause: An attempt was made to read the cookies while checking life of single sign-on session. However, an unexpected Exception was encountered. This message appears when there is no message that gives a more specific reason for the failure.

Action: Make sure that Javascript is enabled for the browser.
Appears: Java console
Level: 99
Trigger: None

FRM-92060: fatal error: cannot connect to the server: bad machine specification: %s:%s
Cause: An attempt was made to connect to the Forms server. The serverURL applet parameter was not specified, so Forms attempted to connect to the specified host machine, on the specified port. (These are derived from the serverHost and serverPort applet parameters, if specified). However, the format of the host/port combination was invalid.

Action: Correct the syntax of the serverHost and/or the serverPort applet parameter, or specify a valid value for the serverURL applet parameter.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92062: fatal error: cannot connect to the server: bad URL specification: %s
Cause: An attempt was made to connect to the Forms server, at the specified URL. (This is derived from the serverURL applet parameter). However, the URL was malformed.

Action: Correct the syntax of the serverURL applet parameter.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92070: internal error: cannot instantiate Java class
Cause: The Forms server requested a Java class by specifying a numeric "handlerClassId", and the Forms Java client found an entry for the specified handlerClassId in the registry. However, the Java class that was specified by the registry entry could not be instantiated.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause (e.g. a configuration problem), correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92080: fatal error: cannot execute command: %s %s. full details: %s
Cause: The Forms application executed the web.showDocument built-in, and the applet defined the clientBrowser parameter, which caused Forms to attempt to start the specified external client browser. However, this attempt encountered an Exception.

Action: If the "full details" messages indicates the problem, correct it. Otherwise, if the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92089: unexpected fatal error while initializing the applet's user interface
Cause: An unexpected Exception was encountered while attempting to initialize the applet's user interface.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, applet status window
Level: 99
Trigger: None

FRM-92090: unexpected fatal error in client-side Java code during startup
Cause: An unexpected Exception was encountered in client-side Java code during startup.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92091: unexpected fatal error in client-side Java code
Cause: An unexpected Exception was encountered in client-side Java code (after startup).

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92095: Cannot obtain server certificate chain
Cause: SSL certificate(s) were not found on the server.

Action: Contact the server admin for SSL certificate(s).
Appears: Java console
Level: 99
Trigger: None

FRM-92102: A network error or server failure has occurred. The Forms client has attempted to reestablish its connection to the Server %s time(s) without success. You will need to restart your application.
Cause: The Forms Java client attempted to communicate with the Forms server. The indicated number of attempts (specified by the networkRetries applet parameter) were made, but each attempt encountered an unexpected Exception. This probably indicates a problem with the network, or with the application server that was hosting the Forms server, or with the server machine that was hosting the application server.

Action: Correct the network problem (if any), or restart the application server or reboot the server host machine (if necessary).
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92103: A network error or server failure has occurred. You will need to restart your application.
Cause: The Forms Java client attempted to communicate with the Forms server. The networkRetries applet parameter did not specify a positive value, so only a single attempt was made. This attempt encountered an unexpected Exception. This probably indicates a problem

with the network, or with the application server that was hosting the Forms server, or with the server machine that was hosting the application server.

Action: Correct the network problem (if any), or restart the application server or reboot the server host machine (if necessary).
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92104: A network error or server failure has occurred. The request was sent to the wrong application server (not the one which created the session). The Forms client has attempted to migrate the session %s time(s) without success. You will need to restart your application.
Cause: The Forms Java client attempted to communicate with the Forms server. The indicated number of attempts were made, but on each attempt, the request was sent to the wrong application server (not the one which created the session). This probably indicates a problem with the network, or with the application server that was hosting the Forms server that initially created the session, or with the server machine that was hosting the application server.

Action: Correct the network problem (if any), or restart the application server or reboot the server host machine (if necessary).
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92110: New passwords do not match. They must be identical. Password change failed.
Cause: In the Change Password dialog, the new password and the retyped new password do not match.

Action: Retry the Change Password dialog, and correctly type and retype the new password.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92120: Fatal error: registry file %s is missing.
Cause: The Forms Java client was unable to read the registry file at the specified URL (on the Forms server machine).

Action: The system administrator should ensure that the registry file exists and is readable.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92130: JavaScript execution is disabled in an Oracle Forms Standalone Application.
Cause: JavaScript integration is not supported in an Oracle Forms Standalone Application.

Action: Use a browser if you want to run JavaScript integration with Oracle Forms.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92134: JavaFX not found, audio support disabled.
Cause: The JavaFX classes needed for audio support were not found.

Action: Install and configure JavaFX to run with your client-side Java installation.

Appears: Java console
Level: 99
Trigger: None

FRM-92135: No source file specified for Play_Audio.
Cause: URI String is null or does not exist.

Action: Check the value specified as media file name, it should not be null and file should exist at specified location.
Appears: Java console
Level: 99
Trigger: None

FRM-92136: Illegal URL format used in media file name %s.
Cause: Either URI has null scheme or its not a proper URI.

Action: Check the value specified as media file name.
Appears: Java console
Level: 99
Trigger: None

FRM-92137: Unsupported protocol specified for Play_Audio %s.
Cause: Unsupported protocol specified for the media source.

Action: Ensure that a supported protocol is being used for Play_Audio.
Appears: Java console
Level: 99
Trigger: None

FRM-92138: Malformed URL used in Play_Audio %s.
Cause: Incorrect/unsupported URL supplied as media source.

Action: Check the media URL and make the required correction in the URL format.
Appears: Java console
Level: 99
Trigger: None

FRM-92140: Media Error - %s MEDIA_CORRUPTED
Cause: The media appears to be invalid or corrupted.

Action: Validate the media file if its a valid media file. Also check, if its supported by javafx.scene.media.MediaPlayer.
Appears: Java console
Level: 99
Trigger: None

FRM-92141: Media Error - %s MEDIA_INACCESSIBLE
Cause: Although the media may exist, it is not accessible.

Action: Check the access rights are given to the user which is trying to access media file.
Appears: Java console
Level: 99
Trigger: None

FRM-92142: Media Error - %s MEDIA_UNAVAILABLE
Cause: Media does not exist or is otherwise unavailable.

Action: Check if media file is physically available at the specified location.
Appears: Java console
Level: 99
Trigger: None

FRM-92143: Media Error - MEDIA_UNSPECIFIED
Cause: Media has not been specified.

Action: Supply a valid media file name.
Appears: Java console
Level: 99
Trigger: None

FRM-92144: Media Error - %s MEDIA_UNSUPPORTED
Cause: Media type is not supported by this platform.

Action: Use one of the supported formats for the source media.
Appears: Java console
Level: 99
Trigger: None

FRM-92145: Media Error - OPERATION_UNSUPPORTED
Cause: An operation performed on the media is not supported by this platform.

Action: Operation user is trying to perform on media is not supported on this platform.
Appears: Java console
Level: 99
Trigger: None

FRM-92146: Media Error - PLAYBACK_ERROR
Cause: A playback error which does not fall into any of the other pre-defined categories.

Action: Contact your system administrator.
Appears: Java console
Level: 99
Trigger: None

FRM-92147: Media Error - PLAYBACK_HALTED
Cause: An unrecoverable error which has resulted in halting playback.

Action: Media player is in a halted state. Restart the application if media support is needed.
Appears: Java console
Level: 99
Trigger: None

FRM-92148: Media Error - %s UNKNOWN
Cause: An error has occurred for an unknown reason.

Action: Contact your system administrator.
Appears: Java console
Level: 99
Trigger: None

FRM-92150: Fatal error: web client version is too new.
Cause: The version of the client is newer than the version of the Server.

Action: The system administrator should ensure that the Forms product is correctly installed on the Forms server machine.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92160: Fatal error: web client version is too old.
Cause: The version of the client is older than the version of the Server.

Action: The system administrator should ensure that the Forms product is correctly installed on the Forms server machine.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92170: JavaScript execution is disabled during Java Web Start session.
Cause: JavaScript integration is not supported when running Java Web Start.

Action: Use a browser if you want to run JavaScript integration with Oracle Forms.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92180: Fatal error: JavaScript is unable to obtain the server URL. This can occur if legacy_lifecycle=true and JavaScript has been disabled. If so, you will need to reenable JavaScript, restart the browser, and restart your application.
Cause: The serverURL applet parameter specified a value of "?", which indicates that the serverURL should be obtained from an element outside of the applet. (In a page that's generated from a standard base HTML file, this occurs when legacy_lifecycle=true in formsweb.cfg). Forms attempted to obtain the serverURL using JavaScript, but the attempt failed, probably because JavaScript has been disabled.

Action: Reenable JavaScript and restart the browser. If that does not solve the problem, examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92190: JavaScript is unable to evaluate expression.
Cause: The Forms application executed the web.JavaScript_Eval_Expr built-in, but an invalid Javascript expression was specified as the first argument.

Action: Correct the Javascript expression that is passed to the web.JavaScript_Eval_Expr built-in.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92192: Target %s for JavaScript evaluation does not exist.
Cause: The Forms application executed the web.JavaScript_Eval_Expr built-in, but an invalid target was specified as the second argument.

Action: Correct the target that is passed to the web.JavaScript_Eval_Expr built-in.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92193: error executing JavaScript while terminating single sign-on session
Cause: An attempt was made to terminate the single sign-on session using Javascript. However, an unexpected Exception was encountered.

Action: If the problem persists, contact Oracle Support Services.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92210: invalid value %s for lookAndFeel applet parameter. Defaulting to %s.
Cause: An invalid value was specified for the lookAndFeel applet parameter.

Action: The system administrator should ensure that a valid value was specified for the lookAndFeel applet parameter.
Appears: Java console, applet status window
Level: 99
Trigger: None

FRM-92211: invalid value %s for colorScheme applet parameter. Forms will use the default colorScheme for the specified lookAndFeel.
Cause: An invalid value was specified for the colorScheme applet parameter.

Action: The system administrator should ensure that a valid value was specified for the colorScheme applet parameter.
Appears: Java console, applet status window
Level: 99
Trigger: None

FRM-92212: unable to process %s required for custom color scheme. Forms will not use custom color scheme.
Cause: Unable to locate registry file or it had some format issue.

Action: The system administrator should ensure that registry file is available and valid.
Appears: Java console
Level: 99
Trigger: None

FRM-92213: color parameter %s not defined in %s. Forms will not use custom color scheme.
Cause: A color parameter required for custom color scheme was undefined in registry file.

Action: The system administrator should ensure that registry file has all the required color parameters when custom color scheme configured.
Appears: Java console
Level: 99
Trigger: None

FRM-92214: invalid value for color parameter %s. Forms will not use custom color scheme.
Cause: An invalid value was defined for color property in registry file.

Action: The system administrator should ensure that color parameters has valid values defined.
Appears: Java console
Level: 99
Trigger: None

FRM-92215: invalid value for color parameter %s. Parameter will be ignored.
Cause: An invalid value was defined for color property in registry file.

Action: The system administrator should ensure that color parameters has valid values defined.
Appears: Java console
Level: 99
Trigger: None

FRM-92220: access to system clipboard denied
Cause: The system clipboard is locked by some other application. This generally indicates a problem with the other application. Note: If the allowAlertClipboard applet parameter is set to 'false', this message appears only on the Java console.

Action: Determine which application is locking the system clipboard, and terminate it (or terminate the action that's locking the clipboard). Then, if possible, correct the application so that it does not erroneously lock the system clipboard.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92290: HTTP response code %s received when attempting to download archive file
Cause: An error occurred on the attempt to download the file specified in the preceding 'Downloading archive file' message. The Oracle Forms Standalone Launcher was terminated.

Action: Check network connectivity. For a response code of 404, also verify that the file exists. If the problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92291: Unsupported Java version detected. Contact system administrator or download a supported Java version from: %s
Cause: The java version on client machine is not allowed.

Action: Install a supported java release or have the system administrator change the value of the fsalJavaVersion applet parameter
Appears: Java console
Level: 99
Trigger: None

FRM-92292: Required parameter fsalJavaVersion not set or is invalid.
Cause: The fsalJavaVersion parameter is not set or the format is incorrect.

Action: Check for the parameter and the format of it.
Appears: Java console
Level: 99
Trigger: None

FRM-92293: Unsupported Java version detected. Contact system administrator.
Cause: The java version on client machine is not allowed.

Action: Install a supported java release or have the system administrator change the value of the fsalJavaVersion applet parameter
Appears: Java console
Level: 99

Trigger: None

FRM-92310: Wrong password, try again
Cause: Password entered for TrustStore was wrong.

Action: Try again with new password.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92311: Wrong password, retry limit exceeded
Cause: Retry limit for entering TrustStore password was exceeded.

Action: Try again by restarting the Oracle Forms Standalone Application.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-92312: Password can not be blank
Cause: An empty password was entered.

Action: Try again with proper password.
Level: 99
Trigger: None

FRM-92313: Password changes only permitted when running in a terminal window
Cause: System console for accepting password was not available.

Action: Try running the application from a command prompt/terminal where console is available.
Level: 99
Trigger: None

FRM-92314: Unable to load certificate details
Cause: Certificate details failed to load.

Action: Try running the FSAL application with switch -showDetails 2, to check the details of SSL/TLS certificate. Meanwhile contact Oracle Support Services.
Level: 99
Trigger: None

FRM-92410: EndUserMonitoring initialization has failed. Verify that %s (specified by the applet parameter %s) is a valid URL.
Cause: EndUserMonitoring initialization failed, probably due to an invalid URL specified by the specified applet parameter (typically the EndUserMonitoringURL parameter). The applet parameter was ignored; EndUserMonitoring was then disabled and execution continued.

Action: The system administrator should ensure that a valid URL was specified for the specified applet parameter. If that does not solve the problem, examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92411: EndUserMonitoring has failed, and will be disabled.
Cause: EndUserMonitoring was enabled, but a subsequent attempt to send a message to the EndUserMonitoring monitor encountered an unexpected Exception. EndUserMonitoring was then disabled, and execution continued.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92412: failure to send EndUserMonitoring data
Cause: EndUserMonitoring was enabled, but a subsequent attempt to send data to the EndUserMonitoring monitor was unsuccessful. Execution continued.

Action: If the problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92413: Update is needed, but automatic updating has been disabled by the administrator.
Cause: The checksum matching failed and autoupdate of Oracle Forms Standalone Launcher is disabled by the admin.

Action: Contact your system administrator to allow autoupdate of Oracle Forms Standalone Launcher.
Appears: Java console
Level: 99
Trigger: None

FRM-92420: could not find listener class %s
Cause: The class specified by the formsMessageListener applet parameter could not be found. The applet parameter was ignored, and execution continued.

Action: The system administrator should ensure that a valid class name was specified for the formsMessageListener applet parameter.
Appears: Java console
Level: 99
Trigger: None

FRM-92421: could not instantiate listener class %s
Cause: The class specified by the formsMessageListener applet parameter could not be instantiated. The applet parameter was ignored, and execution continued.

Action: The system administrator should ensure that a valid class name was specified for the formsMessageListener applet parameter.
Appears: Java console
Level: 99
Trigger: None

FRM-92422: could not initialize listener class %s
Cause: An unexpected Error was encountered while attempting to find and instantiate the class specified by the formsMessageListener applet parameter. The applet parameter was ignored, and execution continued.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92430: warning: invalid value %s ignored for parameter %s - defaulting to %s
Cause: The specified applet parameter (typically the asyncEventDelay parameter) did not specify a valid decimal number. A default value (5 seconds) was substituted, and execution continued.

Action: The system administrator should ensure that a valid decimal number was specified for the specified applet parameter.
Appears: Java console
Level: 99
Trigger: None

FRM-92431: warning: heartbeat or maxeventwait parameter value is less than 250ms
Cause: The value specified for the heartbeat or maxeventwait parameter specified a duration of under 250 milliseconds, which might cause problems.

Action: The system administrator should consider increasing the value to a more appropriate duration.
Appears: Java console
Level: 99
Trigger: None

FRM-92440: Thread %s has been interrupted while waiting for a message from the server.
Cause: The specified thread (in the Forms Java client) was interrupted while waiting for a message from the Forms server. This message identifies the thread, for diagnostic purposes. A fatal error subsequently occurs.

Action: No action is required for this message. (But action may be required for the subsequent fatal error).
Appears: Java console
Level: 99
Trigger: None

FRM-92450: Thread %s has been interrupted while waiting for a dialog to appear.
Cause: The specified thread was starting a dialog, and was interrupted while waiting for a dialog to appear. The wait was restarted.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92460: Thread %s has been interrupted while waiting for the LOV data fetching thread to die.
Cause: The specified thread had been fetching data for an LOV, but the end user accepted or canceled the LOV, so the thread requested a graceful death. But it was interrupted while the request was underway.

Action: No action is required.
Appears: Java console
Level: 99

Trigger: None

FRM-92470: unable to load image %s for image item
Cause: A requested image could not be loaded. A default image (indicating that the load failed) was substituted, and execution continued."

Action: If the error message specifies the name of the image file, verify that it exists, and is readable, and is in a valid image format.
Appears: Java console
Level: 99
Trigger: None

FRM-92471: unable to load image %s for iconic button item
Cause: A requested image could not be loaded. A default image (indicating that the load failed) was substituted, and execution continued."

Action: If the error message specifies the name of the image file, verify that it exists, and is readable, and is in a valid image format.
Appears: Java console
Level: 99
Trigger: None

FRM-92472: unable to load image %s for checkbox item
Cause: A requested image could not be loaded. A default image (indicating that the load failed) was substituted, and execution continued."

Action: If the error message specifies the name of the image file, verify that it exists, and is readable, and is in a valid image format.
Appears: Java console
Level: 99
Trigger: None

FRM-92473: unable to load image %s for radiobutton item
Cause: A requested image could not be loaded. A default image (indicating that the load failed) was substituted, and execution continued."

Action: If the error message specifies the name of the image file, verify that it exists, and is readable, and is in a valid image format.
Appears: Java console
Level: 99
Trigger: None

FRM-92474: unable to load image %s for alert
Cause: A requested image could not be loaded. A default image (indicating that the load failed) was substituted, and execution continued."

Action: If the error message specifies the name of the image file, verify that it exists, and is readable, and is in a valid image format.
Appears: Java console
Level: 99
Trigger: None

FRM-92480: Property %s: specified value has caused %s.
Cause: An attempt to set the value of an item property failed because the value was not of the proper type. The property was left unchanged, and execution continued. This error most commonly occurs when the application executes the SET_CUSTOM_PROPERTY built-in.

Action: Correct the value that is being passed to the SET_CUSTOM_PROPERTY built-in.
Appears: Java console
Level: 99
Trigger: None

FRM-92491: Unable to fetch archive file from server.
Cause: An Oracle Forms Standalone Application was attempting to fetch an archive file from the server but an error occurred.
It may fail due to following reasons:
1. Archive file does not exist.
2. Communication error.

Action: Ensure the following:
1. formsweb.cfg must have the parameter ARCHIVE defined.
2. Check network connectivity.
3. Change timeout value.
Appears: Java console
Level: 99
Trigger: None

FRM-92492: An erroneous applet definition containing <html> or <jnlp>, or with no configuration parameters, was received from the server. Its contents are shown below for diagnostic purposes:
Cause: The basesaafile configuration parameter apparently specified the wrong base text file, or raw HTML containing an error message from OHS or OAM was generated.

Action: Contact your system administrator.
Appears: Java console
Level: 99
Trigger: None

FRM-92493: Cannot open URL, java.awt.Desktop is not supported on this platform.
Cause: The java.awt.Desktop class is not supported on the platform.

Action: Run the Oracle Forms Standalone Application on a platform where the java.awt.Desktop class is supported.
Appears: Java console
Level: 99
Trigger: None

FRM-92494: Warning: Invalid timeout value. Continuing with default timeout value of %s milliseconds.
Cause: An invalid timeout value was encountered in the command line arguments for the Oracle Forms Standalone Launcher.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92495: Warning: Ignoring invalid archive URL %s.
Cause: The archive configuration or URL parameter specifies a single or a comma-separated list of values. The single value or one of the values was invalid, e.g. an absolute URL beginning with 'http:' or 'https:', but containing no '/'.

Action: Correct or remove the invalid archive URL.
Appears: Java console

Level: 99
Trigger: None

FRM-92496: error while generating checksum of Oracle Forms Standalone Launcher file
Cause: An I/O error occurred while generating the checksum.

Action: Check if Oracle Forms Standalone Launcher file exists.
Appears: Java console
Level: 99
Trigger: None

FRM-92497: Server returned HTTP response code %s
Cause: The Oracle Forms Standalone Launcher received an HTTP error response code when attempting to connect to the server, for example, 'Server returned HTTP response code 403--Forbidden'.

Action: Submit a valid request.
Appears: Java console
Level: 99
Trigger: None

FRM-92498: Warning: Invalid showDetails value. Continuing with default value of 0.
Cause: An invalid showDetails value was encountered in the command line arguments for the Oracle Forms Standalone Launcher.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92501: There is no active WebSocket session
Cause: An error occurred while trying to access the Session object.

Action: Initiate a WebSocket session.
Appears: Java console
Level: 99
Trigger: None

FRM-92502: An instance of WebSocket server is already running
Cause: An error occurred while starting a WebSocket server.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92503: Port number %s is in use. Please try a different port number.
Cause: An error occurred while trying to start WebSocket server on given port number.

Action: Provide a different port number.
Appears: Java console
Level: 99
Trigger: None

FRM-92504: A WebSocket session already exists
Cause: An error occurred while trying to initiate another WebSocket client session.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92505: Unable to initiate a WebSocket client session
Cause: An error occurred while trying to connect to the WebSocket server.

Action: Start WebSocket server if it has not yet started or check port number.
Appears: Java console
Level: 99
Trigger: None

FRM-92506: Given port number %s is invalid. Using default port number %s
Cause: An error occurred while trying to validate the given port number.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92507: WebSocket server is not running.
Cause: An error occurred while trying to access the WebSocket server object.

Action: Start WebSocket server.
Appears: Java console
Level: 99
Trigger: None

FRM-92508: Unable to download new Oracle Forms Standalone Launcher file.
Cause: A checksum mismatch error occurred while attempting to download new jar.

Action: A network interruption may have occurred. Try again or contact your administrator.
Appears: Java console
Level: 99
Trigger: None

FRM-92509: Unable to start the Oracle Forms Standalone Launcher process.
Cause: An error occurred while trying to start the Oracle Forms Standalone Launcher process.

Action: Try again and check file permissions.
Appears: Java console
Level: 99
Trigger: None

FRM-92510: Couldn't insert initial text.
Cause: An exception was thrown while attempting to render the UI dialog.

Action: Try again by relaunching the Oracle Forms Standalone Application and if problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92511: Unexpected error: %s
Cause: An unexpected exception was thrown while dealing with SSL/TLS.

Action: Try again by relaunching the Oracle Forms Standalone Application and if problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92512: Unable to load TrustStore
Cause: An exception was thrown while attempting to load or create a TrustStore file.

Action: Try again by relaunching the Oracle Forms Standalone Application and if problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92513: Access denied
Cause: The file did not have write permission.

Action: User should change the file permissions.
Appears: Java console
Level: 99
Trigger: None

FRM-92514: Cache directory %s not found or access to it failed
Cause: The directory was not present or did not have write permission.

Action: User should change the folder permissions.
Appears: Java console
Level: 99
Trigger: None

FRM-92522: forcing use of the native HTTP implementation
Cause: The useURLConnection applet parameter was specified as 'true' or 'yes'.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92523: forcing use of the Forms HTTP(S) implementation
Cause: The useURLConnection applet parameter was specified as 'false' or 'no'.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92530: error closing socket: %s
Cause: An unexpected Exception was encountered while attempting to close a socket. Execution continued.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92540: reallocating output buffer for native HTTP implementation
Cause: The native HTTP 'write' method was requested to write a block of data larger than its current output buffer. The buffer was reallocated, and execution continued.

Action: No action is required.
Appears: Java console
Level: 99
Trigger: None

FRM-92550: negative content-length on a response to a GET to URL %s
Cause: The Forms Java client issued a GET request to the Forms servlet. The result was a response with negative content-length. The fatal error FRM-92052 will subsequently appear.

Action: No action is required for this message. (But action may be required for the subsequent fatal error).
Appears: Java console
Level: 99
Trigger: None

FRM-92551: negative response (%s) to a read on behalf of a GET to URL %s
Cause: The Forms Java client issued a GET request to the Forms servlet. The result was a negative response. The fatal error FRM-92052 will subsequently appear.

Action: No action is required for this message. (But action may be required for the subsequent fatal error).
Appears: Java console
Level: 99
Trigger: None

FRM-92572: Forms HTTP connect has failed - giving up after %s attempts.
Cause: A Forms HTTP connect failed. The fatal error FRM-92050 (or possibly FRM-92060) will subsequently appear.

Action: No action is required for this message. (But action may be required for the subsequent fatal error).
Appears: Java console
Level: 99
Trigger: None

FRM-92574: Forms HTTP read has failed: %s
Cause: An unexpected Exception was encountered while attempting a native HTTP 'read'.

Action: Examine the stack trace that accompanies this message. If the stack trace indicates a possible cause, correct it. If the problem persists, contact Oracle Support Services.
Appears: Java console
Level: 99
Trigger: None

FRM-92575: SSL/TLS hostname verification of %s failed. Continuing at user''s request.
Cause: The user has requested to continue running (insecurely) even though SSL/TLS hostname verification failed.

Action: No action is required for this message but correcting the invalid SSL/TLS certificate is strongly encouraged.
Appears: Java console
Level: 99

Trigger: None

FRM-92576: SSL/TLS hostname verification has been disabled at user''s request.
Cause: The user specified the option "-bypassHostnameVerification true" when invoking the Forms Standalone Launcher.

Action: No action is required for this message, but specifying this option in a production system is strongly discouraged.
Appears: Java console
Level: 99
Trigger: None

FRM-93110: No Forms Servlet configuration file is specified.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93111: Cannot find Forms Servlet configuration file %s.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93112: error reading Forms Servlet configuration file %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93114: Forms Servlet configuration file %s contains invalid data.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93121: Cannot find environment variable configuration file %s.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93122: error reading environment variable configuration file %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93124: Environment variable configuration file %s contains invalid data.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93130: No base HTML file is specified.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93131: Cannot find base HTML file %s.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93132: error reading base HTML or JNLP file %s
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93134: No base JNLP file is specified.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93135: Cannot find base JNLP file %s.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93136: no base TXT file specified for Oracle Forms Standalone Application
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93143: Oracle Forms Standalone Launcher checksum failed.
Cause: The checksum received from the client did not match the Oracle Forms Standalone Launcher jar.

Action: Contact your system administrator or get the latest Oracle Forms Standalone Launcher file.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93144: error while generating checksum of Oracle Forms Standalone Launcher file
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93146: Oracle Forms Standalone Launcher version does not match server version %s.
Cause: Oracle Forms Standalone Launcher jar is outdated.

Action: Get the latest Oracle Forms Standalone Launcher file.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93150: The following restricted parameters cannot be specified in the URL: %s
Cause: The indicated parameters were specified by the end user (in a Forms Servlet URL).

Action: Remove the indicated parameters from the URL and resubmit.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93151: Restricted characters cannot be specified in the URL
Cause: Invalid characters were specified by the end user (in a Forms Servlet URL).

Action: Remove the invalid characters from the URL and resubmit.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93154: The Forms Servlet is not allowing new connections.
Cause: The system administrator has disabled new connections to the Forms Servlet.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93156: The Forms Servlet is unable to contact the Forms Load Balancing Server at %s:%s.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93218: fatal error reading client request content
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93240: Multiple Forms applications cannot share an HTTP session.
Cause: Either (1) the user selected File, then New, then Browser Window (or Ctrl+N) in Internet Explorer while the servlet session was being tracked using cookies, or else (2) the Forms application was configured incorrectly.

Action: In case (1), select File, then New, then Browser Window (or Ctrl+N) in an Internet Explorer window that is not running a Forms application, or start a second instance of Internet Explorer. Otherwise, contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93242: HTTP session not found on target server, probably because it has already been used.
Cause: You probably re-submitted a downloaded JNLP file that was created for Java Web Start with SSO enabled, or with the jnlpMatchIP configuration parameter set to true, or with the jnlpTimeout configuration parameter set to a non-zero value. Such JNLP files can be used only once.

Action: If you re-submitted a downloaded "one-shot" JNLP file, resubmit the original URL (that generated the JNLP file) from the browser. If the error occurred when submitting a URL from the browser, contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93243: Static HTML is not allowed.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93245: HTTP session not found on target server; possible load balancer configuration issue.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93248: connection to the server apparently closed after a request had executed for %s seconds, probably due to an intermediate agent timing out
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93249: launch of browser window to be used for SSO authentication of an Oracle Forms Standalone application timed out after %s seconds
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93260: JNLP file has expired.
Cause: There was excessive delay launching a JNP file after the Web Start request that created it.

Action: Retry the Web Start request.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93261: JNLP file launched from unexpected IP address
Cause: The JNLP file was not launched from the IP address from which it was created (by a Web Start request).

Action: Launch the JNLP file from the IP address from which it was created.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93301: Fatal authentication error: Unable to connect to Oracle Internet Directory.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93302: error running Forms in SSO mode.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93320: unable to obtain application entity credential from Oracle Platform Security Services
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93330: Fatal authentication error: User does not have proper credentials configured.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93331: Fatal authentication error: Invalid Single Sign-On User.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93340: Session requires SSO user authentication.
Cause: The session was not SSO-authenticated or had expired.

Action: Re-login using SSO credentials.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93364: Cannot dynamically create resource in Oracle Internet Directory: URL specifies invalid value "%s" for the config parameter.
Cause: An attempt to dynamically create an SSO resource failed because the user specified a nonexistent configuration section.

Action: Specify the name of a valid configuration section in the base configuration file as the value of the config parameter.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93365: error running Forms in SSO mode
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93366: error creating Forms application's Resource Access Descriptor in Oracle Internet Directory.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93367: invalid parameters %s passed to the Forms Rad Servlet
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93368: Forms Rad Servlet is in an invalid state.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93369: Direct user access of Forms RadServlet is not allowed.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93370: error reading the Resouce Access Descriptor for update
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.

Appears: Java console, alert
Level: 99
Trigger: None

FRM-93380: unexpected error while attempting to redirect to Oracle Access Manager or the Forms Rad servlet
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93382: Oracle Access Manager login processing for an Oracle Forms Standalone application timed out after %s seconds
Cause: Oracle Access Manager login was not completed within the indicated time interval.

Action: If you aborted the application by closing the page, or intended to abort the application by closing the page, no action is required. If you completed the login after receiving this error message, the application was subsequently aborted, but you can retry launching it. [The login page won't be presented because you are already logged in]. If you intended to complete the login, retry launching the application and redoing the login within the indicated time interval. If that interval is insufficient, contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93383: Resource Access Descriptor processing for an Oracle Forms Standalone application timed out after %s seconds
Cause: Submit or Cancel was not pressed in the Resource Access Descriptor page within the indicated time interval.

Action: If you aborted the application by closing the page, or intended to abort the application by closing the page or pressing Cancel, no action is required. If you intended to enter data (pressing Submit to complete the action), retry launching the application and reentering the data within the indicated time interval. If that interval is insufficient, contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93384: SSO login for an Oracle Forms Standalone application completed, but the application was terminated.
Cause: This message may appear in a browser window which is performing SSO authentication for a Standalone application if you completed the SSO login after the SSO login page timed out or an error occurred in the startup process. A more specific message should appear in the terminal window from which the Standalone application was launched.

Action: Refer to the message in the terminal window from which the Standalone application was launched. If no such message appears or the Forms session is not being run as a Standalone application, and the problem persists, contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93385: Resource Access Descriptor processing for an Oracle Forms Standalone application timed out, or was not performed due to an error in the startup process.
Cause: This message may appear in a browser window which is performing SSO authentication for a Standalone application if you pressed pressed Submit or Cancel after RAD processing timed out or an error occurred in the startup process. A more specific message should appear in the terminal window from which the Standalone application was launched.

Action: Refer to the message in the terminal window from which the Standalone application was launched. If the Forms session is not being run as a Standalone application, and the problem persists, contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93500: unexpected error while attempting to create the runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93520: Runtime process not created: Maximum permissible number of runtime processes is exceeded.
Cause: The number of currently executing runtime processes has reached the limit that was set by the system administrator.

Action: Retry the application when the system is less heavily loaded.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93531: cannot create runtime process: unable to switch to working directory
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93535: cannot create runtime process: unable to execute startup command
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93536: cannot create runtime process: unable to create temporary logging file
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.

Appears: Java console, alert
Level: 99
Trigger: None

FRM-93543: cannot connect to runtime process: unable to get I/O streams from newly created runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93550: cannot connect to runtime process: no response from newly created runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93552: cannot connect to runtime process: Newly created runtime process has terminated abnormally.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93553: cannot connect to runtime process: unable to establish a socket connection to newly created runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93558: cannot connect to runtime process: error reading data from newly created runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93600: unexpected error while attempting to communicate with the client or the runtime process or an external service
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93618: fatal error reading data from runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93628: fatal error writing data to runtime process
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93652: The runtime process has terminated abnormally.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93830: Test cookie set. Details:%s
Cause: This is the normal output of the setcookie and setcookiesess commands.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93832: Found test cookie: %s=%s
Cause: The test cookie was found.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93834: Test cookie not found.
Cause: The test cookie was not found.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93840: Proctest: Warning: nProcs configuration parameter is nonnumeric - using 1.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93841: Proctest: Process test starting with %s processes.
Cause: The Proctest command started.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93842: Proctest: Creating process %s.
Cause: The Proctest command started creating a runtime process.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93843: Proctest: Failure to create process %s - aborting test.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93844: Proctest: Connecting to process %s.
Cause: The Proctest command started connecting to a newly created runtime process.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93845: Proctest: Failure to connect to process %s - aborting test.
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93846: Proctest: unexpected error in process test
Cause: A fatal error occurred in the Forms server, which will require the attention of your system administrator.

Action: Contact your system administrator.

Appears: Java console, alert
Level: 99
Trigger: None

FRM-93847: Proctest: Number of processes started and connected to is %s.
Cause: The Proctest command created the requested number of runtime processes, and successfully connected to them.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93848: Proctest: Stopping the %s runtime processes.
Cause: The Proctest command reached the point in its processing where it was about to stop the runtime processes that it had previously created.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93849: Proctest: Process test complete.
Cause: The Proctest command completed.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93860: em_result=%s|000
Cause: This is the normal output of the status command.

Action: No action is required.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93890: Trace: invalid Xlate parameter %s
Cause: The URL that contains the trace command also specifies a parameter that is not recognized by the Xlate utlity.

Action: Correct the URL.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93891: Trace: outputClass parameter value "%s" for Xlate cannot contain a ".".
Cause: The URL that contains the trace command specifies an invalid value for the outputClass parameter.

Action: Correct the URL that specifies the invalid parameter value.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93892: Trace: PID parameter value "%s" for Xlate cannot contain a "%s".
Cause: The URL that contains the trace command specifies an invalid value for the PID parameter.

Action: Correct the URL that specifies the invalid parameter value.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-93893: Trace: Xlate encountered an IO error
Cause: The Xlate utility encountered an IOException.

Action: Ensure that a valid PID parameter is specified.
Appears: Java console, alert
Level: 99
Trigger: None

FRM-99999: Error %s occurred. See the release notes file (relnotes) for information about this error.
Cause: An error occurred; the error is documented in the release notes file.

Action: See the release notes file.
Level: 25
Trigger: None

Forms Java EE Application Deployment Descriptors

The Forms Services Java EE application EAR (Enterprise Archive) file `formsapp.ear` is deployed to the WLS_FORMS (Oracle WebLogic Managed Server) when you configure Oracle Forms.

This results in the creation of a directory structure under `$DOMAIN_HOME /servers/WLS_FORMS/tmp/_WL_user/formsapp_12.2.1/<random_string1>/APP-INF` directory that is similar to the following:

```
./APP-INF
./APP-INF/lib
./APP-INF/lib/frmconfig.jar
./APP-INF/lib/frmconfigmbeans.jar
./META-INF
./META-INF/application.xml
./META-INF/jazn-data.xml
./META-INF/jps-config.xml
./META-INF/mbeans.xml
./META-INF/weblogic-application.xml
```

This following directory structure is created under `$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_12.2.1/<random_string2>/war/WEB-INF` directory.

```
./WEB-INF
./WEB-INF/lib
./WEB-INF/lib/frmsrv.jar
./WEB-INF/web.xml
```

```
./WEB-INF/weblogic.xml
```

> **Note:**
>
> The sub-directories in `$DOMAIN_HOME/servers/WLS_FORMS/tmp/_WL_user/formsapp_12.2.1` are created by the nostage deployment process of Oracle WebLogic Server. They are named with a random string. For example, e18uoi, wb1h9e and so on.

Deployment descriptors:

- `application.xml` and `weblogic-application.xml` define the structure of the EAR file.
- `web.xml` defines the aliases `frmservlet` and `lservlet` for the Forms servlet and the Forms Listener servlet.
- `weblogic.xml` defines the context parameters and any user defined virtual directory mappings.