

Oracle® Fusion Middleware

Upgrading Oracle Forms 6i to Oracle Forms 12c

12c (12.2.1.1)

E71740-01

June 2016

This document explains how to upgrade your Oracle Forms applications to Oracle Forms 12c.

Oracle Fusion Middleware Upgrading Oracle Forms 6i to Oracle Forms 12c, 12c (12.2.1.1)

E71740-01

Copyright © 2015, 2016, Oracle and/or its affiliates. All rights reserved.

Primary Author: Arup Roy

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

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

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

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface	vii
Intended Audience.....	vii
Documentation Accessibility	vii
Related Documents	vii
Conventions	viii
1 Why Upgrade?	
1.1 Overview	1-1
1.2 About Upgrading Your Forms Applications to 12c Release (12.2.1.1).....	1-1
1.3 Forms 10g Features Removed from Oracle Forms.....	1-2
1.4 Forms 6i Features Removed from Oracle Forms.....	1-2
1.5 Components of 6i Developer Product Suite Removed.....	1-2
1.6 How are Obsolete Item Types Handled when Upgrading Forms 6i Applications?	1-3
1.7 Are Tools Available to Assist with the Migration of Obsolete Features?.....	1-4
2 Using the Oracle Forms Migration Assistant	
2.1 What Does the Oracle Forms Migration Assistant Do?.....	2-1
2.1.1 Multiple Log Support.....	2-2
2.2 Editing the converter.properties File	2-2
2.3 Editing the search_replace.properties File	2-3
2.3.1 Adding Search-and-Replace Strings	2-3
2.3.2 Modifying Warnings for Obsolete Built-Ins	2-3
2.4 Starting the Oracle Forms Migration Assistant	2-4
2.4.1 About the Migration Assistant in Batch Mode.....	2-5
2.4.2 Starting the Migration Assistant in Batch Mode	2-6
2.4.2.1 Starting the Migration Assistant in batch mode in Windows.....	2-6
2.4.2.2 Starting the Migration Assistant in Batch Mode in UNIX.....	2-6
2.4.3 Running the Wizard Version of the Forms Migration Assistant	2-6
2.4.3.1 Starting the Wizard Version of the Forms Migration Assistant	2-6
2.4.3.2 Setting Advanced Converter Options	2-7
3 Converting Forms 6i FMTs to Oracle Forms FMBs	
3.1 Converting a Forms 6i FMT to an Oracle Forms FMB	3-1

4	Built-ins, Packages, Constants, and Syntax	
4.1	Obsolete Menu Built-ins	4-1
4.2	Other Obsolete Built-ins.....	4-2
4.3	Obsolete Built-in Packages	4-4
4.4	Obsolete Constants	4-4
4.5	Obsolete Syntax.....	4-5
5	Triggers	
5.1	Obsolete Triggers	5-1
5.2	Stricter Enforcement of Triggers.....	5-1
6	Properties	
6.1	Obsolete Properties.....	6-1
7	Changes to Client/Server Deployment and Forms Runtime	
7.1	Overview	7-1
7.2	Effect on Forms Development.....	7-1
7.3	Obsolete Forms Runtime Command Line Options	7-1
7.4	Obsolete Character Mode Runtime	7-2
8	Item Types	
8.1	Obsolete Item Types	8-1
8.2	Item Types Specific to Operating Systems	8-1
9	Logical and GUI Attributes	
9.1	Use Visual Attributes to Replace Logical and GUI Attributes.....	9-1
9.2	Obsolete Logical and GUI Attributes.....	9-1
10	List of Values (LOVs)	
10.1	Obsolete List of Values (LOVs).....	10-1
11	User Exits	
11.1	Obsolete V2 User Exits	11-1
12	Menu Parameters	
12.1	Predefined Menu Parameters.....	12-1
12.2	User-Defined Menu Parameters	12-1
13	Java-Related Issues	
13.1	Using Pluggable Java Components and Other Custom Java	13-1
13.2	JDK Versions and Font-Rendering Issues	13-1

14 Integration with Oracle Reports

14.1	Overview	14-1
14.2	About Integration with Oracle Reports	14-1
14.2.1	Displaying Reports in Oracle Forms.....	14-1
14.2.2	Example.....	14-2
14.2.3	Notes About the Example.....	14-3
14.2.4	Using Parameter Lists in RUN_REPORT_OBJECT	14-3
14.2.5	Upgrade Steps	14-4

15 Upgrading Client/Server Applications to the Web

15.1	Guidelines for Upgrade	15-1
15.2	About Client/Server-Based Architecture.....	15-2
15.3	About Web-Based Architecture	15-3

16 Upgrading from Pre-Forms 6i Applications to Oracle Forms

16.1	Upgrading a Form	16-1
16.2	Upgrading files saved in Database.....	16-1
16.3	Compatibility with Earlier Versions of PL/SQL.....	16-2
16.4	Forms Developer Runtime Behavior.....	16-2

Preface

Welcome! This manual describes:

- Features and functionality that have been removed from Forms Developer and Forms Services
- Information about upgrade events that automatically occur when you open or deploy a Forms 6i application in Oracle Forms 12c.
- Information about the Oracle Forms Migration Assistant, a tool to help you convert your applications.
- Information about steps that developers, system administrators, and DBAs need to take to upgrade Forms applications from Forms 6i to Oracle Forms 12c.

Intended Audience

This manual is intended for developers, system administrators, and DBAs who develop and deploy Oracle Forms applications.

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 customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Related Documents

For more information, see the following manuals and online help:

- *Oracle Fusion Middleware Release Notes for Linux x86*
- *Oracle Fusion Middleware Forms Services Deployment Guide*
- Oracle Forms Developer Online Help, available from the Help menu in Oracle Forms Developer.

In addition, you will find white papers and other resources at <http://www.oracle.com/technetwork/developer-tools/forms/documentation/index.html>.

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.
<i>italic</i>	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.

Why Upgrade?

This chapter contains the following sections:

- [Section 1.1, "Overview"](#)
- [Section 1.2, "About Upgrading Your Forms Applications to 12c Release \(12.2.1.1\)"](#)
- [Section 1.3, "Forms 10g Features Removed from Oracle Forms"](#)
- [Section 1.4, "Forms 6i Features Removed from Oracle Forms"](#)
- [Section 1.5, "Components of 6i Developer Product Suite Removed"](#)
- [Section 1.6, "How are Obsolete Item Types Handled when Upgrading Forms 6i Applications?"](#)
- [Section 1.7, "Are Tools Available to Assist with the Migration of Obsolete Features?"](#)

1.1 Overview

Oracle Forms Builder and Oracle Forms Services have been upgraded to simplify the development and deployment of Forms applications on the Web. A number of features have been added. In restructuring the Oracle Forms product, some legacy features have been dropped or scaled back. The future of Forms includes improving the Java-based Web user interface and extending product "openness" by allowing Java integration on all three tiers.

1.2 About Upgrading Your Forms Applications to 12c Release (12.2.1.1)

To upgrade your Oracle Forms applications, open your Forms *6i* source files (FMB, MMB, PLL, and so on) in the Oracle Forms 12c (12.2.1.1) Builder, save them, then compile them. You can also use the Oracle Forms Compiler to upgrade your Forms *6i* applications.

You can use the Forms Migration Assistant to perform batch upgrades. For more information, see [Chapter 2, "Using the Oracle Forms Migration Assistant"](#).

Note: You must compile `rp2rro.pll` into `rp2rro.plx` if your Form depends on this library. `rp2rro.pll` can be found in `ORACLE_HOME/forms/rp2rro.pll`. The resulting `plx` should be in your `FORMS_PATH`.

1.3 Forms 10g Features Removed from Oracle Forms

The following features have been dropped from Oracle Forms 10g and later:

- Graphics Integration
- Chart Items

1.4 Forms 6i Features Removed from Oracle Forms

The following features have been dropped from Oracle Forms 9.0.2 and later:

- Client-server Runtime
- Character mode Runtime
- Various runform command line options
- Character mode properties and logical attributes
- Item types that are specific to operating systems
- Various Built-ins
- Various properties
- Various menu features including:
 - Character mode menu properties
 - Obsolete types from the Menu-Items command type property
 - Menu parameters
 - Menu Built-ins
 - Full screen menu style
 - Bar (Lotus) menu style
- Forms version 2 style triggers and list of values (LOVs)
- Graphics Chart Wizard

In addition, rules enforcing trigger usage have become stricter.

1.5 Components of 6i Developer Product Suite Removed

The following components have been removed.

Table 1–1 Components Removed from the Developer Suite

Obsolete Component	Upgrade Notes
Oracle Graphics	If your applications use the Graphics Web Cartridge or Oracle Graphics Runtime, you should rewrite your applications and redevelop the graphics using other means such as Java, Bi Beans. For more information, see the demos at http://www.oracle.com/technetwork/developer-tools/forms/downloads .

Table 1–1 (Cont.) Components Removed from the Developer Suite

Obsolete Component	Upgrade Notes
Oracle Forms Listener and Load Balancing Components	Use the Forms Listener Servlet to manage Forms sessions on the Web. The Forms Listener Servlet provides: <ul style="list-style-type: none"> ■ Improved security because all traffic is directed through standard Web server HTTP or HTTPS ports, with no extra ports open through the firewall. ■ Compliance with standards that can be used for load balancing techniques. ■ Broader firewall and proxy support. ■ Less administration because the listener and load balancing processes do not need to be managed. ■ Simplified HTTPS support because a separate Web server SSL certificate for the Forms listener is not required.
Oracle Forms Server Cartridge and CGI	Use the Forms Servlet. The functionality available with the Oracle Forms Server cartridge and CGI was incorporated into the Forms Servlet, which was first available in Oracle Forms Release 6i patchset 2.
Oracle Procedure Builder	Use the facilities for editing and debugging local and server-side PL/SQL code in Forms Developer, which has been considerably improved for this release.
Oracle Project Builder	No upgrade path or replacement functionality.
Oracle Translation Builder	Use TranslationHub to translate resource strings in Forms modules to deploy the modules in multiple languages.
Oracle Query Builder/Schema Builder	No upgrade path or replacement functionality.
Oracle Terminal	The resource files used by Web-deployed forms are text based and can be edited using a conventional text editor. As a result, Oracle Terminal is no longer required for the product.
Open Client Adapters (OCA)	In order to provide platform-independent access to a wider range of non-Oracle data sources, use the Oracle Transparent Gateway and Generic Connectivity solutions instead of OCA.
Tuxedo Integration	No upgrade path or replacement functionality.
Performance Event Collection Services (PECS)	No upgrade path. Use Forms Trace and Oracle Trace, which are documented in the book <i>Oracle Fusion Middleware Forms Services Deployment Guide</i> .

1.6 How are Obsolete Item Types Handled when Upgrading Forms 6i Applications?

When you open a Forms application, obsolete item types are listed after the item-type poplist in the Property Palette. The property values for obsolete items are indicated as obsolete. For example, the property value for VBX would be "VBX Control (Obsolete)".

Note: If you are upgrading from pre-6i you must upgrade to 10g and then to 12c. If you are upgrading from 6i or later, you can go directly to 12c. See [Chapter 16, "Upgrading from Pre-Forms 6i Applications to Oracle Forms"](#) for more information.

You can use the Oracle Forms Migration Assistant, described in [Chapter 2, "Using the Oracle Forms Migration Assistant"](#), to resolve many upgrade issues.

1.7 Are Tools Available to Assist with the Migration of Obsolete Features?

The Oracle Forms Migration Assistant is provided with Oracle Forms to help you upgrade your Forms 6*i* applications. See [Chapter 2, "Using the Oracle Forms Migration Assistant"](#) for details.

Using the Oracle Forms Migration Assistant

This chapter contains the following sections:

- [Section 2.1, "What Does the Oracle Forms Migration Assistant Do?"](#)
- [Section 2.2, "Editing the converter.properties File"](#)
- [Section 2.3, "Editing the search_replace.properties File"](#)
- [Section 2.4, "Starting the Oracle Forms Migration Assistant"](#)

2.1 What Does the Oracle Forms Migration Assistant Do?

The Oracle Forms Migration Assistant updates obsolete usage in your PL/SQL code to upgrade your Forms *6i* applications to Oracle Forms 12c. The tool issues warnings when it cannot make the required changes automatically. This tool has a command line and a wizard version. The Oracle Forms Migration Assistant does the following for all Forms module types (including object libraries and PL/SQL libraries):

- Updates PL/SQL code where possible, for example:
 - Updates RUN_PRODUCT to the RUN_REPORT_OBJECT built-in when used to call Reports.
 - Updates CHANGE_ALERT_MESSAGE to the SET_ALERT_PROPERTY built-in.
- Provides a list of obsolete code usage, including code that the tool cannot change when there is not a straight-forward equivalent for upgrade, for example:
 - Provides warnings when specific obsolete built-ins are used at run time, such as ITEM_ENABLED.

Note: The Oracle Forms Migration Assistant replaces built-ins and issues warnings about built-ins that exist within code comments.

- Provides warnings when obsolete functionality is encountered, such as when obsolete item types are included in the code.
- Provides warnings about triggers defined at incorrect levels.
- Replace simple one-for-one code strings such as OHOST to HOST, MENU_CLEAR_FIELD to CLEAR_ITEM, and MENU_FAILURE to FORM_FAILURE.
- Performs more complex substitutions such as CHANGE_ALERT_MESSAGE to SET_ALERT_PROPERTY built-in, DISABLE_ITEM to

SET_MENU_ITEM_PROPERTY built-in, ITEM_ENABLED to GET_ITEM_PROPERTY built-in, and ENABLE_ITEM to SET_MENU_ITEM_PROPERTY built-in.

- Raises a warning in the Form module’s log if V2-style triggers are found.
- Raises a warning in the Form module’s log if built-ins related to obsolete object types are found such as VBX.FIRE_EVENT, VBX.GET_PROPERTY, and VBX.GET_VALUE_PROPERTY.

You run the Oracle Forms Migration Assistant in batch mode. You can again enter the utility as needed to run the upgrade process on a Forms application more than once. You can also use the wizard version of the Migration Assistant to upgrade multiple modules.

By editing the converter.properties file, you can set options before you start the batch migration. By editing the search_replace.properties file you can specify the strings that the Oracle Forms Migration Assistant searches for and replaces, and edit the warnings that are issued when an obsolete built-ins is encountered.

The tool creates a log file so that you can navigate to problem areas in the application and make modifications manually.

2.1.1 Multiple Log Support

The Forms Migration Assistant allows you either to write all log information into a single log file or span multiple log files. If the Forms Migration Assistant spans multiple log files, the Forms Migration Assistant generates individual log files for each module that is processed.

You must specify the directory in which the Forms Migration Assistant writes the log files. The name of the log file which is generated is `modulename_moduletype.log`.

For example, if you process a module by name, such as `test.fmb`, the name of the log file is `test_fmb.log`. If you select `test.fmb` and `test.mmb` for upgrade, the Forms Migration Wizard does not overwrite the log files. However, if you converted two modules `test.fmb` from two different directories, the log files are overwritten. The Forms Migration Assistant generates two log files: `test_fmb.log` and `test_mmb.log`.

2.2 Editing the converter.properties File

To change upgrade options, edit the `converter.properties` file in a text editor. You can set the following upgrade options:

Table 2–1 Oracle Forms Migration Assistant converter.properties File Options

Option	Description
Log File Name (default.logfilename)	Specifies the file name and location for log information.
Reports Queue Table Installed (default.usequeuetables)	When using web-based reports, these queue tables helps to monitor queued and processed reports. When used with the Oracle Forms Migration Assistant, the queue table provides detailed error messages when installed in the application schema. (For example, if a report cannot run due to uncompiled PL/SQL, you can use the queue table to query for full error messages.) The resulting report is automatically printed. See the Reports Services documentation for more information about the Reports Queue Table.

Table 2–1 (Cont.) Oracle Forms Migration Assistant converter.properties File Options

Option	Description
Reports Servlet Directory (default.servletdir)	Specifies the name that has been defined for the virtual path used for the Reports Servlet, which is used for running reports on the web. This setting is required when converting Run_Product calls to Run_Report_Object.
Reports Servlet Name (default.servletname)	Specifies the name for the Reports Servlet used for running reports on the web. This setting is required when converting Run_Product calls to Run_Report_Object.
Reports Server Host (default.reports_servername)	The name or IP address of the system running the Reports Server. This setting is required when converting Run_Product calls to Run_Report_Object.
DESTYPE (default.destype)	The type of destination device that receives the report output. See the <i>Oracle Fusion Middleware Reports Services Publishing Reports to the Web</i> for details.
DESFORMAT (default.desformat)	The printer driver to be used when DESTYPE is FILE. See the <i>Oracle Fusion Middleware Reports Services Publishing Reports to the Web</i> for details.
DESNAME (default.desname)	The name of the file, printer, e-mail ID, or distribution list to which the report output is sent. See the <i>Oracle Fusion Middleware Reports Services Publishing Reports to the Web</i> for details.
Reports Server Host (default.reportshost)	The name or IP address of the system running the Reports Server. This setting is required when converting Run_Product calls to Run_Report_Object.

2.3 Editing the search_replace.properties File

The search_replace.properties file contains the strings that the Oracle Forms Migration Assistant searches for and replaces. It also contains a list of obsolete built-ins that generate warnings.

2.3.1 Adding Search-and-Replace Strings

You can edit this file to add your own search-and-replace strings, as follows:

1. Open the search_replace.properties file in a text editor.
2. Go to the end of the list of search-and-replace strings.
3. Add a string to search for and replace using the following syntax:

```
SearchString|ReplaceString
```

4. Save the search_replace.properties file.

2.3.2 Modifying Warnings for Obsolete Built-Ins

The warnings for built-ins have the following syntax:

```
<class>.Message=<WarningMessage>
<class>.Warning1=<BuiltIn1>
<class>.Warning2=<BuiltIn2>
<class>.Warning3=<BuiltIn3>
etc.
```

For example, for the class `obsoleteMenuParam`, the warnings are coded as follows:

```
obsoleteMenuParam.Message=Menu Parameters are no longer supported, the parameter
and usage of %s should be replaced using a Forms parameter or global variable.
obsoleteMenuParam.Warning1=MENU_PARAMETER
obsoleteMenuParam.Warning2=QUERY_PARAMETER
obsoleteMenuParam.Warning3=TERMINATE
```

`<class>` is a group of built-ins that have a common warning. `<WarningMessage>` can contain one variable string (`%s`).

When the Oracle Forms Migration Assistant finds a built-in for which a warning should be issued, it logs the warning and replaces the variable string (`%s`) with the built-in name.

Classes included in the `search_replace.properties` file are:

- `obsoleteItemTypeBuiltin`
- `obsoleteBuiltin`
- `obsoleteMenuParam`
- `obsoleteItemTypeConstantProp`
- `obsoleteConstantProp`
- `obsoleteConstant`
- `obsoleteHardCodedUserExit`
- `obsoleteComplexBuiltin`
- `DataParameterWithReports`
- `NoErrorOrWarningFromForms`

You can add more built-in warnings to an existing class, or create classes and warnings:

1. Open the `search_replace.properties` file in a text editor.
2. Go to the end of the list of warning messages.
3. Add a warning to an existing class or create classes and warnings using the syntax described.
4. Save the `search_replace.properties` file.

Note: Do not delete the last two commands in the `search_replace.properties` file.

2.4 Starting the Oracle Forms Migration Assistant

You should upgrade the common modules that a form relies upon and, ensure that these are available in the `FORMS_PATH`, before running the migration assistant. The following should be upgraded in this order using the FMA:

- `olb`
- `.pll`
- `.mmb`
- `.fmb`

Note: For UNIX, an xterm display is required to run the Oracle Forms Migration Assistant.

Note: The forms\rp2rro.pll and forms\EnableDisableItem.pll files should be in the FORMS_PATH to convert RUN_PRODUCT to RUN_REPORT_OBJECT.

In UNIX, set the variable FORMS_PATH, such as
 setenv FORMS_PATH \$ORACLE_HOME/forms.

Table 2–2 Oracle Forms Migration Assistant Command Line Parameters

Parameter	Description
module (required in batch mode only)	Specifies the module to upgrade. The module name parameter can take only one value. For more information on upgrading multiple modules at a time, see "Running the Wizard Version of the Forms Migration Assistant" .
log (optional)	Specifies the log file to which the upgrade results are written. If not specified, the default values from the converter.properties file are used.
mode (optional)	This option has two values (batch and wizard). Use mode=batch to run the Migration Assistant in Batch mode, mode=wizard to run the wizard.

Information about the progress of the upgrade process is displayed on the screen. It is also saved to the log file that you specify in the converter.properties file. (To change upgrade options, see [Editing the converter.properties File](#).)

Check the log file for information about required upgrade steps that the tool did not modify. Manually make these changes to your application.

2.4.1 About the Migration Assistant in Batch Mode

You can convert multiple applications by running the Forms Migration Assistant in batch mode from the command line. Batch mode is useful for converting multiple Forms applications. For example, on Windows, create a batch file (for example upgrade.bat) that contains the following:

```
for %%f in (%1) do frmplsqlconv module=%%f userid=<connect_string>
```

Then run the batch file as follows:

```
upgrade *.fmb
```

or

```
upgrade foo*.mmb
```

On UNIX, create a shell script (for example upgrade.sh) that contains the following:

```
for file in $*
do
    frmplsqlconv.sh module=$ff
done
```

Then run the shell script as follows:

or

```
upgrade.sh foo*.mmb
```

2.4.2 Starting the Migration Assistant in Batch Mode

You can run the Migration Assistant in batch mode on Windows and UNIX computers.

2.4.2.1 Starting the Migration Assistant in batch mode in Windows

In a Command window, issue the following command:

```
frmplsqlconv mode=batch module=<filename> log=<logname>
```

Note: A dialog window appears recommending that you backup your files before converting them. Select the **Show me this again** box to enable this warning in the future.

where <filename> is the name of the file you want to convert, and <logname> is the name of the log file that is generated.

The Forms Migration Assistant looks for the file `test.fmb` in the `d:\temp` directory and names the generated log file `test.log` and places it in the `d:\temp` directory. You can give the log file any name you want and specify any location you want. By default the log file is written in `ORACLE_HOME\bin`. The module parameter can take only one value.

In the following example,

```
frmplsqlconv mode=batch module=d:\temp\test.fmb log=d:\temp\test.log
```

the output of the conversion is displayed on the screen. It is also saved to the default log file. Notice that in batch mode, all converter options are taken from the `converter.properties` file. To change the converter options, see ["Editing the converter.properties File"](#).

2.4.2.2 Starting the Migration Assistant in Batch Mode in UNIX

In a UNIX shell, issue the following command:

```
frmplsqlconv.sh mode=batch module=<filename> log=<logname>
```

The output of the conversion is displayed on the screen. It is also saved to the default log file. Notice that in batch mode, all converter options are taken from the `converter.properties` file. To change the converter options, see ["Editing the converter.properties File"](#).

2.4.3 Running the Wizard Version of the Forms Migration Assistant

You can run the wizard version of the Forms Migration Assistant and edit converter options.

2.4.3.1 Starting the Wizard Version of the Forms Migration Assistant

This section describes how to start the Forms Migration Assistant on Windows and UNIX computers.

1. On Windows computers, set the `FORMS_PATH` environment variable:

```
SET FORMS_PATH=%ORACLE_HOME%\forms
```

On UNIX computers, set the variable FORMS_PATH as in the following example:

```
setenv FORMS_PATH $ORACLE_HOME/forms
```

2. On Windows computers, from the Start menu, select **ORACLE_HOME | Forms Developer | Oracle Forms Migration Assistant (GUI Mode)**, or at the command line, start the conversion utility by entering: `frmp1sqlconv.bat mode=wizard`.
in UNIX, enter `frmp1sqlconv.sh mode=wizard`.

The Conversion Wizard **Welcome** dialog displays.

Note: You can get help by typing `-h` after the command, for example, `frmp1sqlconv -h`

3. Click **Next**.
4. In the **Modules** dialog, click the **Add Module(s)** button.
5. Select the modules that you want to convert.
6. Click **Next**.
7. In the **Converter options** dialog, enter the location and name of the log file that is generated. A browse button is provided to assist in choosing a location.
8. If you have Forms that include embedded reports, fill in the remaining fields. Otherwise, you may ignore these fields.
9. For more information about the Reports fields, see *Oracle Fusion Middleware Reports Services Publishing Reports to the Web*. For more information about advanced converter options, see the following section, "[Setting Advanced Converter Options](#)".
10. Click **Next**.
11. In the **Finish** dialog, the modules you selected are listed. Click **Finish** to start the conversion.
12. Progress displays in the Log window. (Log output is stored with the log file name(s) that you specify in the Options dialog. See the following section for information about setting options.)
13. Check the log file(s) for information about required conversion steps that the tool did not modify. Manually make these changes to your application.

2.4.3.2 Setting Advanced Converter Options

Note: For information about setting the `search_replace.properties` file, see "[Editing the converter.properties File](#)".

You can set conversion options before running the wizard:

1. In the **Converter options** dialog (step 2 of the wizard), click **Advanced Options**.
2. Converter properties display in the left column of the dialog. Edit the property values as needed, as described in [Table 2-3, "Converter Properties"](#):

Table 2–3 Converter Properties

Property Value	Description
Display Backup Warning	Determines whether the warning dialog displays to back up files when the application starts.
Log File Name	Specifies the log filename in Single Log mode.
Log Dir	The destination directory to write the log files in multilog mode
Reports Servlet Virtual Directory	Specifies the name that has been defined for the virtual path used to define the Reports Servlet, which is used for running reports on the web. This setting is required when converting Run_Product calls to Run_Report_Object.
Reports Servlet	Specifies the name for the Reports Servlet used for running reports on the web. This setting is required when converting Run_Product calls to Run_Report_Object.
Reports Server	The name or IP address of the system running the Reports Server. This setting is required when converting Run_Product calls to Run_Report_Object.
Default DESFORMAT	Printer driver to be used when DESTYPE is PRINTER (XML, HTML, HTMLCSS, PDF, RTF, delimited).
Default DESTYPE	Type of destination device that receives the report output (cache, printer, file).
Default DESNAME	Name of the file, printer, e-mail ID, or distribution list to which the report output is sent
Default Browser	For UNIX, the browser used to display help for the Migration Assistant. This setting is either <i>firefox</i> , <i>iexplorer</i> , or <i>chrome</i> . In Windows, the system default browser is used.

3. Click **OK** to save the configuration. The data is saved to the `converter.properties` file, and the settings you specify are used in the current *and* future sessions of the utility.

Converting Forms 6*i* FMTs to Oracle Forms FMBs

Because some properties are obsolete in Oracle Forms, you cannot directly convert Forms 6*i* FMTs and MMTs to Oracle Forms FMBs and MMBs using Forms Developer.

3.1 Converting a Forms 6*i* FMT to an Oracle Forms FMB

To convert a Forms 6*i* FMT or MMT to an Oracle Forms 12c FMB or MMB, do the following:

1. Use the Forms 6*i* Builder or Compiler to convert the Forms 6*i* FMT or MMT to a Forms 6*i* FMB or MMB.
2. Then, use Forms Developer to convert the Forms 6*i* FMB or MMB to an Oracle Forms FMB or MMB. Open the Forms 6*i* source files in the Oracle Forms Builder, save them, then recompile them; or use the Oracle Forms Compiler.

Built-ins, Packages, Constants, and Syntax

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, built-ins, constants, packages, and some syntax that are not applicable to Web deployment have been removed.

This chapter contains the following:

- [Section 4.1, "Obsolete Menu Built-ins"](#)
- [Section 4.2, "Other Obsolete Built-ins"](#)
- [Section 4.3, "Obsolete Built-in Packages"](#)
- [Section 4.4, "Obsolete Constants"](#)
- [Section 4.5, "Obsolete Syntax"](#)

4.1 Obsolete Menu Built-ins

Menus associated with full-screen display and character mode have been removed. Code that contains these built-ins are not compiled and should be recoded, although exceptions are mentioned in the following table. Built-ins that are equivalent to obsolete built-ins are also noted in the table.

Table 4–1 *Obsolete Menu Built-ins*

Obsolete Menu Built-in	Upgrade Notes
Application_Menu	No upgrade path or replacement functionality.
Application_Parameter	No upgrade path or replacement functionality. See Chapter 12, "Menu Parameters" .
Background_Menu<n>	No upgrade path or replacement functionality.
Debug_Mode	No upgrade path or replacement functionality. Code containing this Built-in compiles but does not provide functionality.
Disable_Item	Use SET_MENU_ITEM_PROPERTY().
Enable_Item	Use SET_MENU_ITEM_PROPERTY().
Exit_Menu	No upgrade path or replacement functionality.
Hide_Menu	No upgrade path or replacement functionality.
Item_Enabled	Use GET_MENU_ITEM_PROPERTY(<name>, ENABLED). Item_Enabled works in Oracle Forms, but will be removed in a future release.
Main_Menu	No upgrade path or replacement functionality.

Table 4–1 (Cont.) Obsolete Menu Built-ins

Obsolete Menu Built-in	Upgrade Notes
Menu_Clear_Field	Use CLEAR_ITEM.
Menu_Failure	Use FORM_FAILURE flag.
Menu_Help	No upgrade path or replacement functionality.
Menu_Message	Use MESSAGE.
Menu_Next_Field	Use NEXT_ITEM.
Menu_Parameter	No upgrade path or replacement functionality. See Chapter 12, "Menu Parameters" .
Menu_Previous_Field	Use PREVIOUS_ITEM.
Menu_Redisplay	No upgrade path or replacement functionality.
Menu_Show_Keys	Use SHOW_KEYS. The upgrade process makes this change automatically.
Menu_Success	Use FORM_SUCCESS flag.
New_Application	No upgrade path or replacement functionality.
New_User	Use LOGOUT and LOGON.
Next_Menu_Item	No upgrade path or replacement functionality.
OS_Command	Use HOST.
OS_Command1	Use HOST.
Previous_Menu	No upgrade path or replacement functionality.
Previous_Menu_Item	No upgrade path or replacement functionality.
Query_Parameter	No upgrade path or replacement functionality. See Chapter 12, "Menu Parameters" .
Set_Input_Focus	No upgrade path or replacement functionality.
Show_Background_Menu	No upgrade path or replacement functionality.
Show_Menu	No upgrade path or replacement functionality.
Terminate	No upgrade path or replacement functionality. See Chapter 12, "Menu Parameters" .
Where_Display	No upgrade path or replacement functionality.

4.2 Other Obsolete Built-ins

The following Built-ins have been removed. Code that contains these Built-ins does not compile and should be recoded, although exceptions are mentioned in the following table. Built-ins that are equivalent to the obsolete Built-ins are also noted in the table.

Table 4–2 Other Obsolete Built-ins

Obsolete Built-in	Upgrade Notes
BLOCK_MENU	No upgrade path or replacement functionality.
BREAK	Upgrade to DEBUG.SUSPEND.
CALL	Use CALL_FORM.

Table 4–2 (Cont.) Other Obsolete Built-ins

Obsolete Built-in	Upgrade Notes
CHANGE_ALERT_MESSAGE	Use SET_ALERT_PROPERTY(..., ALERT_MESSAGE_TEXT,...);
DISPATCH_EVENT	Applied only to OLE and OCX items. Therefore, no upgrade path or replacement functionality.
(FORMS_OLE.)ACTIVATE_SERVER	No upgrade path or replacement functionality.
(FORMS_OLE.)CLOSE_SERVER	
(FORMS_OLE.)EXEC_VERB	
(FORMS_OLE.)FIND_OLE_VERB	
(FORMS_OLE.)GET_INTERFACE_POINTER	
(FORMS_OLE.)GET_VERB_COUNT	
(FORMS_OLE.)GET_VERB_NAME	
(FORMS_OLE.)INITIALIZE_CONTAINER	
(FORMS_OLE.)SERVER_ACTIVE	
MACRO	No upgrade path or replacement functionality.
OHOST	Use HOST.
PLAY_SOUND	No upgrade path or replacement functionality.
READ_SOUND_FILE	No upgrade path or replacement functionality.
ROLLBACK_FORM	CLEAR_FORM(NO_COMMIT,FULL_ROLLBACK)
ROLLBACK_NR	CLEAR_FORM(NO_COMMIT,FULL_ROLLBACK)
ROLLBACK_RL	CLEAR_FORM(NO_COMMIT,FULL_ROLLBACK)
ROLLBACK_SV	CLEAR_FORM(NO_COMMIT,FULL_ROLLBACK)
RUN_PRODUCT	Valid only for integration with Oracle Graphics. Use RUN_REPORT_OBJECT for integration with Oracle Reports. For all other uses, code compiles but generates run-time errors.
UPDATE_CHART	No upgrade path or replacement functionality.
VBX.FIRE_EVENT	No upgrade path or replacement functionality.
VBX.GET_PROPERTY	
VBX.GET_VALUE_PROPERTY	
VBX.INVOKE_METHOD	
VBX.SET_PROPERTY	
VBX.SET_VALUE_PROPERTY	
WRITE_SOUND_FILE	No upgrade path or replacement functionality.

4.3 Obsolete Built-in Packages

The following Built-in packages have been removed. Code that contains these packages does not compile and should be recoded, although exceptions are mentioned in the following table. Packages that are equivalent to the obsolete packages are also noted in the table.

Table 4–3 Obsolete Built-in Packages

Obsolete Package	Upgrade Notes
DEBUG	No upgrade path or replacement functionality because there is a new debugger. DEBUG.ATTACH and DEBUG.SUSPEND are still supported.
PECS	No upgrade path. Use Forms Trace and Oracle Trace, which are documented in the book <i>Oracle Fusion Middleware Forms Services Deployment Guide</i> .

4.4 Obsolete Constants

The following constants used in the GET_ITEM_PROPERTY and SET_ITEM_PROPERTY Built-ins have been removed. Code that contains these constants does not compile and should be recoded, although exceptions are mentioned in the following table.

Table 4–4 Obsolete Constants

Obsolete Constant	Upgrade Notes
DATE_FORMAT_COMPATIBILITY mode	Used by GET_APPLICATION and SET_APPLICATION properties. This constant is ignored.
COMPRESSION_OFF	No upgrade path or replacement functionality.
COMPRESSION_ON	
HIGHEST_SOUND_QUALITY	
HIGH_SOUND_QUALITY	
LOW_SOUND_QUALITY	
LOWEST_SOUND_QUALITY	
MEDIUM_SOUND_QUALITY	
MONOPHONIC	
ORIGINAL_QUALITY	
ORIGINAL_SETTING	

Table 4–4 (Cont.) Obsolete Constants

Obsolete Constant	Upgrade Notes	
POPUPMENU_CUT_ITEM	No upgrade path or replacement functionality.	
POPUPMENU_COPY_ITEM		
POPUPMENU_DELOBJ_ITEM		
POPUPMENU_INSOBJ_ITEM		
POPUPMENU_LINKS_ITEM		
POPUPMENU_OBJECT_ITEM		
POPUPMENU_PASTE_ITEM		
POPUPMENU_PASTESPEC_ITEM		
SHOW_FAST_FORWARD_BUTTON		
SHOW_PLAY_BUTTON		
SHOW_POPUPMENU		
SHOW_RECORD_BUTTON		No upgrade path or replacement functionality.
SHOW_REWIND_BUTTON		
SHOW_SLIDER		
SHOW_TIME_INDICATOR		
SHOW_VOLUME_CONTROL		
STEREOPHONIC		

4.5 Obsolete Syntax

Using the ampersand (&) as a functional equivalent to NAME_IN() is now obsolete.

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, triggers that are not applicable to Web deployment have been removed. In addition, the functionality of some triggers is being more strictly enforced.

5.1 Obsolete Triggers

Table 5–1 *Obsolete Triggers*

Obsolete Triggers	Upgrade Notes
ON-DISPATCH-EVENT	Applies only to OLE and OCX items. Therefore, no upgrade path or replacement functionality.
All V2-style triggers	When you open FMBs that contain V2-style triggers, the triggers are dropped and a warning message lists the names of the dropped triggers. You should recode V2-style triggers into PL/SQL in Forms 6i, before upgrading to this release.
When-Mouse-Move/When-Mouse-Enter/When-Mouse-Leave Triggers	These triggers are ignored when running on the Web due to the amount of network traffic that would be generated.

5.2 Stricter Enforcement of Triggers

The use of the following triggers is more strictly enforced. These triggers do not execute if they are used incorrectly.

Table 5–2 *Triggers with Restricted Usage*

Trigger	Restricted Usage
WHEN-CLEAR-BLOCK	Allowed at the Block and Form level only.
WHEN-CREATE-RECORD	No longer allowed at the Item level.
WHEN-DATABASE-RECORD	
WHEN-NEW-RECORD-INSTANCE	
WHEN-REMOVE-RECORD	
WHEN-NEW-FORM-INSTANCE	Allowed at the Form level only. No longer allowed at the Block and Item level.

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, properties that are not applicable to Web deployment have been removed.

6.1 Obsolete Properties

Many properties, including those associated with character mode and menus, have been removed. When you open a form that contains these properties, the properties are ignored and do not appear in Oracle Forms. Except as noted, code that attempts to use these properties at run time fails. Refer to [Table 6–1, "Obsolete Properties"](#) for more information.

Table 6–1 Obsolete Properties

Obsolete Property	Applies to	Upgrade Notes
Character Mode Logical Attribute	items, canvases, and so on.	
Command Type	menu items	<p>Note: This property is partially obsolete. The only valid values are Null, PL/SQL, and Menu.</p> <p>If your menu module uses Plus, Form, or Macro, which are no longer valid values, the following PL/SQL code replaces the values in the Command Text property:</p> <pre>Plus: /* HOST('sqlplus <old_code>'); */ null; Form: /* CALL_FORM(<old_code>); */ null; Macro: /* MACRO: <old_code>; */ null;</pre> <p>where <old_code> is the value of the Command Text property before upgrade. The replacement PL/SQL code is commented out so that you can replace the original code with new PL/SQL code.</p>
Data Block Description	blocks	
Fixed Length	items	Use a format mask with the relevant number of placeholders to limit or control the length of data entered for an item.
Help Description	menu items	
Listed in Data Block Menu	blocks	

Table 6–1 (Cont.) Obsolete Properties

Obsolete Property	Applies to	Upgrade Notes
List Type	LOVs	Because all LOVs are now based on record groups, this property is obsolete.
Menu Source	forms	A value of Database is no longer valid. File is the only valid value for this property, which indicates that at run time, Forms uses the normal search path to locate the MMX file.
Runtime Compatibility Mode	forms	Ignored at runtime. 5.0 behavior is always used. (See the Forms Developer online help for a description of run-time behavior.) To allow WHEN-VALIDATE-ITEM to run for NULL items, specify 4.5 for the DEFER_REQUIRED_ENFORCEMENT property. (If your Forms application used "4.5" as the Runtime Compatibility Mode property setting, the Oracle Forms Migration Assistant automatically sets the Defer Required Enforcement property to "4.5".)
Trigger Style	triggers	All triggers are now PL/SQL triggers.
White on Black	items, canvases, and so on.	

Changes to Client/Server Deployment and Forms Runtime

This section contains the following:

- [Section 7.1, "Overview"](#)
- [Section 7.2, "Effect on Forms Development"](#)
- [Section 7.3, "Obsolete Forms Runtime Command Line Options"](#)
- [Section 7.4, "Obsolete Character Mode Runtime"](#)

7.1 Overview

Client/server runtime is obsolete in Forms Developer and Forms Services.

When you use Forms Developer, part of the upgrade process is to upgrade your Forms applications for Web-based deployment.

See [Chapter 15, "Upgrading Client/Server Applications to the Web"](#) for information about how client/server deployment differs from Web-based deployment.

7.2 Effect on Forms Development

The obsolescence of client/server deployment has little to no effect on the development and debugging of Forms applications. You can still run your code in Forms Developer without having to deploy on the Web first. Use the one-button-run* facility, which renders a true WYSIWYG representation of a Web-deployed form.

The PL/SQL debugger has been improved to allow debugging in a three-tier environment.

*One-button Run is available for non *Forms Builder Only* installs only.

7.3 Obsolete Forms Runtime Command Line Options

The following command line options for Runform have been removed because they relate to obsolete features:

- OptimizeSQL
- OptimizeTP
- Keyin
- Keyout

- Output_file
- Interactive
- Block_menu
- Statistics

7.4 Obsolete Character Mode Runtime

Character mode runtime, which was only available on UNIX and VMS platforms, is no longer available. All character mode support has been removed from Forms Developer and Forms Services. See [Chapter 6, "Properties"](#) and [Chapter 9, "Logical and GUI Attributes"](#) for more information.

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, item types that are not applicable to Web deployment have been removed.

8.1 Obsolete Item Types

The following item types are obsolete in Forms Developer and Forms Services.

Table 8–1 *Obsolete Item Types*

Item Type	Upgrade Notes
Chart Item	No upgrade path or replacement functionality.

8.2 Item Types Specific to Operating Systems

The following item types are specific to operating systems and are obsolete in Forms Developer and Forms Services. These items are not removed by the upgrade process. However, any modules that contain them do not compile. Use JavaBeans and Pluggable Java Components for equivalent functionality.

Table 8–2 *Obsolete Item Types Specific to Operating Systems*

Item Type	Upgrade Notes
VBX	Was applicable to 16-bit Windows platforms only. No upgrade path or replacement functionality.
OLE Container	Was applicable to Windows platforms only. Programmatic OLE interaction is supported with external OLE servers on the middle tier.
OCX/ActiveX Controls	Was applicable to Windows platforms only. JavaBean support provides similar functionality.
Sound	No upgrade path. JavaBeans provide equivalent functionality.

Logical and GUI Attributes

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, logical and GUI attributes that are not applicable to Web deployment have been removed.

9.1 Use Visual Attributes to Replace Logical and GUI Attributes

For Web-deployed forms, you can use Visual Attributes for logical and GUI attributes to define the appearance of dynamic items.

9.2 Obsolete Logical and GUI Attributes

Replace any references to the following obsolete logical and GUI attributes in SET_ITEM_PROPERTY, SET_FIELD, or DISPLAY_ITEM with an equivalent Visual Attribute.

Table 9–1 *Obsolete Logical and GUI Attributes*

Obsolete Attribute	Where Used and Upgrade Notes
Alert	Alert text.
AlertBackground	Alert background.
AlertIcon	Icon in an alert window.
AlertMessage	Message text in an alert window.
Boilerplate	Constant text.
Bold	Bold for all items (including check boxes).
Bold-inverse	Inverse bold for all items.
Bold-text	Boilerplate.
Button-current	Current button.
Button-non-current	Non-current button.
Field-current	Color for current text item.
Field-non-current	Color for text item that is not currently selected.
Field-Queryable	Queryable field in Enter-Query mode.
Field-selected-current	Currently selected text item.
Field-selected-non-current	Text item that is not currently selected.
Full-screen-title	Screen title.

Table 9–1 (Cont.) Obsolete Logical and GUI Attributes

Obsolete Attribute	Where Used and Upgrade Notes
ItemQueryDisabled	When a Block goes into Enter-Query Mode, any non-queryable items inherit this set of attributes.
ListItemNonSelect	Unselected item in a text list.
ListItemSelect	Selected item in a text list.
ListPrefix	List prefix.
Listtitle	List of Values (LOV) title.
Menu	Selected menu.
Menu-bottom-title	Current title at bottom of menu.
MenuItemDisabled	Disabled menu item.
MenuItemDisableMnemonic	Mnemonic of a disabled menu item.
MenuItemEnable	Enabled, non-current menu item.
MenuItemEnableMnemonic	Mnemonic of an enabled menu item.
MenuItemSelect	Current menu item.
MenuItemSelectMnemonic	Mnemonic of the current menu item.
Menu-subtitle	Current menu subtitle.
Menu-title	Current menu title.
Normal	Text item.
NormalAttribute	Normal background for windows.
PushButtonDefault	Default or current button.
PushButtonNonDefault	Button that is not default.
Scroll-bar-fill, Inverse, Inverse-underline, Bold-underline, Bold-inverse-underline	These logical attributes are not unique to Forms Developer. As a result, these logical attributes can be overridden by the visual attributes defined by the window manager.
ScrollThumb	Elevator box on scroll bar.
Status-Empty	Controls the look of the empty Status Line.
Status-Hint	Controls the font of any item hint appearing on the Status Line.
Status-Items	Controls the look of the Operator Information Area which contains the LOV lamp, record count, and so on.
Status-Message	Controls the font of any message appearing on the Status Line.
Sub-menu	Selected submenu.
TextControlCurrent	Current field or text editor.
TextControlFailValidation	When an item fails a validation check, it is set to this attribute set.
TextControlNonCurrent	Disabled or non-current field or text editor.
TextControlSelect	Selected text in an enabled field or text editor.
ToolkitCurrent	Generic attribute.

Table 9–1 (Cont.) Obsolete Logical and GUI Attributes

Obsolete Attribute	Where Used and Upgrade Notes
ToolkitCurrentMnemonic	Generic attribute.
ToolkitDisabled	Generic attribute.
ToolkitDisabledMnemonic	Generic attribute.
ToolkitEnabled	Generic attribute.
ToolkitEnabledMnemonic	Generic attribute.
Underline	Underline for all items.
WindowTitleCurrent	Title of active window.

List of Values (LOVs)

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, List of Values (LOVs) that are not applicable to Web deployment have been removed.

10.1 Obsolete List of Values (LOVs)

LOVs based on record groups are still valid. However, "Old-style" LOVs (V2.3-style LOVs) are obsolete in Oracle Forms.

When forms that contain old-style LOVs are upgraded to Forms Developer, the old-style LOVs' Old LOV Text property, which refers to a table and column (such as EMP.ENAME), is converted to a "new-style" LOV by creating a record group based on a query (select <column> from <table>). The new-style LOV is based on the new record group.

As a result of removing old-style (V2) triggers, V2 user exits have also been removed.

11.1 Obsolete V2 User Exits

The following user exits, which are hard-coded callbacks to V2 trigger functionality, have been removed. It is now assumed that any calls to these user exits call a user-defined user exit rather than a built-in one.

You should recode any code that uses these callbacks to PL/SQL:

- COPY
- ERASE
- HOST
- EXEMACRO
- EZ_GOREC
- EZ_CHKREC

Menu Parameters

In order to streamline the tools available and simplify the development process for building Forms applications for the Web, menu parameters, which are not applicable to Web deployment, have been removed.

All menu parameters are removed from your applications when you upgrade to Oracle Forms.

12.1 Predefined Menu Parameters

Predefined menu parameters have names like UN and PW. Using predefined menu parameters, you were able to refer to bind variables, for example, :UN and :PW in PL/SQL code attached to menu items.

When upgrading from previous versions of Forms, use the recommended built-ins as replacements for the obsolete predefined menu parameters in the following table:

Table 12–1 *Obsolete Predefined Menu Parameters*

Obsolete Parameter	Recommended Built-in
:UN	GET_APPLICATION_PROPERTY(USERNAME)
:PW	GET_APPLICATION_PROPERTY(PASSWORD)
:LN	GET_APPLICATION_PROPERTY(USER_NLS_LANG)
:AD	GET_FORM_PROPERTY(NAME_IN('SYSTEM.CURRENT_FORM'),FILE_NAME)
:SO	:SYSTEM.TRIGGER_MENUOPTION
:TT	Only relevant in a character mode environment. This parameter has no replacement.

12.2 User-Defined Menu Parameters

User-defined menu parameters are obsolete for Oracle Forms. Any menu item that calls the MENU_PARAMETER or APPLICATION_PARAMETER built-ins allowed you to define values for menu parameters.

At runtime, an un-customizable Query Parameters dialog box would let you inspect or change the values of menu parameters. Built-ins associated with Query Parameter dialogs, such as TERMINATE, are obsolete as well. See [Chapter 4, "Built-ins, Packages, Constants, and Syntax"](#) for more information.

Therefore, to replace obsolete user-defined parameters, manually redefine them as Global variables (:GLOBAL). The initial value property of parameters can be emulated by initializing your replacement Global variables in your Menu startup code.

For other features, such as the dialog box that pops up using the MENU_PARAMETER built-in, there is no replacement functionality, although you can emulate this functionality by building a dialog using Forms.

Java-Related Issues

This chapter describes upgrade steps to take if your Forms applications use Java-related components.

13.1 Using Pluggable Java Components and Other Custom Java

Pluggable Java Components (PJC)s and JavaBeans use classes that are part of the oracle.ewt framework. The sample PJC)s and JavaBeans provided by Oracle are re-coded to use Swing classes or oracle.ewt classes. When upgrading to Oracle Forms, there are steps you must take to ensure equivalent functionality.

- The Oracle Forms Jar file (fsmall.jar) contains only the EWT classes that are required by the Forms Java Client. Therefore, PJC)s that had been used with Forms 6i may fail at runtime in Oracle Forms because the classes that were available in Forms 6i can no longer be located. Missing oracle.ewt classes are available in ewt.jar, which is supplied with Oracle JDeveloper.
- All client side Java jar files must be signed with a trusted certificate. This includes, but is not limited to, jars created for custom PJC)s, Java Beans, or any other custom resources you provide (such as images, audio, libraries, etc).

13.2 JDK Versions and Font-Rendering Issues

When upgrading Forms applications from JDK 1.1 to JDK 1.3 or higher, you may encounter font height changes. This is because the code that renders fonts underwent significant changes from JDK 1.1 to JDK 1.3. As a result of these changes, the font height for logical fonts of the same size increased in JDK 1.3. For example, a dialog font of size 12 points has a height of 15 points in JDK 1.1 and a height of 17 points in JDK 1.3.

In Forms applications, the font size changes may affect labels, which can overlap text fields. One possible workarounds is to set the following applet parameter to "yes":

```
<PARAM NAME = "mapFonts" VALUE = "yes" >
```

After making this change, check the appearance of the font size to be sure it is acceptable. Modify the form if this workaround does not provide acceptable font sizes. See the *Oracle Fusion Middleware Forms Services Deployment Guide* for details.

Another workaround is that when the font is unspecified, the default font name and size in Registry.dat is used. The default font in registry.dat is Dialog with a default size of 900. The size of this font can be modified to a smaller value in the registry.dat file. In cases where the font is not specified, you can work around the problem without modifying the form. However, use caution because it modifies the font size for the entire application.

Integration with Oracle Reports

This chapter contains the following:

- [Section 14.1, "Overview"](#)
- [Section 14.2, "About Integration with Oracle Reports"](#)

14.1 Overview

Oracle Graphics6i is no longer shipped with Oracle Forms versions 9.0.2 and higher. In addition, the Charting wizard has been removed from Forms. From 12c onwards, the run-time integration of Graphics has also been removed. This chapter describes how you can call existing Reports applications from a form.

14.2 About Integration with Oracle Reports

You can embed new and existing Reports applications in forms that have been upgraded to Oracle Forms 12c.

You can no longer use the Reports client run-time engine to output Reports in the Web. From Forms 5.0 forward, the RUN_REPORT_OBJECT built-in is available in Forms Developer to run integrated reporting in Forms. Using RUN_PRODUCT in Oracle Forms to run integrated Oracle Reports is no longer supported in this release. In 12c, the use of this built-in results in a compilation error.

Oracle Forms and Oracle Reports versions 9.0.2 and higher are now Web-based only, and do not have client/server run-time engines. Therefore, integrated reports in Oracle Forms applications must be recoded to use the RUN_REPORT_OBJECT Built-in and Oracle Fusion Middleware Reports Services.

For additional information about integrating Oracle Forms with Reports, see *Integrating OracleAS Reports in OracleAS Forms* at the Oracle Technology Network at <http://www.oracle.com/technetwork/developer-tools/forms>.

14.2.1 Displaying Reports in Oracle Forms

If your form contains embedded Oracle Reports applications, you can upgrade the form to Oracle Forms by changing the integrated call to Oracle Reports to use:

- RUN_REPORT_OBJECT built-in (Do not use the RUN_PRODUCT built-in to call Reports.)
- WEB.SHOW_DOCUMENT built-in

Using RUN_PRODUCT in Oracle Forms to run integrated Oracle Reports is no longer supported in *this release*. In 12c, the use of this built-in results in a compilation error.

The Migration Assistant is provided to help you upgrade your applications to use RUN_REPORT_OBJECT. See [Chapter 2, "Using the Oracle Forms Migration Assistant"](#) for more information.

14.2.2 Example

The following example runs a report using the RUN_REPORT_OBJECT Built-in. The report_object node defined in Oracle Forms is assumed to be "report_node1". A user-defined Reports parameter "p_deptno" is passed by Forms using the value in the "dept.deptno" field. The Reports parameter form is suppressed. For additional details about the logic used in this example, see [Notes About the Example](#).

```

/* The following example runs a report using the RUN_REPORT_OBJECT Built-in. The
report_object node defined in Forms Developer is assumed to be "report_node1". A
user-defined Reports parameter "p_deptno" is passed by Forms using the value in
the "dept.deptno" field. The Reports parameter form is suppressed */

DECLARE
v_report_id Report_Object;
vc_report_job_id VARCHAR2(100); /* unique id for each Report request */
vc_rep_status VARCHAR2(100); /* status of the Report job */

BEGIN
/* Get a handle to the Report Object itself. */
v_report_id:= FIND_REPORT_OBJECT('report_node1');
SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_COMM_MODE, SYNCHRONOUS);
SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_DESTTYPE,CACHE);

/* Define the Report output format and the name of the Reports Server as well as
a user-defined parameter, passing the department number from Forms to the Report.
The Reports parameter form is suppressed by setting paramform to "no". */
SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_DESFORMAT,
'<HTML|HTMLCSS|PDF|RTF|XML|DELIMITED>');
/* replace <ReportServerTnsName> with the name of the Reports Services as defined
in your tnsnames.ora file */
SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_SERVER, '<ReportServerTnsName>');
SET_REPORT_OBJECT_PROPERTY(v_report_id,REPORT_OTHER, 'p_
deptno='||:dept.deptno||'paramform=no');
/* finally, run the report and retrieve the Reports job_id as a handle to the
Reports process */
vc_report_job_id:=RUN_REPORT_OBJECT(v_report_id);

/*The report output is not delivered automatically to the client, which is okay
because the Web is a request model. Thus the next step is to check if the report
finished. */

vc_rep_status := REPORT_OBJECT_STATUS(vc_report_job_id);
IF vc_rep_status='FINISHED' THEN
/* Call the Report output to be displayed in a separate browser window. The URL
for relative addressing is only valid when the Reports Server is on the same host
as the Forms Server. For accessing a Remote Reports Server on a different
machine, you must use the prefix http://hostname:port/ */
web.show_document ('/<virtual path>/<reports cgi or servlet name>/getjobid='||
vc_report_job_id ||'?server='|| '<ReportServerTnsName>','_blank');
ELSE
message ('Report failed with error message '||rep_status);
END IF;
END;
```

14.2.3 Notes About the Example

- Calling a report synchronously makes the user wait while the report gets processed on the server. For long-running Reports, it is recommended that you start the report asynchronously, by setting the REPORT_COMM_MODE property to asynchronous and the REPORT_EXECUTION_MODE to batch.

```
SET_REPORT_OBJECT_PROPERTY (report_id,REPORT_EXECUTION_MODE,BATCH);
```

```
SET_REPORT_OBJECT_PROPERTY (report_id,REPORT_COMM_MODE,ASYNCHRONOUS);
```

- After calling the RUN_REPORT_OBJECT Built-in, you must create a timer to run frequent checks on the current REPORT_OBJECT_STATUS using a When-Timer-Expired trigger. For performance reasons, the timer should not fire more than four times a minute. After the report is generated, the When-Timer-Expired trigger calls the WEB.SHOW_DOCUMENT Built-in to load the Reports output file, identified by a unique job_id, to the client's browser.

Note: Do not forget to delete the timer when it is no longer needed.

The following example shows the When-Timer-Expired trigger that checks for the Report_Object_Status.

```
(...)
/* :global.vc_report_job_id needs to be global because the information about
the Report job_id is shared between the trigger code that starts the Report
and the When-Timer-Expired trigger that checks the current Report status. */
vc_rep_status:= REPORT_OBJECT_STATUS(:global.vc_report_job_id);
IF vc_rep_status='FINISHED' THEN
web.show_document ('/<virtual path>/<reports cgi or servlet name>/getjobid='||
vc_report_job_id ||'?server='|| '<ReportServerTnsName>','_blank');
ELSIF vc_rep_status not in ('RUNNING','OPENING_REPORT','ENQUEUED') THEN
message (vc_rep_status||' Report output aborted');
END IF;
(...)
```

14.2.4 Using Parameter Lists in RUN_REPORT_OBJECT

Parameter lists that were used with RUN_PRODUCT in client/server mode can also be used with RUN_REPORT_OBJECT calling Oracle Fusion Middleware Reports Services. System parameters must be set by the Set_Report_Object_Property. The syntax for using parameter lists in RUN_REPORT_OBJECT is as follows:

```
report_job_id:=run_report_object (report_id,paramlist_id);
```

where paramlist_id is the same id used with RUN_PRODUCT.

Parameter settings can be any of the following:

- REPORT_COMM_MODE: Batch, Runtime
- REPORT_EXECUTION_MODE: Synchronous, Asynchronous
- REPORT_DESTYPE: FILE, PRINTER, MAIL, CACHE, PREVIEW, FTP, FAX, WEBDAV, ORACLEPORTAL, ORACLEWIRELESS, SECUREPDF
- REPORT_FILENAME: The report filename (not used with CACHE)
- REPORT_DESNAME: The report destination name (not used with Cache)
- REPORT_DESFORMAT: The report destination format

- REPORT_SERVER: The report server name

Other settings are as follows:

- Reports CGI name is "rwcgi.sh" (UNIX) and "rwcgi.exe" (Windows)
- Reports Servlet default name is "rwservlet"
- Reports Servlet virtual path is /reports/

14.2.5 Upgrade Steps

You can use the Forms Migration Assistant described in [Chapter 2, "Using the Oracle Forms Migration Assistant"](#) to change integrated Reports calls in your Oracle Forms modules. The Migration Assistant adds code to your application modules that redirects Run_Product calls to Reports and uses the Run_Report_Object Built-in and Reports Services. The resulting conversion is of the same quality as using Run_Product and the run-time engine in Forms 6i.

To manually upgrade Reports in Oracle Forms, do the following:

1. Find all occurrences of Run_Product.
2. Identify and locate the parameter lists used with these calls.
3. Remove all of the Reports system parameter settings like desname and destype from the parameter lists.
4. Find the Reports node ID for the Reports node name defined in Oracle Forms or the Forms 6i Builder.
5. Create Set_Report_Object_Property codes for DESNAME, REPORT_SERVER, DESFORMAT, DESTYPE, COMM_MODE, and EXECUTION_MODE in your PL/SQL.
6. Use Run_Report_Object(report_node_id, paramlist_id) to reuse your parameter lists that had been created for Run_Product.

Note: Detailed information about how to change calls to Oracle Reports in Forms 6i to use Run_Report_Object are provided in a whitepaper on <http://www.oracle.com/technetwork/developer-tools/forms/documentation/index.html>.

Upgrading Client/Server Applications to the Web

This chapter describes guidelines for upgrading client/server applications to the Web.

This chapter contains:

- [Section 15.1, "Guidelines for Upgrade"](#)
- [Section 15.2, "About Client/Server-Based Architecture"](#)
- [Section 15.3, "About Web-Based Architecture"](#)

In addition, refer to the white papers and other resources at

<http://www.oracle.com/technetwork/developer-tools/forms/documentation/index.html>.

15.1 Guidelines for Upgrade

When upgrading your applications from client/server deployment to the Web, notice that a Web-based application:

- Supports JPEG and GIF image types only, so convert existing images to these formats.
- Supports the use of compressed JAR (Java Archive) files for file transfer. Ensure you JAR the PJC's or Java beans for file transfer. You do not need to JAR files if transferring files from the client to the Forms server as part of the application functionality.
- Does not support ActiveX, OCX, OLE, or VBX controls in the user interface. Instead, use JavaBeans to duplicate functionality in the user interface. Any other Microsoft Windows user interface dependencies should also be replaced with JavaBeans. You can also use WebUtil to restore some of these functions.
- Does not support MouseMove triggers, such as When-Mouse-Enter, When-Mouse-Leave, and When-Mouse-Move.
- Does not natively support write access to the client hard drive. This can be accomplished by the use of Oracle Forms Webutil library.
- Supports Java fonts only, so check applications for the types of fonts used. If necessary, switch to Java fonts. Java uses a font alias list, located in the Registry.dat file. The font aliases described in [Table 15-1](#) are supported:

Table 15–1 Font Support for Web-based Applications

Java font	Windows font	XWindows font	Macintosh font
Courier	Courier New	adobe-courier	Courier
Dialog	MS San Serif	b&h-lucida	Geneva
DialogInput	MS San Serif	b&h-lucidatypewriter	Geneva
Helvetica	Arial	adobe-helvetica	Helvetica
Times Roman	Times New Roman	adobe-times	Times Roman ¹

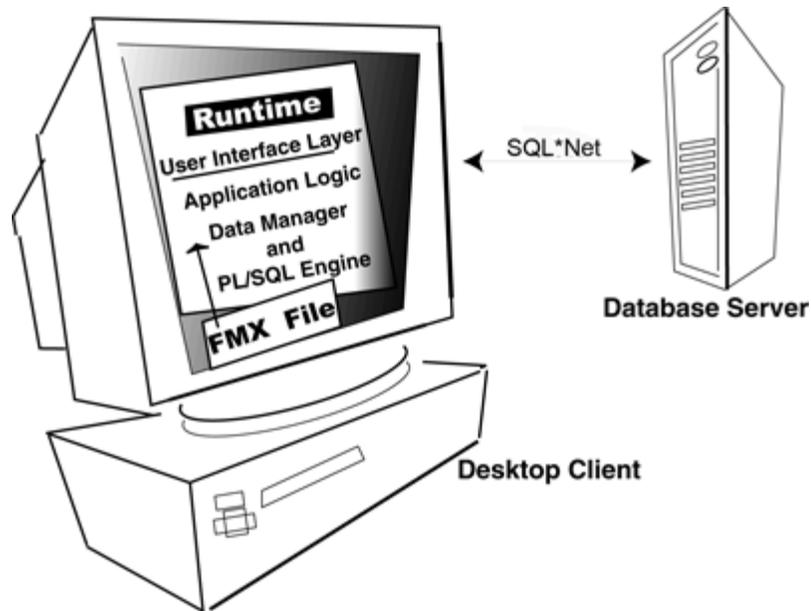
¹ More information about Java fonts can be found at <https://docs.oracle.com/javase/tutorial/2d/text/fonts.html>

- Has some Built-ins and packages that execute only in the application server, but not in the client browser:
 - TEXT_IO
 - HOST
 - ORA_FFI
 - GET_FILE_NAME*
 - READ_IMAGE_FILE
 - WRITE_IMAGE_FILE

*Although modules that use the GET_FILE_NAME Built-in will successfully compile/generate, calls to this built-in will do nothing (NULL is returned). If the functionality of these Built-in and packages is required for the client, use WebUtil.

15.2 About Client/Server-Based Architecture

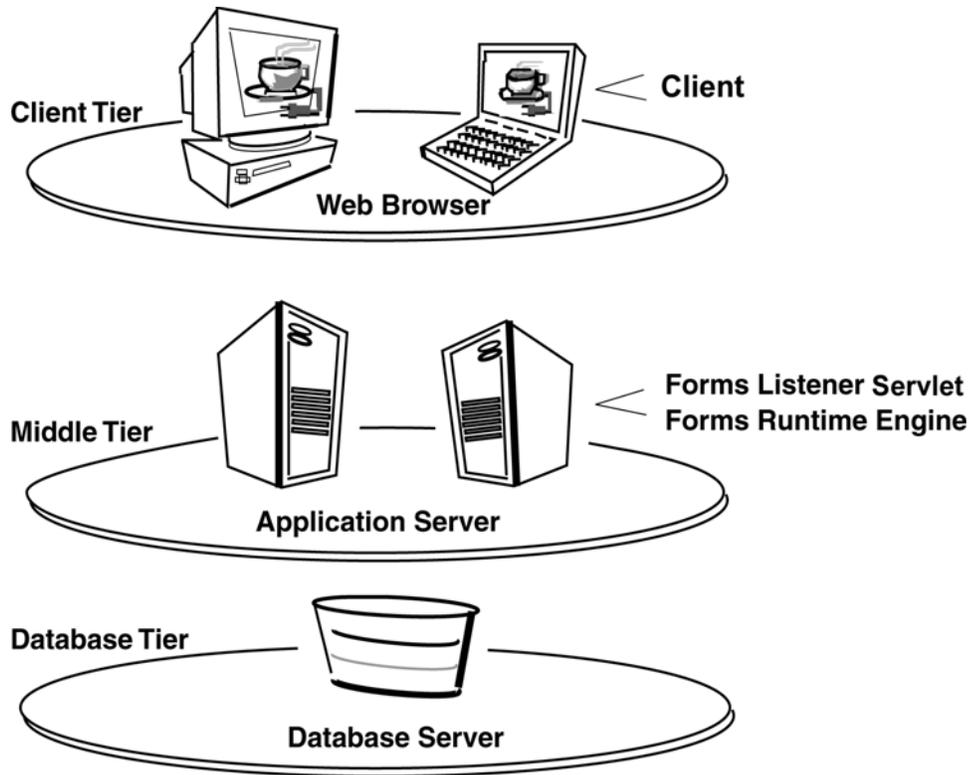
In the client/server-based implementation, shown in [Figure 15–1](#), the Forms Server Runtime Engine and all application logic are installed on the user's desktop machine. All user interface and trigger processing occurs on the client, except for database-server-side triggers and logic that may be included in some applications.

Figure 15–1 Oracle Forms Client/Server Architecture

15.3 About Web-Based Architecture

In a Web-based implementation, shown in [Figure 15–2](#), the Forms Services Runtime Engine and all application logic are installed on application servers, and not on client machines. All trigger processing occurs on the database and application servers, while user interface processing occurs on the Forms client, located on users' systems.

Figure 15-2 Oracle Forms Web Architecture



Upgrading from Pre-Forms 6i Applications to Oracle Forms

Forms Developer is upwardly compatible with earlier versions of Oracle Forms, including Versions 3.0, 4.0, 4.5, and 5.0.

Note: If you are upgrading from releases of Forms before Forms 6i to Oracle Forms 12c, you must first upgrade your applications to Forms 10g (10.1.2), and then upgrade them to Oracle Forms 12c.

Note: Before converting your forms or menus, it is recommended that you first make backup copies of all files. Once you upgrade a module, the module cannot be opened in an earlier version of Forms Developer.

16.1 Upgrading a Form

To upgrade a version 4.x or 5.x Forms application to Forms 10g (10.1.2):

1. Start Forms 10g (10.1.2).
2. Choose **File | Open** to display the file.
3. Choose the module you want to upgrade.
4. Click **OK**.
5. Choose **Program | Compile PL/SQL | All** to compile the upgrade the module's PL/SQL code.
6. Choose **File | Save**.

Note: All form modules and libraries must be upgraded and recompiled.

16.2 Upgrading files saved in Database

In order to streamline the tools available and simplify the development process for building Forms applications, the option of saving files to database has been removed. In Forms 6i, modules could be saved in the database. For all releases after 6i, module files that were saved in database must be saved to the file system.

1. Open the prior version of Forms.

2. Save the module files to the local file system.
3. Upgrade the files as described in [Section 16.1, "Upgrading a Form"](#).

16.3 Compatibility with Earlier Versions of PL/SQL

If you have client-side program units written in PL/SQL V1 or V2, you must convert that code to the new level.

Stored program units can use all the PL/SQL features that are valid for use in client side PL/SQL. Certain PL/SQL features such as the supplied DBMS_LOB routines or DBMS packages that are specified as pragma interface (C, C++, etc) cannot be directly called from Client Side PL/SQL. Forms programs must call a stored procedure which in turn calls the routine in question in these restricted cases.

16.4 Forms Developer Runtime Behavior

The default run-time behavior of forms created with Forms 5.0 through 6i differed from run-time behavior for Forms 4.5. The form-level Runtime Compatibility Mode property could be set to "4.5" to provide Forms release 4.5 behavior. (This happened by default in forms that were upgraded from Forms release 4.5.)

Starting with Forms Developer, 5.0 behavior is used in all cases, and the form-level property "Runtime Compatibility Mode" is ignored.

If you are upgrading a form that specifies 4.5 behavior to Oracle Forms, you must alter the logic, as necessary, to reflect the differences between 4.5 and 5.0 behavior. See the online help for information about the Runtime Compatibility Mode property and the differences between 4.5 and 5.0 behavior.