Oracle9*i* Forms: Features Obsolescence

Statement of Direction
February 2002

ORACLE

Oracle9*i* Forms: Features Obsolescence

PURPOSE

This document describes Oracle Corporation's intention to no longer support certain features in Oracle9*i* Forms.

INTRODUCTION

The Oracle Forms product suite has been around for many years now, and whenever possible, has faithfully maintained compatibility with older product versions. With the move to pervasive Web deployment and the release of Oracle9*i* Forms, we are simplifying the product set and continuing to optimize the product for building and deploying Internet applications. To this end, Oracle9*i* Forms will focus exclusively on Internet deployment. This process will encompass legacy features within the existing products along with the removal or replacement of several features of the product suite.

Oracle recognizes that many features that will no longer be supported are considered to be important parts of the product by many customers. Oracle has, therefore, decided that it is appropriate to treat this transition by using the same obsolescence process as if Oracle were ending the life of an entire major product.

Oracle Forms release 6*i* is the last major release of the Oracle Forms suite to contain the features that are documented here as being planned for obsolescence.

To aid customers who, in light of the features to be dropped, feel unable to migrate to Oracle9*i* Forms, Oracle intends to provide error correction support for Oracle Forms release 6*i* until 2006, with extended support available as an option until 2008. The formal obsolescence process (Maturity, Extended Assistance, and Obsolescence) will be followed for that release.

This document specifically deals with Oracle Forms and its bundled products. Oracle Reports is covered in a separate statement of direction.

Terminology

Throughout this paper, references to Oracle Forms refer to all components bundled with it, with the exception of Oracle Reports.

FEATURES TO BE DROPPED FROM ORACLE FORMS

In order for the product to move forward and evolve, Oracle is refocusing Forms for Web-only deployment. This allows Oracle to put all its efforts into improving the already rich Web user interface based on Java, and to extend the openness of the product by allowing Java integration on all three tiers. The following topics are covered:

- Client Server Runtime
- Character Mode Runtime
- Character Mode Properties and Logical Attributes
- Deprecation of Operating System Specific Item Types
- Obsolete and no longer supported built-ins
- Forms Version 2 Style Triggers and List Of Values
- Properties being Removed
- Command line options to Runform
- Tightening up of Trigger and Built-in Usage
- Use of the Database for Module Storage

Client Server Runtime

Although the Form Builder will continue as a client-server application running on either 32-bit Windows or Motif platforms, Oracle intends to make conventional client-server runtime obsolete in Oracle9*i* Forms.

From the point of view of the development environment, this change will have little effect on day-to-day building and debugging. Developers will continue to run their code on the Form Builder without having to Web-deploy first. This feature is already available in the run on Web facility available within the 6*i* Form Builder, although it will be improved in Oracle9*i* Forms to provide a truly WYSIWYG representation of a Web-deployed form. The PL/SQL debugger is undergoing considerable improvement, one aspect of which is to allow debugging in a three-tier environment.

Users who currently deploy their applications in a client-server environment and who wish to upgrade to Oracle9*i* Forms will have to Web-deploy their applications as part of the upgrade process. In doing so, they can leverage the cost savings inherent in web deployment over client-server.

Character Mode Runtime

In line with the obsolescence of the client-server run-time, Oracle will also remove the character mode runtime that is currently only available on UNIX and VMS Forms platforms.

As character mode support will be removed from the product, the following features, which existed solely for supporting backwards compatibility, will also be removed.

Character Mode Form Properties

The following properties, related to character mode support, will be removed entirely from Forms (and Menus). The upgrade process will simply ignore these properties, making them not appear in the Oracle9*i* Forms Builder. Any code that attempts to use these properties at runtime will fail.

Obsolete Property	**Relates to**
Listed in Data Block Menu	Block
Data Block Description	Block
White on Black	Items, Canvases, etc.
CM Logical Attribute	Items, Canvases, etc.
Help Description	Menu Item (Full Screen Menus)
Menu Source	Form

In Oracle Forms release 6*i* and earlier, the Form level property Menu Source has two possible values: *File* or *Database*. A value of Database for this property indicates that at run-time, Forms should look up the *location* of the menu MMX file in the database. The MMX file itself still had to be present on the file system. A value of File indicates that Forms runtime should use its normal search path to locate the MMX file. In Oracle9*i* Forms the Database option for this property will be dropped, and File will become the access method for menu files.

Character Mode Logical Attributes

Character mode logical attributes provided a way to control features such as field appearance on character mode (and block mode) screens. This generally consisted of definitions such as, “make enterable fields inverse video”. Logical attributes are defined in existing versions of Forms using the Oracle Terminal utility.

For several years, Oracle has recommended that customers utilize named Visual Attributes, rather than Logical Attributes to define their dynamic item appearance characteristics.

All Logical and GUI Visual attributes (listed in Appendix A) will be removed. Any run-time references to them in SET_ITEM_PROPERTY, SET_FIELD, or DISPLAY_ITEM should be altered to use a named Visual Attribute created in the Forms Builder.

Deprecation of Operating System Specific Item Types

As Oracle moves Forms to a Web-only deployment model, we will no longer be able to support item types that are specific to client-server or character mode use. As a result, the following types of items will be deprecated in Oracle9*i* Forms.

Item Type	Notes
VBX	VBX Items are applicable only to 16-bit windows. This is also an obsolete technology.
OLE Container	OLE containers are only applicable to Windows platforms. Oracle has no plans for OLE container support over the Web. Programmatic OLE interaction will still be supported with external OLE servers on the middle tier.
OCX/ActiveX Controls	ActiveX Controls are applicable to Windows Platforms only. Oracle has no plans for ActiveX support over the web. JavaBean support in Forms 6*i* and above provides similar functionality.
Sound Item	Sound Items only function in client-server deployment and will not be supported in Oracle9*i* Forms.

While these items will not be removed by the upgrade process, they will not function in Oracle9*i* Forms. Functionality previously accomplished with these methods can be achieved using JavaBeans or Pluggable Java Components.

Deprecation of Built-ins Relating to Obsolete Operating System Specific Item Types

As the above widgets are removed from the product, the following built-ins become obsolete and using them will result in a compile-time errors:

Item Type	Built-ins
VBX	VBX.FIRE_EVENT VBX.GET_PROPERTY VBX.GET_VALUE_PROPERTY VBX.INVOKE_METHOD VBX.SET_PROPERTY VBX.SET_VALUE_PROPERTY
OLE Container OCX ActiveX Controls	(FORMS_OLE.)[1]ACTIVATE_SERVER (FORMS_OLE.)CLOSE_SERVER DISPATCH_EVENT (FORMS_OLE.)EXEC_VERB

[1] Many of the OLE container related built-ins can be optionally prefixed with the package name FORMS_OLE. You code could be using the built-in, with or without the prefix.

	(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
Sound Item	PLAY_SOUND READ_SOUND_FILE WRITE_SOUND_FILE

Removal of Constants Relating to Obsolete Operating System Specific Item Types

As the above widgets are removed from the product, the following constants become obsolete and using them will result in a compile-time errors. These are all constants used in the GET and SET_ITEM_PROPERTY built-ins:

Item Type	**Built-ins**
OLE Container OCX ActiveX Controls	SHOW_POPUPMENU POPUPMENU_CUT_ITEM POPUPMENU_COPY_ITEM POPUPMENU_PASTE_ITEM POPUPMENU_PASTESPEC_ITEM POPUPMENU_INSOBJ_ITEM POPUPMENU_DELOBJ_ITEM POPUPMENU_LINKS_ITEM POPUPMENU_OBJECT_ITEM
Sound Item	SHOW_PLAY_BUTTON SHOW_REWIND_BUTTON SHOW_FAST_FORWARD_BUTTON SHOW_RECORD_BUTTON SHOW_VOLUME_CONTROL SHOW_TIME_INDICATOR SHOW_SLIDER ORIGINAL_SETTING ORIGINAL_QUALITY MONOPHONIC STEREOPHONIC COMPRESSION_OFF COMPRESSION_ON HIGHEST_SOUND_QUALITY HIGH_SOUND_QUALITY MEDIUM_SOUND_QUALITY LOW_SOUND_QUALITY LOWEST_SOUND_QUALITY

Deprecation of Triggers Relating to Obsolete Operating System Specific Item Types

As part of the deprecation of ActiveX controls, the ON-DISPATCH-EVENT trigger used to handle events from ActiveX controls will also be removed. The WHEN-CUSTOM-ITEM-EVENT trigger used by ActiveX and VBX controls is retained, as this trigger is also used to handle events from Pluggable Java Components and JavaBeans.

Additional Obsolete and No Longer Supported Built-ins and Packages

The following built-ins will be dropped in Oracle9*i* Forms. Code containing calls to these built-ins may fail to compile and should be re-coded.

Built-in / Package Name	**Notes**
ROLLBACK_FORM ROLLBACK_NR ROLLBACK_RL ROLLBACK_SV	These are all undocumented built-ins that should not have been used in your code. Any code using these built-ins should be changed to use CLEAR_FORM instead.
CHANGE_ALERT_ MESSAGE	This is an Undocumented built-in that should be changed to SET_ALERT_PROPERTY(…, ALERT_MESSAGE_TEXT,…);
BREAK	The Break built-in will still compile in Oracle9*i* Forms, but will not function. References to Break should be converted to the built-in Debug.Suspend.
DEBUG (Package)	Most built-ins within the debug package will be rendered obsolete by the new debugger built into Oracle9*i* Forms. The Built-In Debug.Suspend is still valid and should be used instead of the obsolete Break built-in. A new procedure has been added to the Debug Package – Debug.Attach to allow attaching of the Oracle9*i* Forms debugger at runtime.
PECS (Package)	The PECS utility is being removed (see "Components to be removed from the suite"). References to the PECS built-ins should be removed from your code.
BLOCK_MENU	Block Menus are only relevant in Character Mode applications and will be removed in

	Oracle9*i* Forms. Any references to BLOCK_MENU[2] should be removed from your code.
OHOST	The OHOST built-in was an undocumented built-in, formally used on a limited range of platforms such as DOS, where only a single connection could be made to the database. This allowed a hosted session to share the same database connection as the form. In current releases, OHOST simply emulates a HOST() call. OHOST will be removed in Oracle9*i* Forms.
MACRO	The MACRO built-in is just another way of invoking V2 style macro code from PL/SQL. MACRO will be removed in Oracle9*i* Forms.
CALL	The CALL built-in is an undocumented version of CALL_FORM. It will be removed in Oracle9*i* Forms.

[2] The KEY-MENU trigger point will not be removed and will still function. This is in acknowledgement of the wide use of this trigger point for purposes other than the display of the Block Menu.

The following built-ins will be dropped in Oracle9*i* Forms. Code containing calls to these built-ins will compile, but will generate run-time errors and should be re-coded.

Built-in/ Package Name	**Notes**
RUN_PRODUCT	RUN_PRODUCT should only be used in Oracle9*i* Forms for integration with Oracle Graphics. RUN_PRODUCT calls being used to integrate Forms with Reports should be replaced using the newer RUN_REPORT_OBJECT built-in.

V2-Style Triggers

V2-style triggers have been gradually phased out over the past several releases. In Oracle Forms release 4.5, developers were able to create and edit V2-style triggers. In Oracle Forms release 5.0, they were able to edit existing V2-style triggers, but not create new ones. In Oracle Forms release 6*i*, developers can neither create nor edit (or even view) V2-style triggers, which makes them effectively obsolete. In Oracle9*i* Forms, Oracle will take the next step and remove them entirely from the product.

When upgrading .fmb files to Oracle9*i* Forms, the V2-style triggers will be dropped, and a warning message that lists the names of the dropped triggers will appear.

In addition, the ***Trigger Style*** property will go away in Oracle9*i* Forms, as all triggers will be PL/SQL triggers.

As well as dropping V2-style triggers generically, Oracle will also drop several related features:

- V2 hardcoded user exits
- V2 style naming references
- V2 style List of Values

Obsoleted Hardcoded V2 User Exits

The following User Exits are hard-coded callbacks into V2 trigger functionality. They will be removed in Oracle9*i* Forms and any code encountering them will search for a User Exit with the same name in the IAPXTB structure as if for any normal User Exit:

- COPY
- ERASE

- HOST
- EXEMACRO
- EZ_GOREC
- EZ_CHKREC

Any code using these callbacks should be manually re-coded into PL/SQL.

Obsoleted V2 Naming References

One feature of V2-style triggers was the use of ampersand (&) as a functional equivalent to NAME_IN(). This has not been documented since Oracle Forms release 3.0 and will no longer be possible in Oracle9*i* Forms.

V2 Style List of Values (LOVs)

LOVs are one of two types:

- Record Group, or
- Old-Style (also known as V2.3-style)

In Oracle Forms release 6*i*, the Builder can load and compile old-style LOVs, but developers are not allowed to create new ones. Only LOVs based on record groups can be created in Oracle Forms release 6*i*. Oracle Forms release 5.0 was the last release where developers were able to create old-style LOVs. In Oracle9*i* Forms, Oracle will take the next step and drop old-style LOVs.

Oracle will implement an upgrade path for old-style LOVs. Old-style LOVs have a single property called Old LOV Text ,which has a value that refers to a table and a column (that is, EMP.ENAME). These old-style LOVs will be upgraded into new-style LOVs by creating a record group based on a query (select <column> from <table>), and then turning the old-style LOV into a new-style LOV based on that new record group.

In addition, the ***List Type*** property will go away in Oracle9*i* Forms, as all LOVs will be based on record groups.

Other Properties being removed in Oracle9*i* Forms

The following will also be removed from Oracle9*i* Forms.

Item Fixed Length Property

The fixed length property on Forms items is ignored in many circumstances and is considered obsolete. In Oracle9*i* Forms, Oracle will remove this property. To enforce fixed lengths on items that require the ability to limit or control the length of the data entered, use a format mask with the relevant number of placeholders.

Form Runtime Compatibility Mode Property

Oracle will preserve one useful feature of the current 4.5 runtime compatibility mode in Oracle9*i* Forms. This is the option that allows WHEN-VALIDATE-ITEM to run for NULL items, even when the DEFER_REQUIRED_ENFORCEMENT property is set to true. You can accomplish this behavior by specifying the value 4.5 for the DEFER_REQUIRED_ENFORCEMENT property, which is actually introduced with Oracle Forms release 6*i*.

All other aspects of 4.5 compatibility mode will no longer be supported. These behaviors are currently documented in the help topic **Form Builder 4.5 Runtime Behavior** in the Form Builder help for Oracle Forms release 6*i*.

Explicit setting of the Runtime Compatibility Mode is ignored at runtime and the 5.0 behavior is used no matter what value for the mode is specified.

If you are currently using the Forms Runtime Compatibility set to a value of 4.5, you should change the setting to 5.0 and test your application in this mode before upgrading to Oracle9*i* Forms.

Date Format Compatibility Mode

The application property Date_Format_Compatibility_Mode is used in Forms 6i and below, for forms whose Runtime Compatibility Mode is 4.5, to control the way in which strings representations of dates are parsed for certain built-ins, as documented in the 6i online help. With a setting of 4.5 the current date input and output masks are used to convert the string for those built-ins. With a setting of 5.0 (the recommended value), the built-in date format mask is used for all built-ins.

For forms whose Runtime Compatibility Mode is 5.0, the Date_Format_Compatibility_Mode property is ignored: the built-in date format mask is always used for all built-ins. Since Oracle9*i* Forms always behaves as if Runtime Compatibility Mode is 5.0, it will always ignore the Date_Format_Compatibility_Mode property.

Command Line options to Forms Runtime

The following seven Command Line Options for Runform will be removed in Oracle9*i* Forms as they relate to obsolete features:

- OptimizeSQL
- OptimizeTP
- Keyin
- Keyout
- Output_file
- Interactive

- Block_menu
- Statistics

Stricter Enforcement of Existing Triggers and Built-ins

With Oracle Forms release 6*i* and below, you are able to define some triggers at Item and Block level, which make no sense in that context. With Oracle9*i* Forms, Oracle will enforce the rules regarding these triggers and will not execute them if they exist.

The following five triggers will only be allowed at the Block or Form level, and not at the Item level as is currently possible:

- WHEN-CLEAR-BLOCK
- WHEN-CREATE-RECORD
- WHEN-DATABASE-RECORD
- WHEN-NEW-RECORD-INSTANCE
- WHEN-REMOVE-RECORD

The following trigger will, likewise, only be allowed at Form level, not at Block or Item Level:

- WHEN-NEW-FORM-INSTANCE

One aspect of built-in usage in Oracle9*i* Forms that is being changed, is to prevent navigation to non-enabled items. This is allowed in Forms 6*i* using the GO_ITEM() built-in. Trying the same operation in Oracle9*i* Forms will result in the error: **FRM-40112: Attempted go_item to non enabled item** <name>.
An environment variable FORMS90_REJECT_GO_DISABLED_ITEM is available to reverse this behavior change, if absolutely necessary (simply set the value to FALSE to allow navigation to disabled items). However, this environment variable will be removed in a future version of Forms and the new behavior strictly enforced. Therefore we recommend that you correct any code carrying out this action.

Use of the Database for Module Storage

In Oracle Forms 6i and below you are able to store Forms source files into the database as an alternative to saving them directly to the filesystem. You may also be subclassing objects from modules stored in this way.

In Oracle9*i* Forms the option to store and subclass from modules stored in this way will be removed. The only available modes of saving your modules will be directly to the filesystem, or by saving them into source control using Oracle9*i* Software Configuration Manager (Oracle9*i* SCM), using the Check in and Check out capabilities provided by the Form Builder.

Moving From Character-Mode or Client Server to Web Deployment

Although not directly an issue of upgrading from 6*i* to Oracle9*i* Forms, there are some features that are effected or obsoleted as a result of the transition to the Web. These issues would have to be addressed in such a migration solely within 6*i*, but are documented here as a convenience.

Feature	**Notes**
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.
Host Commands / ORA_FFI / User Exits	All execute on the application server tier not on the end user browser machine.
Image Control Palette	The control panel that can be displayed using the Show Palette property on Image items does not display when web deployed
Icons	Icons used for Iconic Buttons need to be available as GIF or JPEG files rather than ICO files.
Get_File_Name built-in	Does not function when web deployed. This functionality can be replaced on the client tier using a JavaBean.

FEATURES TO BE DROPPED FROM MENUS

Like the Forms product, Menu has evolved over the years. What was considered to be a standard menu 10 years ago with a full screen interface, has changed into the drop-down paradigm that we are accustomed to today. Meanwhile, Menu has long ceased to be a stand-alone product and is now integrated as part of Forms. The old menu styles of Full Screen and Bar (Lotus) will no longer be supported in Oracle9*i* Forms, and only the Drop-Down menu style will be available.

The following features will also be dropped:

- Character Mode Menu Properties
- Obsolete types from Menu-Items command type property
- Menu Parameters
- Menu Built-ins

Character Mode Menu Properties

As mentioned in the section on Forms, the following property, related to character mode support, will be entirely removed from Menus. This upgrade process simply ignores this property and it does not appear in the Oracle9*i* Forms Builder.

Obsolete Property	**Relates to**
Help Description	Menu Item (Full Screen Menus)

Menu-Items command type property

The menu items *Command Type* property specifies the nature of a menu item command. This property determines how Forms Builder interprets the text in the Command Text property. In Oracle Forms release 6.0, the possible values for this property include Null, PL/SQL, Menu, Plus, Form, and Macro. The first three are often used, while the latter three are not often used. The documentation explicitly warns against using Plus, Form, and Macro command types. Oracle will remove the Plus, Form, and Macro command types in Oracle9*i* Forms.

Oracle will enable menu modules with these command types to upgrade to Oracle9*i* Forms by automatically replacing Plus, Form, and Macro type menu items with PL/SQL menu items. The new PL/SQL code in the *Command Text* property for these items will be as follows:

Command Type	**Upgraded Code**
Plus	`/* HOST('plus80 <old_code> '); */` `null;`
Form	`/* CALL_FORM(<old_code>);`

	*/ null;
Macro	/* MACRO: <*old_code*> ; */ null;

<*old_code*> is the value of the ***Command Text*** property before upgrade. The replacement PL/SQL code will be commented out to allow developers to replace their previous code with new PL/SQL code.

Menu Parameters

Menu Parameters will be obsolete in Oracle9*i* Forms and deleted by the upgrade process as they are not useful for GUI application development.

Two kinds of menu parameters exist: predefined and user-defined. We will deal with each of these in turn.

Predefined Menu Parameters

The predefined menu parameters have names like UN, PW, etc., which means that you can refer to bind variables: UN, :PW, etc. in PL/SQL code attached to menu items. Alternative ways of accomplishing the same thing are, or will be, provided using Forms built-ins as follows:

Menu Parameter	**Functional Replacement**
:UN	GET_APPLICATION_PROPERTY(USER_NAME)
:PW	GET_APPLICATION_PROPERTY(PASSWORD)
:LN	GET_APPLICATION_PROPERTY(USER_NLS_LANG)
:AD	GET_FORM_PROPERTY(<formname>,FILE_NAME)
:SO	:SYSTEM.TRIGGER_MENUOPTION[3]

One predefined menu parameter :TT is only relevant in a character mode environment (it returns the terminal type in use, for example, VT220). This parameter has no replacement.

The upgrade process to Oracle9*i* Forms will not perform the above conversions for you. This will have to be carried out manually or by using a Forms API program.

[3] :SYSTEM.TRIGGER_MENUOPTION is a new system variable that will be added to Oracle9*i* Forms.

User-Defined Menu Parameters

User-defined menu parameters are created in the Object Navigator. You can define values for menu parameters by attaching PL/SQL code to a menu item that calls either MENU_PARAMETER() or APPLICATION_PARAMETER() built-ins. At runtime this brings up an un-customizable Query Parameters dialog box that lets you inspect or change the value of the menu parameters.

All of the built-ins that are associated with these Query Parameter dialogs, such as TERMINATE, will be obsoleted along with the parameter feature.

In effect, since user-defined parameters act just like Global variables in Forms (:GLOBAL.), the upgrade path will be to manually redefine them as Global variables. The initial value property of parameters can be emulated by initializing your replacement Global variables in your Menu startup code.

There is no replacement for the additional features provided by menu parameters, such as the dialog box that pops up by the MENU_PARAMETER() built-in, but it is simple to emulate this facility in a way that is visually much more appealing by building a simple dialog using Forms.

Obsoleted Menu Built-ins

There are a large number of built-ins within Menus (and Forms) that are principally present to support Full Screen and Character Mode menus. Some of these built-ins already have Forms built-in equivalents. All built-ins mentioned here will be removed and code containing them will not compile, unless mentioned specifically in the notes for a particular built-in.

Menu Built-in	**Notes**
Application_Menu	Full Screen Menu specific.
Application_Parameter	See notes on Menu Parameters.
Background_Menu<n>	Full Screen Menu specific.
Debug_Mode	Long defunct debugging mode used only by menu 5.0. Code containing this built-in will compile in Oracle9*i* Forms but will do nothing.
Disable_Item	Undocumented way of disabling a Menu Item. You can re-code using SET_MENU_ITEM_PROPERTY()[4].

[4] The behavior of Set_Menu_Item_Property is slightly different from the Enable_Item and Disable_Item built-ins when dealing with Menu items that are subject to Menu Security and not currently displayed. Trying to enable or disable these menu items with Set_Menu_Item_Property will generate an FRM-41068 error, which will not be seen with the corresponding Disable/Enable_Item. To prevent this error check that the item is VISIBLE before attempting to Disable or Enable it.

Enable_Item	Undocumented way of enabling a Menu Item. You can re-code using SET_MENU_ITEM_PROPERTY().
Exit_Menu	Full Screen Menu specific.
Hide_Menu	Full Screen Menu Specific.
Item_Enabled	Item Enabled will continue to work after Oracle9*i* Forms upgrade but it will be removed in a future release. You should replace all calls to Item_Enabled with: GET_MENU_ITEM_PROPERTY(<name>, ENABLED) calls
Main_Menu	Full Screen Menu specific.
Menu_Clear_Field	Character Mode specific; use CLEAR_ITEM instead.
Menu_Failure	Undocumented equivalent to FORM_FAILURE flag.
Menu_Help	Full Screen Menu specific.
Menu_Message	Undocumented equivalent to MESSAGE.
Menu_Next_Field	Undocumented equivalent to NEXT_ITEM.
Menu_Previous_Field	Undocumented equivalent to PREVIOUS_ITEM.
Menu_Parameter	See notes on Menu Parameters.
Menu_Redisplay	Character Mode specific.
Menu_Show_Keys	Equivalent to SHOW_KEYS. The upgrade process will make this transformation for you.
Menu_Success	Undocumented equivalent to FORM_SUCCESS flag.
New_Application	Full Screen Menu specific.
New_User	Undocumented built-in. Functionally you can use LOGOUT and LOGON to accomplish the same thing.
Next_Menu_Item	Full Screen Menu specific.
OS_Command	Undocumented equivalent to HOST.
OS_Command1	Undocumented equivalent to HOST.

Previous_Menu	Full Screen Menu specific.
Previous_Menu_Item	Full Screen Menu specific.
Query_Parameter	See notes on Menu Parameters.
Set_Input_Focus	Full Screen Menu specific.
Show_Background_Menu	Full Screen Menu specific.
Show_Menu	Full Screen Menu specific.
Terminate	See notes on Menu Parameters.
Where_Display	Full Screen Menu specific.

COMPONENTS DUE TO BE REMOVED FROM THE SUITE

The following sub-components are planned for removal from the Oracle9*i* Forms installation:

- Oracle Graphics
- Oracle Forms Listener and Load Balancing Components
- Oracle Forms Server Cartridge and CGI
- Oracle Procedure Builder
- Oracle Project Builder
- Oracle Translation Builder
- Oracle Query Builder/Schema Builder
- Oracle Terminal
- Open Client Adapters (OCA)
- Tuxedo Integration
- Performance Event Collection Services (PECS)
- PVCS and Clearcase Integration

Oracle Graphics

Oracle Graphics will not be shipped with Oracle9*i* Forms and the charting wizard will be removed from within the Forms Builder.

It will be possible to upgrade and deploy existing Forms applications that contain embedded Oracle Graphics displays (as chart objects or RUN_PRODUCT calls). For this to work, Oracle Graphics 6*i* will be required and must be installed on the same machine (but in a separate ORACLE_HOME) as Oracle9*i*AS Forms Services.

Customers who have applications which use the Graphics Web Cartridge, or which use Oracle Graphics Runtime should remain on release 6*i*.

ORACLE FORMS LISTENER AND LOAD BALANCING COMPONENTS

With Forms 6*i* patchset 4, Oracle introduced a new way of managing sessions for Forms on the Web using the Forms Listener servlet to create, manage, and clean up the sessions. This removes the requirement to have a separate Forms Listener program to manage sessions. The Forms Listener servlet provides you with several benefits over the previous listener architecture:

- All traffic is directed through the standard Web server HTTP or HTTPS ports, with no extra ports open through the firewall.

- Standard Web server load balancing can be used to distribute the load, alleviating the requirement for a separate load balancing utility proprietary to Forms.
- Broader firewall and proxy support.
- Less administration – the listener and load balancing processes do not need to be managed
- Existing Web server SSL certificate can be used to secure HTTPS traffic. A separate certificate for the Forms listener is not required.
- Support for HTTPS with Internet Explorer 5.x native JVM

Given all of these benefits gained from using the new Forms Listener Servlet we have decided to make this the only supported method of running Oracle9*i* Forms.

Oracle Forms Server Cartridge and CGI

The Oracle Forms Server cartridge (used by the Oracle Application Server - OAS) will no longer be supplied with Oracle9*i* Forms. Its functionality, including load balancing, has been incorporated into the Forms servlet that is introduced with Oracle Forms Release 6*i* patchset 2.

The Forms CGI performs the same function and will also been removed in favor of the servlet

Oracle Procedure Builder

Both the GUI and Line Mode versions of Oracle Procedure Builder will become obsolete in Oracle9*i* Forms. The facilities for editing and debugging local and server-side PL/SQL code are already available in the Form and Reports Builders, which currently share the same PL/SQL engine and debugger as Procedure Builder.

In the Oracle9*i* Forms timeframe, Oracle will considerably improve the PL/SQL debugging facilities within the Forms Builder to further improve user productivity.

Oracle Project Builder

Project Builder will be removed in Oracle9*i* Forms. Oracle has no immediate plans to provide its functionality in the form of another product, but in the long term Oracle intends to change and improve the Form Builder interface to be project-based rather than module-based, thus providing equivalent functionality.

Oracle Translation Builder

Oracle Translation Builder provides a way to translate resource strings within Forms modules so that the same module can be deployed in multiple languages.

Oracle Translation Builder will be replaced by a new translation tool TranslationHub, which is currently in use within Oracle for translating the database and the Oracle Applications Suite.

Oracle Query Builder and Schema Builder

Query Builder and Schema Builder will not be shipped as stand-alone products in Oracle9*i* Forms.

Oracle Terminal

Oracle Terminal is used to create and edit terminal resource files (key mapping files) for client server and character mode versions of Forms. Forms deployed through Oracle9*i* Application Server Forms Services use a flat text file for key mappings and do not require a special tool to edit the file. As Oracle Terminal will no longer be required due to the obsolescence of client-server and character mode, it will not be shipped with Oracle9*i* Forms.

Open Client Adapters (OCA)

The Open Client Adaptors provide a way of running Forms against non-Oracle data sources using ODCB connections. The use of OCA is limited to desktop machines running Microsoft Windows. The Oracle Transparent Gateway and Generic Connectivity solutions should be used to replace the use of OCA.

As well as providing a wider range of non-Oracle data sources over OCA, deployment of the Oracle9*i*AS Forms Services using Oracle Transparent Gateways is not restricted to deployment onto Windows NT or Windows 2000.

Tuxedo Integration

Integration with the Tuxedo TP monitor from BEA systems is provided in the form of an additional operating system library and PL/SQL library that are integrated into client applications that wish to interface with Tuxedo. There has not been sufficient uptake of this component to warrant its further development, and will not be shipped with Oracle9*i* Forms.

Performance Event Collection Services (PECS)

PECS is used as a low impact, performance data collection facility for Forms. PECS will become obsolete in Oracle9*i* Forms.

PVCS and Clearcase Integration

In Oracle9*i* Forms, the only source control capability that will be directly interfaced into the Forms Builder is for Oracle9i SCM. Other source control systems such as PVCS and Clearcase can still be used externally from the Form Builder to provide source control services.

APPENDIX A - LOGICAL AND GUI ATTRIBUTES DROPPED IN ORACLE ORACLE9*i* FORMS

Attribute Name	Context Where Attribute Is Used
ToolkitDisabled	Generic attribute
ToolkitEnabled	Generic attribute
ToolkitCurrent	Generic attribute
ToolkitDisabledMnemonic	Generic attribute
ToolkitEnabledMnemonic	Generic attribute
ToolkitCurrentMnemonic	Generic attribute
NormalAttribute	Normal background for windows
Normal	Text item
Bold	Bold for all items (including check boxes)
Bold-text	Boilerplate
Bold-inverse	Inverse bold for all items
Underline	Underline for all items
Boilerplate	Constant text
WindowTitleCurrent	Title of active window
Menu	Selected menu
Sub-menu	Selected submenu
Full-screen-title	Screen title
Menu-title	Current menu title
Menu-subtitle	Current menu subtitle
Menu-bottom-title	Current title at bottom of menu
MenuItemEnable	Enabled, non-current menu item
MenuItemDisabled	Disabled menu item
MenuItemSelect	Current menu item
MenuItemEnableMnemonic	Mnemonic of an enabled menu item
MenuItemDisableMnemonic	Mnemonic of a disabled menu item
MenuItemSelectMnemonic	Mnemonic of the current menu item

TextControlCurrent	Current field or text editor
Field-current	Color for current text item
TextControlNonCurrent	Disabled or non-current field or text editor
Field-non-current	Color for text item that is not currently selected
Field-Queryable	Queryable field in Enter-Query mode
ItemQueryDisabled	When a Block goes into Enter-Query Mode, any non-queryable items will inherit this set of attributes.
TextControlFailValidation	When an item fails a validation check, it will be set to this attribute set.
TextControlSelect	Selected text in an enabled field or text editor
Field-selected-current	Currently selected text item
Field-selected-non-current	Text item that is not currently selected
PushButtonDefault	Default or current button
PushButtonNonDefault	Button that is not default
Button-non-current	Non-current button
Button-current	Current button
Alert	Alert text
AlertIcon	Icon in an alert window
AlertMessage	Message text in an alert window
AlertBackground	Alert background
Status-Message	Controls the font of any message appearing on the Status Line.
Status-Hint	Controls the font of any item hint appearing on the Status Line.
Status-Empty	Controls the look of the empty Status Line.
Status-Items	Controls the look of the Operator Information Area which contains the LOV lamp, record count, etc.
Listtitle	List of Values (LOV) title
ListItemSelect	Selected item in a text list

ListItemNonSelect	Unselected item in a text list
ListPrefix	List prefix
ScrollThumb	Elevator box on scroll bar
Scroll-bar-fill, Inverse, Inverse-underline, Bold-underline, Bold-inverse-underline	These logical attributes are not unique to Forms Builder. As a result, these logical attributes can be overridden by the visual attributes defined by the window manager.

Oracle9*i* Forms: Features Obsolescence
Version 1.20
February 2002
Author: Duncan Mills
Contributing Authors: Emerson DeLaubenfels

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com

Oracle Corporation provides the software that powers the internet.