

AppsForm

Training Guide

Software Version 6.5.5

© 2005 Logical Apps

All rights reserved. Printed in USA.

Restricted Rights Legend

This software and associated documentation contain proprietary information of Logical Apps. It is provided under a license agreement containing restrictions on use and disclosure and it is also protected by copyright law. Reverse engineering of this software is prohibited.

The information contained in this document is subject to change without notice. Logical Apps does not warrant that this document is error free. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Logical Apps.

Logical Apps provides on-site support as well as remote phone and web support to ensure quick and effective product implementation. To request support, to suggest product enhancements, or to comment on Logical Apps software or documentation, send email to support@logicalapps.com, or contact us at the address or phone number given below.

AppsAccess, AppsAudit, AppsControl, AppsExtend, AppsForm, AppsFlow, AppsRules, and Rapid Compliance are trademarks of Logical Apps. All trademarks and registered trademarks are the property of their respective owners.

Document Version AR004-655A

11/2/05

Logical Apps
15420 Laguna Canyon, Suite 150
Irvine, CA 92618
949.453.9101

Contents

About AppsForm	1
Prerequisites	2
AppsForm in Other AppsRules Applications	2
Determining Internal Names for Items Used in Rule Elements	2
Starting AppsForm.....	3
Creating a Rule and Its Elements.....	5
Creating a Rule.....	6
Beginning to Create a Rule Element	6
Using the Event Tracker	7
Capturing Items from a Form	7
Using the Event Tracker to Set Security	7
Completing the Rule Element Definition.....	8
Configuring Rule Element Details	11
Setting Security	11
Selecting Components	12
Assigning Security Attributes.....	13

Setting Navigation Paths.....	14
Creating Menu Links	15
Special Cases	16
Creating Zooms.....	17
Creating Messages.....	17
Setting Default Values.....	18
Creating and Modifying Lists of Values	19
Altering an Existing LOV	19
Creating a New LOV.....	21
Setting Field Attributes	22
Creating SQL Procedures	25
Running AppsFlow Processes	26
Saving Rule Element Details	28
Creating Subscribers.....	29
Defining a Subscriber.....	30
More Subscriber Fields	31
Subscriber Lists	32
Using the SQL Wizard	33
Starting a SQL Statement.....	34
Using the Definition Panel	34
Using the Return Columns Panel	36
Using the SQL Panel	37
AppsForm Migration	39
Preparing for Migration	39
Stipulations.....	41
Migrating, Exporting, or Copying Rules or Libraries	41
Importing a File Containing a Rule, Elements, or a Library	44
Running AppsForm Utilities	45
Using AppsExtend	45
Creating a New Form.....	46
Linking a New Form to an Existing Form.....	47
Creating LOVs for an AppsExtend Form	47

Collecting Rules in Libraries	48
Creating a Library	49
Adding to Value Sets	50
Using the Mass Associate Utility.....	52
Rules Caching	52
Adding Custom Events	53
Support	55

About AppsForm

AppsForm enables users to write rules that customize Oracle Applications forms, modifying their security, navigation, field, and data properties.

Each AppsForm rule consists of subordinate rules, called “rule elements.” Each of these elements may target a form, a block within a form, or a field within a block, and each specifies an “event” that triggers processing — for example, the act of opening a target form or navigating to a target block or field. Finally, each element defines customizations to the target form, or to its blocks, fields, tabs, or other components. Rule elements can do the following:

- Set security attributes. These can mandate that data entry be required; that updates, insertions, or deletions be prevented; or that items be hidden from view.
- Establish navigation paths from a target form to other Oracle Applications forms or made-to-order forms created through use of a tool called AppsExtend.
- Display messages.
- Define default values for fields, compile lists of values to be selected from fields, or set other field attributes.
- Run structured query language (SQL) statements.
- Execute processes defined in Logical Apps AppsFlow. (For information on creating such processes, see the *AppsFlow Training Guide*.)

Moreover, AppsForm users specify “subscribers.” Each is an entity, such as a user or a responsibility, to which an AppsForm rule (or a rule element) applies. Thus, a form,

block, or field may be customized in varying ways. For example, a security element may hide a field from some users, but present it to others.

Prerequisites

You are assumed to have a basic understanding of the Oracle Applications modules for which your organization deploys AppsForm. Although the creation of AppsForm rules does not require a knowledge of programming languages, AppsForm tools permit the direct manipulation of structured query language (SQL) code, and a knowledge of SQL is helpful. Moreover, you are expected to have some knowledge of the relationships among tables and views (and their primary keys) in your Oracle database.

AppsForm in Other AppRules Applications

AppsForm includes a rule, called ICX: APPSACCESS User Rules, that supports processing in AppsAccess. Do not alter or delete it. AppsForm also provides access to rules created in AppsControl; the name of each begins with the prefix “LA_CC_.” You may create subscribers for these rules (see the *AppsControl Training Guide*), but should not otherwise modify them in AppsForm.

Determining Internal Names for Items Used in Rule Elements

As you work with AppsForm, you will often need to know the “internal” names for Oracle Applications forms, blocks, and fields — for example, APXVDMVD for the Enter Vendor form, VNDR for the block on the Enter Vendor form that contains a Supplier Name field, and VENDOR_NAME_MIR for the field itself. To discover the internal name for an Oracle Applications form, complete these steps:

- 1 Navigate to the form whose internal name you want to know.
- 2 Select Help in the menu bar, then About Oracle Applications in the Help menu.
- 3 An About Oracle Applications window opens. In it, scroll to the Current Form section and take note of the Form Name — in the following example, APXVDMVD.



- 4 Click on the OK button to close the form.

To discover the internal names for blocks and fields, complete these steps:

- 1 Click on a field if you mean to determine either its name or the name of the block that contains it.
- 2 Select Help in the menu bar, then Diagnostics in the Help menu, then Examine.
- 3 An Enable Diagnostics dialog prompts for the Oracle password. Enter the password for your instance and click on the OK button.
- 4 An Examine Field and Variable Values form opens. Take note of the internal block and field names for your selection.

The screenshot shows a dialog box titled "Examine Field and Variable Values". It has three input fields: "Block" with the value "VNDR", "Field" with the value "VENDOR_NAME_MIR", and "Value" which is empty. Each input field has a small menu icon to its right. At the bottom right of the dialog are "OK" and "Cancel" buttons.

- 5 Click on the OK button to close the form.

Starting AppsForm

To open AppsForm:

- 1 Select the Logical Apps AppsRules responsibility in the Oracle Applications list. (Ensure first that the AppsRules responsibility is available to you.)
- 2 A Logical Apps — AppsRules form appears. In it, ensure that the AppsForm panel is active (this is the default selection). If you move to another of the AppsRules panels, you can return to AppsForm by clicking on its tab.

If you close the AppsRules form, you can reopen AppsForm:

- 1 In the Logical Apps Navigator, click on the AppsRules option and then on the Open button. (Or, double-click on the AppsRules option.)
- 2 Once again, ensure that the AppsForm panel is active in the AppsRules form.

Creating a Rule and Its Elements

The first steps in working with AppsForm are to name a rule and to provide basic information for one or more of its elements. That information includes a target and an event that initiate processing of the element.

As the target for an element, you always select, at minimum, an Oracle Applications form; depending on the event you intend to call, you may also specify a block or a field on the form. You can, for example, choose a When New Form event, which triggers the rule element to run each time a user opens a specified form. If you do, you need designate only the form as a target for the element. Or, among several other events, you can choose When New Item, which triggers the rule element to run each time a user navigates to a field. If you do, you typically designate not only a form, but also a block within the form and finally a target field within the block.

To select a block or a field, you must first use a specialized event — the Event Tracker — to “capture” the blocks and fields that belong to the form.

Once you’ve set up the rule and its elements, you configure details for each of the elements — define how each element modifies the a target form, blocks, fields, or other items. You also configure subscribers — determine who or what is affected by the rule. Later chapters discuss the creation of these items.

Logical Apps - AppsRules

AppsForm AppsFlow AppsAudit

Rule Name	Description	Debug	Active	Subscribers
				Exist
Supplier Security	Sets security attributes for the Enter Vendor	☐	☒	☐
		☐	☐	☐

Rule Elements

Seq	Element Name	Form Name	User Form Name	Event	Block Name	Field Name	Debug	Active
10	Enforce Uppercase	APXVDMVD	Enter Vendor	Event Tracker			☐	☒
							☐	☐
							☐	☐
							☐	☐
							☐	☐
							☐	☐
							☐	☐

Description

Rule Subscribers Element Subscribers Details

AppsForm does not recognize blocks, fields, or other items on a form until you run the Event Tracker. If the target of an element is to be a block or field, or if you expect to cite specific items as you define how an element modifies its target, use the fragment of the element you have created so far as a vehicle to run the Event Tracker.

Using the Event Tracker

Use the Event Tracker to “capture” blocks and fields for either of two purposes: for selection in the Block Name and Field Name fields of the main AppsForm window as you set the target of a rule element, or for selection later as you define how the element modifies a target form or items on it. Moreover, as you run the Event Tracker, you can set some security attributes for the target form.

Capturing Items from a Form

To capture blocks, fields, or other form items, complete these steps:

- 1 Fill in the Seq, Element Name, and Form Name fields in a row of the Rule Elements grid (as described in “Beginning to Create a Rule Element,” page 6).
- 2 In the Event list of values, select the value *Event Tracker*.
- 3 Respond to two messages that appear when you select the Event Tracker:
 - The first provides brief instructions on the use of the Event Tracker. After reading the message, click its OK button to close it.
 - In the second, click the Append button to add items to an existing collection of “metadata” (items already captured) for the form you have chosen. Or select the Replace button to discard older metadata and begin a new collection.
- 4 Save the rule: Click on File in the menu bar, then on Save in the file menu.
- 5 Open the Oracle Application that uses the form you have chosen. Navigate to the form; in it, navigate to each block and field you may want to select as you build AppsForm rule elements. Create or update a record and save your work.

By doing so, you capture a reference to each item you touch, as well as to “undocumented” events associated with the form. The items you capture become available in AppsForm in any rule element that targets the form, not only in the element from which you launched the Event Tracker.

Using the Event Tracker to Set Security

When you open an Oracle Applications form for which you are running the Event Tracker, an AppsRules Actions menu provides options for setting security properties for items on the form. (If you set security properties from within AppsForm, though, you have a wider selection of options. See “Setting Security,” page 11.) To use the menu:

- 1 Create an AppsForm rule element, select the Event Tracker, and open the Oracle Applications form that is the target of the element (see “Capturing Items from a Form,” above).

- 2** Click on a field for which you want to set security, or one that exists in a block or tab for which you want to set security.
- 3** Click on AppsRules Actions in the menu bar, and then on any of the following options.
 - Prevent Update to Block: Prevent an existing value from being changed for any field in the block where the cursor is located.
 - Prevent Insert to Block: Prevent an original value from being entered for any empty field in the block where the cursor is located.
 - Prevent Update to Field: Prevent an existing value from being changed for the selected field.
 - Hide Field: Remove the selected field from the form.
 - Make This Field Required: Prevent a user from selecting a new record or closing a form if no value has been saved in the selected field.
 - Enforce Uppercase on This Field: Require that data entered in the selected field be all upper case
 - Hide This Tab: Remove the tab that contains the selected field, and all the fields associated with it, from the form.
 - Get Field Properties: Capture the properties of the selected field. (This is essentially the same as simply navigating to the field with the Event Tracker running.)
- 4** A message informs you that a rule is created in AppsRules. Click on the OK button to close the message

The security attributes you configure through use of the Event Tracker take effect when you complete the definition of the Rule Element from which you are running the Event Tracker. (See the next section.)

Completing the Rule Element Definition

To complete the rule element, open AppsForm (if you've closed it to apply the Event Tracker to an Oracle Applications form) and select the rule with which you want to work. If you've created a fragmentary element for the purpose of running the Event Tracker, select that element. If you're creating a new element from scratch, perform steps 1–3 in "Beginning to Create a Rule Element" (page 6). Then:

- 1** In the Event field, select (or replace the value *Event Tracker* with) an event that determines the circumstances under which the rule element is to be evaluated. Choose among the following:
 - When New Form: The element fires whenever a user opens its target form. If you select this event, you cannot enter values in the Block Name and Field Name fields.

- **When New Block:** The element fires when a user navigates from one block to another in the target form. Or, if you select a value in the Block Name field (which is recommended), the element fires when a user navigates to the specified block.
- **When New Item:** The element fires when a user navigates from one field to another in the target form. Or, if you select a value in each of the Block Name and Field Name fields (which is recommended), the element fires when a user navigates to the specified field.
- **When New Record:** The element fires whenever a user navigates from one record to another (whether new or existing). You may select a block name if you want to restrict the firing to the selection of a record within the specified block.
- **When Validate Record:** The element fires whenever a user saves a record. You may select a block name if you want to restrict the firing to the saving of a record while the cursor is located in the specified block.
- **Zoom Special:** This special Logical Apps event makes a zoom regardless of subscribers. It ignores the subscribers until the moment the zoom is pressed. So it enables conditional use of the zoom to navigate to different entities. In essence the zoom shows up regardless of the subscribers, but does not function when a user tries to zoom and the subscriber evaluates as false.
- **Undocumented events:** Undocumented events associated with the target form appear in the Event list of values if they have been captured by the Event Tracker.



Note

You can also capture undocumented events manually (see page 53). No matter how such events are captured, however, Oracle does not support them and the installation of a patch may cause them to disappear.

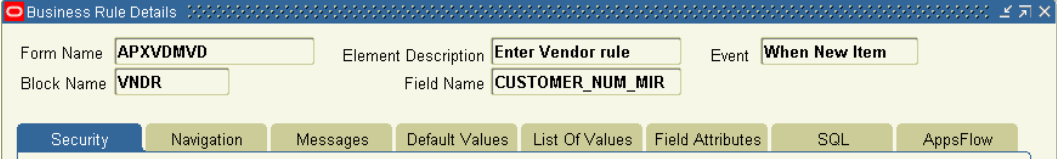
- **Audit:** The Audit event is no longer supported. Do not select it.
- 2** If you want the element to target a block or a field, and you've selected an event that allows it to do so, select the block in the Block Name field, which offers a selection of values captured by the Event Tracker.
 - 3** If you want the element to target a field, and you've selected an event that allows it to do so, select the field in the Field Name field, which offers a selection of values that have been captured by the Event Tracker and that exist in the block you selected in step 2.
 - 4** Select the Debug text box to cause AppsForm to display messages as the rule element is being run. Or clear the text box to prevent the display of such messages.
 - 5** Select the Active check box to make the element active, or clear the text box to hold the element in reserve.
 - 6** Save the rule. Click on File in the menu bar, then on Save in the File menu.

Configuring Rule Element Details

Once you've created a rule element — selected its target form, block, or field, and chosen the event that triggers its use — you need to define what it does.

Click on the element in the Rule Elements grid of the main AppsForm window, and then click on the Details button. A Business Rule Details form appears. Click on its tabs to expose panels in which you can assign security attributes; set navigation paths; create messages; define default values, lists of values, or other field attributes; run SQL statements; or run processes defined in AppsFlow.

First, however, note that certain fields are already completed, reflecting the selections you made for the rule element in the main AppsForm window. These include the Form Name, Element Description, Event, Block Name, and Field Name fields near the top of the form. You can alter these values only indirectly, by changing rule-element values in the main AppsForm window.



Setting Security

You can assign security attributes to forms, blocks, tabs, fields, and descriptive flexfields (DFF). Attributes are available to each of these components in varying

combinations. You can restrict the ability to update, insert, or delete data; require that data be entered or that text entries be in upper or lower case; or hide screen elements.

To set these security attributes, use the Security panel, which is selected by default when you open the Business Rule Details form. If you navigate away from it, you can click on the Security tab to return to it:

Type	Block/Tab	Field Name	Case	Required	No Update	No Insert	No Delete	Hide	Active
Field	VNDR	VENDOR_NAME_MIR		☐	☐	☐	☐	☐	☒
				☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐
				☐	☐	☐	☐	☐	☐



Notes

You can use the Event Tracker to set *some* security attributes (see page 7). If you have done so, each of the settings occupies a row in the Security panel.

You can also set security attributes for field instances, but to do so, you would use the Field Attributes panel (see page 22), not the Security panel.

Selecting Components

In each row of the security grid, select a component whose security attributes you want to set.

- 1 In the Type list box, choose whether you want to set security attributes for the target Form, or for a Block, Tab, Field, or DFF on the form.
- 2 In the Block/Tab and Field Name fields, select the component whose type you identified in step 1:
 - If you selected the Form type, leave both fields blank.
 - If you selected the Block, Tab, or DFF type, choose a value in the Block/Tab field and leave the Field Name field blank.
 - If you selected the Field type, choose values in both fields.

Alternatively, if you intend to assign security attributes to a number of fields, you can select them all at once:

- 1 Click on Tools in the menu bar, and then on AppsRule Form Elements in the Tools menu. The Form Elements window appears, as shown in the illustration at the top of the next page.

Block Name	Prompt Text	Field Name	Item Type	Item Canvas	Tabbed Canvas	Include Flag
VNDR	Distribution Set	DISTRIBUTION_SET_NAM	TEXT ITEM	ACT		☐
VNDR	Allow & Withholding T	ALLOW_AWT_FLAG_MIR	CHECKBOX	AWT		☐
VNDR	&One Time	ONE_TIME_FLAG_MIR	CHECKBOX	CLASS		☐
VNDR	&Woman Owned	WOMEN_OWNED_FLAG	CHECKBOX	CLASS		☐
VNDR	Minority Owned	MINORITY_GROUP_DISP	TEXT ITEM	CLASS		☐
VNDR	SIC	STANDARD_INDUSTRY_C	TEXT ITEM	CLASS		☐
VNDR	Small & Business	SMALL_BUSINESS_FLAG	CHECKBOX	CLASS		☐
VNDR	Type	VENDOR_TYPE_DISP_MIR	TEXT ITEM	CLASS		☐

Buttons: Select All, De-Select All, Accept, Close

This form displays a selection of fields that depends on the choices you made as you created the rule element in the main AppsForm window:

- If you left the Block Name and Field Name fields blank as you created the rule element in the main AppsForm window, the Form Elements window shows all fields captured by the Event Tracker, from all blocks on the target form.
 - If you selected a Block Name but not a Field Name as you created the rule element in the main AppsForm window, the Form Elements window shows all the fields from the selected block that were captured by the Event Tracker.
 - If you selected a Block Name and a Field Name as you created the rule element in the main AppsForm window, the Form Elements window shows only the selected field.
- 2 For each field you want, click on the Include Flag check box. Or, to select all fields, click on the Select All button. (The De-Select All button removes check marks from all check boxes.)
 - 3 When you are satisfied with your selection, click on the Accept button. The Form Elements window closes, and the fields you chose appear in the Security grid of the Business Rule Details form. (You can click on the Close button to close the Form Elements window without accepting any selected fields.)

Assigning Security Attributes

For each of the components you've selected, assign security attributes. For the most part, these are controlled by the Case field and the check boxes, which are available to the component types in varying combinations:

Attribute	Description	Available to				
		Field	DFF	Block	Form	Tab
Case	Field values must be entered in upper case (Upper), lower case (Lower), or mixed case (blank)	×				
Required	If the check box is selected, a field value must be entered	×				

Table continues on the next page.

Attribute	Description	Available to				
		Field	DFF	Block	Form	Tab
No Update	If the check box is selected, existing values cannot be changed	×	×	×	×	
No Insert	If the check box is selected, new values cannot be entered	×		×	×	
No Delete	If the check box is selected, existing values cannot be deleted			×	×	
Hide	If the check box is selected, the screen component is hidden from user's view	×	×			×
Active	If the check box is selected, the attributes selected in the Case field and other check boxes become active	×	×	×	×	×

For the Block component type, you can also enter values in two text boxes that appear only when that type is selected:

- **Default Where:** Enter a SQL “where” statement that creates a filter — the block can display only records for which a field is set to a value specified in the SQL statement. For example, where *vendor_type_disp* = 'Employee'
- **Order By:** A block may present multiple records. A user may, for example, execute a query — search for records with a field set to a value that matches a search value, then load records into the form one by one for display. Or, a block may present data in a grid format, with each column corresponding to a field and each row containing related values for a set of fields — a record.

For such occasions, you can designate an order for data records returned to the block. Choose a field, and its values are arranged in alphanumeric order; the values for other fields are appropriately arranged so that records remain intact. Choose a second field to determine the sort order for records in which values for the first field are identical. Continue specifying any number of fields. Use the following syntax: *Order by field_1, field_2, ... field_x*

Setting Navigation Paths

You can create entries in the Tools, Actions, or Reports menu of a target form, each of which, when clicked, opens another form (or, in a special case, executes an AppsForm rule element). You can also create “zooms” — similar links that are activated when a user clicks on the Zoom button in the tool bar.

Typically, such a link becomes active when a form is first opened, and so you would create such links for rule elements that use the When New Form event. Moreover, a navigational link works only if the source and destination forms are both available within a single responsibility. If a user does not have access to a form, a navigational link created in AppsForm will not take him there.

To create navigation links, click on the Navigation tab in the Business Rule Details form:

Business Rule Details

Form Name: Element Description: Event:

Block Name: Field Name:

Security | **Navigation** | Messages | Default Values | List Of Values | Field Attributes | SQL | AppsFlow

Sequence	Label	To Function	Parameters	Icon Name	Apprules	Disable	Special	Menu	Active
Tools10	Vendor Online Audit	AppsAudit Online	P_TABLE_NAME=PO_VEND		☐	☐	☐	☐	☒
					☐	☐	☐	☐	☐

Zooms

From Block	To Function	Parameters	Active
			☒
			☐

Done

Creating Menu Links

To add a navigation link to a menu in the target form, complete a row in the Menus section:

- 1 In the Sequence field, select a sequence number prefixed by the name of the menu to which you want to add the link. (The higher the number you select, the more remote is the possibility of overwriting an existing menu option.)
- 2 In the Label field, type a name for the link. This name will appear as an option in the menu you selected in step 1.
- 3 In the To Function list of values, select the user function name that corresponds to the form to which you are creating a link. To ascertain the user function name:
 - a Determine the internal name for the form to which you are creating a link (see page 2).
 - b Switch to the Application Developer responsibility and select the Application/Form option. Using the form name you determined in step a, query on the Form field and make a note of the corresponding value in the User Form Name field. (To query, press the F11 key. Type the value for which you are querying in the appropriate field, and then press Ctrl+F11.)
 - c Still in the Application Developer responsibility, select the Application/Function option. In the Form field of its Form tab, query on the user form name value you determined in step b. Then click on the Description tab and make a note of the value in the User Function Name field.
- 4 If your function takes any parameters, the Parameters field displays a template indicating what those parameters are. If so, replace the placeholders (text surrounded by angle brackets) with actual values. If the Parameters field remains

blank after you select a function, you need not supply parameters; you have the option, however, of entering `QUERY_ONLY="YES"` to make the destination form open in query-only mode.

- 5 In the Icon Name field, accept the default value.
- 6 Ensure that the Active check box is selected.

Special Cases

Two checkboxes enable you to adapt navigation links to specialized purposes. First, a Disable Menu option turns off an existing menu item, even one supplied in a standard Oracle menu:

- 1 Ensure that you are working with a rule element that targets the form whose menu option you want to disable, and that it uses the When New Form event.
- 2 In the Sequence field, select the value assigned to the menu option you want to disable. (You can determine the appropriate number by opening the menu in question and observing the position of the option.)
- 3 In the Label field, type the label assigned to the menu option you want to disable.
- 4 In the To Function field, select the user function name that corresponds to the form whose menu option you want to disable. (See step 3 in the procedure just before this one.)
- 5 Select the Disable Menu check box. (Make sure also that the Active check box is selected.)

Second, you can create a link to an AppsForm rule element — for example, one that runs a SQL script — so that users can click on a menu option to run that element:

- 1 Ensure that you are working with a rule element that targets the form from which you want the menu option to appear, and that it uses the When New Form event.
- 2 In the Sequence and Label fields, select a sequence number prefixed by the name of the menu to which you want to add the option, and the label for the option, as normal.
- 3 Leave the To Function, Parameters, and Icon Name fields blank.
- 4 Select the Appsrules Special check box. (Make sure also that the Active check box is selected.)
- 5 Return to the main AppsForm window to create a new rule element. You will discover that the selection of events available to that element includes the value *Special*, followed by the sequence number you selected in step 2. Choose that event, and leave the Block Name and Field Name fields blank. Use the Business Rule Details form to define what you want that rule element to do.

When that process is complete, a user can click on the menu item you created to execute the rule element you created in step 5.

Creating Zooms

A Zoom enables a user to move from a block to another form by clicking on the Zoom button in the Oracle Applications tool bar. You can create only one zoom per block. To do so, use the Zooms section of the Navigation panel:

- 1 In the From Block list of values, select the block from which you want to enable the zoom. The LOV presents all of the blocks for the target form that you have captured through use of the Event Tracker.
- 2 In the To Function list of values, select the user function name that corresponds to the form for which you are creating a link. (Once again, you can use Application Developer features to determine the correct value; see page 15.)
- 3 Once again, if the function takes any parameters, the Parameters field displays a template indicating what those parameters are. If so, replace the placeholders (text surrounded by angle brackets) with actual values. If the Parameters field remains blank after you select a function, you need not supply parameters; you have the option, however, of entering `QUERY_ONLY="YES"` to make the destination form open in query-only mode.
- 4 Ensure that the Active check box is selected.

Creating Messages

You can write messages that appear when a user performs an action corresponding to the event you have chosen for a rule element — for example, opening a form, navigating to a field, or saving a record. Click on the Messages tab in the Business Rule Details form:

Business Rule Details

Form Name: **APXVDMVD** Element Description: **Enter Vendor Attributes** Event: **When New Form**

Block Name: Field Name:

Security Navigation **Messages** Default Values List Of Values Field Attributes SQL AppsFlow

Sequence	Description	Message Type	Active
10	Remind users that new business rules are in place	Note	☒
			☐
			☐

Message

New business rules have been implemented, and affect this form. You may no longer be able to update some fields. If you need to alter data in such a field, contact your supervisor.

Done

To create a message:

- 1 In the Sequence field, enter a number that reflects the order in which you want this message to appear in relation to other messages you may create in other rows.

- 2 In the Description field, briefly explain the purpose of the message.
- 3 In the Message Type list box, select one of the following:
 - Note: The message appears, but allows the user to continue work.
 - Error: The message appears and prevents the user from saving a record. As a result, select this type for messages associated with rule elements that use the When Validate Record event.
- 4 Write the message in the Message text box. A message can contain not only text, but also field names; at run time, the names are replaced by values associated with the currently selected record. Use this syntax: `#:BLOCK.FIELD_NAME#`



Note

A message can be made to appear only when certain data conditions are met — for example, a promotional message appears when a user enters a certain item on an order. To make this happen, create the message for a rule element based on the When New Item event; create an element subscriber with a Data filter type so that the message appears only when the correct data is entered.

Setting Default Values

You can set the default values of fields in the form that is the target of a rule element. To do so, click on the Default Values tab in the Business Rule Details form:

The screenshot shows the 'Business Rule Details' window with the 'Default Values' tab selected. The form configuration is as follows:

Block	Field	Default Type	Default Value	Active
VNDR	VENDOR_NAME_ALT_M	Form	VNDR.VENDOR_NAME_MIR	☒
VNDR	CUSTOMER_NUM_MIR	Static	909090909	☒
GLOBAL	V_LA_ORG_ID	SQL	select fnd_profile.values('org_id') from dual	☒
				☐
				☐
				☐
				☐
				☐

At the bottom of the window is a 'Done' button.

Regardless of the event you select to trigger the rule element, you can set values for any number of fields in any number of blocks on the form (providing, of course, that the fields and blocks have been captured through use of the Event Tracker). Devote one row in the grid to each field:

- 1 In the Block list of values, select the block that contains the field for which you want to set a default value.
- 2 In the Field list of values, select the field for which you want to set a default value.

- 3** In the Default Type list box, select one of these values:
 - Static. The default value is a constant.
 - Form. The default value is a copy of the value entered for another field on the form.
 - SQL. The default is a value returned by a SQL statement.
- 4** In the Default Value field, type an entry appropriate for the selection you made in the Default Type list box:
 - If you selected Static, type the value that serves as the default.
 - If you selected Form, identify the field that returns a default value. Use the format `BLOCK_NAME.FIELD_NAME` — the internal names (see page 3) for the block that contains the field and the field itself.
 - If you selected SQL, type a SQL statement that returns values for use as defaults.
- 5** Ensure that the Active check box is selected.

Creating and Modifying Lists of Values

You can both alter existing lists of values or create new LOVs. Before you do so, you must run the Event Tracker on fields for which you want to create or modify LOVs.

Altering an Existing LOV

To alter an existing LOV is to select the field and then modify the SQL statement that compiles the values displayed in the field. You cannot, however, modify the “select” portion of the SQL statement, which identifies the database columns that return values to the LOV. You can alter only the “where” and “sort by” portions of the statement, which specify the conditions under which records are selected to be returned, and the order in which they are arranged.

- 1** In AppsForm, ensure that you have created a rule and rule element that you want to use to modify the LOV. The element must select, as a target, the form on which the LOV exists. (It’s often the case that this element would use When New Form as an event; if so, the form is all you need select as a target. If you choose an event that requires you to do so, however, also choose a block and/or a field.)
- 2** Navigate to the form that contains the LOV you want to change, and click in the LOV.
- 3** Run a trace file:
 - a** Click on Help in the menu bar, then Diagnostics in the Help menu, and then Trace in the Diagnostics submenu. Select the Regular Trace radio button. (If you have not already used a Diagnostics option, an Enable Diagnostics dialog prompts you for your Oracle password. Enter it, and click on the OK button

- to clear the dialog. A note informs you that tracing is activated and provides the path and name of a trace file. Click on the OK button to clear the note.)
- b** In the LOV you want to change, select any value.
 - c** Click on Help in the menu bar, then Diagnostics in the Help menu, and then Trace in the Diagnostics submenu. Select the No Trace radio button. (Another message informs you that tracing is deactivated and provides the path and name of a trace file. Click on the OK button to clear the note.)
- 4** Open AppsForm and use a utility called TKProf to examine your trace file:
- a** Click on Logical Apps Utilities in the menu bar, and then on AppsRules TKProf Utility in the Utilities menu.
 - b** A concurrent request runs, and a message informs you of its identification number. Make a note of the number and click on the OK button to close the message.
 - c** Click on View in the menu bar, and then on Requests in the View menu.
 - d** A Find Requests form opens. Click on the Specific Requests radio button and, in the Request ID field, enter the ID number for your request. Click on the Find button.
 - e** A Requests form appears; its grid contains an entry for your request. When its status is Completed (you may need to click the Refresh Data button), click on the View Log button.
 - f** In the log file, search for the SQL statement that generates values for the LOV. (Typically, it begins, “select displayed_field,description,lookup_code from”). Leave the log file open.
- 5** In AppsForm, select the rule and element you want to use to modify the LOV. Click on the Details button and, in the Business Rule Details form, click on the List of Values tab.

Business Rule Details

Form Name: **APXVDMVD** Element Description: **LovTest** Event: **When New Form**

Block Name: Field Name:

Block Name	Field Name	Record Group	LOV Name	Active
VNDR	VENDOR_TYPE_DISP_MIR	LA_VENDOR_TYPES	VENDOR_TYPES	☒
				☐
				☐

SQL Text

```
select displayed_field,description,lookup_code
from
po_lookup_codes where lookup_type = 'VENDOR TYPE' and sysdate <
nvl(inactive_date, sysdate + 1) order by upper(displayed_field)
```

Done

- 6 In the Block Name field, select the block where the LOV exists; in the Field Name field, select the field where the LOV exists. Accept default values in the Record Group and LOV Name fields, and be sure the Active check box is selected.
- 7 In the log file, copy the SQL statement for your LOV (highlight it and press Ctrl+C). Then, in the AppsForm Business Rule Details form, click on the SQL Text area of the List of Values panel. Press Ctrl+V to paste the SQL statement there.
- 8 Close the log file (click on the × symbol in its upper right corner). In the AppsForm List of Values panel, edit the SQL statement as you desire. Remember that you can modify only the “where” and “sort by” clauses. If bind variables exist in the statement (they may appear as :1 or :5), you may have to open the form to identify the actual SQL that is being executed.

Creating a New LOV

To create a new LOV is to convert an existing text-entry field for use as a list of values. The process involves identifying the field (after first having used the Event Tracker to capture it) and creating a SQL statement that compiles values the field is to display.

- 1 Ensure that you have created an AppsForm rule and rule element that you want to use to create the LOV. The rule element must specify, as a target, the field you intend to make into a list of values (and therefore, of course, the block and form that contain the field), and it must use the When New Item event.
- 2 In AppsForm, select the rule and element, click on the Details button and, in the Business Rule Details form, click on the List of Values tab.

The screenshot shows the 'Business Rule Details' window with the 'List of Values' tab selected. The form contains the following fields and sections:

- Form Name:** APXVDMVD
- Element Description:** LovCreate
- Event:** When New Form
- Block Name:** VNDR
- Field Name:** STANDARD_INDUSTRY_4

The 'List of Values' tab is active, showing a table with the following columns: Block Name, Field Name, Record Group, LOV Name, and Active.

Block Name	Field Name	Record Group	LOV Name	Active
VNDR	STANDARD_INDUSTRY_CL	LA_STANDARD_INDUSTRY	APPCORE_ZOOM	☒
				☐
				☐

Below the table is the 'SQL Text' area, which contains the following SQL statement:

```
SELECT 'NAME' NAME, 'VALUE' VALUE FROM DUAL
```

A 'Done' button is located at the bottom right of the form.

- 3 In the Block field, select the block where the LOV exists; in the Field Name field, select the field where the LOV exists. The Record Group field defaults to a value that begins with “LA_” and the LOV Name field defaults to “APPCORE_ZOOM”; accept these values. Be sure the Active check box is selected.
- 4 The SQL Text field displays a “stub” SQL statement. You may edit it or replace it entirely, to create either a static list or one that returns values determined at runtime. Keep the following in mind:

- The SQL statement can specify only two return columns, with the aliases NAME and VALUE.
- The template constitutes one line of a SQL statement that returns one value. To create multiple values in a static LOV, use the UNION statement. For example, the following SQL statement returns the values High, Medium, and Low:

```
SELECT 'High' NAME, 'High' VALUE FROM DUAL
UNION
SELECT 'Medium' NAME, 'Medium' VALUE FROM DUAL
UNION
SELECT 'Low' NAME, 'Low' VALUE FROM DUAL
```

- In the template, the value *DUAL* is a placeholder for a table name. To create a static list of values, leave it as is; to create a list of values determined at runtime, replace it with the name of the table that supplies values.

Setting Field Attributes

You can designate the display properties of blocks and fields, such as the positioning, color, size, and weight of items. You can also set security properties for field instances. To do so, click on the Field Attributes tab in the Business Rule Details form:

The screenshot shows the 'Business Rule Details' window with the 'Field Attributes' tab selected. The form is configured for the 'APXVDMVD' form, 'Enter Vendor Attributes' element, and 'When New Form' event. The 'Field Attributes' tab displays a table with the following data:

Seq	Type	Block Name	Field Name	Property	Value	Enabled Flag
1	Field	VNDR	EMPLOYEE_NAME_MIF	Background Color	Red	☒
						☐
						☐
						☐
						☐
						☐
						☐

A 'Done' button is located at the bottom right of the form.

Set an attribute for one block, field, or field instance in each row of the grid:

- 1 In the Seq field, type a number that reflects the order in which you want this attribute to be set with respect to other attributes listed in the grid.
- 2 In the Type list box, choose whether you want to set an attribute for a block, field, or field instance.
- 3 Select the component whose type you identified in step 2. If you selected the Block type, choose a value in the Block Name list of values and leave the Field Name list of values blank. If you selected the Field or Field Instance type, choose values in both the Block Name and Field Name LOVs. (The LOVs display blocks and fields you have captured through use of the Event Tracker.)



Note

A “field” is a set of like values, while a “field instance” is an individual value for a field. For example, when a form presents a grid, a “field” is an entire column in the grid, and a “field instance” is an individual cell in the column. Field attributes apply no matter what the value of a field is. If you create field-instance attributes, however, you need to define the particular instances to which the attributes apply. You would do this by creating a data subscriber that targets the field you selected in step 3; the attributes you create would take effect only when the subscriber definition evaluates to true. Take care that that field-instance security attributes you define here do not conflict with field security attributes defined in the security panel

- 4 In the Property list box, select the attribute you want to set for the component you have identified. The array of attributes varies by component type; see the following table for descriptions of the attributes you can select for each type.
- 5 In the Value field, type or select the value that sets the attribute. For example, if you selected Background Color in the Property field, you would select a specific color — for example, red — in the value field.
- 6 Select the Enabled Flag check box to activate the attribute, or clear the check box to deactivate the attribute.

The following table explains the properties you can select for blocks, fields, and field instances:

Type	Property	Value
Block	Blockscrollbar X Pos: Sets the horizontal starting point for the scroll bar, from the left of the block.	Type a number of pixels.
	Blockscrollbar Y Pos: Sets the vertical starting point for the scroll bar, from the top of the block.	Type a number of pixels.
	Current Row Background Color: For the row on which the cursor is focused, sets the color of the space surrounding field entries.	Select from six colors.
	Current Row Font Size: Sets the type size for entries in the row on which the cursor is focused.	Select a number from 1 to 18 points.

Table continues on the next page.

Type	Property	Value
Block	Current Row Font Weight: Sets the thickness of type for entries in the row on which the cursor is focused.	Select from nine weights.
	Current Row Foreground Color: For the row on which the cursor is focused, sets the color of field entries.	Select from six colors.
	Next Navigation Block: Sets the block to which the cursor moves if a user presses the Tab key from the last field in the current block.	Type a block name.
	Previous Navigation Block: Sets the block to which the cursor moves if a user presses Shift+Tab from the first field in the current block.	Type a block name.
	Query Allowed: Determines whether a user can query fields in the block — search for records with a field set to a value that matches a search value.	Select TRUE or FALSE.
	Query Data Source Name: Sets the database table or view searched for records in response to a query.	Type the name of a database table or view.
Field	Background Color: Sets the color of space surrounding field entries. (For a field instance in a selected row, the Current Row Background Color setting for the block takes precedence.)	Select from six colors.
	Conceal Data: Presents asterisks rather than actual entries for a field.	TRUE (To set the value to FALSE, do not select this attribute.)
	Font Size: Sets the type size for field entries. (For a field instance in a selected row, the Current Row Font Size setting for the block takes precedence.)	Select a number from 1 to 18 points.
	Font Weight: Sets the thickness of type for field entries. (For a field instance in a selected row, the Current Row Font Weight setting for the block takes precedence.)	Select from nine weights.
	Foreground Color: Sets the color of field entries. For a field instance in a selected row, the Current Row Foreground Color setting for the block takes precedence.)	Select from six colors.
	Format Mask: Imposes formatting on numeric or date fields.	
	Social Security Number:	999"-"99"-"999
	Monetary value, US currency up to \$1 billion:	\$999,999,999.99
	Numeric with thousand separators, up to four decimals:	999,999,999.9999
	Numeric with thousand separators, up to two decimals:	999,999,999.99
	Local currency with thousand separators, two decimals:	L99G999D99
	Number, up to four digits, with leading zeros:	0999
	Date, with month spelled out:	DD-MONTH-YYYY
	Date and time, with month abbreviated:	DD-MON-YYYY HH24:MI:SS

Type	Property	Value
Field	Height: Sets the vertical dimension of the field.	Type a number of pixels.
	Hint Text: Creates a message that provides information about the field in a status bar at the bottom of the screen	Type the message of up to 30 characters.
	Next Navigation Item: Sets the field to which the cursor moves if the user presses the tab key	Select from a list of field names.
	Previous Navigation Item: Sets the field to which the cursor moves if the user presses Shift+Tab	Select from a list of field names.
	Prompt Text: Creates a "user" field name that identifies the field on screen.	Type the label of up to 30 characters.
	Width: Sets the vertical dimension of the field.	Type a number of pixels.
	X Pos: Sets the horizontal starting point for the field, from the left of its block.	Type a number of pixels.
Field Instance	Y Pos: Sets the vertical starting point for the field, from the top of its block.	Type a number of pixels.
	Insert Allowed: Determines whether a user may enter data if the field instance is blank.	Select TRUE or FALSE.
	Navigable: Determines whether a user may select the field instance.	Select TRUE or FALSE.
	Required: Determines whether a user must enter data in a field instance.	Select TRUE or FALSE.
	Update Allowed: Determines whether a user may alter existing data in a field instance.	Select TRUE or FALSE.

Creating SQL Procedures

To create SQL statements that are executed when a user performs an action corresponding to the event you have chosen for a rule element, click on the SQL tab in the Business Rule Details form. Create any number of statements, one per row in the grid.

Business Rule Details

Form Name: Element Description: Event:

Block Name: Field Name:

Security Navigation Messages Default Values List Of Values Field Attributes **SQL** AppsFlow

Sequence Num	Procedure Name	Active
		☒
		☐
		☐

SQL Text

```

v_variable   varchar2(30);
BEGIN
  NULL;
--Sample Code to change an item property
--set_item_property('BLOCK.FIELD',REQUIRED,PROPERTY_TRUE);
--Sample Code to post a message
--fnd_message.set_string('Hello World');
--fnd_message.show;

```

Compile All Active SQL Rules

Done

- 1** In the Sequence Num field, type a number that reflects the order in which you want this SQL statement to be executed in relation to other statements you may create in other rows.
- 2** In the Procedure Name field, type a name for the SQL statement.
- 3** Ensure that the Active check box is selected to use the statement, or clear the check box to hold the statement in reserve.
- 4** The SQL Text field displays a template. Substitute actual values for placeholder values in the template, or replace the template entirely with a statement of your own.
 - You may not reference form bind variables directly in this window.
 - To reference a field, you must use the “name_in” function.
 - Declare variables before the BEGIN keyword. Do not use the word *Declare* in the SQL text.
- 5** Click on the Compile All Active SQL Rules button. (Or, as an alternative, click on Tools in the menu bar and then on AppsRules Compile All Active SQL Rules in the Tools menu.) This has two effects:
 - A concurrent request runs to compile the code. A message informs you of its identification number. Make a note of the number and click on the OK button to close the message.
 - A validation procedure determines whether the SQL is syntactically correct.
- 6** Review the concurrent program log for errors.
 - a** Click on View in the menu bar, and then on Requests in the View menu.
 - b** A Find Requests form opens. Click on the Specific Requests radio button and, in the Request ID field, enter the ID number for your request. Click on the Find button.
 - c** A Requests form appears; its grid contains an entry for your request. When its status is Completed (you may need to click the Refresh Data button), click on the View Log button.
- 7** If successful, exit Oracle Applications and log back in. SQL rules are implemented via the custom library, which may be cached when you log in. To test recently compiled rules, log out and log in to the application.

Running AppsFlow Processes

Logical Apps AppsFlow defines and implements business processes — sets of actions to be completed in specified sequences. AppsFlow can send approval requests or notification messages, enforce exceptions and constraints, and run concurrent programs and SQL scripts.

An AppsFlow process may be configured to run in response to a “triggering” event, typically the insertion or updating of a record in a specified database table. For such

a process, an AppsForm rule may instead define the event that triggers the AppsFlow process to run. (An AppsFlow process may be configured to run on a regular schedule rather than in response to a trigger; if so, the process cannot be run from within AppsForm.)

To prepare an AppsFlow process to be called from an AppsForm rule, do the following:

- Among your entries in the Process Rule Details panel of the main AppsForm form, select Trigger in the Event/Periodic field. In the Primary Keys panel, select the table with which the process is linked, and its primary keys. In the Display Table/View Columns panel, choose the appropriate table and display column settings. Set start and end dates in the Effectivity Dates panel. (For details, see “Creating a Process” in the *AppsFlow Training Guide*.)
- Leave launch criteria unconfigured.
- Create “process flows” to define the actions that the process sets in motion. (See “Configuring Elements Called by Rules” and “Creating Process Flows” in the *AppsFlow Training Guide*.)

Then, in AppsForm, create the rule element that is to launch the process. The form that you select as the target of this element must correspond to the table with which the AppsFlow process is linked. (That is, the table must be the one that stores the data accepted or displayed by the form.)

In the Business Rule Details form, click on the AppsFlow tab. Complete one row of the grid for each AppsFlow process you want to run. As you do, you will not only identify the process, but also specify one or two primary-key values to be passed to the AppsFlow process. (These serve to identify a record from the table with which the process is linked, and upon which the process acts.)

Business Rule Details

Form Name: **APXVDMVD** Element Description: **Enter Vendor Change A** Event: **When Validate Rec**

Block Name: Field Name:

Security Navigation Messages Default Values List Of Values* Field Attributes SQL **AppsFlow**

Sequence	Process	Type	Disposition Id1	Disposition Id2	Active
1	Send Approvals	Form	VNDR.VENDOR_ID		☒
					☐
					☐
					☐
					☐
					☐
					☐

Done

- 1 In the Sequence field, type a number that reflects the order in which you want this process to run with respect to other processes listed in the grid.

- 2** In the Process list of values, select the process you want to run.
- 3** In the Type list box, select Form if you intend to have primary-key values supplied by fields from the form that is the target of the rule element. Select Static if you intend set constants as primary-key values.
- 4** If you selected Form in step 3, use the Disposition Id1 field to provide the name of the form field that corresponds to the first primary key in the table with which the AppsFlow process is linked. If the table has a second primary key, provide the name of its corresponding form field in the Disposition Id2 field. Use the format `BLOCK_NAME.FIELD_NAME`. (As a reminder, the procedure for determining the internal names of blocks and fields is on page 3.)

If you selected Static in step 3, type a constant value in the Disposition Id1 field; if the AppsFlow process is linked with a table that has two primary keys, enter a second constant in the Disposition Id2 field. In each case, make sure the value is of the data type defined for its corresponding primary key column.
- 5** Ensure that the Active check box is selected to run the process. Or clear the check box to prevent the process from being run.

Saving Rule Element Details

When you are finished configuring details for a rule element, click on the Done button to close the Business Rule Details form. Then save your work: Click on File in the menu bar and then on Save in the File menu.

Creating Subscribers

An AppsForm rule (or rule element) may be fired selectively — it may produce results in some circumstances but not in others. A “subscriber” defines those circumstances; it is a filter that selects the users, responsibilities, or other entities to which an AppsForm rule (or element) applies.

To define a subscriber, you work from within a selected rule or element, and so focus that rule or element on the subscriber you define. You can create a rule without configuring a subscriber for it; in that case, the rule applies universally. Similarly, you can create an element without a subscriber; if so, the element applies to all of the rule’s subscribers.

As you create subscribers, you can filter on the following entities:

- Responsibility
- Profile
- Operating Unit
- Inventory Organization
- User
- Data
- Subscriber List
- Function
- SQL

Defining a Subscriber

To define a subscriber for a rule, select the rule in the main AppsForm window and click on the Rule Subscribers button. To define a subscriber for a rule element, select the element in the AppsForm window and click on the Element Subscribers button. In either case, the following form appears:

Filter Type	Filter Name	Operator	Value	Data Type	Field Type	Dependant Value	And/Or	Group	Allow Reversals
Responsibility		Equal		Varchar	Static		Or	1	☒
									☐
									☐
									☐
									☐
									☐
									☐

View Statement Done

To begin the definition, select a filter type, an operator, and a value. A rule (or element) applies whenever a logical statement defined by those three entities evaluates to true — for example when a user's responsibility (filter type) equals (operator) Purchasing Super User (value).

If you wish to use the Responsibility, Profile, Operating Unit, Inventory Organization, User, or Subscriber List filter type, make up to three selections:

- 1 Select the type you want to use from the Filter Type list of values.
- 2 Select an operator from the Operator list of values. Your options are Equal, Not Equal, Is Null, Is Not Null, Greater Than, and Less Than.
- 3 If you choose the Equal, Not Equal, Greater Than, or Less Than operator, select an appropriate value from the Value LOV.

For example, you might select Responsibility, Equal, and Purchasing Super User to make a rule apply only to users logged on with the Purchasing Super User responsibility. Or you might select User, Not Equal, and DAbbot to have a rule apply to all users other than DAbbot.

If you wish to use the Data, Function, or SQL filter type, you need to make a different set of selections. (Note that the Function and SQL filters are available only for rule element subscribers.)

- 1 Select the type you want to use from the Filter Type list of values.
- 2 Depending on the selection you made in step 1, choose a block and field, a database function, or a SQL statement in the Filter Name list of values. (The SQL statement may be one prepared through use of the SQL Wizard; see page 33.)
- 3 Select the operator you want from the Operator list of values. If you select Is Null or Is Not Null, you're finished. If you select Equal, Not Equal, Greater Than, or Less Than, continue.

- 4 Choose a data type — VarChar, Number, Date, DateTime, or Boolean — in the Data Type list of values. The type should match the type stored in, or returned by, the item you selected in the Filter Name list of values (step 2).
- 5 In the Field Type list box, select Static if you are constructing a logical statement that compares a constant value to values stored in, or returned by, the item you selected in the Filter Name list of values (step 2). Or select Form Field if you want to compare these values to other values returned by an Oracle Applications Field.
- 6 Depending on the selection you made in step 5, type a constant value in the Dependent Value field, or select a field name.

For example, you might select the Data filter type and an Oracle Application block and field in the Filter Name list of values. You might also select the Is Null operator, in which case the rule would apply whenever the field you selected contains no value. Or, you might select the Equal operator, the Static Field Type, and a constant value as the Dependent Value; if so, the rule would apply whenever the value in the field is the specified constant.



Note

If you select the Data filter type, the Data Type entry is Number, and the Field Type entry is Static, AppsForm validates the Dependent Value entry — determines whether it actually is a number and, if not, presents an error message. If the Data Type entry is any other or the Field Type is Form Field, AppsForm displays hint text advising the user to ensure that the Dependent Value entry is the correct data type. (This hint text appears in the status bar near the bottom of the screen.)

More Subscriber Fields

You can complete multiple rows in the Subscribers form to create a subscriber definition that consists of several criteria. If so, values in the And/Or and Group fields come into play. (Note, however, that you need to complete these fields even if your subscriber definition consists only of a single row; in that case, you would typically accept default values.)

- **Group:** Enter a number that reflects the sequence in which a row is to be evaluated with respect to other rows. You can enter the same number in more than one row. If you do, the rows are grouped together — enclosed in parentheses in a SQL statement that is generated from the values you enter.
- **And/Or:** Select a value that determines whether a row has an AND (both must be true) or OR (either may be true) relationship with the next row (or group, or row within a group).

In addition, you may use the following controls:

- **Allow Reversals:** If this check box is not selected when a rule has been enforced, the rule continues in force even if the next record does not meet the subscriber criteria.

- **View Statement:** Open a window that displays the SQL statement generated from the selections you make in the Subscribers form. (To close this window, click on its Close button.)
- **Done:** Click on this button to close the Subscribers form. Save the subscriber before closing the form: Click on File in the menu bar and Save in the File menu.

Subscriber Lists

Among the filters you can select as you create subscribers is Subscriber List, which is itself a selection of users, responsibilities, or other entities. As you create a subscriber, for example, you might select Subscriber List as the filter, Equal as the operator, and List1 as the value. The AppsForm rule (or element) would then apply to all the members of a subscriber list called “List1.”

To create a subscriber list, click on Tools in the menu bar, and then AppsRules Subscriber Lists in the Tools menu. The following form appears:

Subscriber List Name	Description	Start Date	End Date
List 1		19 AUG 2005	

And/Or	When	Profile	Condition	Value	Grouping
And	Responsibility		Equal	Purchasing	1
And	User		Not Equal	BKARMICO	1
Or	Operating Unit		Equal	Vision Portugal	2

View Statement Done

In the grid at the top of the form, create a new subscriber list in each row.

- 1 Create a name for the list in the Subscriber List Name field.
- 2 Optionally, provide a brief explanation for the purpose of the list in the Description field.
- 3 In the Start Date field, select a date on which the list is to take effect.
- 4 In the End Date field, select a date on which the list is to expire. Or, leave the field blank to allow the list to remain in effect indefinitely.

In the Subscribers Details area, enter values that define the members of whatever list is currently selected in the upper grid. In each row:

- 1 In the When field, select a filter: Responsibility, Profile, Operating Unit, Inventory Organization, or User. (Select a value in the Profile field only if you select Profile as a filter.)
- 2 In the Condition field, select an operator: Equal or Not Equal.

- 3 In the Value field, select a constant that completes a logical statement. For example, if you select Responsibility in the When field, Equal in the Condition field, and Purchasing in the Value field, users who log on in the Purchasing responsibility are members of the subscriber list (subject to refinements in other rows).

In a special case, if you have selected the value *Profile* in the When field and identified a specific profile in the Profile field, include a valid setting for the profile in the Value field. For example, the Profile field might hold the value *AudifTrail:Activate*, the Condition field *Equal*, and the Value field *Yes*.

- 4 In the Group field, enter a number that reflects the sequence in which a row is to be evaluated with respect to other rows. You can enter the same number in more than one row. If you do, the rows are grouped together — enclosed in parentheses in a SQL statement that is generated from the values you enter.
- 5 In the And/Or field, select a value that determines whether a row has an AND (both must be true) or OR (either may be true) relationship with the next row (or group, or row within a group).
- 6 When you finish making entries in rows, save the subscriber list: Click on File in the menu bar and Save in the File men.

You can click on the View Statement button to open a window that displays the SQL statement generated from your selections (and then the Close button in that window to close it). Click on the Done button to close the Subscriber List form.

Using the SQL Wizard

When you select SQL as a filter type for an element subscriber, the Filter Name field lists (and you can therefore select) SQL statements created in a tool called the SQL Wizard.

Suppose, for example, you were to create a subscriber that applies a rule element to vendors with more than two open purchase orders. You would first use the SQL Wizard to create a SQL statement — named, let's say, Open POs — that counts vendors' open purchase orders. You would then use the Subscribers form to define the subscriber, setting the Filter Type field to *SQL*, Filter Name to *Open POs*, Operator to *Greater Than*, Data Type to *Number*, Field Type to *Static*, and finally Dependent Value to 2.

The SQL Wizard consists of up three panels, each accessible from a tab. These panels include:

- Definition, in which you select a “driving table” and may select other tables related or joined to it. The driving table provides information needed for the SQL statement to be configured. (It and tables joined to it might, for example, provide filtering criteria or a return value.)
- Return Columns, in which you select a column from the driving table or a joined table. This column supplies a value that the SQL statement returns to the subscriber. (You can select only one such column.)

- SQL, in which you can generate SQL code automatically from the values you selected in the Definition and Return Columns panels. In the SQL panel, you can also edit code manually and verify it.

Starting a SQL Statement

To use the SQL Wizard to create a SQL statement:

- 1 With AppsForm running, click on LogicalApps Utilities in the menu bar, and then on Advanced Rules Wizard in the LogicalApps Utilities menu. The SQL Wizard opens with the Definition panel active.
- 2 In the Rule Name field, type a unique name for the element you are creating.
- 3 In the Description field, type a brief explanation of the element you are creating.

Using the Definition Panel

Use the Definition panel to select tables from which an SQL statement is to draw information:

The screenshot shows the SQL Wizard window with the Definition panel active. The Rule Name is 'Open POs' and the Description is 'Returns the number of open POs for...'. The Driving Table is 'PO_HEADERS_ALL' with Table Alias 'PH'. Disposition ID1 and ID2 are empty. Below are sections for Related Tables, Table Joins, and Additional Conditions.

Additional Conditions							
Group	Table	Alias	Column	Condition	Value Type	Value	And/Or
10	PO_HEADERS_ALL	PH	VENDOR_ID	equal	FOR...	:VNDR.VENDOR	AND
20	PO_HEADERS_ALL	PH	STATUS_LOOKUP_CODE	equal	STATIC	Open	AND

Always select a driving table and values related to it:

- 1 In the Driving Table list of values, select a table upon which the SQL statement is based.
- 2 In the Table Alias field, create an alias for the table.
- 3 Optionally, in the Disposition ID 1 and 2 fields, enter the names of columns that distinguish a given record in the table from others (typically the primary keys).

If the statement is to use information stored in a table related to the driving table, select appropriate values in the Related Tables and Table Joins grids:

- 1** Devote one row of the Related Tables grid to each table you want to specify: Select its name from the Table Name list of values, create an alias in the Table Alias field, and optionally provide a description in the Description field.
- 2** For each row in the Related Tables grid, enter values in a corresponding row of the Table Joins grid. These values specify how each of the related tables is joined to the driving table or one of the other tables. Select the names and aliases of the two joined tables as well as the column in each that contains join values.

In the Additional Conditions grid, you may construct logical statements specifying values that table cells may hold. Such statements establish criteria by which some records in the driving table (or related tables) are distinguished from others — they either contain the specified values, or they don't.

Conditional statements are appropriate for SQL statements whose purpose is to select records that meet certain criteria. (In the illustrated example, a SQL statement is being created for a subscriber to a rule element that targets the Enter Vendor form; PO_HEADERS_ALL, the driving table, stores purchase-order information; and the first of two conditions causes the SQL statement to select PO_HEADERS_ALL records whose vendor IDs match IDs loaded in the Enter Vendor form. A second condition selects records of open POs.)

To create a condition:

- 1** In the Table field, select the name of the driving table or a related table. (If the LOV does not contain values, save the SQL statement — click on File in the tool bar and Save in the File menu — and then try again.) The Alias field defaults to the alias you've already created for the table.
- 2** In the Column list of values, select a column from the table.
- 3** In the Condition field, select a logical operator — Equal, Not Equal, Greater, Lesser, Is Null, or Is Not Null.

Either of the Is Null or Is Not Null entries completes a logical statement (so if you select either, a third field, Value, does not accept input). For example, if you select Is Null, the condition evaluates to true for records whose cells within the specified column contain no content.

- 4** If you selected the Equal, Not Equal, Greater, or Lesser operator, specify a value to be compared with values contained in the column you selected in step 2.
 - To specify a constant value, select Static in the Value Type list box and type the constant value in the Value field.
 - To specify the value of a field in an Oracle Applications form, select Form Field in the Value Type list box. Then select the name of the block and field in the Value LOV, which lists fields from the form that is the target of the rule element for which you are creating a subscriber.

For example, if you selected a STATUS_LOOKUP_CODE column in step 2 and the Equal operator in step 3, you might now select the Static value type and the value

Open. The condition evaluates to true for records whose cells in the STATUS_LOOKUP_CODE column are set to Open.

- 5 Enter a number in the Group field. All rows with the same number are grouped together (enclosed in parentheses in the SQL statement).
- 6 If you create more than one condition, select AND or OR to determine how each condition (row) relates to other rows (or groups, or rows within groups). If two entities are joined by AND, both must be true; if by OR, either may be true.
- 7 Before continuing to the next panel, save the rule: Click on File in the menu bar and Save in the File menu.

Using the Return Columns Panel

Use the Return Columns panel to select a column that returns data from the a table chosen in the Definition panel:

SQL Wizard

Rule Name: **Open POs**

Description: **Returns the number of open POs for**

Definition | **Return Columns** | SQL

Seq	Table Name	Alias	Column Name
1	PO_HEADERS_ALL	PH	VENDOR_ID

Launch Process Name:

Disposition Id1:

Disposition Id2:

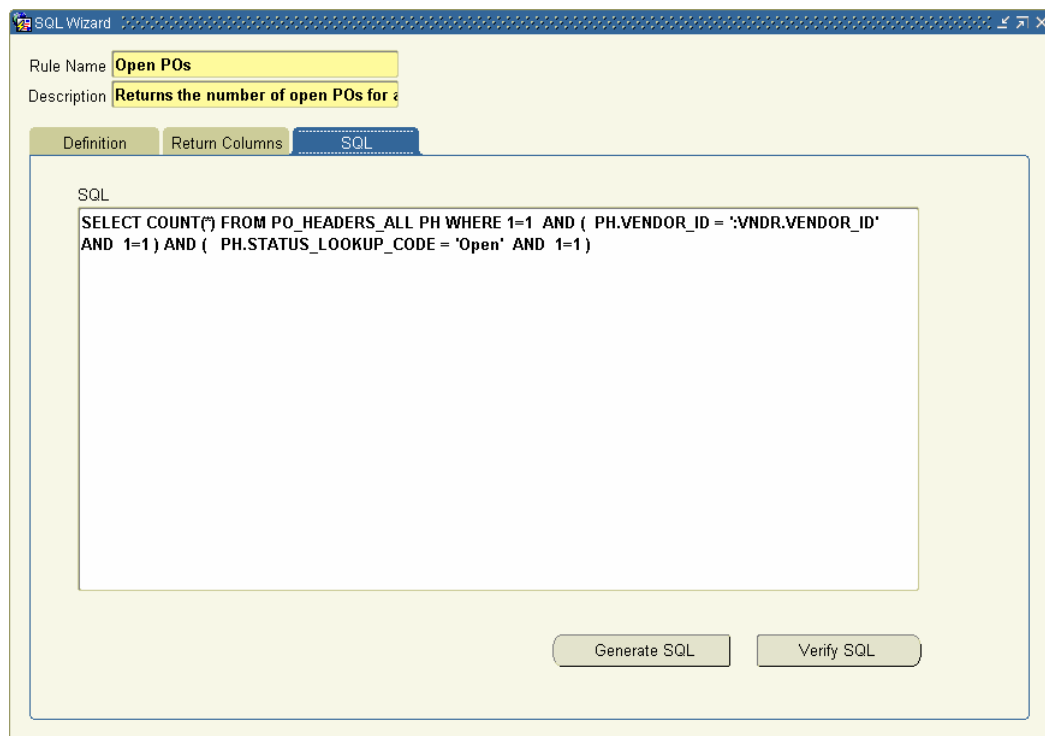
You can select only one column, so you would enter values in only one row of the grid:

- 1 In the Seq field, type a number.
- 2 In the Table Name list of values, select one of the tables you specified in the Definition panel.
- 3 In the Alias field, the Wizard supplies automatically the alias you created for the table in the Definition panel.

- 4 In the Column Name list of values, select a table column that contains information you want to use in the advanced rule.
- 5 Before continuing to the next panel, save the SQL statement: Click on File in the menu bar and Save in the File menu.

Using the SQL Panel

Use SQL panel to generate a SQL SELECT statement automatically from the values you chose in the Definition and Return Columns panels:



- 1 Click on the Generate SQL button. A SELECT statement appears, and a pop-up message informs you that its syntax is valid. Click on the OK button to clear the message.
- 2 Edit the SELECT statement as you see fit.

In the illustrated example, the values selected in the Definition and Return Columns panels would have produced this statement:

```
SELECT PH.VENDOR_ID FROM PO_HEADERS_ALL PH WHERE 1=1
AND ( PH.VENDOR_ID = ' :VNDR.VENDOR_ID' AND 1=1 ) AND
( PH.STATUS_LOOKUP_CODE = 'Open' AND 1=1 )
```

However, because the SQL statement is supposed to return a count rather than a vendor ID, the term “PH.VENDOR_ID” has been replaced with “COUNT(*)”.

- 3 Whenever you edit the SQL statement, click on the Verify SQL button. A pop-up message either informs you that the SQL statement is valid or explains any errors it contains. Click on the OK button to clear the message.



Note

The SQL statement incorporates the aliases you created in the Definition panel. If you generate and save a SQL statement, then change an alias in the Definition panel, the alias is not automatically updated in the SQL panel. You must update it manually.

- 4** Save the SQL Statement: Click on File in the menu bar and Save in the File menu.

AppsForm Migration

Once you have created AppsForm rules and rule elements for an instance of Oracle Applications, you can “migrate” them — copy a rule or element directly to another Oracle Applications instance. Alternatively, you can export individual rules or elements to, or import them from, XML files, or copy them under new or modified names on the source instance. You can also perform these operations on libraries — collections of rules created in AppsForm and AppsFlow.

Preparing for Migration

Before you can migrate rules or elements, you need to specify connection information in all the environments to and from which you plan to transmit data. You need to know the host name, instance SID, and database instance port for each environment. This information is found in the `TNSNAMES.ora` file, which is located in `ORACLE_HOME/network/admin`.

Once you’ve gathered this information, use the Logical Apps Migration Utility to perform the connectivity configuration:

- 1 With AppsForm open, click on LogicalApps Utilities in the menu bar, then on Migration Setup in the Utilities menu. A Migration Utility form appears (as shown at the top of the next page).
- 2 Ensure that the Setup Host Names tab is selected.

Host Name	Description
newport.whq.logicalapps.com	Dev
cypress.whq.logicalapps.com	QA

- 3 In the Host Name column, enter the host name (machine name) for each of the machines hosting the database and involved in the migration.
- 4 In the Description column, you may enter a description for each host name. (This step is optional.)
- 5 Click on the Setup Instances tab. The following form appears:

Host Name	Instance	Port Number	User Name	Password
newport.whq.logicalapps.com	visdev	1525	apps	apps
cypress.whq.logicalapps.com	visdb	1521	apps	apps

Connect String

- 6 In the Host Name column, select the host name for each of the machines from the list of values. (The entries are those defined in the Setup Host Names tab.)
- 7 In the Instance and Port Name columns, type the instance name and port number that corresponds to each host name.
- 8 Under User Name, type the value apps for each entry. Under Password, enter the password for the apps user.

- 9 Click on File in the menu bar and Save in the File menu. When the configuration is saved, the system automatically generates and displays a connection string.
- 10 Close the Migration Utility. Click on the × symbol in the upper right corner of the form.

Stipulations

The following conditions apply to migration, export, and import operations:

- If you migrate an entire rule, its elements move with the rule to the target instance or XML file. If you migrate an element, it moves to the destination instance without its “parent” rule.
- For an element to be migrated, the rule that contains it must already exist on the destination instance.
- A log file gathers information about a migration, export, or import operation. If an operation fails and you are unable to determine why, rerun the operation with the debug level changed from low to high and evaluate the log data.

Migrating, Exporting, or Copying Rules or Libraries

To migrate a rule, element, or library to another instance, or to copy one either to an XML file or on the source instance, complete the following steps:

- 1 Open AppsForm and select the rule you want to use as the source for a migration, export, or copy operation.
- 2 Click on LogicalApps Utilities in the menu bar, then on Migrate Rules in the Utilities menu. The Migrate Form Rules form appears:

The screenshot shows the 'Logical Apps - Migration Utility' window with the 'Migrate Form Rules' tab selected. The 'Action Type' dropdown is set to 'Migrate to Another Instance'. The 'Source Rule' section has 'Library' and 'Rule Element' as empty text boxes, while 'Rule Name' contains 'DacTest'. The 'Destination' section has 'Instance' and 'Apps Passwd' as empty text boxes. The 'Copy Options' section has 'Type' as a dropdown menu and 'New Rule' as an empty text box. The 'Debug/File Options' section has 'Debug Level' as a dropdown menu set to 'Low', 'File name' as an empty text box, and 'Directory' as a text box containing '/tmp'. At the bottom are three buttons: 'Cancel', 'Clear', and 'Migrate'.

- 3** In the Action Type list box, select the operation you want to perform:
 - Migrate to Another Instance (the default) if you want to migrate a rule, element, or library.
 - Export to File if you want to export a rule, element, or library to an XML file.
 - Copy within the Same Instance if you want to copy a rule, element, or library under a new or modified name on the source instance.
- 4** In the Source Rule block, identify an item to be migrated, exported, or copied:
 - The Rule Name list of values displays the rule you selected in step 1. Retain the entry to work with that rule (or its elements); or, select another. Or, delete the Rule Name entry in order to select a library. (If the Rule Name field contains an entry, the Library field does not accept input.)
 - If you retain an entry in the Rule Name field, do one of the following in the Rule Element list of values:

Leave the field blank to migrate, export, or copy the full AppsForm rule you selected in the Rule Name field. This includes the rule and all its elements.

Select the value All Elements to migrate, export, or copy all the elements that belong to the rule, but not the rule itself.

Select a single element belonging to the rule to migrate, export, or copy that element, but not the rule itself.
 - Alternatively, select a library rather than a rule or element. Delete the default entry in the Rule Name field, then click in any other active field. The library field becomes active; in it, select the library you want to migrate, export, or copy. (When the Library field contains an entry, the Rule Name and Rule Element fields do not accept input.)
- 5** If you are performing a migration, make entries in the Destination block:
 - In the Instance list of values, select a destination instance for the migration.
 - In the Apps Passwd text box, type the apps password for the destination instance if you are prompted to do so. (This prompt appears if the XXLAAPPS: Enable for Migration Security profile option is set to Yes on the source instance. If the option is set to No, the prompt does not appear and a password need not be entered.)

If you are performing a file export or copying to the source instance, fields in the Destination block do not apply and do not accept input.
- 6** If you are copying to the source instance, make entries in the Copy Options block:
 - If you are copying an element or all elements (if you entered any value in the Rule Element field of the Source Rule block), the Type field defaults to Add to Existing Rule no matter what value you select there. In the second field, select the existing rule into which you want to copy the elements. The copied elements overwrite any identically named elements in the destination rule.

- If you are copying an entire rule (if you selected a Rule Name entry but left the Rule Element field blank in the Source Rule block), select one of three values in the Type field: Copy as a New Rule if you want to assign a completely new name to the copy, or Add a Prefix or Add a Suffix if you want to assign the copy a name that consists of the original rule name with text added at the beginning or end. In the second field, type the text you want to use as a new rule name or as a prefix or suffix to the original name.
- If you are copying a library (if you selected an entry in the Library field of the Source Rule block) select either of two values in the Type field: Add Suffix to All Rules or Add Prefix to All Rules. In the second field, type the text you want to use as a prefix or suffix to the original rule names.

If you are performing a migration or a file export, fields in the Copy Options block do not apply and do not accept input.

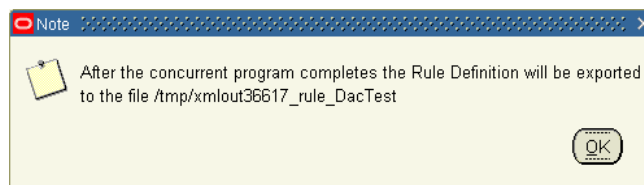
7 Make selections in the Debug/File Options block:

- In the Debug Level list box, select a level of detail for error reporting to a log. Ordinarily, select Low; select High instead if you need to uncover the cause of a failed migration, export, or copy.
- In the Directory text box, type the path that designates a temporary staging file location for XML files to be generated and, in the case of migration, copied to the destination instance.
- The File Name field does not accept input.

8 Click on a button that launches the process. Its label varies depending on your selection in step 3: Migrate if you chose Migrate to Another Instance, Export if you chose Export to File, or Copy if you chose Copy within the Same Instance.

9 Review several messages:

- The system launches a concurrent program to implement the migration, export, or copy. The first message provides an ID number. Click on the OK button to clear the message.
- If you have performed a file export, a message similar to the following one displays the name of the export file you have generated.



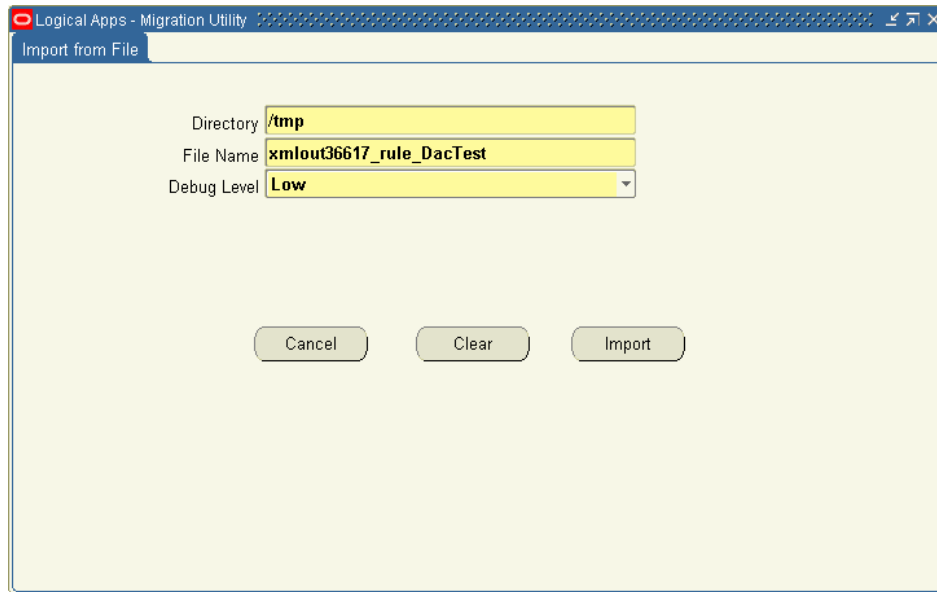
In the file name, the term *xmlout* designates XML output, a number (*36617* in this example) uniquely identifies the export operation, and final phrases (in this example, *rule* and *DacTest*) identify the type and name of the export items. Make a note of the file name and location, and then click the OK button.

- Finally, a dialog prompts you to perform another migration. Click Yes to do so or No to close the Migration form.

Importing a File Containing a Rule, Elements, or a Library

To import an XML file containing a rule, elements, or a library, complete these steps:

- 1 Transmit the exported file via FTP to the destination OS for import.
- 2 With AppsForm open, click on LogicalApps Utilities in the menu bar, then on Import from File in the Utilities menu. An Import From File form appears:



The screenshot shows a window titled "Logical Apps - Migration Utility" with a sub-tab "Import from File". Inside the window, there are three input fields: "Directory" containing "/tmp", "File Name" containing "xmlout36617_rule_DacTest", and "Debug Level" with a dropdown menu currently set to "Low". Below these fields are three buttons: "Cancel", "Clear", and "Import".

- 3 In the Directory box, type the path to the folder that contains the import file.
- 4 In the File Name box, type the name of the file you want to import. This would be a name displayed by a message at the culmination of a file export (see above).
- 5 Select a value for Debug Level. Ordinarily, select Low; select High instead if you need to uncover the cause of a failed import.
- 6 Click on the Import button. A concurrent request message displays the ID number of the concurrent program that implements the import. Click on the OK button to clear the message.
- 7 If you need to import more than one file, click the Clear button to remove current settings and repeat this process as necessary.

Running AppsForm Utilities

In addition to features described elsewhere in this manual, the Tools and LogicalApps Utilities menus provide access to several utility features. You can:

- Use the AppsExtend tool to create new forms, to be opened from (and so extend the capabilities of) existing Oracle Applications forms.
- Gather AppsForm and AppsExtend rules into libraries.
- Associate Logical Apps form functions with menus or responsibilities.
- Display AppsRules version information, and refresh a cache of rule and form data.
- Add custom events.

Using AppsExtend

AppsExtend enables you to create forms, each of which can contain text-entry, list-of-values, or date fields. Having created a form, you would then create an AppsForm navigation rule to make the new form accessible from an existing Oracle Applications form.

Creating a New Form

To create a form in AppsExtend, ensure that AppsForm is open; click on Tools in the menu bar, and then AppsExtend in the Tools menu. The following AppsExtend form appears:

The screenshot shows the 'Logical Apps - AppsExtend' window. It contains two main sections:

- Top Section:** A table with two columns: 'Extension' and 'Description'. The first row is selected and contains 'Vendor Extension Values' and 'Adds fields accessible from the Ent'. Below this are two empty rows.
- Bottom Section:** A section titled 'Extension Elements' containing a table with five columns: 'Attribute', 'Label', 'Enabled', 'Required', and 'Enable LOV'.

Attribute	Label	Enabled	Required	Enable LOV
Attribute1	Credit Rating	☒	☐	☐
Date1	Rating Date	☒	☐	☐
		☐	☐	☐
		☐	☐	☐
		☐	☐	☐

In the unlabeled grid at the top, use one row for each form you create:

- 1 Type a name in the Extension field. This is the display name, which appears in the title bar of the form you are creating. It is also a value you will use as you create the navigation rule that links your AppsExtend form to an existing Oracle Applications form.
- 2 Optionally, in the Description field, briefly explain the purpose of the form.

For the row that is currently selected in the top part of the AppsExtend form, use the Extension Elements grid to define the fields your form is to contain, one field per row:

- 1 In the Attribute list box, select a value labeled *Attribute* or *Date*, followed by a number. Selecting Attribute creates a text-entry or LOV field, while selecting Date creates a date field, on the form. The number sets the order in which the field is listed with respect to others; the lower the number, the higher the position. Attribute fields appear before date fields on the form.
- 2 In the Label field, type a display name, which identifies the field on the form.
- 3 Select the Enabled check box to place the field on the form (or clear the check box to remove the field from the form).
- 4 Select the Required check box if you want to require that data be entered into the field, or clear the check box if you want to make data-entry optional.
- 5 Click the Enable LOV check box if you want the field to display a list of values. If you select this check box, you must use AppsForm to create an LOV for the field. If you chose a Date value in the Attribute field, you cannot select the Enable LOV check box.

- 6 When you finish adding fields to the form, save your work: Click on File in the menu bar, and then on Save in the File menu. Close the AppsExtend form: Click on the × symbol in its upper right corner.

Linking a New Form to an Existing Form

To provide access to the new form, create a navigation rule element that makes it available as a menu option on an existing Oracle Applications form. See “Setting Navigation Paths” (page 14) for detailed information. As you create this rule element, use the following values:

- Use the When New Form event.
- In the To Function field on the Navigation panel of the Business Rule Details form, select AppsExtend Value Form.
- When you navigate to the Parameters field, it presents the following statement:

```
EXTENSION_TYPE="<EXTENSION NAME HERE>" DISPOSITION_ID=
"#<BLOCK.FIELD1>#" S_DISPOSITION_ID=""#<BLOCK.FIELD2>#"
```

In this statement, replace placeholder values (those enclosed in angle brackets, and the angle brackets themselves) with the following values.

- <EXTENSION NAME HERE>: Type the extension name you created for your form in the upper grid of the AppsExtend form (see page 46).
- <BLOCK.FIELD1>: Type the internal names for the block and field on the existing form that correspond to the first primary key of the database table that supports the form (see page 3).
- <BLOCK.FIELD2>: If that table has a second primary key, type the internal names for the block and field on the existing form that correspond to that second key. If the table does not have a second primary key, delete the entire parameter (S_DISPOSITION_ID=""#<BLOCK.FIELD2>#"") pertaining to that key.

For example, suppose that the extension name for the new form is Vendor Extension Values, and you intend to navigate to it from the Enter Vendor form. The table that supports the Enter Vendor form has only one primary key, and it corresponds to a block and field named VNDR.VENDOR_ID. The appropriate parameters would be:

```
EXTENSION_TYPE="Vendor Extension Values" DISPOSITION_ID=
"#VNDR.VENDOR_ID#"
```

- Accept the default for Icon Name.

Creating LOVs for an AppsExtend Form

If, as you created AppsExtend form, you selected the Enable LOV check box for any of its fields, you must create an LOV for each of those fields. To do so, create an AppsForm rule element that generates the list of values. Creating this element involves:

- Running the Event Tracker on the new form.

- Creating a rule element subscriber (without which, the LOV would be attached to all AppsExtend forms).
- Setting the rule element details.

In the Rule Elements grid on the main AppsForm screen, create the element, using the value LAAEELM in the Form Name field and selecting the Event Tracker in the Event field. Then save the rule, open the new form (by opening the Oracle Applications form to which it is linked and clicking on the menu option you created for it), and run the Event Tracker.

Next, reopen AppsForm and select the rule element with which you are working. In the Rule Elements grid of the main screen, retain LAAEELM as the form name and insert these values:

- In the Event field, When New Item.
- In the Block Name field, LA_EXTENSION_VALUES.
- In the Field Name field, the Attribute value you selected for the field in the AppsExtend form (see page 46). This would be the word *Attribute* followed by a number.

Then, create an element subscriber of the data filter type. See “Creating Subscribers” (page 29) for detailed information. As you create the subscriber, use these values:

- In the Filter Type field, Data
- In the Filter Name field, PARAMETER.EXTENSION_TYPE
- In the Operator field, Equal
- In the Dependent Value field, the Attribute value you selected for the field in the AppsExtend form (see page 46).
- In other fields, retain the default values.

Finally, configure the rule-element details. See “Creating a New LOV” (page 21) for detailed information. As you configure the details in the List of Values tab of the Business Rule Details form, use these values:

- In the Block Name field, LA_EXTENSION_VALUES
- In the Field Name field, the Attribute value you selected for the field in the AppsExtend form (see page 46).
- In the Record Group and LOV Name fields, accept default values.
- In the SQL Text field, create the SQL statement that selects the values you want the LOV to display.

Collecting Rules in Libraries

You can gather AppsForm rules and AppsExtend forms (as well as process rules created in AppsFlow) into libraries.

As you do, you can characterize the contents of each library by assigning it a theme, category, or module. You have the opportunity to define each class of values in any way you choose. For example, a theme might be a broadly defined set of items (such as rules that implement Sarbanes Oxley controls), a category might be a more narrowly defined set of those items (such as SOX-related AppsForm rules called by AppsAccess conflict rules), and a module might be a class of applications to which the library items apply. You can also assign each of the items you place in a library to a user. To create themes, categories, or modules, or to identify users to whom library items can be assigned, you would add values to “value sets” that have been created for use with libraries.

Creating a Library

To create a library, click on Logical Apps Utilities in the menu bar, and then on AppsRules Libraries in the Utilities menu. A LogicalApps Libraries form appears:

Library Name	Description	Version	Theme	Category	Module
APPSRULES_TEST15	AppsRules TEST 15---2	11.5.10	ALL	ALL	ALL

Seq	Type	Rule Name	Process Name	AppsExtend
1	Form Rules	TEST_QA_15		
2	Form Rules	TEST_QA_16_17_18_19		
3	Apps Extend			DacScreen

- 1 Select a row in the upper grid. Use any of these methods:
 - If the grid contains any empty rows, click in the first one.
 - Click on the New button, which is first on the left in the tool bar.
 - Click on File in the menu bar, then on New in the File menu.
- 2 In the Library Name field, type a unique name.
- 3 In the Description field, type a description of the library.
- 4 In the Version field, type a version number. Note that it can, but need not, reflect the version number of the software whose components you are collecting in a library.
- 5 In the Theme, Category, and Module lists of values, select values you have defined for characterizing the contents of the library. Or, if you have not defined values, ALL is the only value available to you; select it.

To add items to the library, use the Library Elements grid:

- 1** In the Seq field, type a sequence number.
- 2** In the Type list box, select Form Rules to add an AppsForm rule to the library, or select Apps Extend to add an AppsExtend form to the library.
- 3** If you chose Form Rules in the Type field, select an AppsForm rule in the Rule Name list of values. Or, if you chose Apps Extend in the Type field, select an AppsExtend form in the AppsExtend field. Then slide the scroll bar to the right to reveal additional fields.
- 4** In the Complexity list box, select Low, Medium, or High to assess the relative intricacy of the item you are adding. (Your site should develop its own definitions of low, medium, and high complexity.)
- 5** In the Assigned To list of values, select a user from a set of users to whom a library might be assigned. Or, if you have not identified such users, the field does not accept any entry; leave it blank.
- 6** Optionally, record the state of the library item by selecting appropriate check boxes: Developed, Documented, Tested, QA, or Completed.
- 7** Optionally, type a brief explanation of this library item in the Description field.
- 8** Repeat these steps to add as many additional rules or forms to the library as you like, one per row.
- 9** Save the library: Click on File in the menu bar and then Save in the File menu.

Adding to Value Sets

To define themes, categories, or modules that classify the contents of libraries, or to identify users to whom library items can be assigned, add entries to value sets that have been created for use with AppsRules libraries:

- 1** With the LogicalApps Libraries form open, click on Tools in the menu bar, and then on Value Sets in the Tools menu. A pair of forms open, with a Find Value Set form initially active:

The screenshot shows a 'Find Value Set' dialog box. It features a 'Find Values By' section with four radio buttons: 'Value Set' (selected), 'Key Flexfield', 'Descriptive Flexfield', and 'Concurrent Program'. To the right of this section is a 'Name' text field containing 'la%' with a dropdown arrow. Below these are three stacked text fields labeled 'Independent Value', 'Value', and 'Description'. At the bottom right are two buttons: 'Clear' and 'Find'.

- 2 In the Find Values By area, ensure that the Value Set radio button is selected. Then type the string *la%* in the Name box and click on the Find button.
- 3 A Flexfield Value Sets window opens. In it, click on an entry for the type of values you want to create: LAAR_LIBRARY_THEME corresponds to the Theme field in the Libraries form, LAAR_LIBRARY_CATEGORY to the Category field, LAAR_LIBRARY_MODULES to the Modules field, and LAAR_LIBRARY_ASSIGNED_TO to the Assigned To field. Then click on the OK button.
- 4 A Segment Values form becomes active. In the Effective Values panel, complete one row for each value you want to create.
 - In the Value field, type the name of the value you are creating — for example, the name of a theme if you selected LAAR_LIBRARY_THEME as you opened the form. The Translated Value field defaults to the same entry; you can't change it.
 - In the Description field, type a brief explanation of the value.
 - Ensure that the Enabled check box is selected for the value to take effect.
 - To define a limited period for the value to remain in effect, enter starting and ending dates in the From and To fields. For each, select a date in the pop-up calendar that appears when you click on the list-of-values icon, or type a date in the format configured for your instance of Oracle Applications. Or, to allow the value to remain in effect immediately and indefinitely, leave the From and To fields blank.

The screenshot shows the 'Segment Values' window with the 'Value Set' radio button selected. The 'Name' field contains 'LAAR_LIBRARY_THEME' and the 'Translated Value' field contains 'Library Theme'. Below this, the 'Values (LAAR_LIBRARY_THEME)' panel is active, showing a table with columns: Value, Translated Value, Description, Enabled, From, and To. The first row is pre-filled with 'ALL', 'ALL', 'Theme to include all rules', and the 'Enabled' checkbox is checked. There are several empty rows below it. At the bottom of the window, there are three buttons: 'Define Child Ranges', 'Move Child Ranges', and 'View Hierarchies'.

Value	Translated Value	Description	Enabled	From	To
ALL	ALL	Theme to include all rules	☒		
			☒		
			☐		
			☐		
			☐		
			☐		
			☐		

- 5 When you finish entering values, save your work: Click on File in the menu bar, and then Save in the File menu.

Using the Mass Associate Utility

A Mass Associate utility enables you to add Logical Apps form functions to Oracle Navigator menus or to responsibilities with which they are not already associated:

- 1 Click on LogicalApps Utilities in the menu bar, and then on Mass Update function in the Utilities menu. The following form appears:

- 2 In the Function Name field, select the Logical Apps form function you want to associate with menus or responsibilities.
- 3 Click on the Menu or Responsibility radio button. The Associate Function list then displays all menus or responsibilities not already associated with the function selected in the Function Name field.
- 4 Click the Associate Flag check box corresponding to each menu or responsibility you want to associate with that function.
- 5 Click on the Submit button. Users with access to the newly associated menus or assigned to the newly associated responsibilities then have access to the function.

Rules Caching

Rather than re-evaluate a rule each time a form affected by the rule is opened, AppsForm maintains a cache of rule-evaluation data. This greatly improves performance.

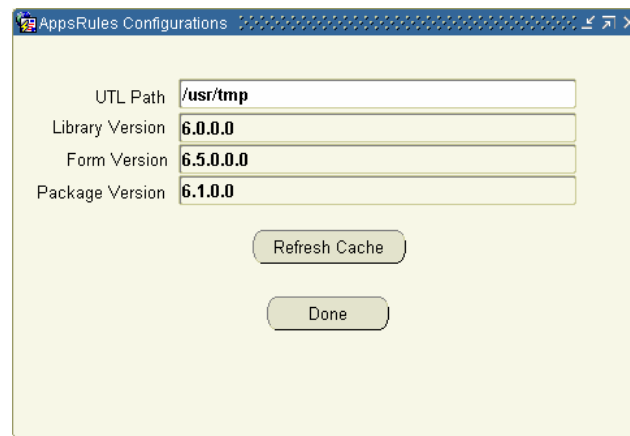
- As rules are created, AppsForm maintains a record in a table — LA_BR_CONFIGURATIONS — that stores a string comprising the names of all forms against which rules have been written. AppsForm updates the table whenever a user closes the AppsRules form, and reads from the table to update memory whenever a user opens an Oracle form associated with a rule.
- Similarly, as rules are created, AppsForm stores a record for each rule element in a table called LA_BR_FORM_RULE. AppsForm updates the table whenever a user

closes the AppsRules form, and reads from the table to update memory whenever a user opens an Oracle form associated with a configured rule detail.

It is important that the cache be refreshed when you test rules. The best way to do this is, once you have created a rule you want to test, close AppsRules and navigate to the Oracle Applications form against which you have written the rule. You should not keep AppsRules and the target Oracle Applications form open simultaneously, as this may cause inconsistencies in caching. However, you can refresh the cache manually. Moreover, you need to refresh the cache manually when you migrate rules to a target instance.

To refresh the cache manually:

- 1 With AppsForm open, click on Tools in the menu bar, and then AppsRules Configuration in the Tools menu. The following AppsRules Configurations window opens.



- 2 Click on the Refresh Cache button. Note that the window also provides information about the versions of AppsRules components you are running.
- 3 Click on the Done button to close the window.



Note

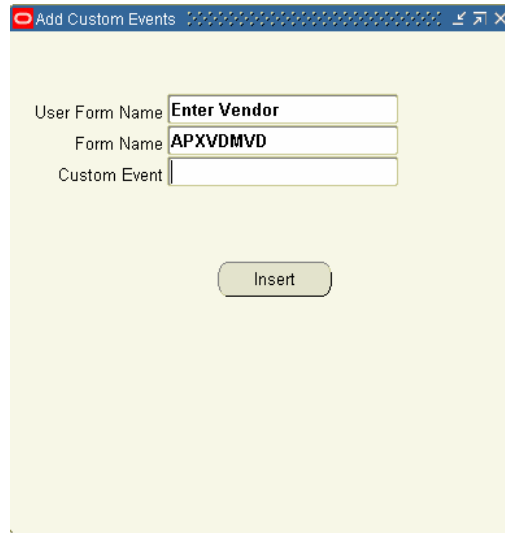
Because of caching, there is a limit of 500 forms to which rules can be attached.

Adding Custom Events

As you use the Event Tracker to capture blocks, fields, and other form components for use in AppsForm rules, you may also capture “undocumented” events, which you may then associate with a rule element to trigger its being processed. An alternative procedure enables you to capture such events for use as you create a rule element. You need to know the name of the event, and the form must call the event.

- 1 In the main AppsForm window, click on the rule element with which you want to use the event.
- 2 Click on Tools in the menu bar, and then on Add Custom Events in the Tools menu.

- 3** A note appears, informing you that using the Event Tracker is the recommended way to capture both standard and custom events. Click on the OK button to clear the note.
- 4** An Add Custom Events form appears:



The screenshot shows a window titled "Add Custom Events". Inside the window, there are three text input fields. The first field is labeled "User Form Name" and contains the text "Enter Vendor". The second field is labeled "Form Name" and contains the text "APXVDMVD". The third field is labeled "Custom Event" and is currently empty. Below these fields is a button labeled "Insert".

- 5** Select the form that uses the event. Do this in either of two ways:
 - In the User Form Name list of values, select the “user friendly,” external name for the form. AppsForm then supplies a corresponding value in the Form Name field.
 - In the Form Name list of values, select the name by which internal program code recognizes the form. AppsForm then supplies a corresponding value in the User Form Name field.
- 6** In the Custom Event field, type the name of the undocumented event.
- 7** Click on the Insert button.

Support

Logical Apps offers many services to assist you with the AppsForm implementation. From on-site support to remote phone and web support, our team of experienced professionals provides the help and information you need to ensure quick and effective implementation. The Logical Apps team includes a Technical Support Representative, an Account Manager, and a Logical Apps staff consisting of consultants and support specialists.

Feedback

Thank you for using Logical Apps AppsForm. We value your comments and feedback. Mail your comments to the following address, or call us directly at (949) 453-9101.

Logical Apps
Attn: Documentation Management
15420 Laguna Canyon Road
Suite 150
Irvine, CA 92618
U.S.A.

Or send email to support@Logical Apps.com.

