ORACLE 11.5.10 FORM PERSONALIZATION
CONTROL YOUR APPLICATIONS

Martin Sugg
Abaris, Inc
Ports America Group

Overview – Oracle introduced Form Personalization with 11.5.10 application code. Form Personalization is a declarative approach to alter the behavior of Oracle screens without customizing Oracle base code. Very similar to the CUSTOM.pll library, Form Personalization allows developers to control the Oracle Application at runtime.

The features in the Form Personalization include changing item properties, creating menu entries, showing messages and using built-ins. With Form Personalization, the Oracle goal remains: Extend Applications with No Customizations.

Limitations – However, there are still limitations.

- Only the features allowed at runtime
- Cannot change an item's datatype
- Cannot create new items
- Cannot move items from canvas to canvas, tab to tab
- Cannot display items not on a canvas
- Although it may be possible to use other Form Events, typically the available ones are:
  - WHEN-NEW-FORM-INSTANCE
  - WHEN-NEW-BLOCK-INSTANCE
  - WHEN-NEW-RECORD-INSTANCE
  - WHEN-NEW-ITEM-INSTANCE
  - WHEN-VALIDATE-RECORD
  - MENU1-10
  - SPECIAL1-45

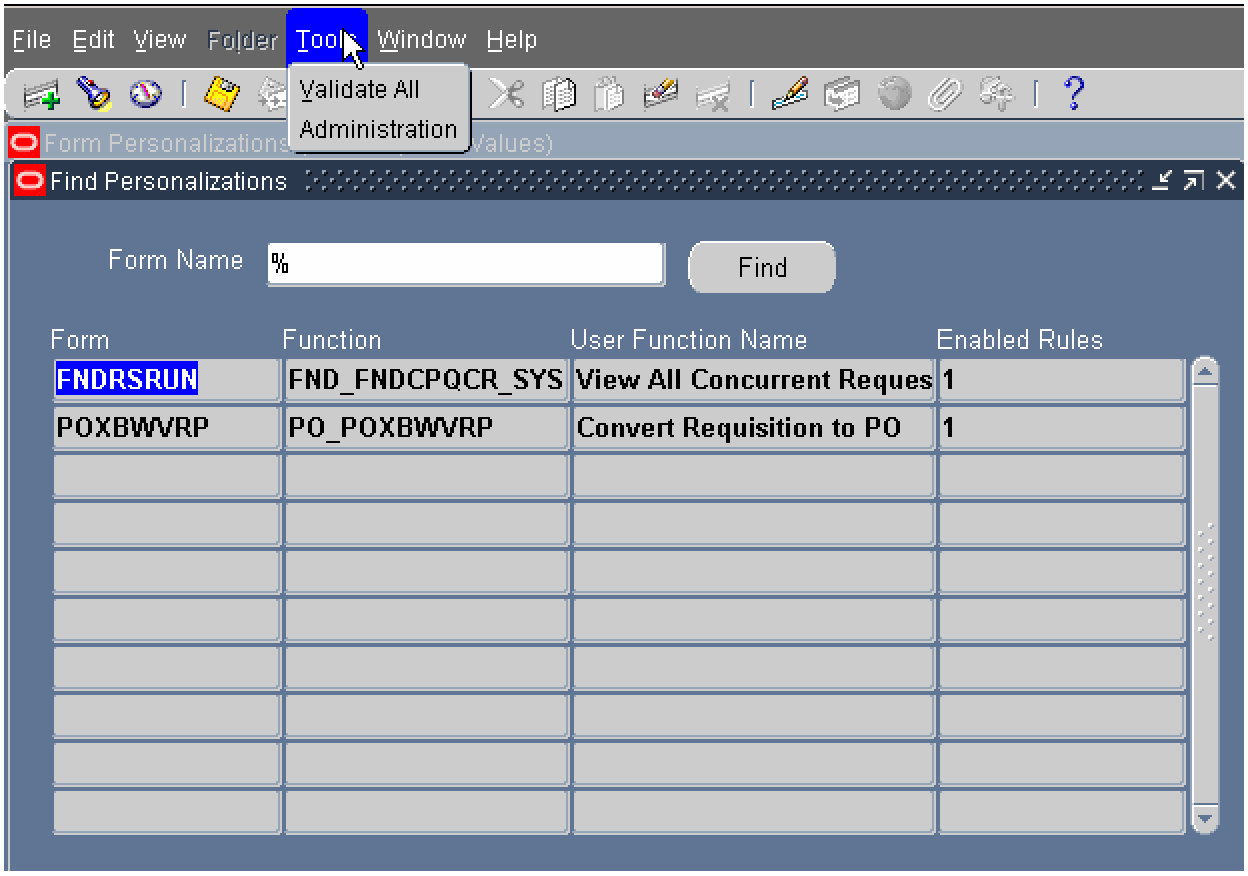
Profiles and Access – Access to Form Personalization is intended for users with experience in Forms Development, PL/SQL and the Oracle Applications Development guidelines. Therefore, access should be restricted.

There are two profile options designed to handle access to Form Personalization as well as access to other Diagnostic tools.

- Utilities:Diagnostics – enable or disable the APPS password for Help > Examine
  - Yes – APPS password is not required by this user, responsibility or site.
  - No – The APPS password is required by this user, responsibility or site.
- Hide Diagnostics Menu Entry – enable or disable the Help > Diagnostic tools. This does not disable the entire Help menu.
  - Yes – the Diagnostics menu is hidden
  - No – the menu is available

Navigation to the Form Personalization is the same for every Oracle Form. To create a new or view existing Form Personalizations, navigate to the Oracle Form in question. Then, from the top menu, navigate to Help > Diagnostics > Custom Code > Personalize.

Once inside a Form Personalization, users can navigate to the Tools menu and select Tools > Administration. This will display all the Oracle Forms with Personalizations.



Form Name %

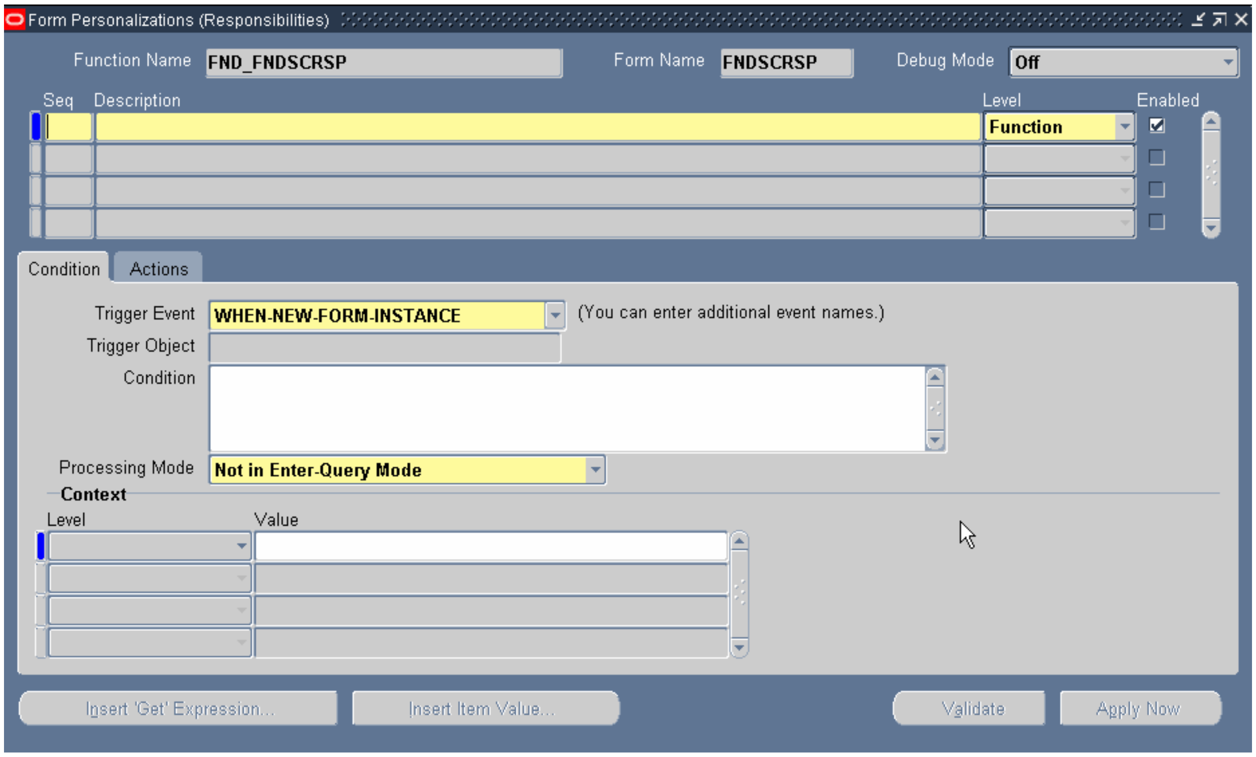
Form	Function	User Function Name	Enabled Rules
FNDRSRUN	FND_FNDCPQCR_SYS	View All Concurrent Reques	1
POXBWVRP	PO_POXBWVRP	Convert Requisition to PO	1

Fig 1: Illustration of the Administration Form Personalization screen

In addition, the following tables store Form Personalization:

- FND_FORM_CUSTOM_RULES
- FND_FORM_CUSTOM_ACTIONS
- FND_FORM_CUSTOM_SCOPES
- FND_FORM_CUSTOM_PROP_LIST
- FND_FORM_CUSTOM_PROP_VALUES
- FND_FORM_CUSTOM_PARAMS
- FND_INDUSTRIES

Anatomy of the Form Personalization – The Form Personalization has basic criteria which must be met. There must be a condition which results to TRUE, and there must be an action associated to that TRUE condition.


Form Personalizations (Responsibilities)

Function Name FND_FNDSCRSP Form Name FNDSCRSP Debug Mode Off

Seq	Description	Level	Enabled
		Function	✓

Condition | Actions

Trigger Event WHEN-NEW-FORM-INSTANCE (You can enter additional event names.)

Trigger Object

Condition

Processing Mode Not in Enter-Query Mode

Context

Level	Value

Fig 2: Form Personalization rule and condition

In the above screen, the Sequence and Description make up the "Rule". Oracle processes each rule in sequential order, lowest to highest. In planning your Form Personalization, make sure a lower rule does not get cancelled by a higher sequenced rule.

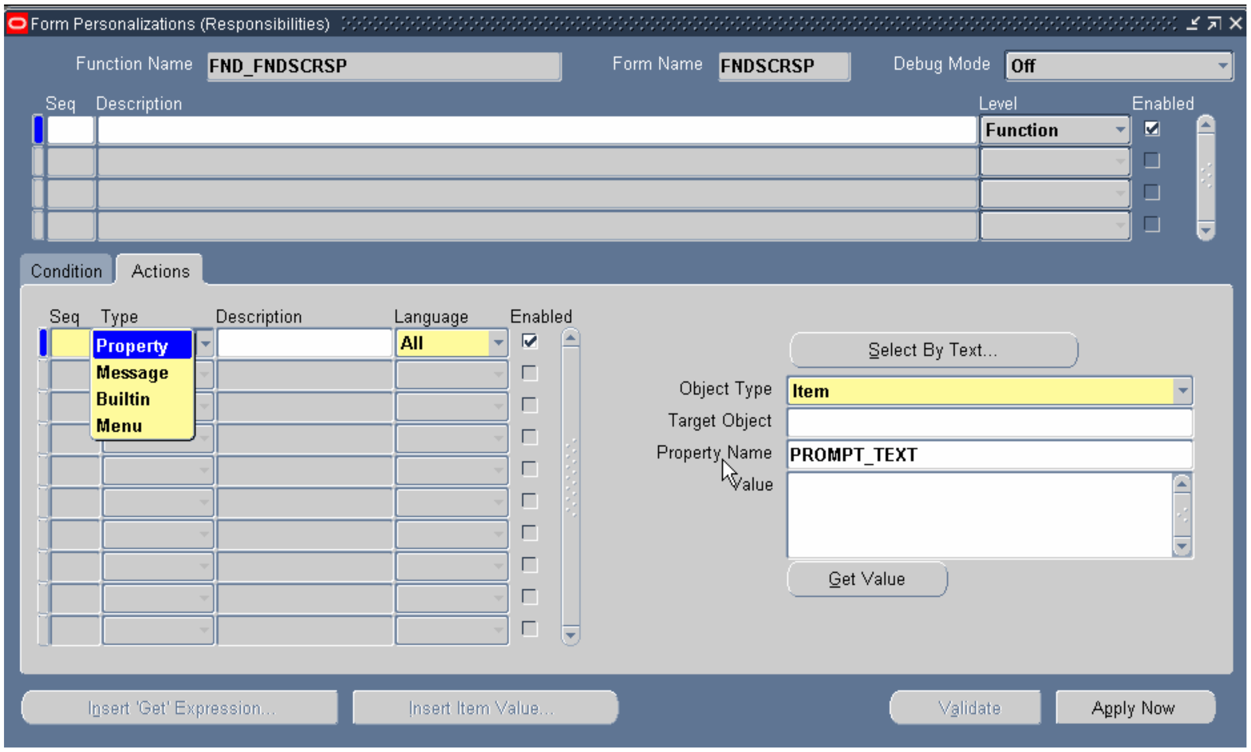
On the Condition Tab, the Trigger Event is set to one of the triggers listed above. The WHEN-NEW-FORM-INSTANCE does not require a target object. Other events need a target object to fire the condition. The WHEN-NEW-BLOCK-INSTANCE requires a trigger object of a Block, and so on…

The Condition Text Box includes logic to evaluate at runtime. If the Condition Text Box is null, then the rule will fire at each trigger event. Use of the Condition is described later in several examples. The Condition can include SQL operators like AND, OR, DECODE, NVL, TO_NUMBER, TO_CHAR, etc. as well as the use of bind variables (:block.field).

When the focus is on the Condition text box, the Insert 'Get' Expression and the Insert Item Value buttons are available. By using the Insert Item Value button, choose an item in the Form from the pop-up menu and the button will insert the :block.field configuration. By using the Insert 'Get' Expression and using the pop-up menu to find the inventory_item_id and its value, the button will return the formula necessary to evaluate the inventory_item_id value at runtime; ${item.related_items_pln_info.inventory_item_id.value}

The Processing Mode is used to determine if the rule is evaluated during normal screen entry, or in query mode, or both. This is rarely changed, however, it is mentioned below in the Zoom example.

The Context region in the Condition Tab contains information on when this personalization will apply. The choices are Site, Responsibility and User. The Industry choice is available but reserved for future use. At the Site level, this rule will fire every time this form is accessed. At the Responsibility level, the Value field is available and the Developer can choose from a list of values for the responsibility needed. There can be multiple responsibilities declared. The User level acts the same way. Responsibility and User can be used together for extreme tight control, however, once the Site level is chosen, then the other level context are ignored.


Form Personalizations (Responsibilities)

Function Name: FND_FNDSCRSP
Form Name: FNDSCRSP
Debug Mode: Off

Seq	Description	Level	Enabled
		Function	✓

Condition | Actions

Seq	Type	Description	Language	Enabled
	Property		All	✓

Object Type: Item
Target Object:
Property Name: PROMPT_TEXT
Value:

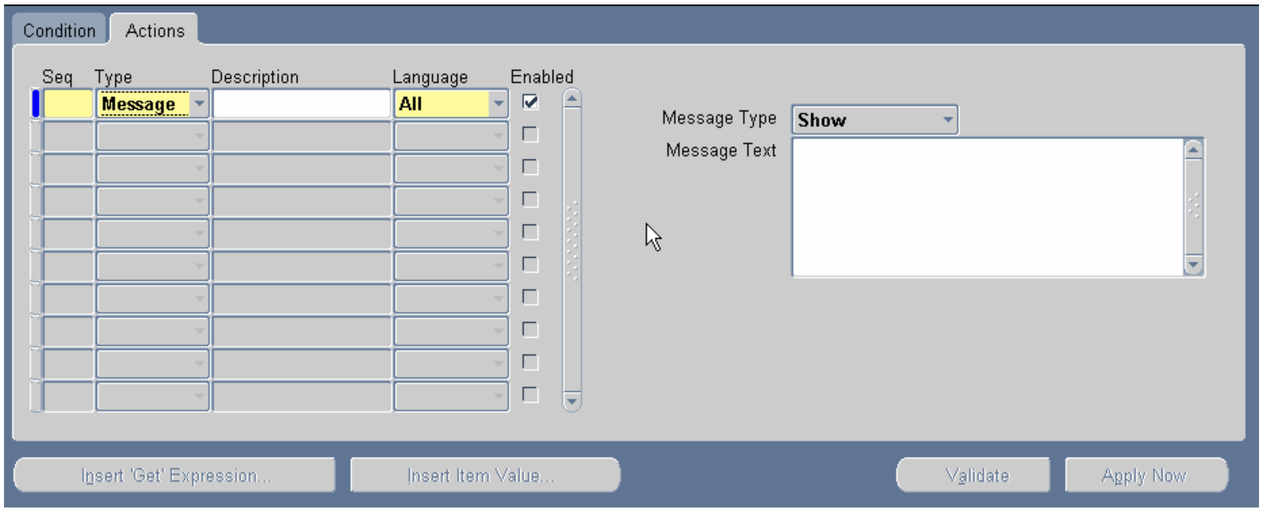
Fig 3: Form Personalization Action

In the Action tab, the action is created for the TRUE condition. A sequence number is assigned first. At runtime, the actions will fire in that sequence from lowest to highest.

There are four Types of actions: Property, Message, Builtin and Menu. Choose the action that is needed for this sequence. The language and Enabled flag default. However, in multi-language environments, Actions can be at the language level. This provides even more granularity for detail.

In the above diagram, the Action is for a Property value. In the Object Type field to the right, there is a pull down with several choices of object you can set properties values, including Item, LOV, Window, Block, Radio Button, Global and Local variables. Depending on the object type, the Target Object list of values will display what is available in this form. The property name LOV will display the available properties developers can set. The Get Value button will retrieve the property value of the target object on the current record in the form. This is helpful to determine how you want to set the value.

The Select By Text… button allows you to search the entire form for objects that can be altered. Once you select the object you need, the three fields will default for you.



Type: Message; Language: All; Enabled: ✓

Message Type: Show

Message Text:

Fig 4: Message Action

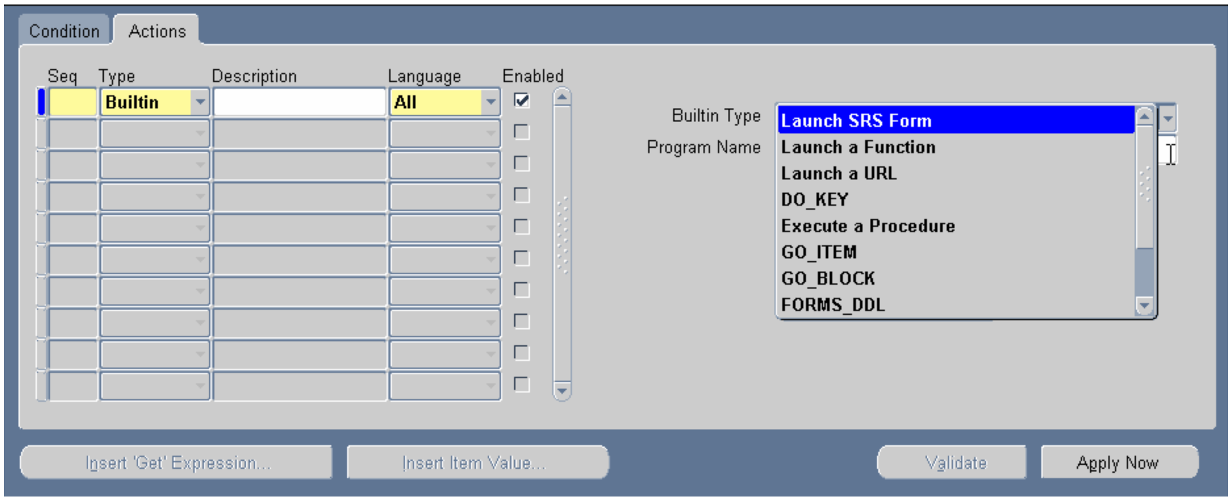
The above illustration shows the Message options for Action Type of Message. Message Types include Show, Debug, Error, Hint and Warn.

Show messages will display a pop up message if the condition evaluates to TRUE.

The Debug message will only display if the Debug Mode is set to Show Debug Messages in the header of the rule. It is next to the Function Name, Form Name fields in the top of the screen.

Error and Warn messages will cause a Form Failure Trigger if the user selects the Cancel option.

The Hint message will display at the bottom of the form as standard hint messages.



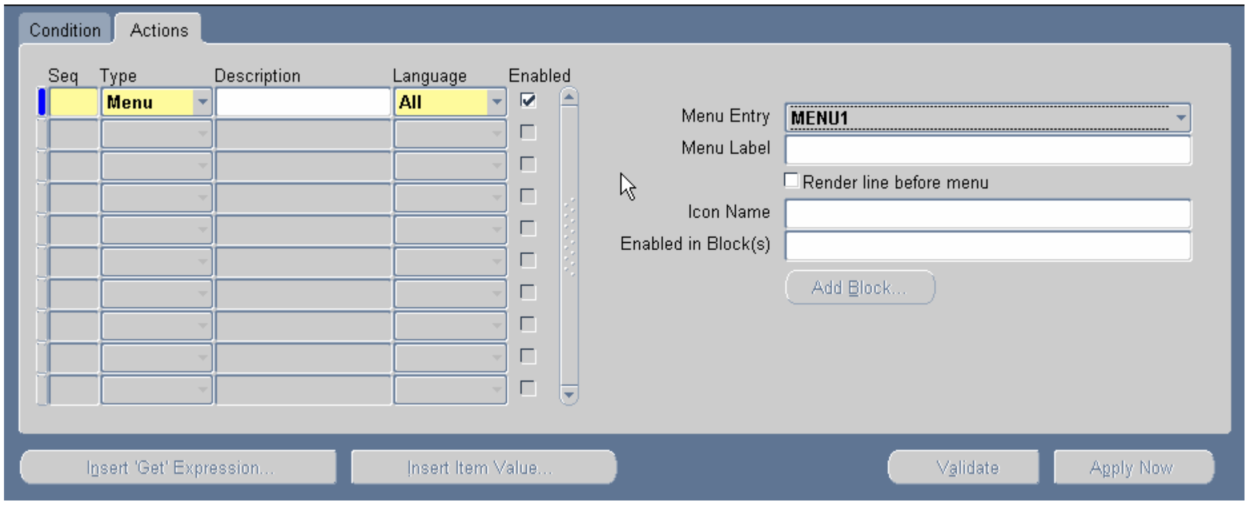
Type: Builtin; Language: All; Enabled: ✓

Builtin Type: Launch SRS Form

Program Name:

Fig 5: Builtin Action

This illustration shows some of the options for Builtins. Once an option is chosen, the Personalization form will display necessary parameters needed for the option. The Launch SRS Form will allow the rule to display the standard SRS submission screen with a chosen Concurrent Program and parameters available. The Launch a Function option will allow developers to choose a new Form to launch similar to an Oracle Zoom. This is discussed in great detail below. Launch a URL will launch a website location. The DO_KEY built-in allows developers to control form functionality by starting or executing a query, clearing a form, etc and is discussed in the example section below. GO_ITEM and GO_BLOCK will control the form to move to the ITEM or BLOCK as indicated. The FORMS_DDL option allows developers to write dynamic SQL, such as an INSERT statement into a custom table or calling a package procedure or function. There are limitations to called packages and functions. Please review the Metalink notes provided below.



Seq	Type	Description	Language	Enabled
	Menu		All	✓

Menu Entry: MENU1
Menu Label:
☐ Render line before menu
Icon Name:
Enabled in Block(s):

Fig 6: Menu Actions

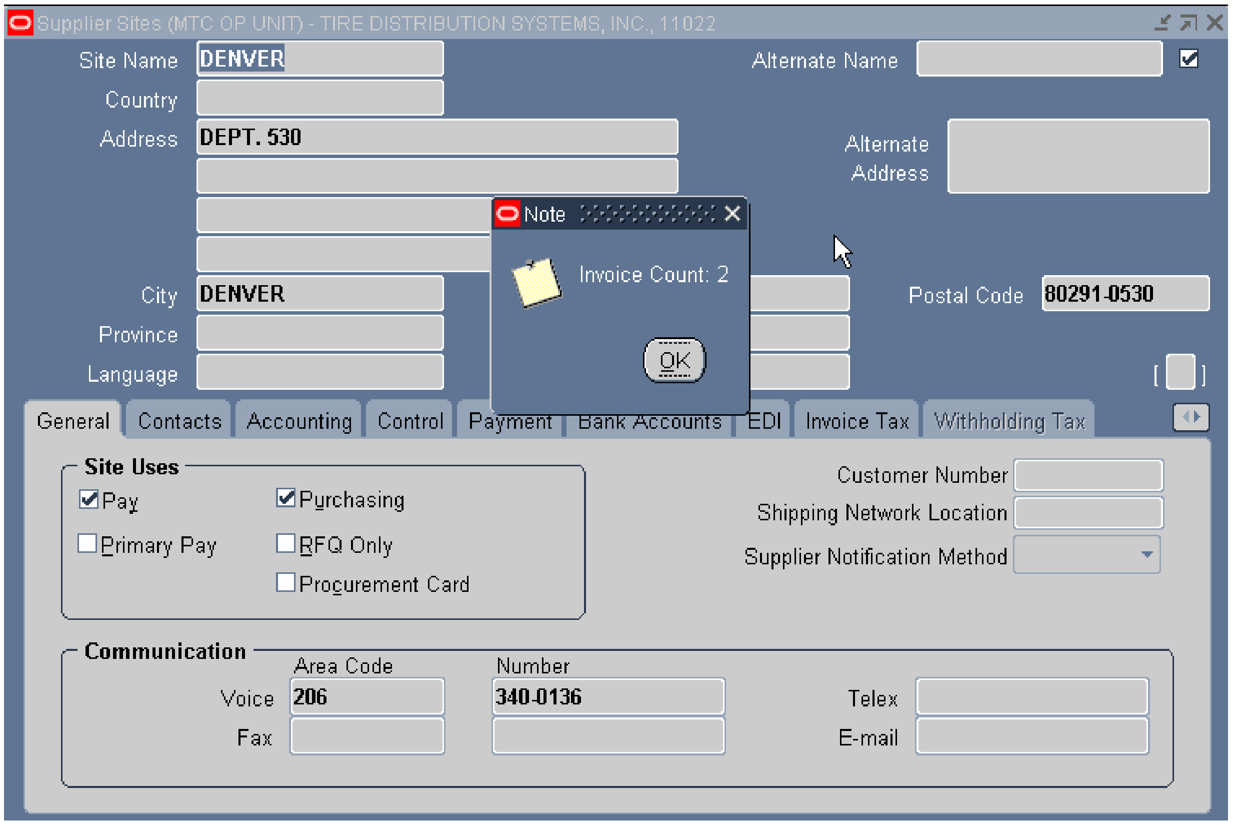
The Menu action type allows developers to choose the menu entry to use, assign a label and determine which block to display the menu if appropriate. Some Oracle Forms already have certain menu entries reserved. The Form Personalization will display which menu entries are already in use.

Menus 1 – 10 display in the Tools menu in Oracle. Special 1 – 15 will display under the Tools heading, Special 16 – 30 will display under the Reports heading and Special 31 – 45 will display under the Actions heading. Certain forms have Tools, Reports and Action heading.

When creating a menu option, there is a two step process. First, you must display the menu. Second, a Form Personalization rule must be created for when the user selects the menu option. This is discussed in several examples below.

Example – Using SELECT in a Message

In this first example, you will learn how to use a SELECT statement in a pop-up message:



Supplier Sites (MTC OP UNIT) - TIRE DISTRIBUTION SYSTEMS, INC., 11022

Site Name	DENVER
Country	
Address	DEPT. 530
City	DENVER
Province	
Language	
Alternate Name	
Alternate Address	
Postal Code	80291-0530

Note: Invoice Count: 2

Site Uses: Pay [✓], Purchasing [✓], Primary Pay [ ], RFQ Only [ ], Procurement Card [ ]

	Area Code	Number
Voice	206	340-0136
Fax		

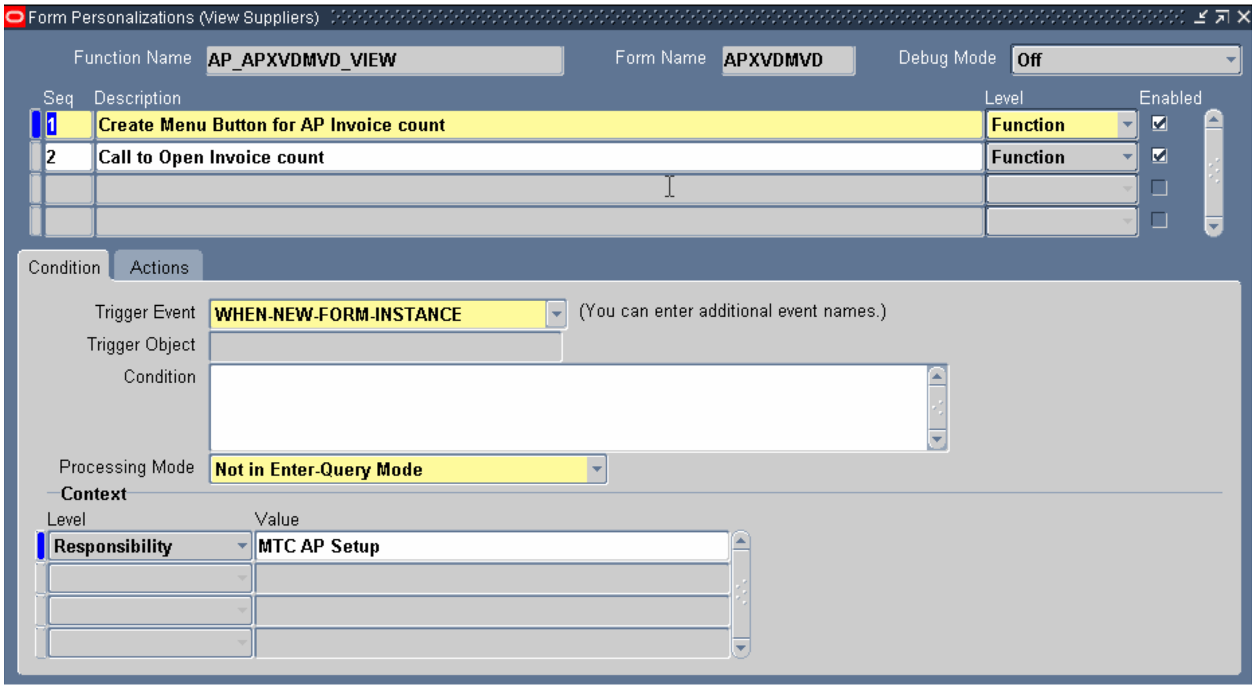
Fig 7: Open Invoice Count message

In this Form Personalization, when a user navigates to the PO Supplier Sites inquiry screen, there will be a Tools > Menu option to quickly view open invoice counts.   This is key for the users to know if any open invoices exist prior to changing a supplier's address or default values.

Navigate to the Supplier Inquiry screen in a Payables responsibility; Suppliers > Inquiry.

Query a Supplier and navigate to its Supplier Site by pressing the Sites button.

Once in the Form, navigate to Help > Diagnostics > Custom Code > Personalize.


Form Personalizations (View Suppliers)

Function Name: AP_APXVDMVD_VIEW | Form Name: APXVDMVD | Debug Mode: Off

Seq	Description	Level	Enabled
1	Create Menu Button for AP Invoice count	Function	✓
2	Call to Open Invoice count	Function	✓

Condition | Actions

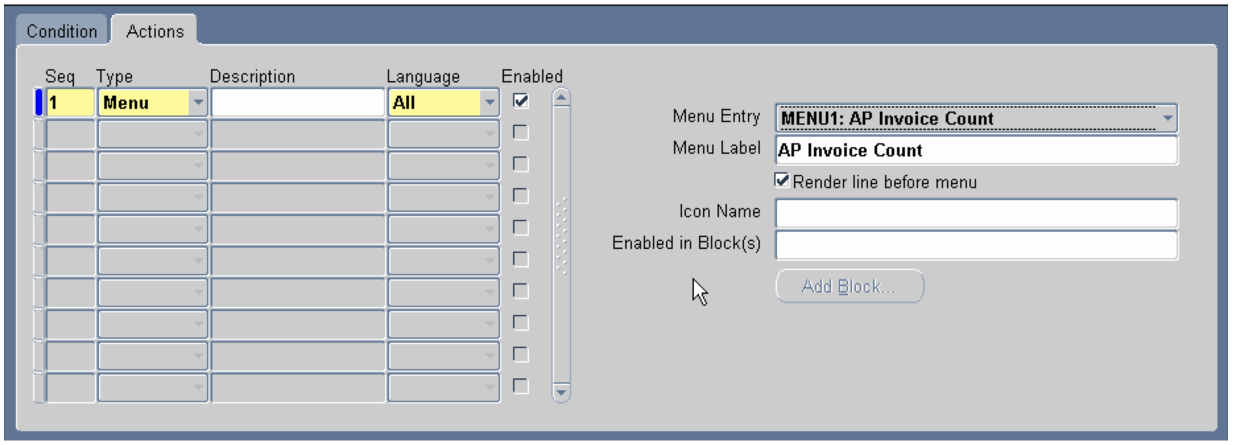
Trigger Event: WHEN-NEW-FORM-INSTANCE (You can enter additional event names.)
Trigger Object:
Condition:
Processing Mode: Not in Enter-Query Mode

Context

Level	Value
Responsibility	MTC AP Setup

Fig 8: Personalization form for the View Suppliers form APXVDMVD

First, create the menu item. Before a user can select our message from the Tools menu, a Menu option must be created. Create a rule in Seq 1. This is a WHEN-NEW-FORM-INSTANCE trigger with no condition. In other words, the finished menu option will always display. Optionally, set to the Responsibility context for a particular responsibility.

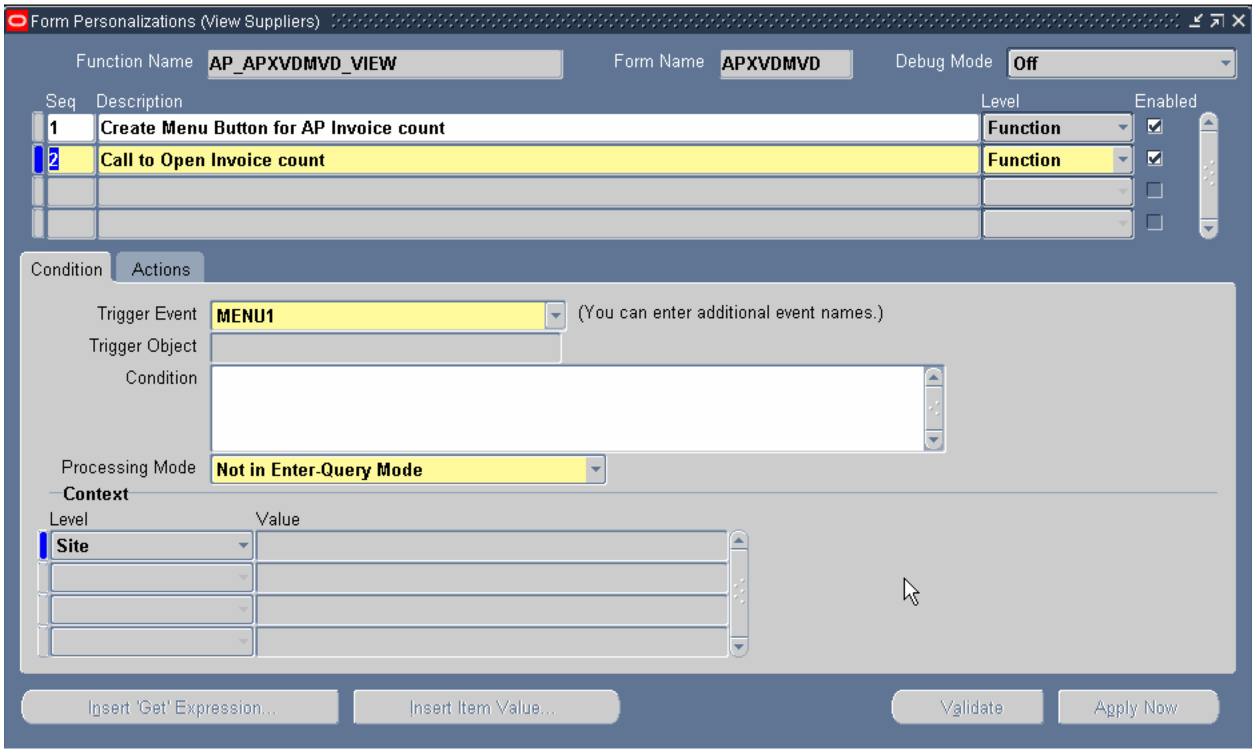

Condition | Actions

Seq	Type	Description	Language	Enabled
1	Menu		All	✓

Menu Entry: MENU1: AP Invoice Count
Menu Label: AP Invoice Count
✓ Render line before menu
Icon Name:
Enabled in Block(s):

Fig 9: Menu Action

Next, create a Menu action to label MENU1. In this screen shot, MENU1 has already been labeled. Check the Render line before menu option to draw a line above this menu choice. This helps separate standard menu options from Personalized menu options.



Form Personalizations (View Suppliers)

Function Name	AP_APXVDMVD_VIEW	Form Name	APXVDMVD	Debug Mode	Off

Seq	Description	Level	Enabled
1	Create Menu Button for AP Invoice count	Function	✓
2	Call to Open Invoice count	Function	✓

Condition | Actions

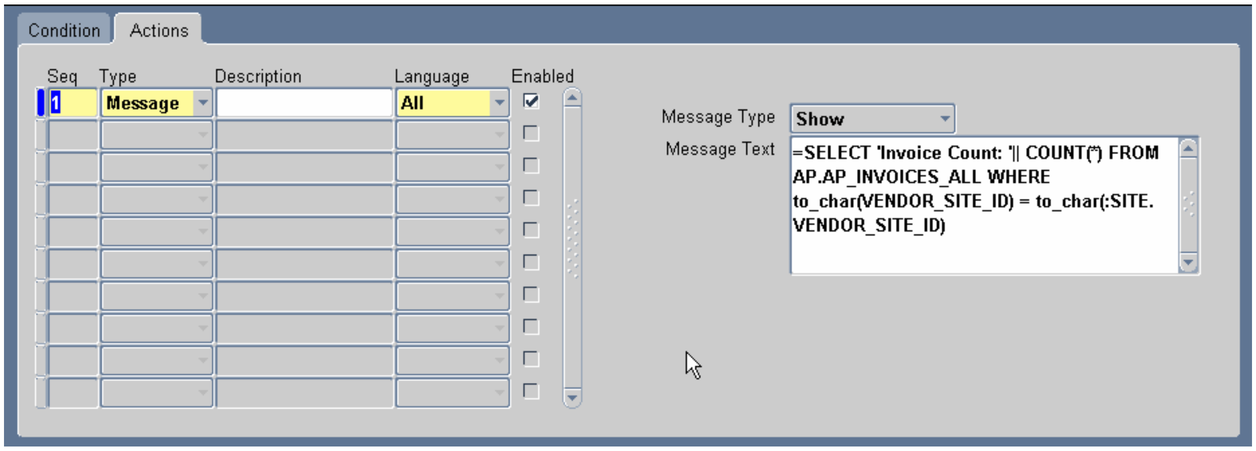
Trigger Event: MENU1 (You can enter additional event names.)
Trigger Object:
Condition:
Processing Mode: Not in Enter-Query Mode

Context

Level	Value
Site	

Fig 10: Rule for Menu option

Next, create Rule Seq 2 for when the user clicks the menu option. The trigger event is Menu1 which was enabled in the first step. When the user selects this menu option from the Tools menu, the action in the next step will fire.



Condition | Actions

Seq	Type	Description	Language	Enabled
1	Message		All	✓

Message Type: Show
Message Text: =SELECT 'Invoice Count: '|| COUNT(*) FROM AP.AP_INVOICES_ALL WHERE to_char(VENDOR_SITE_ID) = to_char(:SITE.VENDOR_SITE_ID)

Fig 11: Message Action

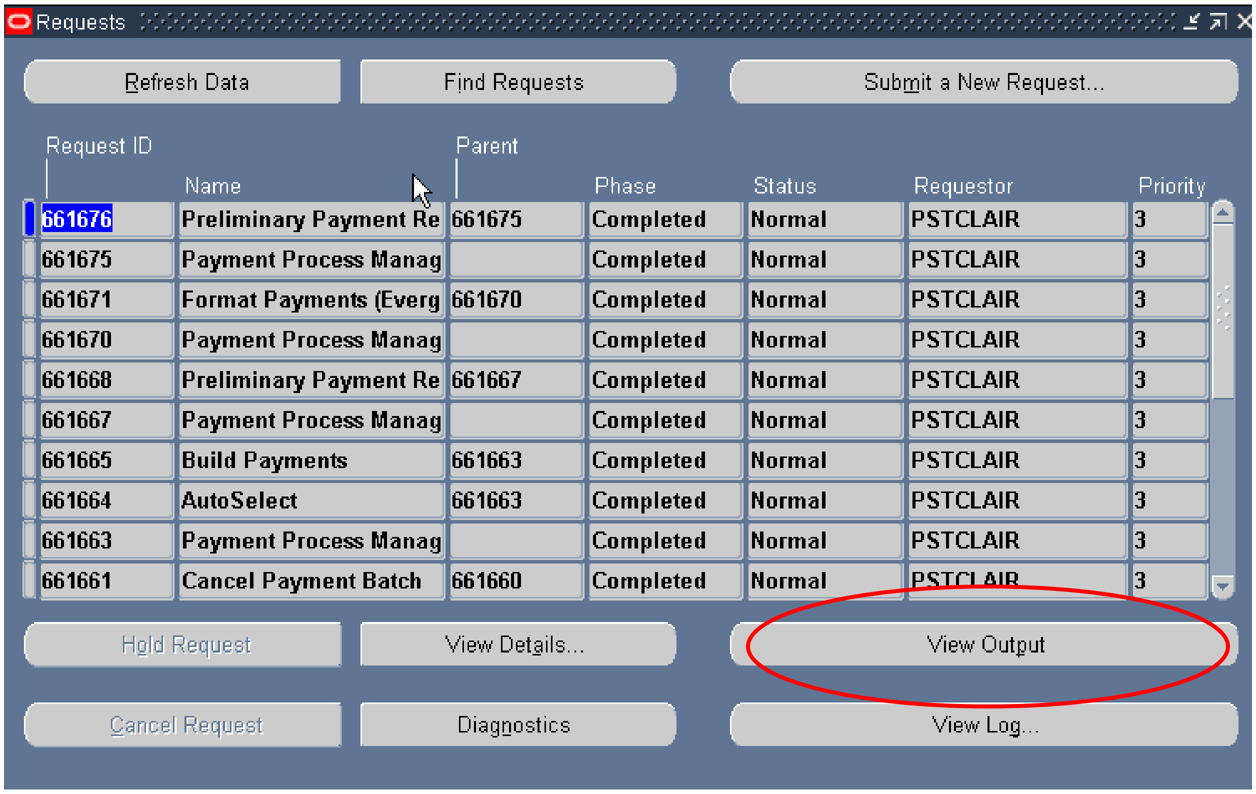
Finally, create a Message action for when the menu option is chosen. The use of a SELECT statement has to start with =SELECT. Notice the to_char use in the VENDOR_SITE_ID comparison. The use of a to_char statement is needed to compare a number datatype to a :block.field reference.

Query used in message:
=SELECT 'Open Invoice Count: '|| COUNT(*) FROM AP.AP_INVOICES_ALL WHERE to_char(VENDOR_SITE_ID) = to_char(:SITE.VENDOR_SITE_ID) and invoice_amount - nvl(amount_paid,0) > 0

Save your work. To test the message, press the Validate button. This will run the select statement for the supplier site currently queried in the form. The pop-up message should display the count of open invoices for that supplier site. Test the message by querying multiple Suppliers and Sites and review their open invoice count.

Example – Enable the View Output Button

In this example, you will learn how to enable a button.



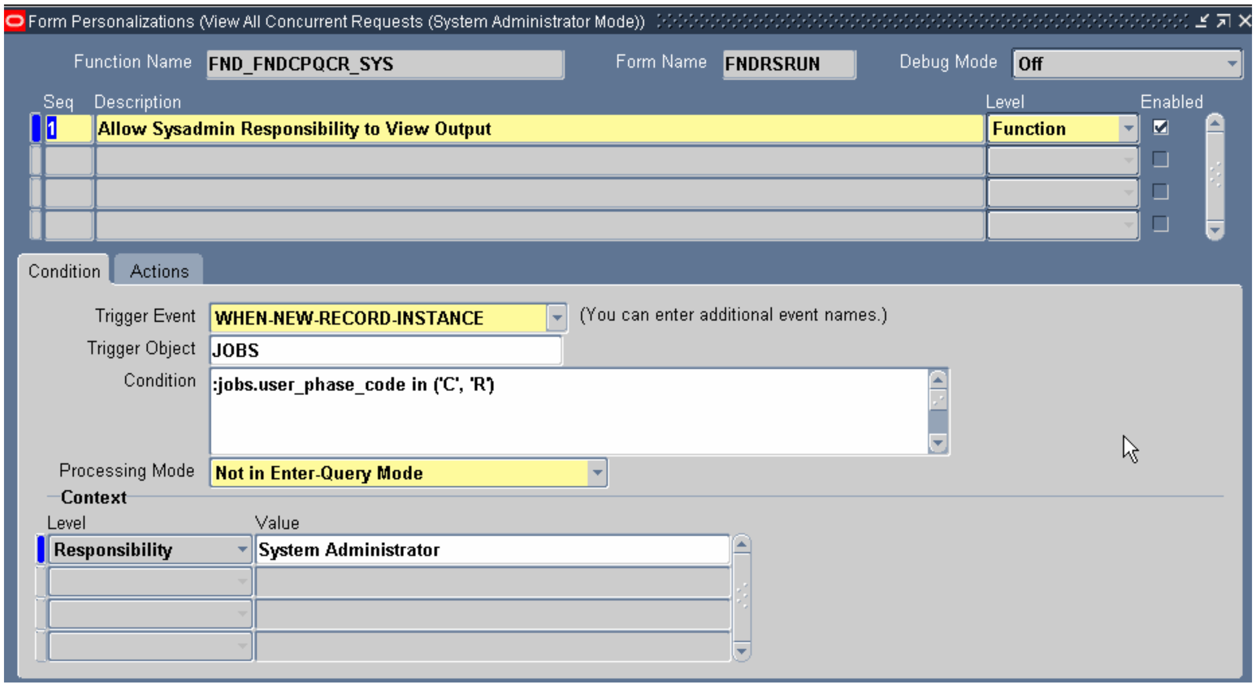
Request ID	Name	Parent	Phase	Status	Requestor	Priority
661676	Preliminary Payment Re	661675	Completed	Normal	PSTCLAIR	3
661675	Payment Process Manag		Completed	Normal	PSTCLAIR	3
661671	Format Payments (Everg	661670	Completed	Normal	PSTCLAIR	3
661670	Payment Process Manag		Completed	Normal	PSTCLAIR	3
661668	Preliminary Payment Re	661667	Completed	Normal	PSTCLAIR	3
661667	Payment Process Manag		Completed	Normal	PSTCLAIR	3
661665	Build Payments	661663	Completed	Normal	PSTCLAIR	3
661664	AutoSelect	661663	Completed	Normal	PSTCLAIR	3
661663	Payment Process Manag		Completed	Normal	PSTCLAIR	3
661661	Cancel Payment Batch	661660	Completed	Normal	PSTCLAIR	3

Fig 12: View Output enabled

In the world of Oracle Application support, many people need to view the log and output files of their users for troubleshooting purposes. The View Log button has always been enabled, however, this personalization will allow only people with System Administrator responsibility to view the output of other users. Again, this is intended for troubleshooting purposes.

Login to the System Administrator responsibility and navigate to Request > View. In the find screen, choose the Specific Request radio button. Next, enter a Requester name that is not your own. This will show concurrent request from another user. Notice the View Output button is visible but disabled.

Navigate to Help > Diagnostics > Custom Code > Personalize.


Form Personalizations (View All Concurrent Requests (System Administrator Mode))
Condition
Actions

Function Name: FND_FNDCPQCR_SYS
Form Name: FNDRSRUN
Debug Mode: Off

Seq	Description	Level	Enabled
1	Allow Sysadmin Responsibility to View Output	Function	✓

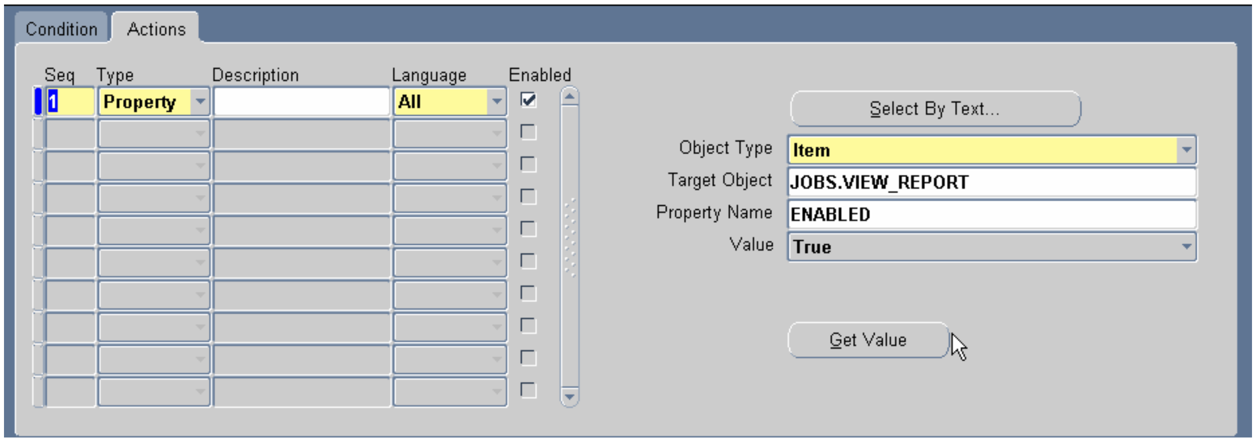
Trigger Event: WHEN-NEW-RECORD-INSTANCE (You can enter additional event names.)
Trigger Object: JOBS
Condition: :jobs.user_phase_code in ('C', 'R')
Processing Mode: Not in Enter-Query Mode

Context

Level	Value
Responsibility	System Administrator

Fig 13: View Request Personalization form

Create a rule Seq 1 set at the WHEN-NEW-RECORD-INSTANCE with the following condition - :jobs.user_phase_code in ('C', 'R'). With each advance to a new line on the form, the condition must be true to enable the button. C is Completed, R is Running. Some concurrent request produce output while running. Set the Context Level to Responsibility and choose System Administrator. This personalization will only be available to those who are trusted with System Administrator responsibilities.



Seq	Type	Description	Language	Enabled
1	Property		All	✓

Object Type: Item
Target Object: JOBS.VIEW_REPORT
Property Name: ENABLED
Value: True

Fig 14: View Request action

There is only one Action for this condition: Enable it! Set a Property Action in Seq 1. Use the Select By Text button to choose the Item and target object of JOBS.VIEW_REPORT. Set the Property Name to ENABLED and value to TRUE.

While in this screen, you can move the form out of the way to see the View Request form. Press the Apply Now button and the View Output button should become enabled. Instant gratification.

Save your work.

Other examples include:

- Disable a button by setting the ENABLED property to False
- Hide a button by setting the DISPLAYED property to False
- Show a button by setting the DISPLAYED property to True
- Make a field required by setting the REQUIRED property to True
- Hide a Tab by setting the DISPLAYED property to False

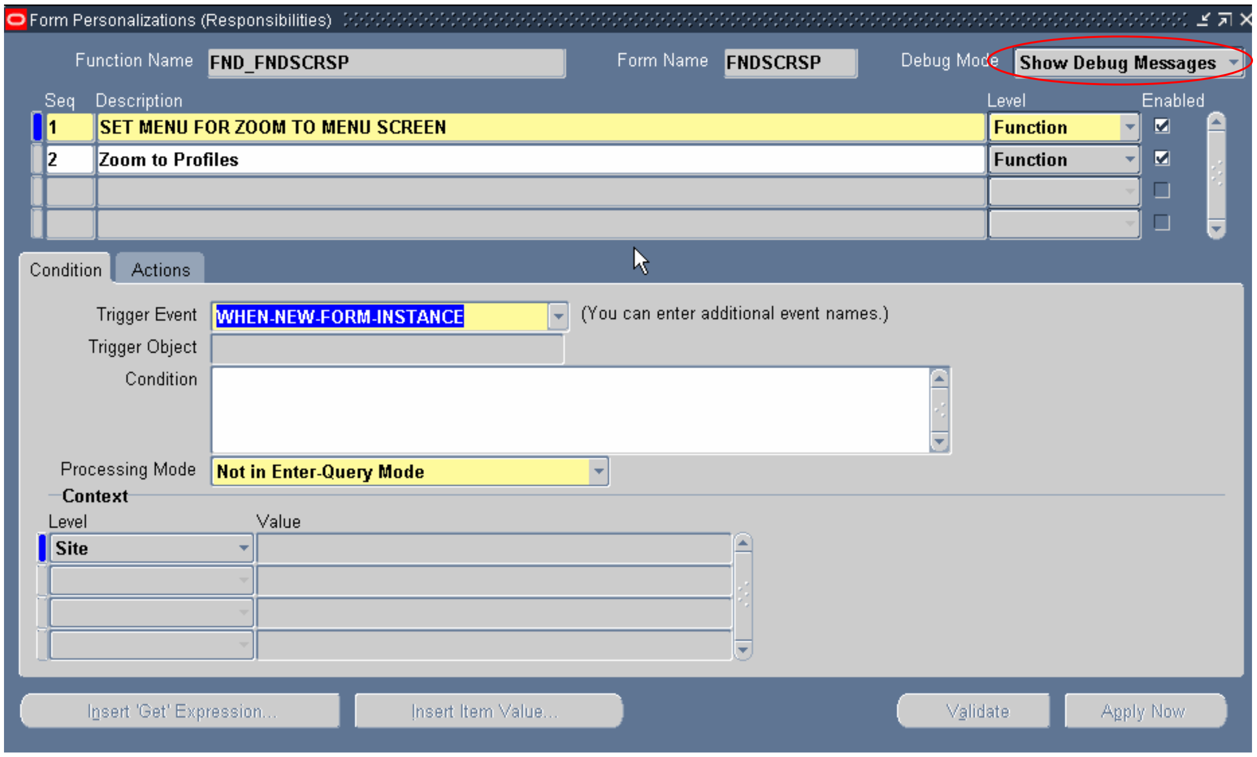
You can test this personalization by logging into another responsibility and navigating to View Request. Find a request submitted by another person and validate the View Output button is not enabled since you are not logged in with System Administrator responsibility.

Example – Zoom!

A common request in the CUSTOM.pll is to create Zooms. This allows a user to “zoom” from one screen to another which may not have been available in standard Oracle functionality. For instance, zoom to the employee inquiry screen from the PO supplier screen when the Vendor Type is Employee. Typically, a Zoom event coded in the CUSTOM.pll took time and development expertise. Also, in 11.5.10 applications, there is a feature to enable multiple Zooms on a block. However, this requires additional coding to the LOV event in the Custom Library.

By using Form Personalization, any number of Zooms can be created from any Form, Block, Record or Item. The only limit is the 10 menus or the 45 specials if the current Form does not reserve any of the menus or specials. Using Form Personalization to setup Zooms is much easier then CUSTOM.pll coding, however, the developer should have a good understanding of Form Functions and Global Variables. The standard Zoom rules still apply. The user’s responsibility menu still needs to include both Forms; the zoom-from Form and the zoom-to Form.

Navigate to the Responsibility form in the System Administrator responsibility.


Form Personalizations (Responsibilities)
Condition
Actions
Insert 'Get' Expression...
Insert Item Value...
Validate
Apply Now

Function Name: FND_FNDSCRSP   Form Name: FNDSCRSP   Debug Mode: Show Debug Messages

Seq	Description	Level	Enabled
1	SET MENU FOR ZOOM TO MENU SCREEN	Function	✓
2	Zoom to Profiles	Function	✓

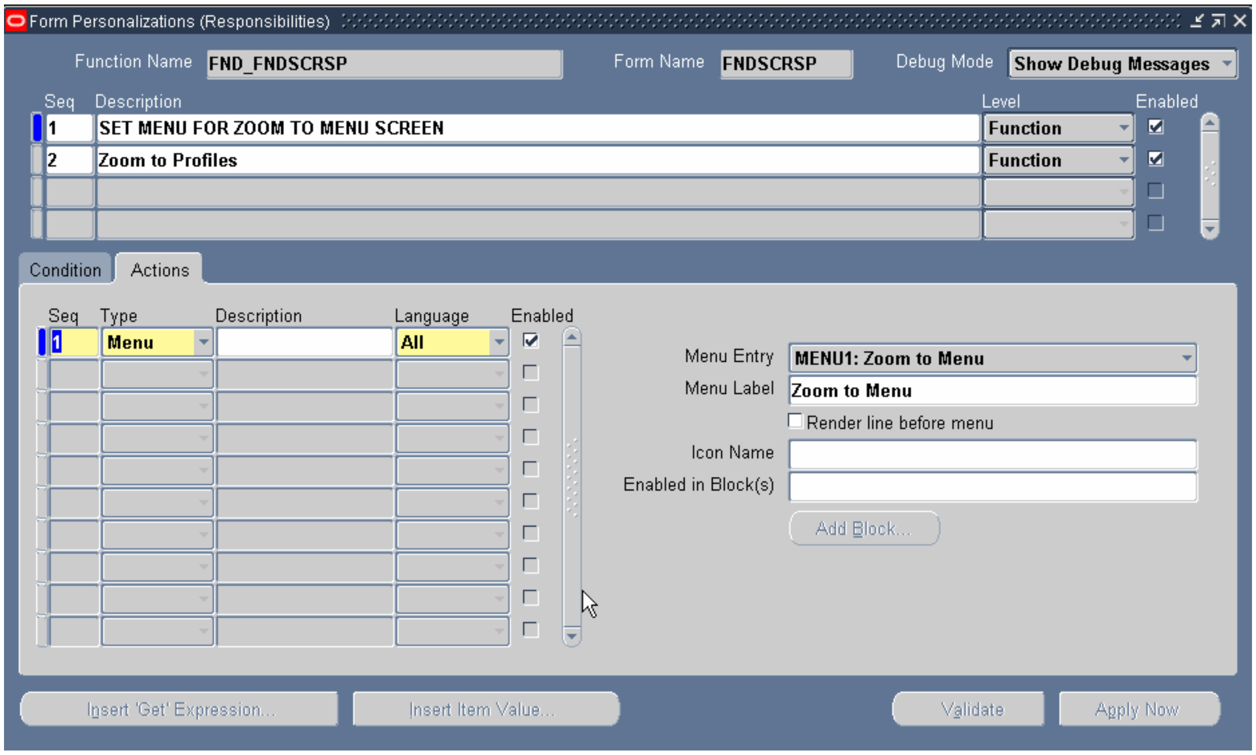
Trigger Event: WHEN-NEW-FORM-INSTANCE (You can enter additional event names.)
Trigger Object:
Condition:
Processing Mode: Not in Enter-Query Mode

Context

Level	Value
Site	

Fig 15: Responsibility form personalization screen with Show Debug Messages selected

In this example, we will create a Zoom from the Responsibility form to the Menu form, querying the Menu associated with the Responsibility. Just like the CUSTOM.pll zoom, it is a two step process.   First, turn it on. Second, do the action. Here we are creating a rule for the WHEN-NEW-FORM-INSTANCE to enable Menu1 in the Tools Menu. This is set at the Site level and with no condition. Also, notice the Show Debug Message is enabled. This will help us display the menu name before Zooming to the Menu form, for debugging purposes. This can be disabled later.


Form Personalizations (Responsibilities)
Condition
Actions
Add Block...
Insert 'Get' Expression...
Insert Item Value...
Validate
Apply Now

Function Name: FND_FNDSCRSP
Form Name: FNDSCRSP
Debug Mode: Show Debug Messages

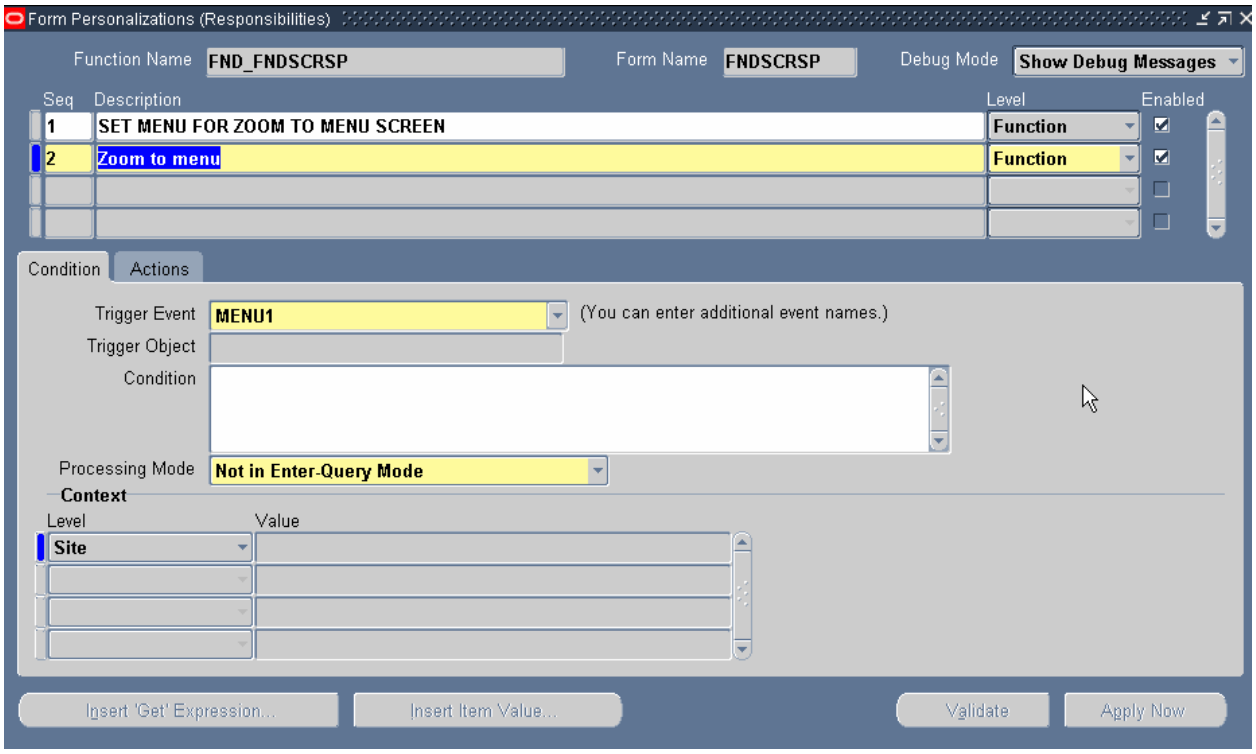
Seq	Description	Level	Enabled
1	SET MENU FOR ZOOM TO MENU SCREEN	Function	✓
2	Zoom to Profiles	Function	✓

Seq	Type	Description	Language	Enabled
1	Menu		All	✓

Menu Entry: MENU1: Zoom to Menu
Menu Label: Zoom to Menu
☐ Render line before menu
Icon Name:
Enabled in Block(s):

Fig 16: Menu Action for Zoom

For Rule 1, the Action is to enable the MENU1 option with a label of Zoom to Menu. Once you save this form, the MENU1 name is changed to MENU1: Zoom to Menu. Optionally check the box to Render line before menu and / or Enabled in Blocks parameter to only allow the Menu to be available in certain form blocks.


Form Personalizations (Responsibilities)

Function Name: FND_FNDSCRSP
Form Name: FNDSCRSP
Debug Mode: Show Debug Messages

Seq	Description	Level	Enabled
1	SET MENU FOR ZOOM TO MENU SCREEN	Function	✓
2	Zoom to menu	Function	✓

Condition | Actions

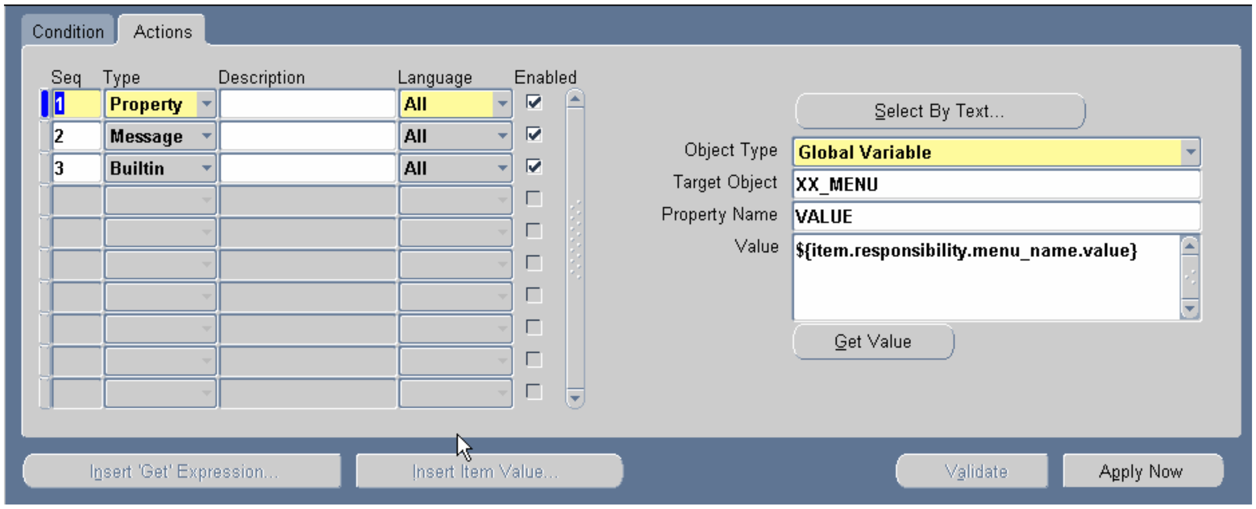
Trigger Event: MENU1 (You can enter additional event names.)
Trigger Object:
Condition:
Processing Mode: Not in Enter-Query Mode

Context

Level	Value
Site	

Fig 17: Rule 2 of the Zoom personalization

The second rule is created for when the user selects MENU1 from the Tools > Menu1 option. This will be labeled Zoom to Menu. The condition Trigger Event is indicating the user has selected the menu.


Condition | Actions

Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Message		All	✓
3	Builtin		All	✓

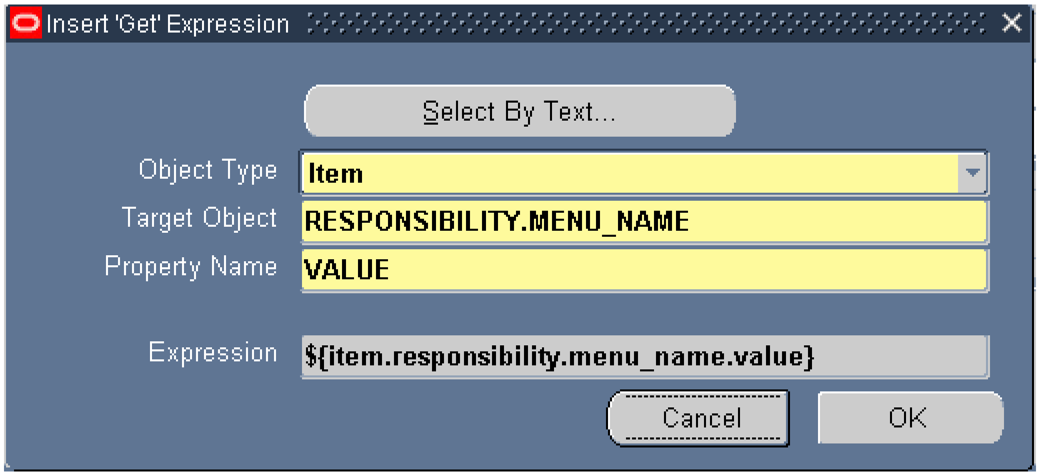
Object Type: Global Variable
Target Object: XX_MENU
Property Name: VALUE
Value: ${item.responsibility.menu_name.value}

Fig 18: Rule 2 Actions

First, we create a Global Variable to collect the Menu name for query. Choose Object Type of Global Variable. In the Target Object field, enter a variable name such as XX_MENU. In the Property Name field, choose VALUE.

Place your cursor in the Value Text Box. The "Insert 'Get' Expression…" button will be enabled. Press the button and use the Select By Text button to search for the Menu entry. You will see the Menu text name with the Object name to the right (RESPONSIBILITY.MENU_NAME). Select this item. In the

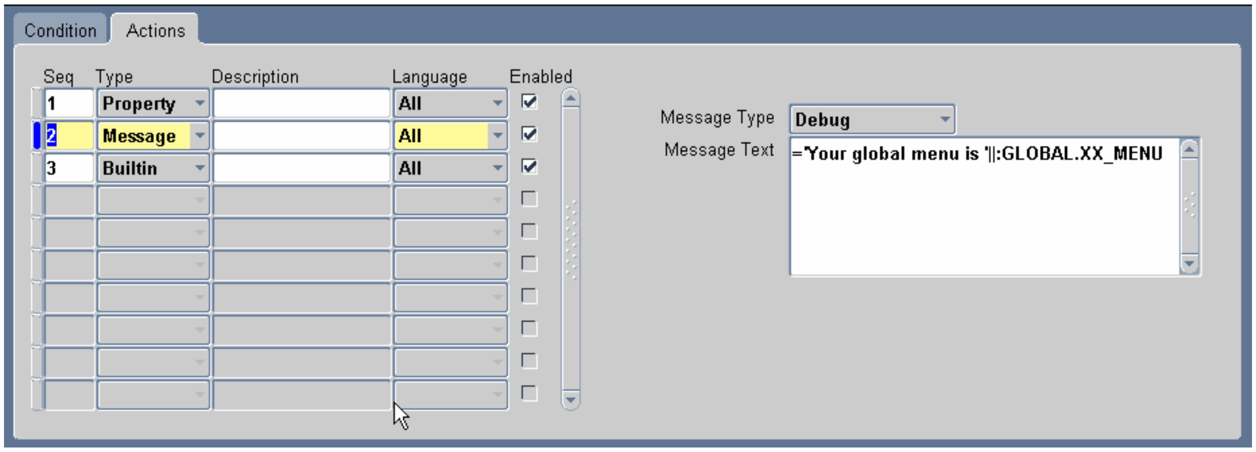
Property Name field, use the pull-down to select VALUE. The Expression should now read: ${item.responsibility.menu_name.value} Press the OK button to move this to the Value text box as shown above.



Object Type	Item
Target Object	RESPONSIBILITY.MENU_NAME
Property Name	VALUE
Expression	${item.responsibility.menu_name.value}

Fig 19: Insert 'Get' Expression screen

The above illustration depicts the correct values in the Insert 'Get' Expression window.

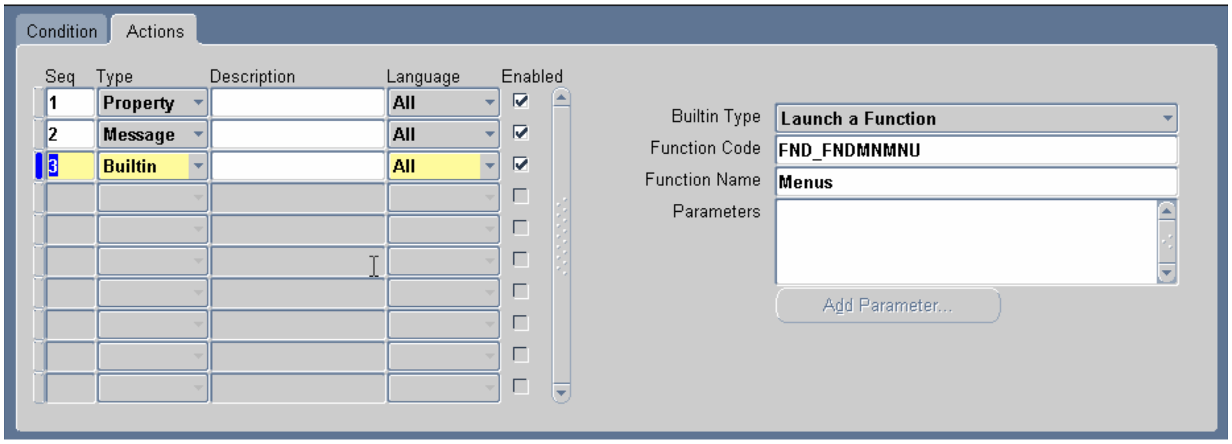


Seq	Type	Description	Language	Enabled
1	Property		All	☑
2	Message		All	☑
3	Builtin		All	☑

Message Type	Debug
Message Text	='Your global menu is '||:GLOBAL.XX_MENU

Fig 20: Action sequence 2

Create a debug message to show the value of the Global Variable for debugging purposes. The Rule should be set to Show Debug Messages as mentioned above. Once testing is complete, this can be disabled. In the first step creating the rule, the illustration shows the Show Debug Messages enabled for this Form Personalization.



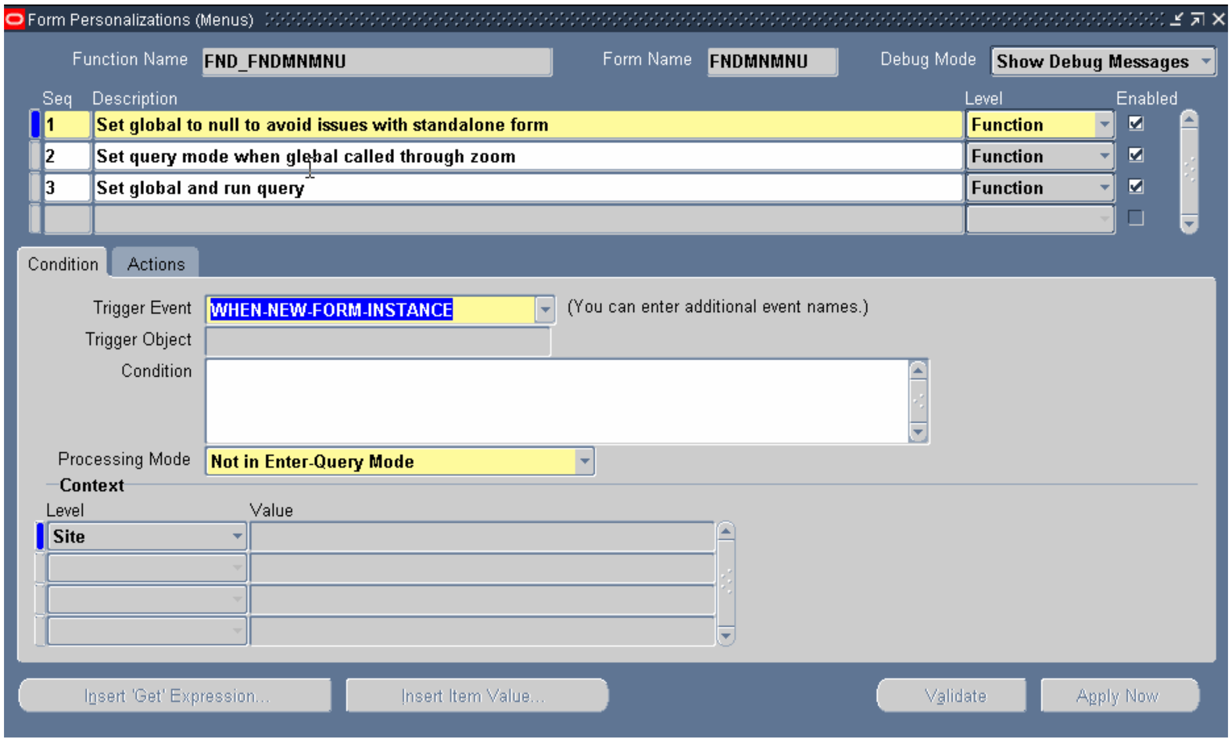
Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Message		All	✓
3	Builtin		All	✓

Builtin Type	Launch a Function
Function Code	FND_FNDMNMNU
Function Name	Menus
Parameters	

Fig 21: *Action Sequence 3 – using the Builtin to launch a form function*

Use the Builtin to launch the Form. Although there were no parameters listed in the Add Parameter button, some forms can use the QUERY_ONLY=YES parameter. By using Forms Developer 6i, a developer can open the Menu form and see if there are other parameters to pass to try and get the form to query upon opening. This is a different approach and would require testing. It is not covered in this white paper.

Next, the form being called for the Zoom needs to include personalizations to know what to do with the Global Variable and to query the parameter passed over. In this case, it is the Menu form.



Function Name FND_FNDMNMNU
Form Name FNDMNMNU
Debug Mode Show Debug Messages

Seq	Description	Level	Enabled
1	Set global to null to avoid issues with standalone form	Function	✓
2	Set query mode when global called through zoom	Function	✓
3	Set global and run query	Function	✓

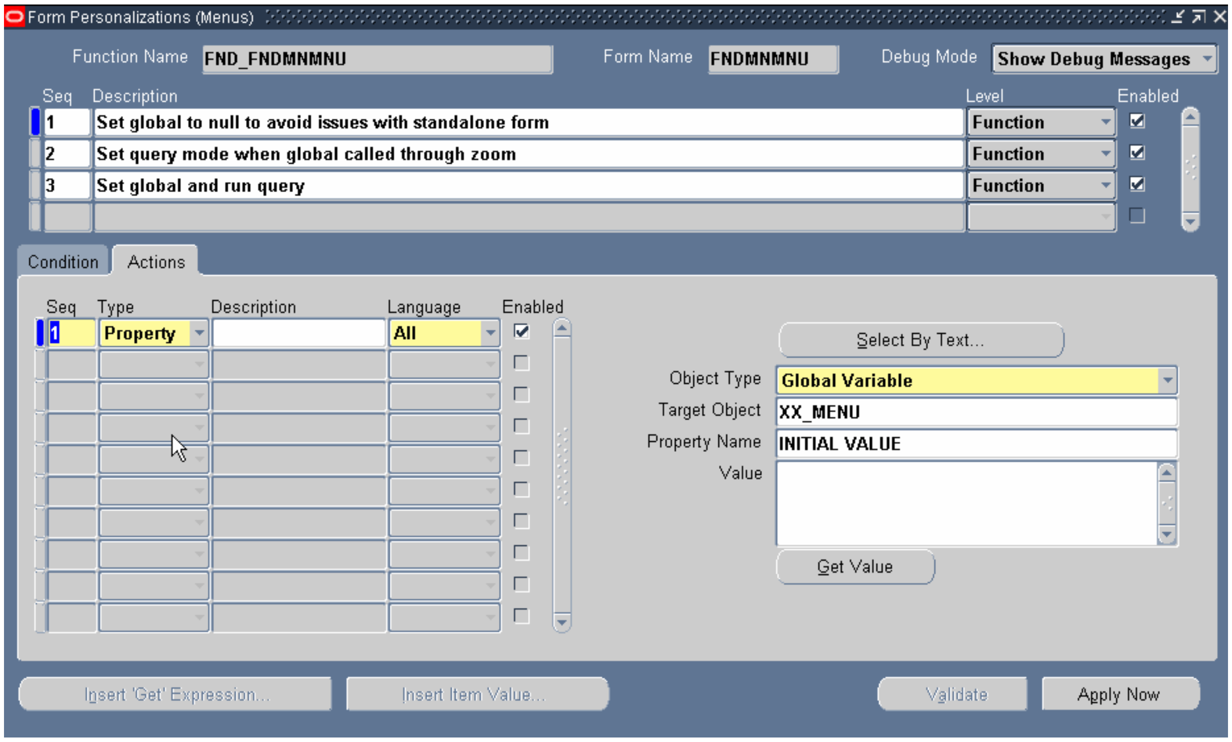
Trigger Event WHEN-NEW-FORM-INSTANCE (You can enter additional event names.)
Trigger Object
Condition
Processing Mode Not in Enter-Query Mode

Context

Level	Value
Site	

Fig 22: Open the Menu form and navigate to Help > Diagnostics > Custom Code > Personalize

There are three personalization rules needed to complete this process. Rule 1 will have 1 action. In this screen shot, we are creating a WHEN-NEW-FORM-INSTANCE rule to set the global variable to null. This will allow the form to function correctly when called directly from the Sysadmin menu (i.e. standalone mode). Because we have to use other WHEN-NEW-FORM-INSTANCE rules, it is necessary to create this null global variable.



Function Name: FND_FNDMNMNU
Form Name: FNDMNMNU
Debug Mode: Show Debug Messages

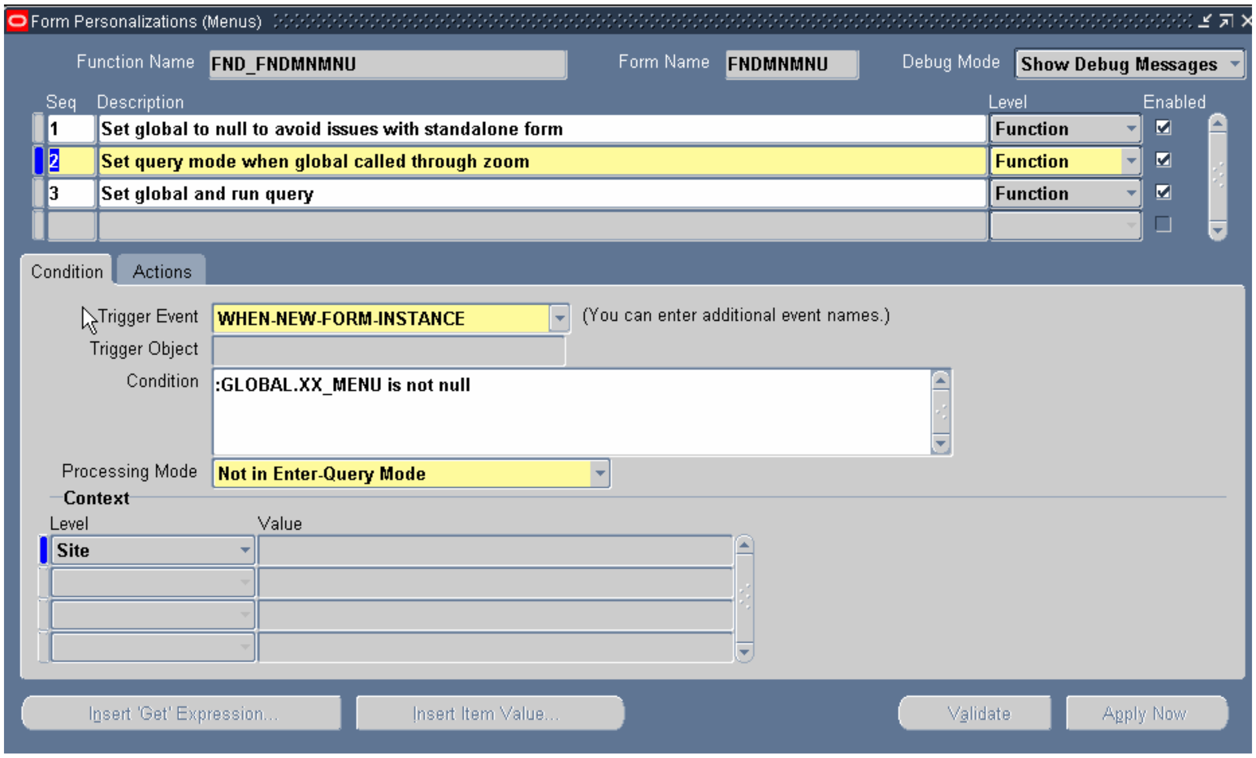
Seq	Description	Level	Enabled
1	Set global to null to avoid issues with standalone form	Function	✓
2	Set query mode when global called through zoom	Function	✓
3	Set global and run query	Function	✓

Seq	Type	Description	Language	Enabled
1	Property		All	✓

Object Type: Global Variable
Target Object: XX_MENU
Property Name: INITIAL VALUE
Value:

Fig 23: Create and initialize a global variable for stand alone operation

Because rules 2 and 3 will work with our Zoom and the Global Variable XX_MENU, Rule 1 should initialize the same Global Variable so the form will work in stand alone mode.



Function Name FND_FNDMNMNU   Form Name FNDMNMNU   Debug Mode Show Debug Messages

Seq	Description	Level	Enabled
1	Set global to null to avoid issues with standalone form	Function	✓
2	Set query mode when global called through zoom	Function	✓
3	Set global and run query	Function	✓

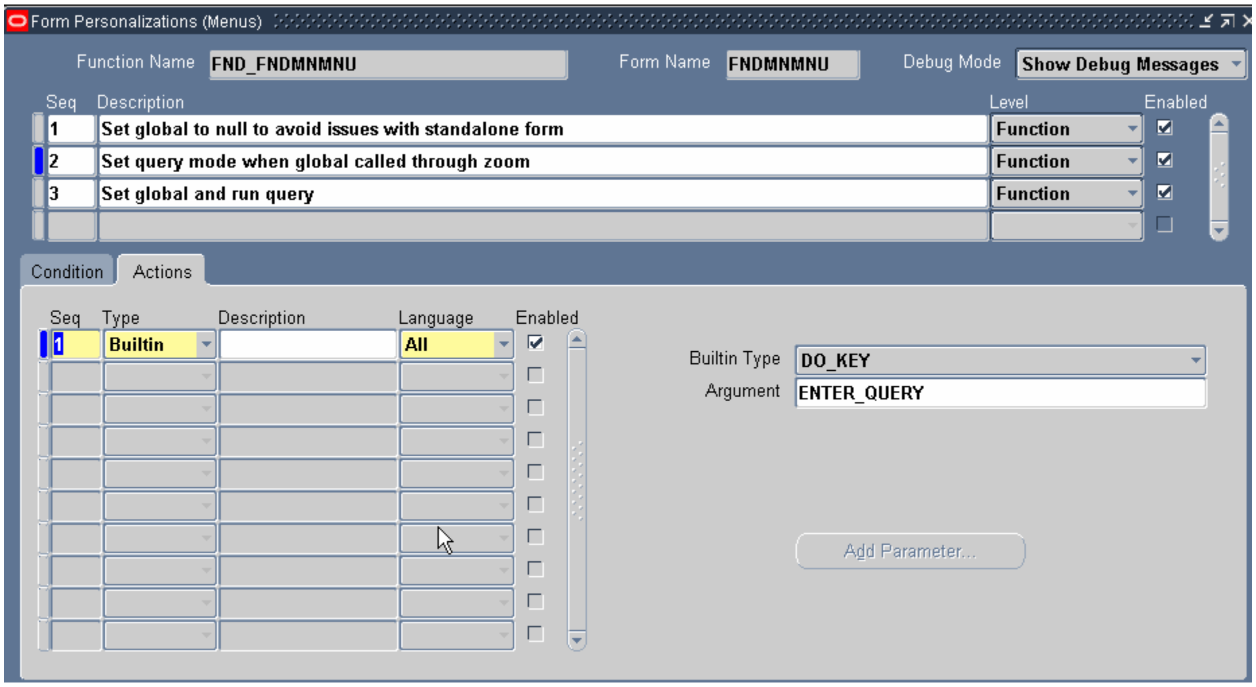
Trigger Event: WHEN-NEW-FORM-INSTANCE (You can enter additional event names.)
Trigger Object:
Condition: :GLOBAL.XX_MENU is not null
Processing Mode: Not in Enter-Query Mode

Context

Level	Value
Site	

Fig 24: Starting the query

Rule 2 sets the Menu form to Query mode when the condition is true; global variable is not null.



Function Name FND_FNDMNMNU   Form Name FNDMNMNU   Debug Mode Show Debug Messages

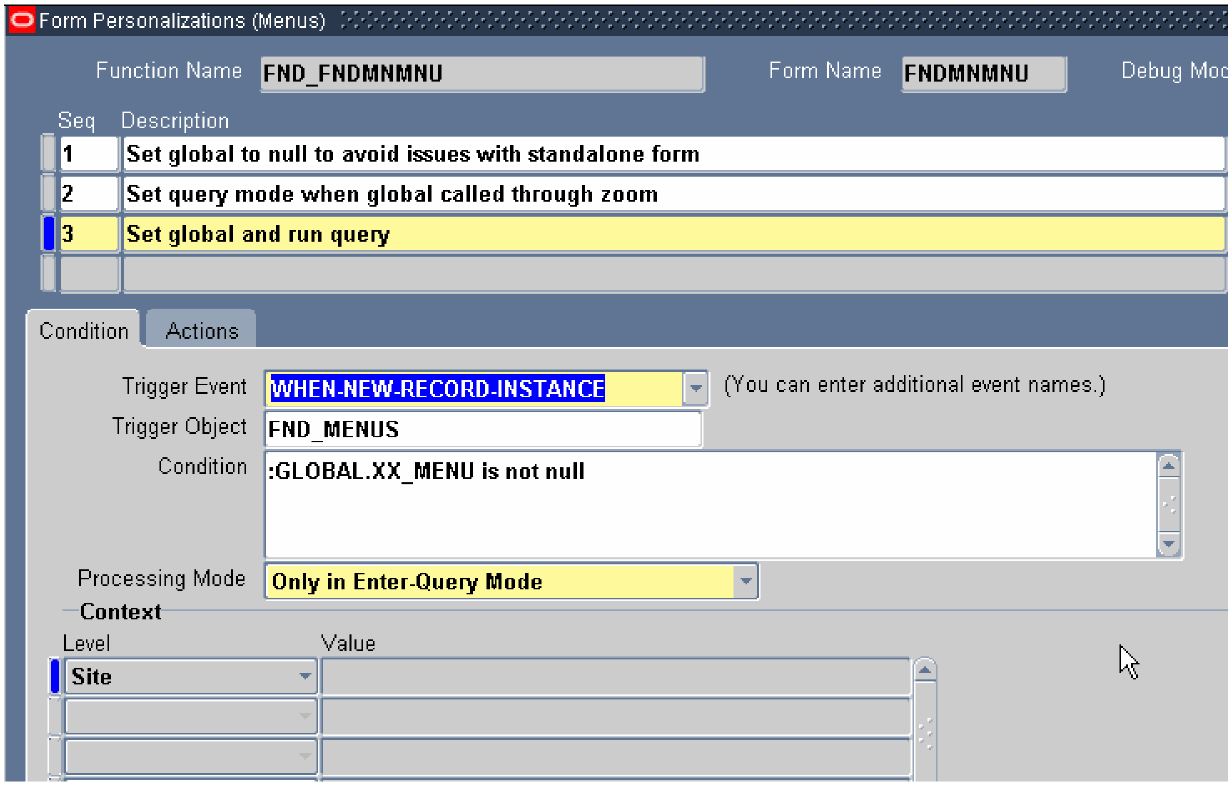
Seq	Description	Level	Enabled
1	Set global to null to avoid issues with standalone form	Function	✓
2	Set query mode when global called through zoom	Function	✓
3	Set global and run query	Function	✓

Seq	Type	Description	Language	Enabled
1	Builtin		All	✓

Builtin Type: DO_KEY
Argument: ENTER_QUERY

Fig 25: Action to call the DO_KEY query

Use a Builtin Action of DO_KEY to set the query mode. Notice this is rule is on the WHEN-NEW-FORM-INSTANCE. Once you use a personalization to set the form to query mode, all other actions have to be in a different rule. The ENTER_QUERY mode disables any other action sequences on this rule.


Form Personalizations (Menus)
Condition
Actions
(You can enter additional event names.)
Context
Debug Mod

Function Name: FND_FNDMNMNU
Form Name: FNDMNMNU

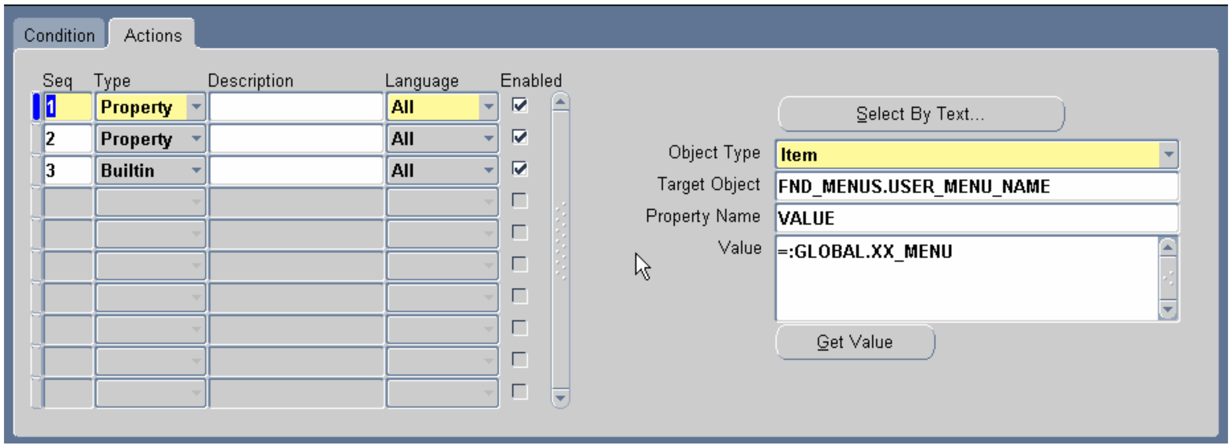
Seq	Description
1	Set global to null to avoid issues with standalone form
2	Set query mode when global called through zoom
3	Set global and run query

Trigger Event: WHEN-NEW-RECORD-INSTANCE
Trigger Object: FND_MENUS
Condition: :GLOBAL.XX_MENU is not null
Processing Mode: Only in Enter-Query Mode

Level	Value
Site	

Fig 26: Rule 3 condition for Query Mode

The third rule is set at the WHEN-NEW-RECORD-INSTANCE and the Processing Mode is Only in Enter-Query Mode. The condition still must evaluate to true; the global variable is not null.

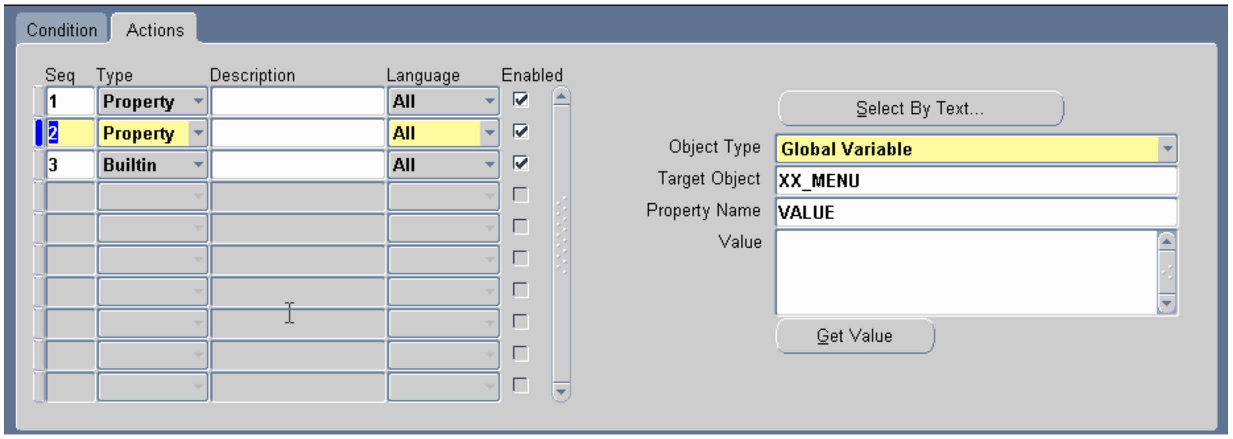


Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Property		All	✓
3	Builtin		All	✓

Object Type: Item
Target Object: FND_MENUS.USER_MENU_NAME
Property Name: VALUE
Value: =:GLOBAL.XX_MENU

Fig 27: Action 1 for Rule 3 – setting the menu value for the query

The first action is to set the USER_MENU_NAME value. Set it to the Global Variable. This should be the same menu name displayed in our Debug message when the user started the Zoom process.

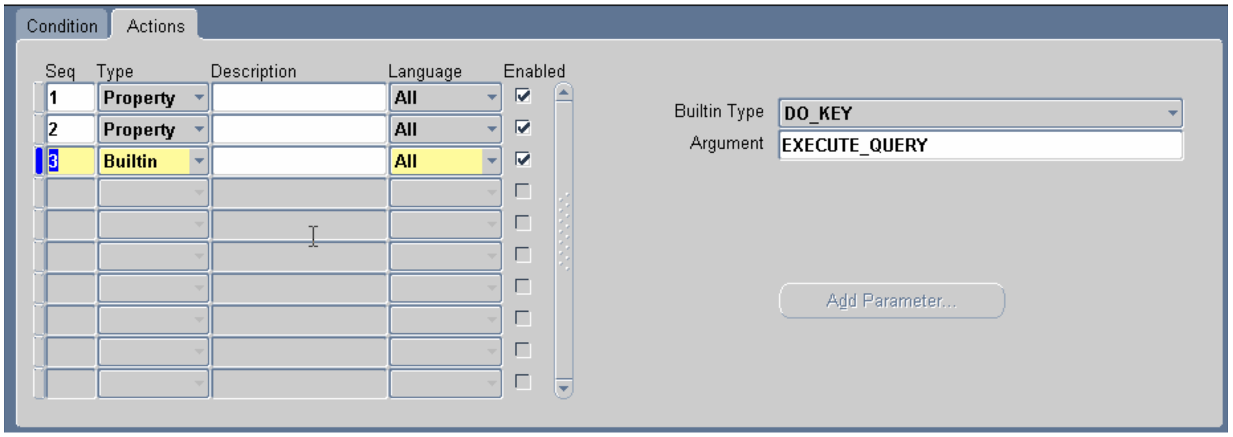


Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Property		All	✓
3	Builtin		All	✓

Object Type	Global Variable
Target Object	XX_MENU
Property Name	VALUE
Value	

Fig 28: Action 2 for Rule 3 – set global variable to null

For Action 2, set the Global Variable to null to avoid any conflicts with your forms sessions. After we set the Menu Name value in action 1, we remove the value in the Global Variable in action 2.



Seq	Type	Description	Language	Enabled
1	Property		All	✓
2	Property		All	✓
3	Builtin		All	✓

Builtin Type	DO_KEY
Argument	EXECUTE_QUERY

Fig 29: Action 3 for Rule 3 – Run the Query

Action 3 uses the DO_KEY Builtin to EXECUTE_QUERY. This will finish the Zoom call.

Before moving the Personalization to Production, thorough testing should be done with different responsibilities, menus and in standalone mode.

References and Patches

- Note:279034.1 Information About the Oracle Applications Form Personalization Feature in 11i
- Note:420518.1 Limitations of Forms Personalization
- Note:429604.1 How to Use Parameters in Forms Personalizations?
- Note:342501.1 When-Validate-Record Trigger firing multiple times
- Note:421999.1 How To Insert Or Update A Database Column Using Forms Personalization?

Form Personalizations can be applied on earlier 11i instances as long as they have ATG_PF.H Patch 3438354 (ref. Note 284086.1) and the latest 11.5.10 ATG RUP patch (ref. Note 296154.1) on top of ATG.H. Note: ATG.H includes FND.H

Conclusion

Form Personalization expands the Oracle Applications to new control standards. With practice, development time and thorough testing, Form Personalization can greatly increase your users experience while maintaining a strict control on validation and navigation.

Other suggested ideas:

- Dynamically change the WHERE clause in View Only forms
- Dynamically change the record group in a Form LOV
- Track changes by storing data in a custom table for Auditing purposes
- With a combination of Default Folders, control navigation on a Form to enforce company procedures and policies.

About the Author:

Martin Sugg has over 15 years experience with Oracle Applications. His roles have included implementation, application support, technical developer and team lead. Throughout his years at Oracle and in the field of consulting, he has worked with many of the toolsets such as Forms, Reports, OA Framework, Portal, Discoverer and the custom library. With his prior accounting experience, he is able to actively participate in end user and technical discussions, offering advice and recommendations. Martin also enjoys developing and leading training sessions for technical team members as well as functional users. His previous presentations have included Discoverer and Portal presentations for the AZ OAUG, and Applied Technology workshops for Oracle customers focusing on Oracle Approvals Manager, Workflow, Portal integration and the PL/SQL Web Toolkit.

Martin's client list includes Guthy Renker, Mesa Airlines, Rockford Corporation, City of Chandler, Factual Data, ZCoil Shoes and Times Microwave.

Martin Sugg is currently a remote consultant for Abaris, inc as well as Senior Business / Systems Analyst for Ports America Group. He lives in Phoenix AZ and is an active musician.