



ILP PROGRAM - ORACLE APPLICATIONS

Tata Consultancy Services

Oracle Forms Study Guide – Day3

Author: [MD. Nazmul Hoda \(TCS\)](#)
Creation Date: Jan 15, 2015
Last Updated: Jan 15, 2015
Document Ref: [ILP/ORACLEAPPS/FORMS/01](#)
Version: DRAFT 1A

Approvals:

<Approver 1>





Santanu Sarkar (TCS)



<Approver 2>

Shubho Chakraborty(TCS)

Contents

Table of Contents

Document Control	4
How to use this manual	5
1. Object Moving, Resizing, Alignment, Grouping continued.....	6
1.1 Associating text with an item prompt.....	6
1.2 Modifying the Appearance of a Text Item	7
1.3 Windows and Canvases	9
1.4 Forms Compiling, Running and File types	12
 Video6: Script: Components of a layout wizard	12
2. Data Block Customization	14
7.1 Creating a List of Values	14
7.2 Defining Check Box.....	19
7.3 Text Item	26
7.4 Radio Button and Radio Groups	27
 Video7: Script: Customizing data block with available features	28
8 Triggers	29
Interface Event Triggers:.....	30
Master/Detail Triggers:	30
Message-Handling Triggers:.....	31
Validation Triggers:	31
Navigational Triggers:.....	31
Transactional Triggers:.....	32
Query-Time Triggers:	32
 Video8: Script: Triggers	33
9. Message and Alerts.....	34
 Video9: Script: Message and Alert	36

10. Menu.....	37
 Video10: Script: Message and Alert	43
11. Custom Forms in E-Business Suite	44
11.1 Preparing the Desktop for Custom Forms Development	45
11.2 Steps for Developing Custom Forms in E-Business Suite.....	45
 Video11: Script: Making custom forms for Oracle Applications.....	46
APPENDIX A: Further Reading.....	47

Document Control

Change Record

Date	Author	Version	Change Reference
15-Jan-15	MD. Nazmul Hoda	Draft 1a	No Previous Document

Reviewers

Name	Position

Distribution

Copy No.	Name	Location
1	Library Master	Project Library
2		Project Manager
3		
4		

Note to Holders:

If you receive an electronic copy of this document and print it out, please write your name on the equivalent of the cover page, for document control purposes.

If you receive a hard copy of this document, please write your name on the front cover, for document control purposes.

How to use this manual

This guide is continuation of “Oracle Forms Study Guide – Day2”. Continue reading it in the same way as Day1 guide. This is the last document of the series “Oracle Forms Study Guide”










In this section you will continue learning how to fix the problems with the layout, which we left in the Day12manual. You may prefer to go to the Day1 manual and recap where you left it.

1. Object Moving, Resizing, Alignment, Grouping continued..

Object Moving, Resizing, Alignment, Grouping

Originally this canvas tool is very similar to MS-Word pattern. Even non SQL or Oracle person can create a Screen.

The canvas handling is very simple one.

- Resizing and Alignment tools are highlighted by the Horizontal Square RED Box.
- Font type - Select the object on Canvas and use either Font List or go to property plate as similar on previous page Size ,Block, Underline, Italian
- Use the property plate for the corresponding Object  and - are increase of decrease the size as a view purpose.
- Use the property plate for the corresponding Object , the bar symbols are Left, Right, To, Down and Center based alignments of objects (refer icons -      )
- Grouping tools inside the Vertical Square RED Box, have lot of beatification tools :
 - Icon Creation
 - Select box Creation
 - Text creation
 - Select button Creation
 - Color setting
 - Advance level of graph creations
 - Moving On inside Canvas are
 - Block level Frames
 - Field Name
 - Fields as values or type

1.1 Associating text with an item prompt

The Forms Builder Layout Editor has a tool called Associate Prompt which enables you to create a prompt for an item using any boilerplate text displayed in the editor. To create a prompt-item association using the Associate Prompt tool, perform the following steps:

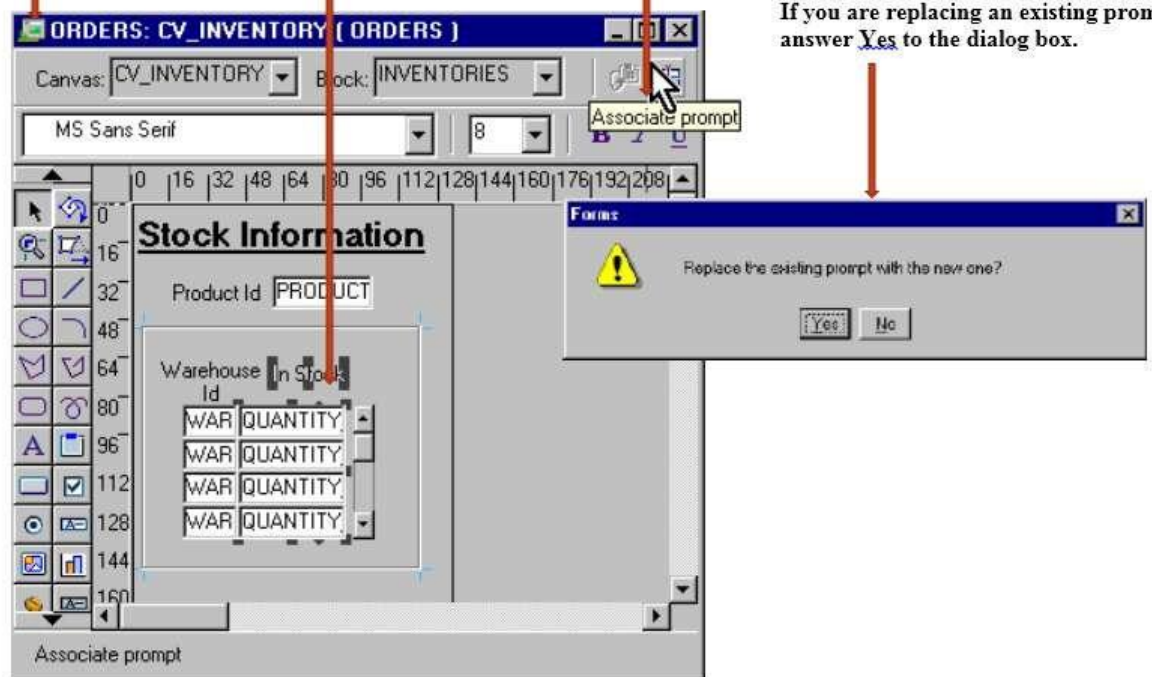
1. Open the Layout Editor window.
2. Select the item and boilerplate text you want as the item's prompt in the editor. ☐ Click the Associate Prompt tool.
3. If you are replacing an existing prompt, answer Yes to the dialog box.

Open the Layout Editor window.

Select the item and boilerplate text you want as the item's prompt in the editor.

Click the Associate Prompt tool.

If you are replacing an existing prompt, answer Yes to the dialog box.



TASK5: Create a boilerplate with Graphics and Video

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

1.2 Modifying the Appearance of a Text Item

Visual Attributes: You set the Visual Attribute Group for an item or its prompt to specify how the visual attributes are derived (select DEFAULT or a named Visual Attribute).

Color: The properties in this section specify the foreground color, background color, and fill pattern for the item. You select these from color and pattern pickers.

Font: The properties in this section determine the font used for the item, along with its size, weight, style, and spacing. You can double-click the Font group to display a Font dialog enabling you to set all these properties at

once, or you can click the individual properties to select each from an appropriate control, such as a pop-up list or LOV.

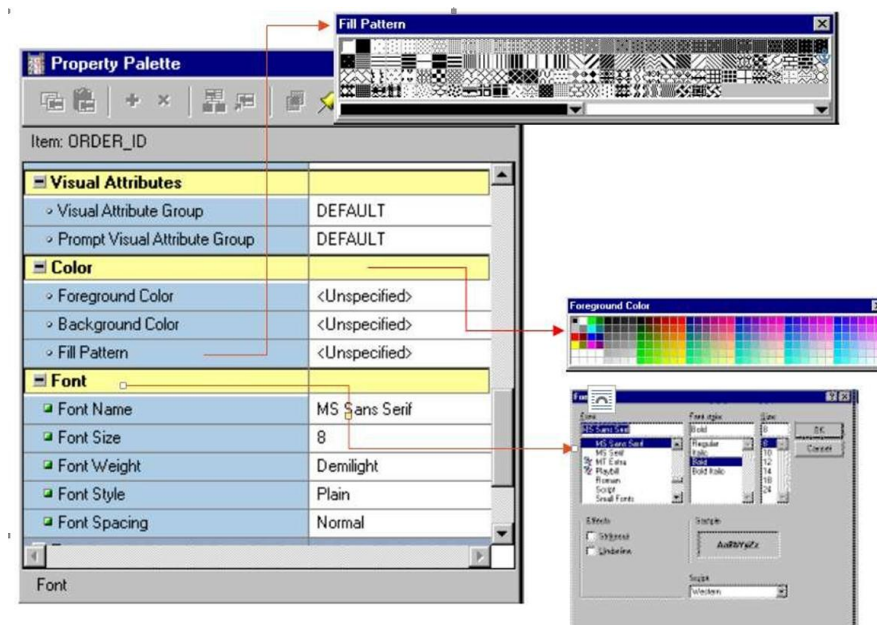


Note: When the form module does not contain any named visual attribute objects, the pop-up list for the Visual Attribute Group property shows only Default or Unspecified. An item that has the Visual Attribute Group property set to default, or that has individual attribute settings left unspecified, inherits those settings from the canvas to which it is assigned.

Use the properties in data group to control the text item. Some of these properties are:

Data Type: Chose from Char, Date, Datetime, and Number

Data Length Semantics: This will be needed when your data contains multibyte character set (ex: Chinese language characters).



The screenshot shows the 'Data' tab for the item 'CUST_FIRST_NAME'. The properties are as follows:

Property	Value
Data Type	Char
Data Length Semantics	CHAR
Maximum Length	5
Initial Value	
Required	Yes
Format Mask	
Lower Allowed Value	
Highest Allowed Value	
Copy Value from Item	
Synchronize with Item	<Null>

Below the properties, a visual representation shows how the same 5-character value is stored in different character sets:

- US7ASCII** (VARCHAR2 (5 CHAR)): 5 single-byte slots.
- JA16SJIS** (VARCHAR2 (5 CHAR)): 10 double-byte slots.
- UTF8** (VARCHAR2 (5 CHAR)): 3 single-byte slots followed by an ellipsis, indicating variable-length storage.

Maximum Length: Specifies the maximum length of the data value that can be stored in an item. If the Maximum Length exceeds the display width of the item, Forms automatically enables the end user to scroll horizontally.



Note: In the example on the slide, whether the form operator is using a single-, double-, or variable-byte character set, the right amount of storage is allocated. To hold the same value if the Data Length Semantics had been set to BYTE, the Maximum Length would have needed to be 5 for single-byte, 10 for double-byte, and an unknown value for a variable-byte character set.



TASK6: Create a fancy text item

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

1.3 Windows and Canvases

With Forms Builder you can display an application in multiple windows by using its display objects—windows and canvases.

What Is a Window?

A window is a container for all visual objects that make up a Forms application. It is similar to an empty picture frame. The window manager provides the controls for the window that enable such functionality as scrolling, moving, and resizing. You can minimize a window. A single form may include several windows.

What Is a Canvas?

Canvas is a surface inside a window container on which you place visual objects such as interface items and graphics. It is similar to the canvas upon which a picture is painted. To see a canvas and its contents at run time, you must display it in a window. A canvas always displays in the window to which it is assigned.



Note: Each item in a form must refer to no more than one canvas. An item displays on the canvas to which it is assigned, through its Canvas property. Recall that if the Canvas property for an item is left unspecified, that item is said to be a Null-canvas item and will not display at run time

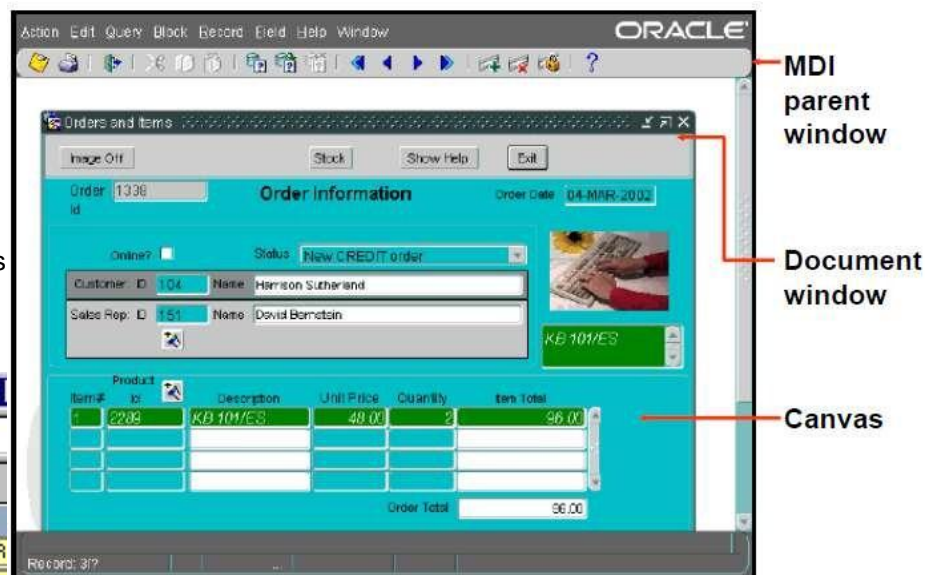
- **Window** Container for Forms Builder visual objects
- **Canvas** Surface on which you “paint” visual objects
- To see a canvas and its objects, display the canvas in a window.

Property Palette

Canvas: CV_CUSTOMER

General	
Name	CV_CUSTOMER
Canvas Type	Content
Subclass Information Comments Help Book Topic	
Functional	
Raise on Entry	No
Popup Menu	<Null>
Physical	
Visible	Yes
Window	WIN_CUSTOMER
Viewport X Position on Canvas	0
Viewport Y Position on Canvas	0
Width	462
Height	204

Window



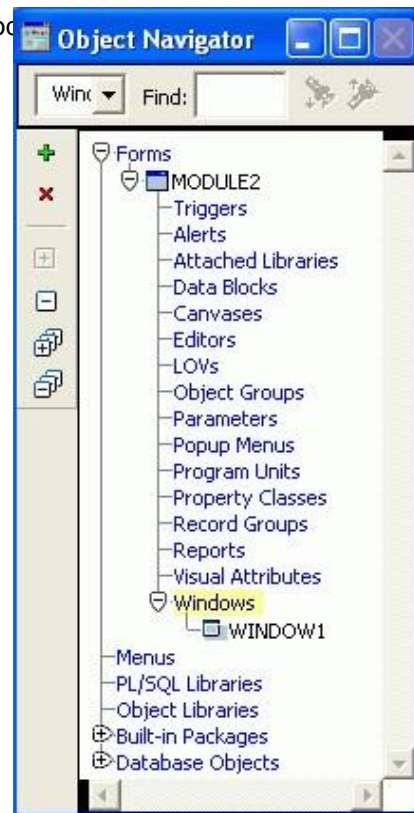
The Relationship between Windows and Content Canvases



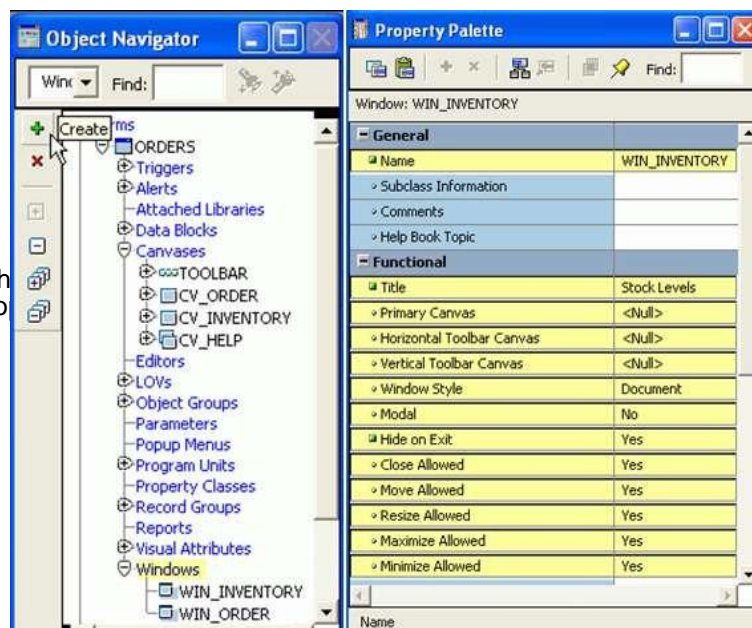
- WINDOW1 is created by default with each new form module.
- It is modeless.
- You can delete, rename, or change its attributes.

Displaying a Form Module in Multiple Windows

- Use additional windows to:
 - Display two or more content canvases at once
 - Switch between them without replacing the initial one
 - Modularize form contents
 - Take advantage of the window manager
- Two types of windows:
 - Modal
 - Modeless



Object Navigator: Click Create with Windows node selected and Properties PaletteSet properties



TASK7: Create a Multi Canvas Form

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

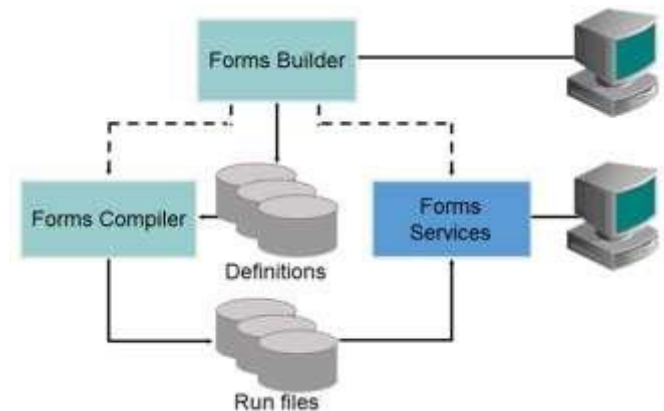
1.4 Forms Compiling, Running and File types

Because Forms applications are Web based, it is not possible to run them directly from the command line. Instead, they are invoked by entering a URL, directed to Forms Services, in a browser. The files used at run time must already have been compiled by the Forms Compiler component. These files must reside on the middle-tier machine in a directory accessible to the Forms Runtime Engine (in FORMS_PATH).

To test your applications, you can also access Forms Services directly from Forms Builder by setting certain preferences, as described later in this lesson.

Once the compilation is complete and there is no error, then **.FMB** file created. This is development file or Source code file. At the same time **.FMX** also create. This is only for execute.

This file is only enough for Apps Server to implement the customization

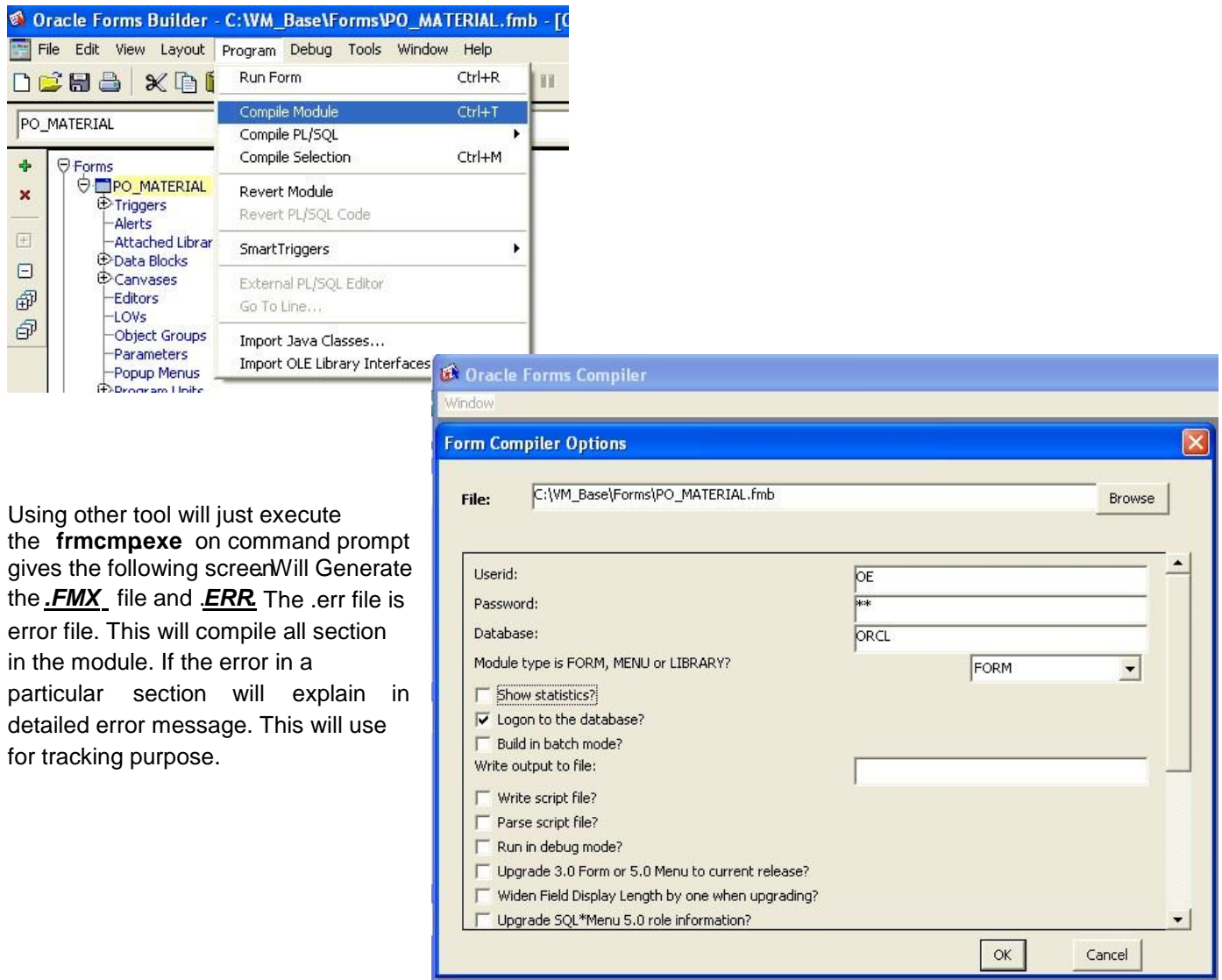


One Way of Compilation is Compile module is just compile the entire Form/Module. Another way is Run Form will compile the entire Form/Module and start execute the same.



Video6: Script: Components of a layout wizard

Screenplay explaining different components of layout editor, how to create a canvas, Explain what is a PL/sql editor, How to resize a, align and group objects on a canvas, handling text items, difference between canvas and window, How to compile and run a form.



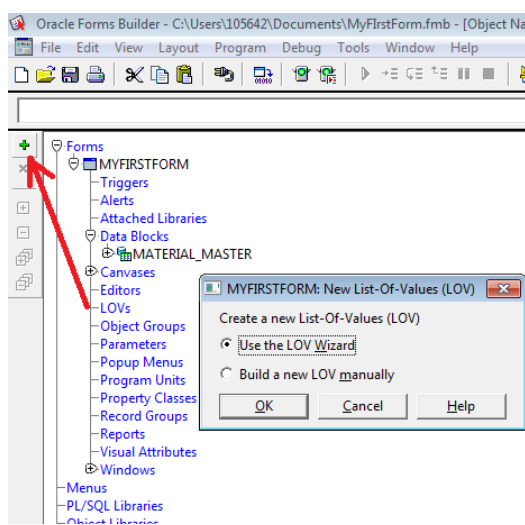
Using other tool will just execute the **frmcmpexe** on command prompt gives the following screen. Will Generate the **.FMX** file and **.ERR**. The .err file is error file. This will compile all section in the module. If the error in a particular section will explain in detailed error message. This will use for tracking purpose.

2. Data Block Customization

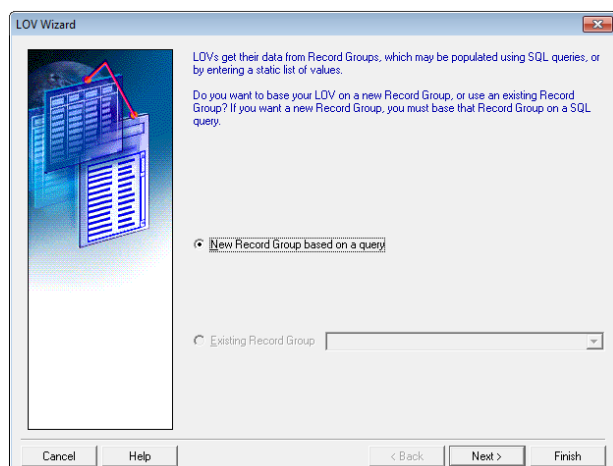
7.1 Creating a List of Values

Often it makes more sense to allow the users to select a field value from a list of predefined values. We will see how to create an LOV and associate it with a field. We will use the same example of MyFirstForm.fmb discussed earlier.

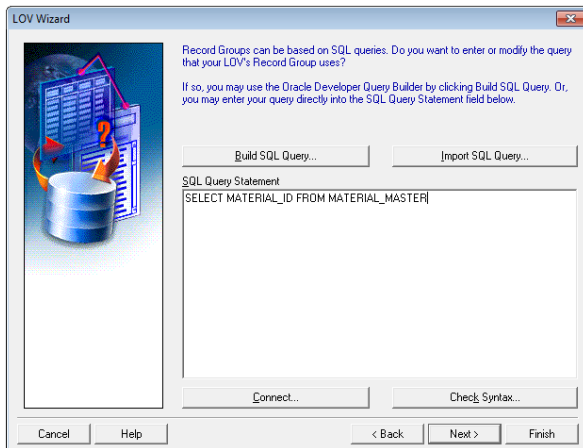
Step 1: On the object Navigator, click on the LOVs and then click the (+) icon, select the LOV



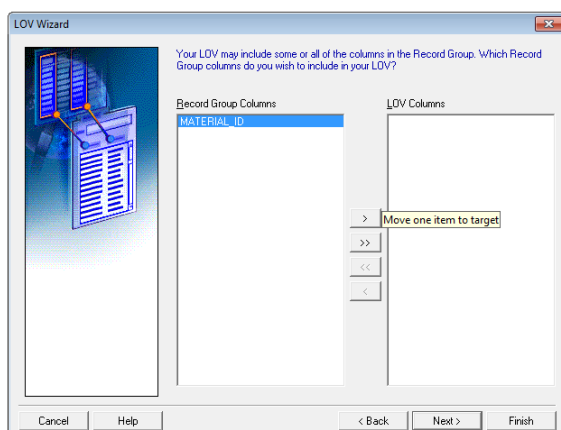
Step 2: Select the New Record Group, Click Next. A record group is container to hold records.



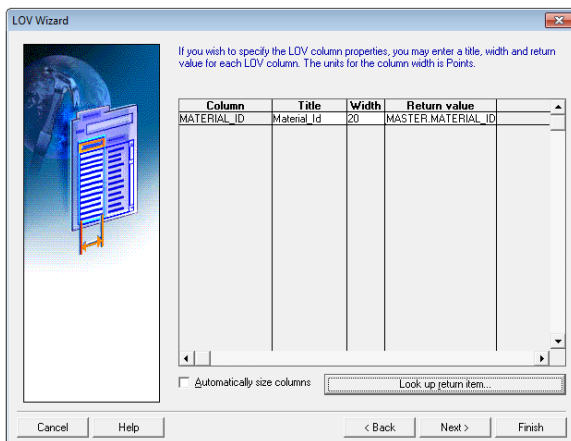
Step 3: Type the sql query to populate the record group.



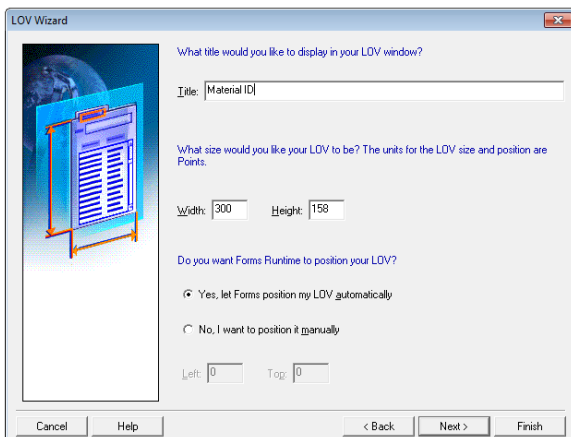
Step 4: A record group may select multiple columns from the table. Out of these, we can assign one column to the LOV. Here, since we have selected only one column in the query, we will select the MATERIAL_ID column,



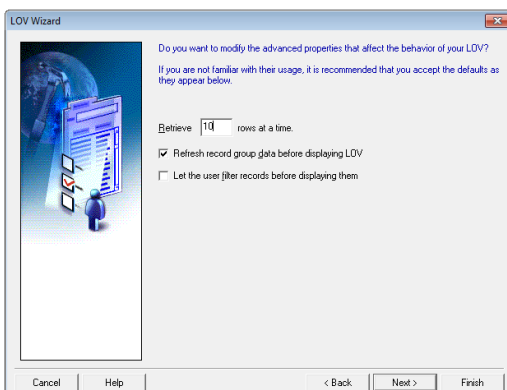
Step 5: Edit the field size if needed, and select the return value. This return value is a parameter or global variable name to which the column value should be assigned. The default value is NULL. This is an optional item, where the value selected from the LOV will go. You can click the 'Look up return item'



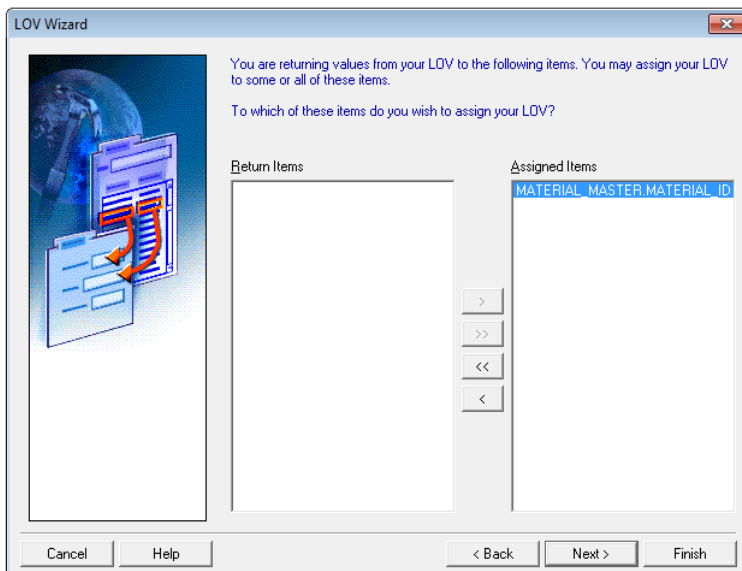
Step 6: Provide a title and optionally change the attributes of the LOV as shown below:



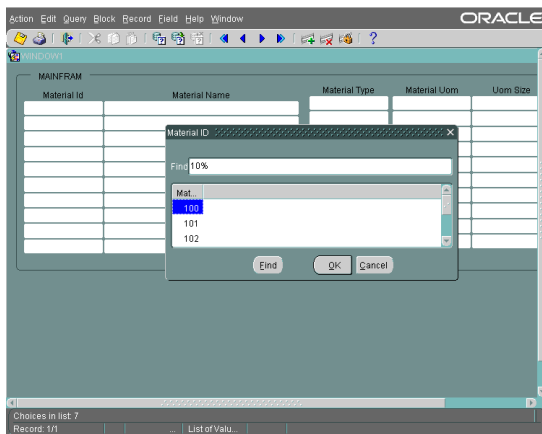
Step 7: Optionally change the number of records to fetch from the database at a time:



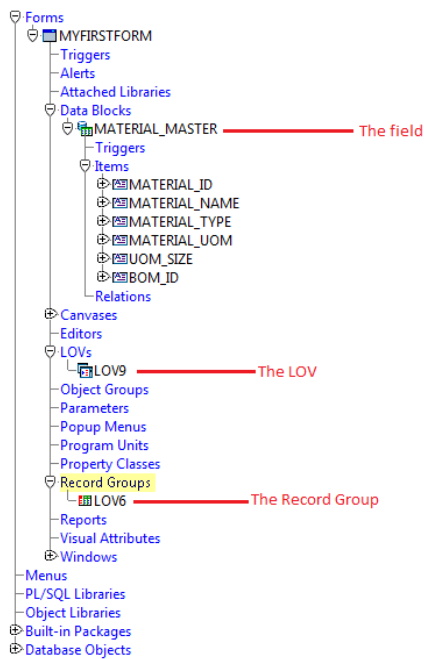
Step 8: Select the target field on the form, where the value selected from LOV should go. We have selected the MATERIAL_MASTER.MATERIAL_ID (Block. Field) here. This means the LOV gets attached to this form field. So, at runtime, if you press Ctrl+L, when your cursor is on this field, the LOV will pop up and upon selecting a value from the LOV, this field will be populated with the selected value.



Now run the form, go to the field – Material ID, Press Ctrl+L, select a value, press OK, and see, if the field gets the value.



Now let us go to the Object navigator, and see, what changes has been done:



Note the following:

- One Record Group Named LOV6 has been created.
- One LOV named LOV9 has been created

Let us see what is inside them, and how does the field know, which LOV to invoke, when user presses Ctrl+L

1 The field

2

3

The LOV

The Record Group

Item: MATERIAL_ID	
Lock Record	No
List of Values (LOV)	
List of Values	LOV6
List X Position	1

LOV: LOV6	
General	
Name	LOV6
Subclass Information	
Comments	
Functional	
Title	LOV6
List Type	Record Group
Record Group	LOV6
Column Mapping Properties	

Record Group: LOV6	
General	
Name	LOV6
Subclass Information	
Comments	
Functional	
Record Group Type	Query
Record Group Query	SELECT MATERIAL_ID FROM MATERIAL_MASTER
Record Group Fetch Size	10
Column Specifications	

The field property shows LOV6 as the LOV name associated with this field. LOV6 again refers back to the Record group LOV6. This is self-explanatory in the diagram above.



Note: To enable the above binding you need to perform step5 and 8, even though they are not mandatory.



TASK8: Create a mailing label report as described below:

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

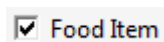
<Tasks are being Synched with the Master Case study, Details to be provided soon>

7.2 Defining Check Box



In this section you will learn how to create a check box

A check box is a field (or a set of fields) in a form, which can be checked and unchecked. It looks like the below:



When you click it, a tick mark will appear and disappear. This is a popular way to get user's selection. In this example, we will be using this field to capture if the items entered in the table is a food item or not. A food item will get some tax benefits.

For this purpose, we will add another field in the table:

```
SQL> alter table material_master add food_item char(1);
```

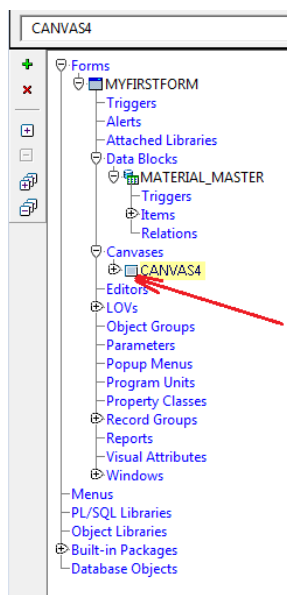
Table altered.

```
SQL> desc material_master
```

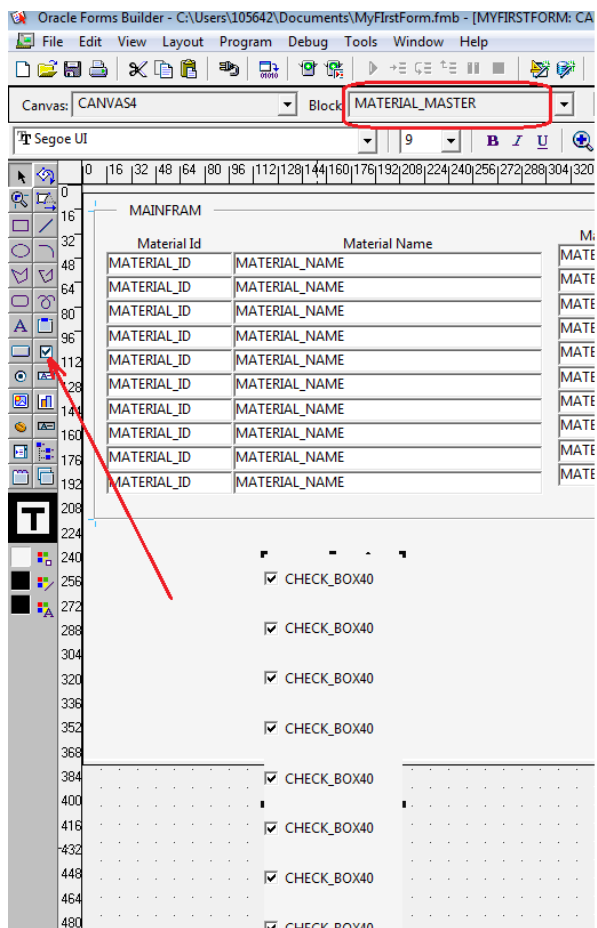
Name	Null?	Type
-----	-----	-----

MATERIAL_ID	NUMBER
MATERIAL_NAME	VARCHAR2 (200)
MATERIAL_TYPE	VARCHAR2 (20)
MATERIAL_UOM	VARCHAR2 (10)
UOM_SIZE	NUMBER
BOM_ID	NUMBER
FOOD_ITEM	CHAR (1)

Step1: Open the existing file – MyFirstForm.fmb and open the object Navigator and double click the canvas.



Step2: In the canvas, click the check box icon, then drag your mouse on the canvas and release.

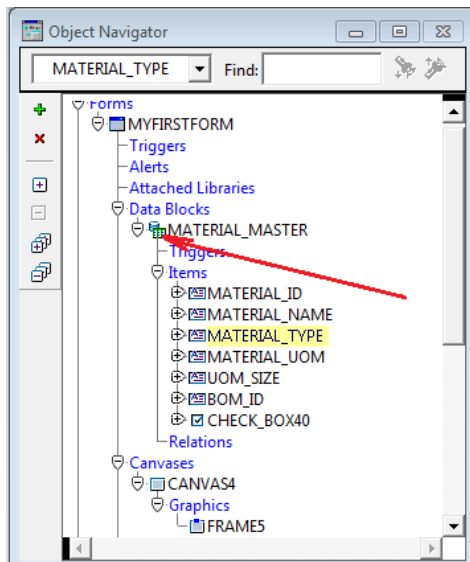


This has created checked boxes, but 10 of them. You cannot delete them individually, it will delete together.

We wanted to place a single check box, which will say, if all the items on this page are food items or not.

Let us see, why we are getting 10 records.

Look at the above diagram, and notice the Block Name as MATERIAL_MASTER. So the check box we are trying to create belongs to the block MATERIAL_MASTER. Let us see the property of this block, click on the block MATERIAL_MASTER in the object navigator, the property palette will open:



The Block - MATERIAL_MASTER

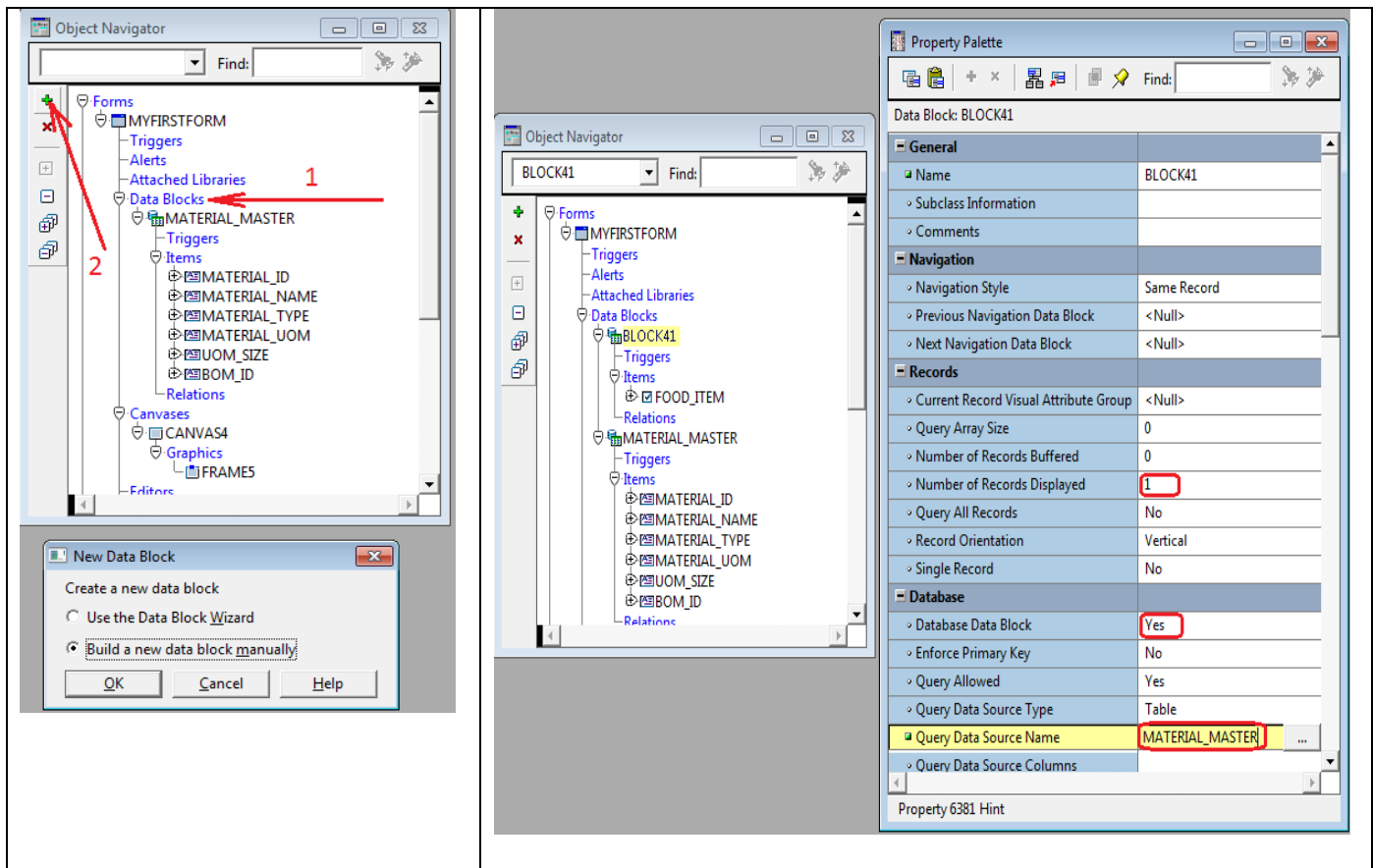
Data Block: MATERIAL_MASTER	
General	
Name	MATERIAL_MASTER
Subclass Information	
Comments	
Navigation	
Navigation Style	Same Record
Previous Navigation Data Block	<Null>
Next Navigation Data Block	<Null>
Records	
Current Record Visual Attribute Group	<Null>
Query Array Size	0
Number of Records Buffered	0
Number of Records Displayed	10
Query All Records	No
Record Orientation	Vertical
Single Record	No
Database	

MATERIAL_MASTER block property

Notice that the block is supposed to display 10 records. This explains why we are getting 10 check boxes.

So it is evident that we will need to create a new block which will have the Number of Records displayed as 1.

Step3: On the object Navigator, click the 'Data Block' and click +. A New data Block creation wizard will appear. Select the manual creation. It will create the new block and give it a new name (here BLOCK41). You can change the name.

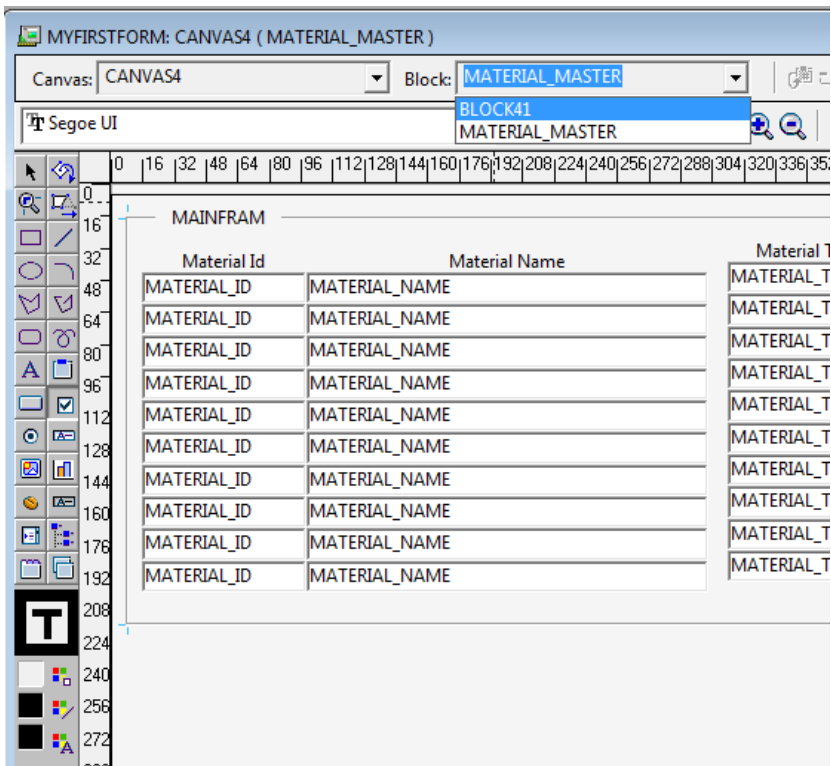


Make the Number of Records displayed as 1.

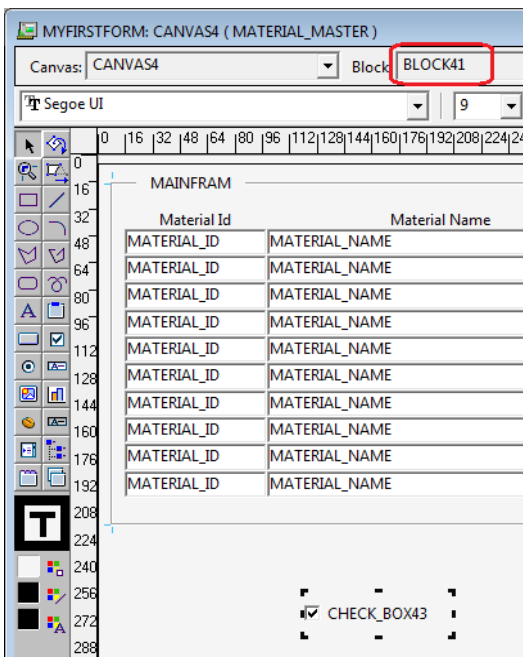
Notice the following:

- The 'Database Data Block' section is 'Yes'. This means whenever we will commit this block, the value will be written back to the database.
- The Field Query Data Source name: It will blank. But if you want to associate it with the table, type the name of the table.

Step4: Go to the canvas once again, and now select the new block we have created:



Now if you try to create a check box, you will get a single record:



Double click on the check box to change the properties as required.

**TASK9: Create a mailing label report as described below:**

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

7.3 Text Item

This type of item is used to display a text dynamically when a form runs. Note its difference from the Boilerplate text that the latter is frozen to the canvas, but the former can display different text depending on the situation.

It supports scrolling and/or querying. You can control many behaviors of a text item, the important ones are listed below:

1. Insert/Edit Allowed/Not allowed
2. Navigable Allowed/Not allowed
3. Mandatory/Optional
4. Calculation
5. Database Item (Y/N)
6. LOV
7. Editor

You can create a text item in the same way as we did while creating the check box, but by selecting the Text

Item dialog box - .



TASK10: Create a mailing label report as described below:

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

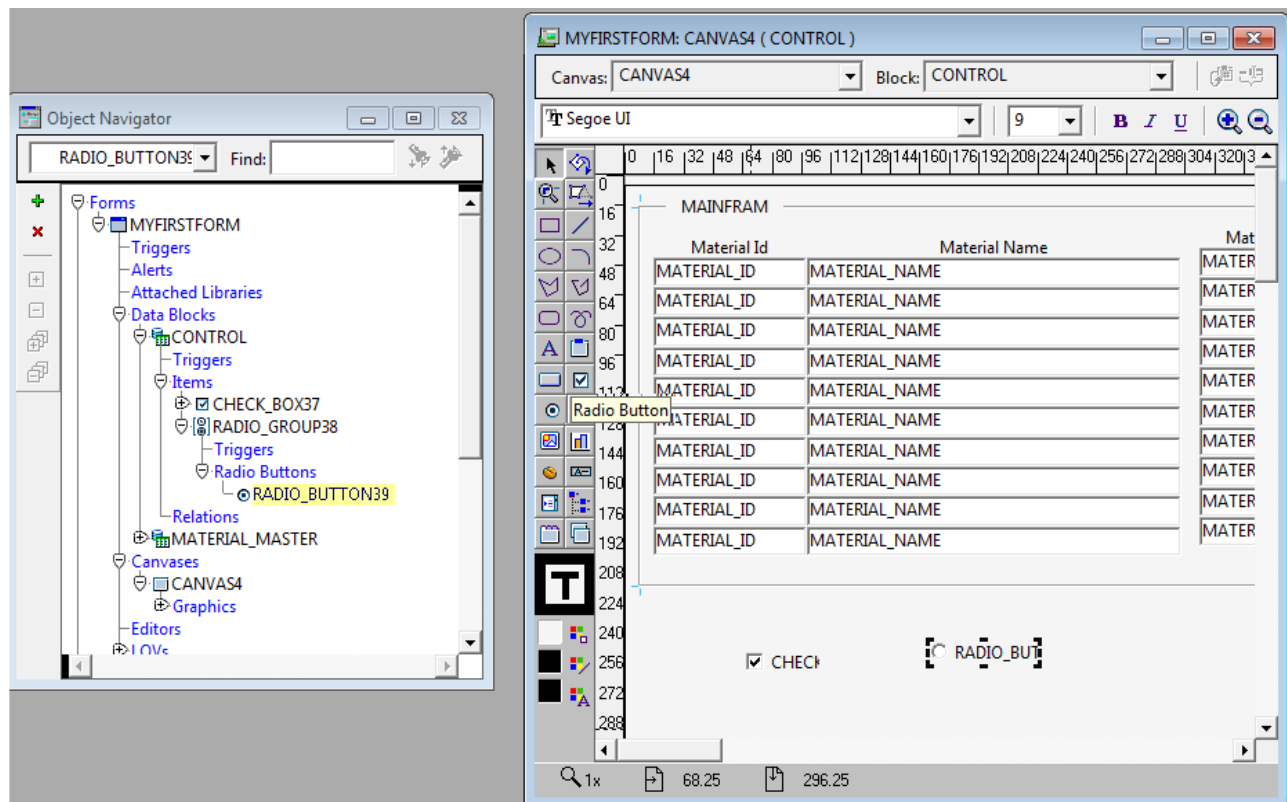
7.4 Radio Button and Radio Groups

Radio Buttons are the fields which accept either Yes or No value. This is useful when a user either has to supply a Yes/No type response. Radio buttons cannot be associated with a database table. You cannot write a trigger on a radio button.

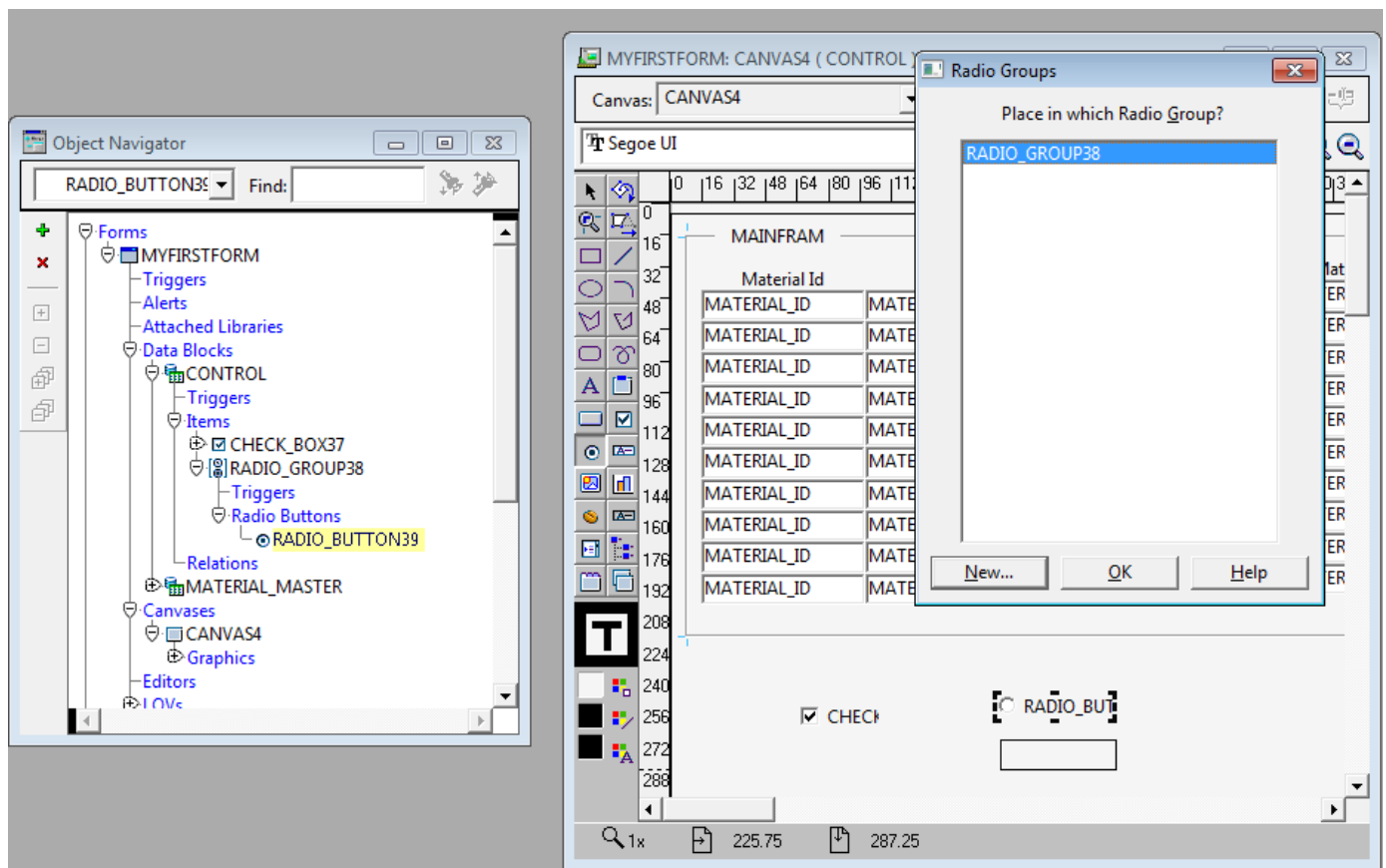
A set of radio buttons can be grouped together in a Radio Group. This is useful when the user response is single choice type (from a list of options). A radio group can be of Database type, meaning the database column can be updated depending on the choice of radio buttons a user makes. A radio group can have a trigger also.

Let us see how to create them:

The process is same as check box creation. Click on the radio button icon on the Canvas view, and drag it on the canvas. When you create the first radio button, a radio group is automatically created. In this case radio group named RADIO_GROUP38 has been created.



When you create the next radio button, a prompt will appear asking to select the radio group. You can select an existing radio group or create a new one.



TASK11: Create a mailing label report as described below:

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>



Video7: Script: Customizing data block with available features

Screenplay explaining How to create and attaching LOV to a field, creating a checkbox, Text Item, Radio Buttons and Groups.

8 Triggers

Triggers are blocks of PL/SQL code that are written to perform tasks when a specific event occurs within an application. For example, in the form we have developed (MyFirstForm.fmb) in previous sections, if we want that the moment a user enters a value in the Material ID field, and presses Tab, a piece of code should fire and check if he has not entered all zeroes. We have to write a trigger on the field MATERIAL_ID. Inside the trigger, we can write the code which will validate if he has entered all zeros.

Material Id	Material Name	Material Type	Material Uom	Uom Size	Bom Id

The code may be as simple as this:

```
IF :MATERIAL_MASTER.MATERIAL_ID = 0
THEN
    MESSAGE ('MATERIAL ID CANNOT BE ZERO');
    RAISE FORM_TRIGGER_FAILURE;
END IF;
```



Note: The field MATERIAL_ID is associated with the table column – MATERIAL_MASTER.MATERIAL_ID, which is of type number. So, oracle database will not allow to insert or update a non-numeric value in this column. However the database level validation will occur only, when the user commits the data into the database. This will not happen when the user jumps from one field to the other inside the form. Hence field level validations has to be controlled by triggers or in the definition of the field. Small validations can be controlled by using the field attributes like – Data Type, Maximum Length, Format Mask, and Lowest Allowed Value, and Highest Allowed Value, Required etc.

When validation becomes complex, we need to use trigger.

When you define a trigger, Forms developer will ask you what kind of trigger you want to create. In the above case, we will chose WHEN-VALIDATE-ITEM type.

Let us understand the types of triggers by classification:

Block Processing Triggers:

Block processing triggers fire in response to events related to record management in a block.

- **When>Create-Record** Perform an action whenever Oracle Forms attempts to create a new record in a block.
- **When-Clear-Block** Perform an action whenever Oracle Forms flushes the current block; that is, removes all records from the block.
- **When-Database-Record** Perform an action whenever Oracle Forms changes a record's status to Insert or Update, thus indicating that the record should be processed by the next COMMIT_FORM operation.

Interface Event Triggers:

Interface event triggers fire in response to events that occur in the form interface. Some of these triggers, such as When-Button-Pressed, fire only in response to operator input or manipulation. Others, like When-Window-Activated, can fire in response to both operator input and programmatic control.

- **When-Button-Pressed** Initiate an action when an operator selects a button, either with the mouse or through keyboard selection.
- **When-Checkbox-Changed** Initiate an action when the operator toggles the state of a check box, either with the mouse or through keyboard selection.
- **When-Image-Activated** Initiate an action whenever the operator double-clicks an image item.
- **When-Image-Pressed** Initiate an action whenever an operator clicks on an image item.
- **When-Radio-Changed** Initiate an action when an operator changes the current radio button selected in a radio group item.
- **When-Window-Activated** Initiate an action whenever an operator or the application activates a window.
- **When-Window-Closed** Initiate an action whenever an operator closes a window with the window manager's Close command.
- **When-Window-Deactivated** Initiate an action whenever a window is deactivated as a result of another window becoming the active window.

Master/Detail Triggers:

Oracle Forms generates master/detail triggers automatically when a master/detail relation is defined between blocks. The default master/detail triggers enforce coordination between records in a detail block and the master record in a master block. Unless developing custom block-coordination schemes, you do not need to define these triggers.

- **On-Check-Delete-Master** Fires when Oracle Forms attempts to delete a record in a block that is a master block in a master/detail relation.
- **On-Clear-Details** Fires when Oracle Forms needs to clear records in a block that is a detail block in a master/detail relation because those records no longer correspond to the current record in the master block.
- **On-Populate-Details** Fires when Oracle Forms needs to fetch records into a block that is the detail block in a master/detail relation so that detail records are synchronized with the current record in the master block.

Message-Handling Triggers:

Oracle Forms automatically issues appropriate error and informational messages in response to runtime events. Message handling triggers fire in response to these default messaging events.

- **On-Error** Replace a default error message with a custom error message, or to trap and recover from an error.
- **On-Message** To trap and respond to a message; for example, to replace a default message issued by Oracle Forms with a custom message.

Validation Triggers:

Validation triggers fire when Oracle Forms validates data in an item or record. Oracle Forms performs validation checks during navigation that occurs in response to operator input, programmatic control, or default processing, such as a Commit operation.

- **When-Validate-Item**
- **When-Validate-Record**

Navigational Triggers:

Navigational triggers fire in response to navigational events. Navigational triggers can be further sub-divided into two categories: Pre- and Post- triggers, and When-New-Instance triggers. Pre- and Post- Triggers fire as Oracle Forms navigates internally through different levels of the object hierarchy. When-New-Instance-Triggers fire at the end of a navigational sequence that places the input focus on a different item.

- **Pre-Form** Perform an action just before Oracle Forms navigates to the form from “outside” the form, such as at form startup.
- **Pre-Block** Perform an action before Oracle Forms navigates to the block level from the form level.
- **Pre-Record** Perform an action before Oracle Forms navigates to the record level from the block level.
- **Pre-Text-Item** Perform an action before Oracle Forms navigates to a text item from the record level.
- **Post-Text-Item** Manipulate an item when Oracle Forms leaves a text item and navigates to the record level.
- **Post-Record** Manipulate a record when Oracle Forms leaves a record and navigates to the block level.

- **Post-Block** Manipulate the current record when Oracle Forms leaves a block and navigates to the form level.
- **Post-Form** Perform an action before Oracle Forms navigates to “outside” the form, such as when exiting the form.
- **When-New-Form-Instance** Perform an action at form start-up. (Occurs after the Pre-Form trigger fires).
- **When-New-Block-Instance** Perform an action immediately after the input focus moves to an item in a block other than the block that previously had input focus.
- **When-New-Record-Instance** Perform an action immediately after the input focus moves to an item in a different record.
- **When-New-Item-Instance** Perform an action immediately after the input focus moves to a different item.

Transactional Triggers:

Transactional triggers fire in response to a wide variety of events that occur as a form interacts with the data source.

- On-Delete
- On-Insert
- On-Update
- On-Logon
- On-Logout
- Post-Database-Commit
- Post-Delete
- Post-Insert
- Post-Update
- Pre-Commit
- Pre-Delete
- Pre-Insert
- Pre-Update

Query-Time Triggers:

Query-time triggers fire just before and just after the operator or the application executes a query in a block.

- **Pre-Query** Validate the current query criteria or provide additional query criteria programmatically, just before sending the SELECT statement to the database.
- **Post-Query** Perform an action after fetching a record, such as looking up values in other tables based on a value in the current record. Fires once for each record fetched into the block.

**TASK12: Create a mailing label report as described below:**

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

**TASK13: Create a mailing label report as described below:**

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>

**TASK14: Create a mailing label report as described below:**

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

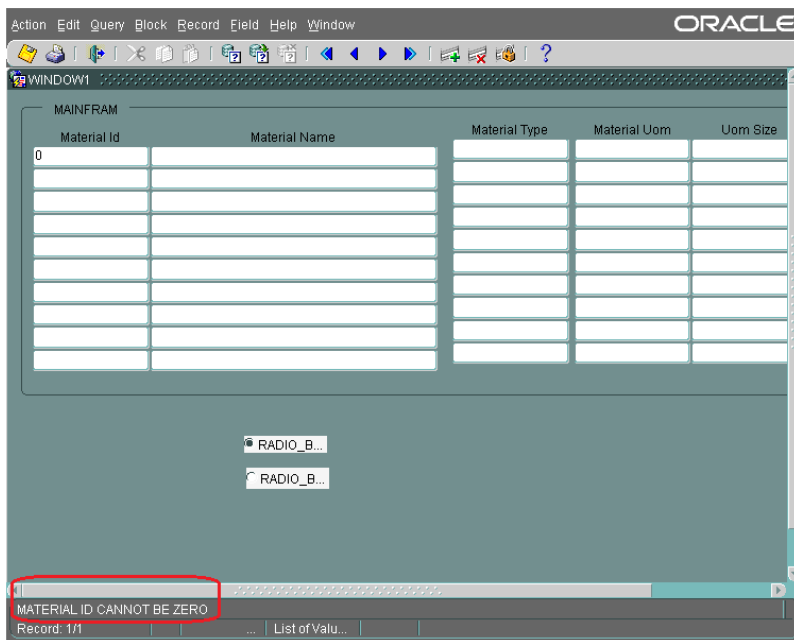
<Tasks are being Synched with the Master Case study, Details to be provided soon>

**Video8: Script: Triggers**

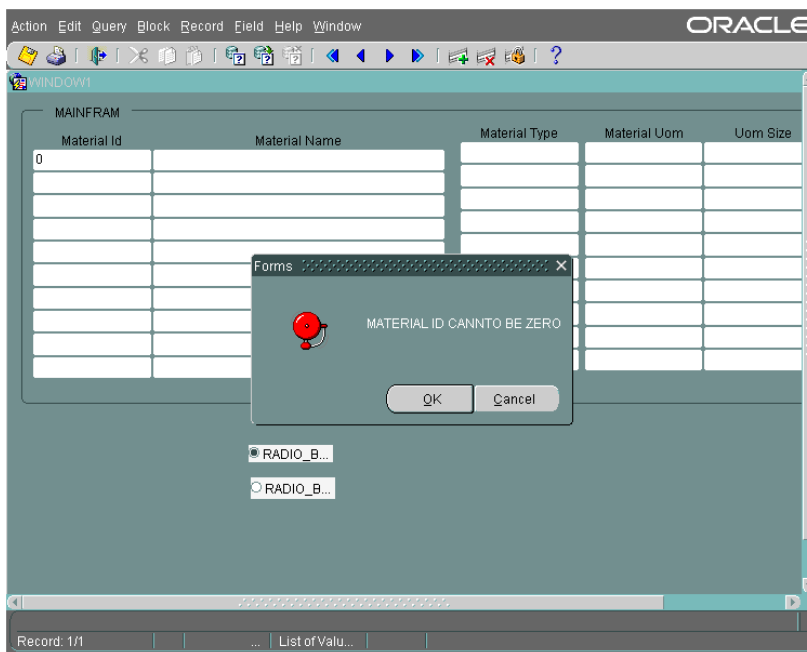
Explaining different kind of triggers and their usage and practical example.

9. Message and Alerts

We have used the message function in the Trigger section of this document above. The below diagram shows the message appearing on the last line of the form, when user types 0 in the Material ID field. This is called a message. The location and behavior of the message is such that it does not create much of an impact.



A better approach may be to force the user's attention to the problem. This can be accomplished by creating a pop-up dialog box that forces the user to read the message and click an OK button (or some other button). In Oracle Forms, this is called an Alert. Let us see, how the same will look if converted to an alert:



To set up an Alert:

1. Use the Object Navigator to display the Alerts area. Pull down the Navigator menu and choose Create. Click on the default name that is given for the new alert (something like ALERT4). You can rename it to a valid name.
2. Bring up the Property Palette for this Alert and fill in the following properties:

Title: ALERT4

Message: MATERIAL ID CANNTO BE ZERO

Alert Style: STOP

Button 1 Label: OK

Leave the **Button 2 Label** and the **Button 3 Label** blank

Default Alert button: Button 1

Leave all of the other properties with their defaults and close the Property palette.

3. Change the above WHEN-VALIDATE-ITEM trigger on the MATERIAL_ID field to:

```
DECLARE
return_alert NUMBER;
BEGIN
  IF (:MATERIAL_MASTER.MATERIAL_ID = 0) THEN
    return_alert := SHOW_ALERT ('ALERT41');
    RAISE FORM_TRIGGER_FAILURE;
  END IF;
```

END;

The SHOW_ALERT procedure calls up the specified alert and obtains the return value (based on the button the user clicks on) to assign to a local variable called return_alert.

In general, Alerts may provide several different buttons for the user to click on. For example, to confirm exiting a form, a POST-FORM trigger might invoke an “Are you sure you want to Exit” Alert with “Yes” and “No” buttons. Or, to confirm deleting a record, a PRE-COMMIT trigger might invoke an “Are you sure you want to delete this Employee?” Alert with “Yes” and “No” buttons. Based on the return_alert value, either a commit or rollback might be issued.



TASK15: Create a mailing label report as described below:

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>



Video9: Script: Message and Alert

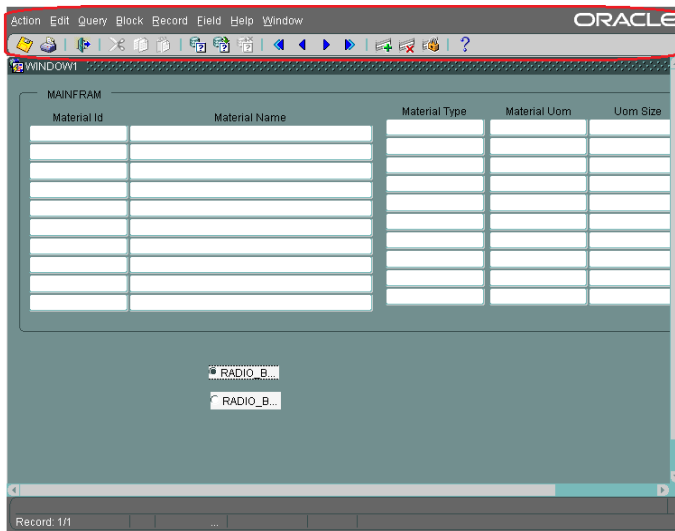
Screenplay showing how to difference between message and alert and their specific usage, How to create a message and alert.

10. Menu



In this section you will learn how to create a custom menu in a form.

You might have noticed that when we run any form, there is a default menu appears, though we never called any menu from our form. This is shown in red circle below:



This is the default menu, which gets called by default from any form you develop. Let us see the source of this.

Open the form, click on the small box next to the form as shown in the diagram below, the property palette will appear. If you go the 'Menu Module' section, you can see the name 'DEFAULT&SMARTBAR'. This is the name of the menu shows up in the above diagram. Let us make some changes to see the effect. First, we will delete this value (re: diagram below), and run the form, and notice that the menu has disappeared (diagram 10.3)

Forms

- MYFIRSTFORM
 - Triggers
 - Alerts
 - ALERT41
 - Attached Libraries
 - Data Blocks
 - Canvases
 - CANVAS4
 - Graphics
 - Editors
 - LOVs
 - Object Groups
 - Parameters
 - Popup Menus
 - Program Units
 - Property Classes
 - Record Groups
 - Reports
 - Visual Attributes
 - Windows

Form Module: MYFIRSTFORM	
General	
Name	MYFIRSTFORM
Subclass Information	
Comments	
Help Book Title	
Functional	
Title	MODULE1
Console Window	WINDOW1
Menu Module	DEFAULT&SMARTBAR
Initial Menu	
Defer Required Enforcement	No
Menu Security	
Menu Role	
Navigation	
Mouse Navigation Limit	Form
First Navigation Data Block	<Null>
Records	
Current Record Visual Attribute Group	<Null>
Database	
Validation Unit	Default

Form Module: MYFIRSTFORM	
General	
Name	MYFIRSTFORM
Subclass Information	
Comments	
Help Book Title	
Functional	
Title	MODULE1
Console Window	WINDOW1
Menu Module	
Initial Menu	
Defer Required Enforcement	No
Menu Security	
Menu Role	
Navigation	
Mouse Navigation Limit	Form
First Navigation Data Block	<Null>
Records	
Current Record Visual Attribute Group	<Null>
Database	
Validation Unit	Default

TATA CONSULTANCY SERVICES

TCS Internal

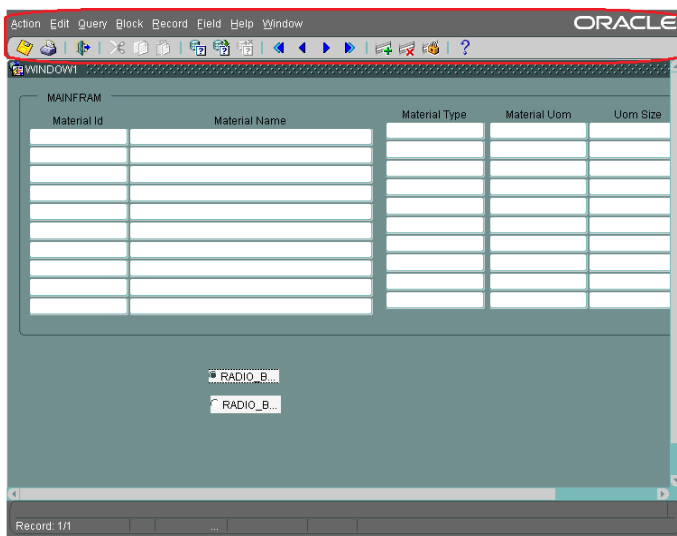


Diagram 10.2: With Default Menu

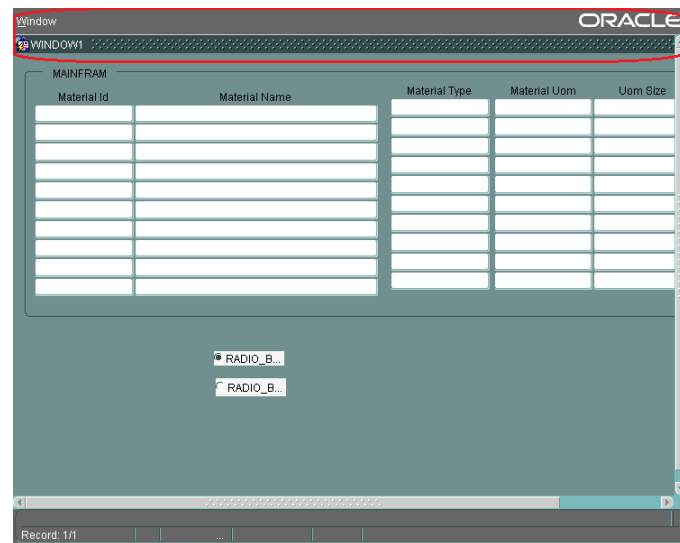
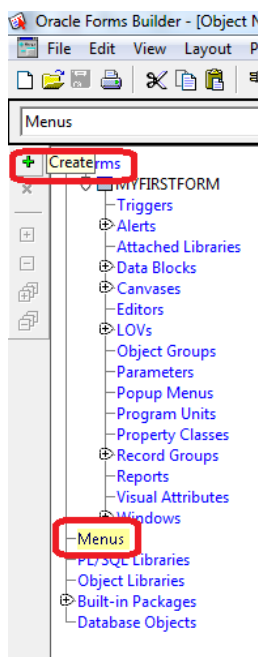
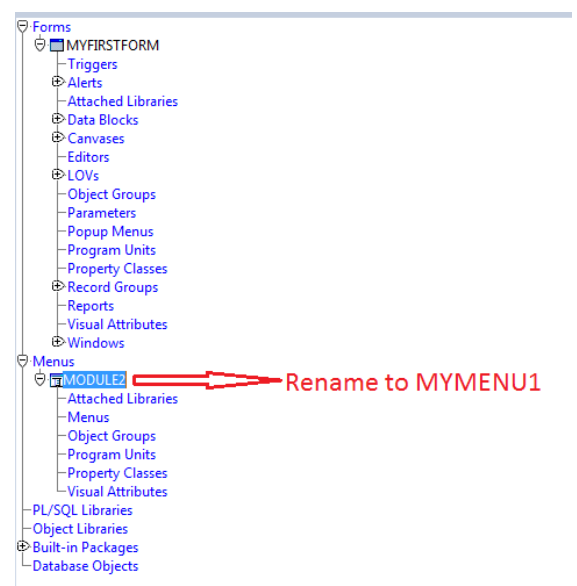


Diagram 10.3: With No Menu

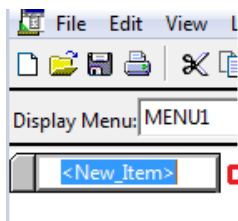
Now, we will create our custom menu.



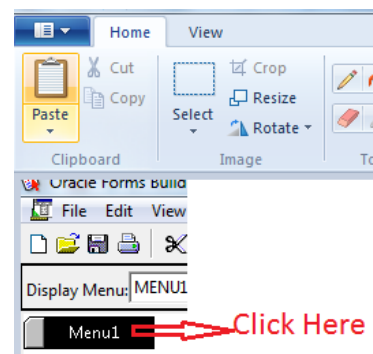
A. Click on Menus and '+'



B. It will create a menu called MODULE2, rename it.



➡️ **Raname to Menu1**



➡️ **Click Here**

C. This will open the menu dialogue box, rename the default display as 'Menu1'

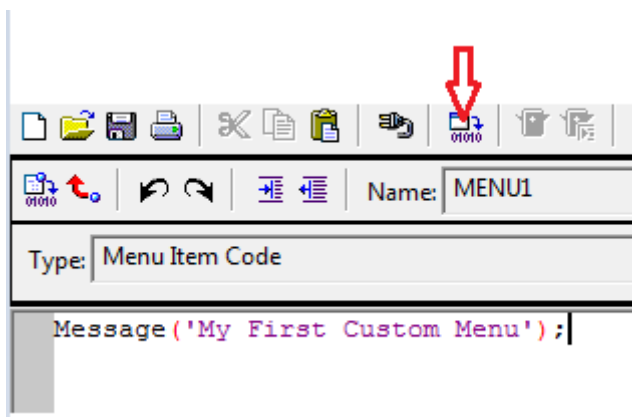
D. Double Click on the Black portion to open the Menu properties.

Details of the menu item will appear:

Menu Item: MENU1	
General	
Name	MENU1
Subclass Information	
Comments	
Functional	
Enabled	Yes
Label	Menu1
Menu Item Type	Plain
Magic Item	None
Menu Item Radio Group	
Command Type	PL/SQL
Menu Item Code	More...
Keyboard Accelerator	

Click on the More button as above, the Menu item code window will appear, type the following:

```
Message('My First Custom Menu');
```



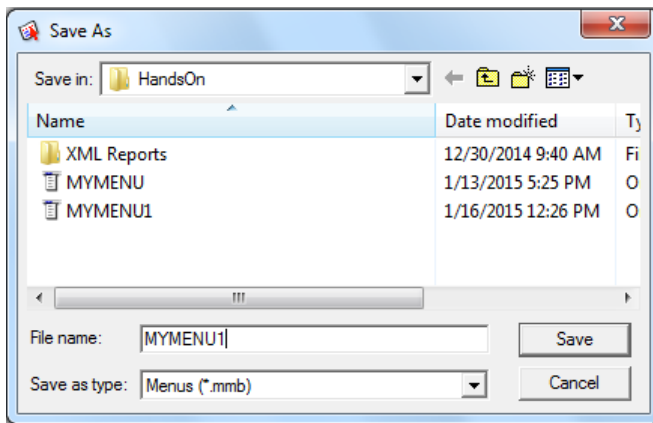
01010

Name: MENU1

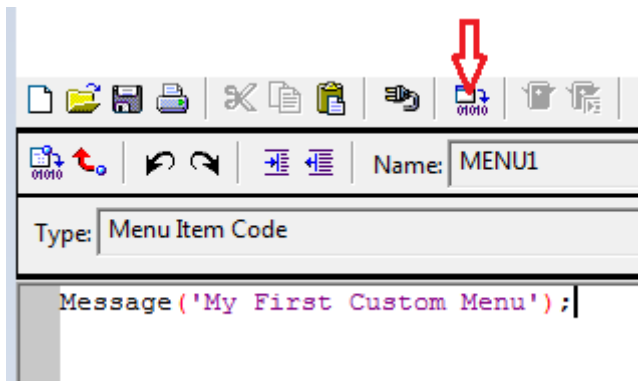
Type: Menu Item Code

```
Message('My First Custom Menu');
```

File > Save As and save the menu as .mmb file:

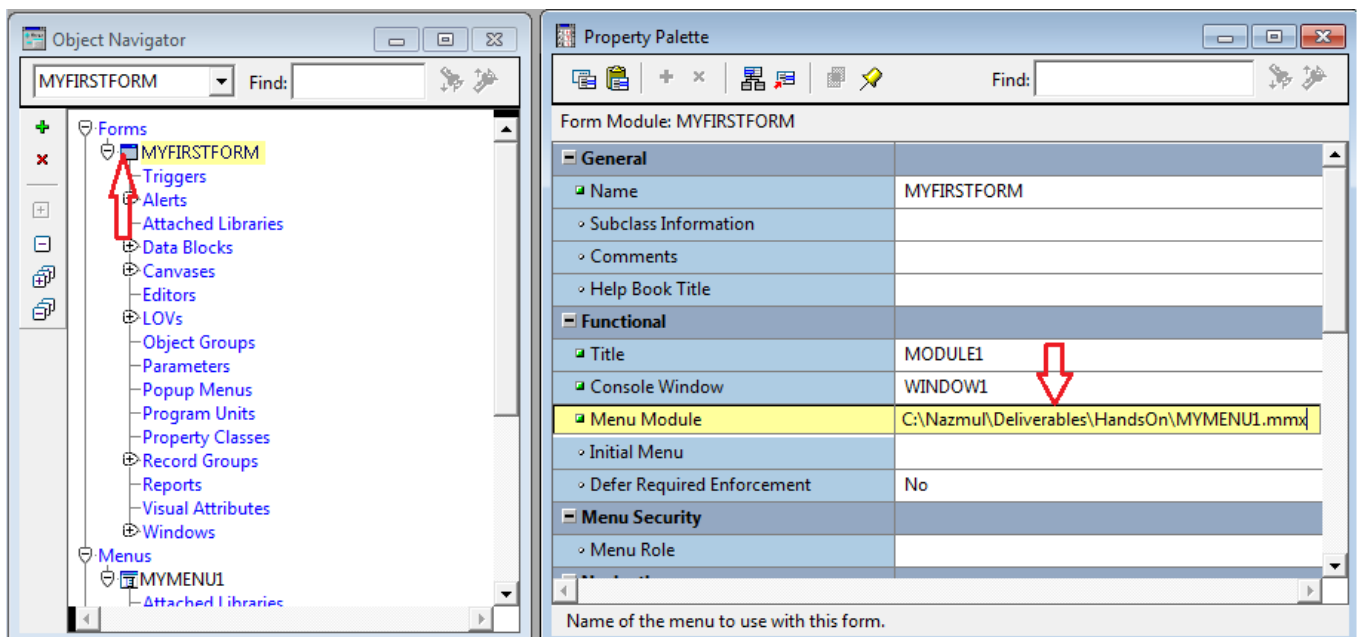


And hit the compile button:

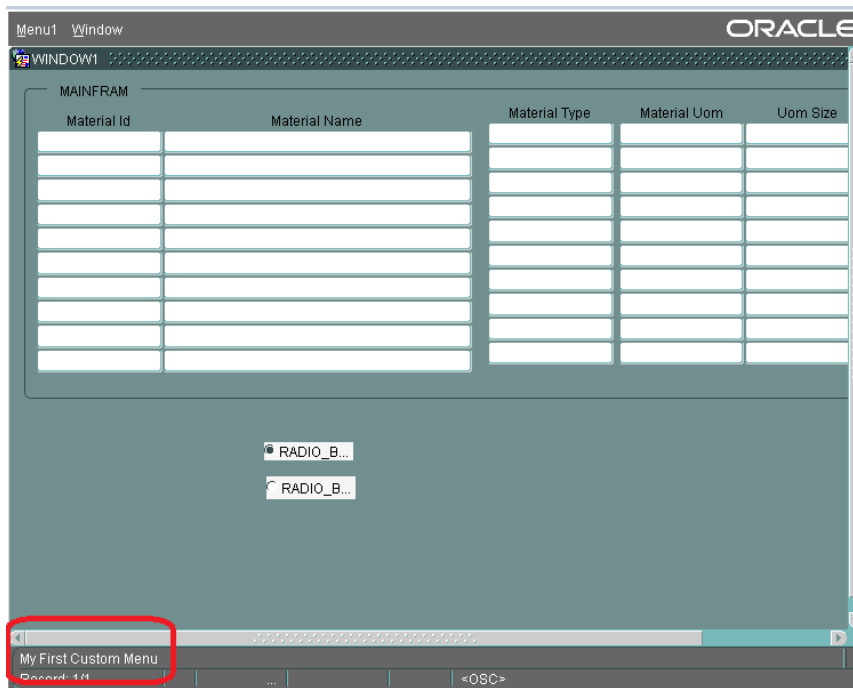


Your menu is ready to be used. Now we need to invoke it.

Go to the Object Navigator and open the form property dialogue, and type the name of the form file you have save, with full path. The extension of the file should be .mmx:



Now run the form and click on the menu item – 'Menu1' to see that the message you have put inside the menu appears at the bottom of the screen.



In the above example we have asked the menu to display a message, however in practical scenario, you need to call a form/function on this click. You can use the below code snippet to call a form namely employee.fmb

```
BEGIN
```

```
  RUN_PRODUCT(FORMS, 'employee', SYNCHRONOUS, RUNTIME, FILESYSTEM, ", ");
```

```
END;
```

You can use a variety of functions inside the menu call, a complete list is available at Oracle® Applications Developer's Guide Release 12 available at Oracle's website.



TASK16: Create a mailing label report as described below:

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>



Video10: Script: Message and Alert

Screenplay showing how to create a custom menu and attach it to a form, calling functions from the menu.

11. Custom Forms in E-Business Suite



So far we have developed standalone forms which has nothing to do with Oracle Applications. However, when we develop a form to be ported into Oracle Applications, we need to follow some standards and guidelines. In this section we will discuss how to make this form.

Our purpose is to create a form which have look and feel and basic functionality like a standard Oracle Applications form.

Our custom Form	Oracle Application Seeded form

I have put the custom form we have developed, and Oracle Seeded form side by side to notice the difference. You can notice the layout differences and the professional look of the seeded form, But what you cannot see is the difference in the functionality (e.g.: navigations, messages etc.), which the seeded form offers.

For this purpose, Oracle has given a starting form called `TEMPLATE.fmb`, which includes these features to be used to develop a form for Oracle Application porting. The `TEMPLATE.fmb` file can be found in the `$AU_TOP/forms/US` directory on the server hosting Oracle Applications.

```

oraapp12@tcs01hw325310-vm03:/oracle/R1212/apps/apps_st/appl/au/12.0.0/forms/US
/oracle/R1212/apps/apps_st/appl/au/12.0.0/forms/US>cd $AU_TOP/forms/US
/oracle/R1212/apps/apps_st/appl/au/12.0.0/forms/US>ls -l TEMPLATE.fmb
-rwxrwxr-x 1 oraapp12 dba 385024 Jan 3 2007 TEMPLATE.fmb
/oracle/R1212/apps/apps_st/appl/au/12.0.0/forms/US>

```

The `TEMPLATE` form includes an example window with sample blocks, as well as many referenced objects such as windows, canvases, blocks, Lists of Values (LOVs), and other objects. It inherits the object groups from the

APPSTAND form, which is a platform-specific form. By starting out with the TEMPLATE form when developing custom forms in E-Business Suite, developers ensure that they will get the same look and properties of the standard product forms. Examples of such properties specific to Oracle Applications are the toolbar, the menu, the calendar, applications property classes, required Form Level triggers, required procedures, LOVs, parameters, and many others.

11.1 Preparing the Desktop for Custom Forms Development

In order to begin development, you must base your custom form development on the TEMPLATE form (TEMPLATE.fmb). This ensures that your custom form will inherit the shared core form components and libraries. The TEMPLATE form has attached several libraries such as FNDSQF, APPCORE, APPDAYPF, and others that contain many of the Application Object Library utilities.

Let us assume that you want to store all your FMB and PLL files in the folder C:\code\forms on your Windows desktop. Transfer TEMPLATE.fmb from \$AU_TOP/forms/US from the middle-tier machine to the C:\code\forms directory on your desktop. Now in Oracle Forms try to open TEMPLATE.fmb. You will get a message “FRM-18108: Failed to load the following objects”. If you click on OK, you will get another error listing the missing libraries (FRM-10102 errors). At this stage, you must not save TEMPLATE.fmb once it is opened.

The reason we see those errors is because the TEMPLATE form itself depends on other forms and libraries. Before you begin to transfer all the dependent objects from the server, you must set the FORMS_PATH environment variable on your computer. In order to do so, right-click My Computer, click on Properties, and in the advanced setting, click on the button labeled Environment Variable. For R12, name of the environment variable will be FORMS_PATH and its value set to C:\code\forms. For Release 11i, you can change the value of FORMS60_PATH in the registry to include your directory, which is C:\code\forms in our example. After changing the environment variable, you must restart the forms builder for this change to take effect.

In order to open TEMPLATE.fmb successfully, we have two options. The first option is to transfer everything from the \$AU_TOP/forms/US and \$AU_TOP/resource directories from the middle-tier machine to the C:\code\forms directory on the desktop. However, in R12 instance, this will equate to transferring files over 3GB to your desktop. Therefore, the second option is to be selective for the files being transferred from the server. From \$AU_TOP/forms/US, transfer the APP*.fmb and FND*.fmb forms to folder C:\code\forms. Similarly, from \$AU_TOP/resource, transfer the APP*.pll, FND*.pll, VERT*.pll, PS*.pll, HR*.pll, GMS*.pll, FV*.pll, IGI*.pll, GLOBE.pll, JA.pll, JE.pll, JL.pll, VERT.pll, GHR.pll, PQH_GEN.pll, PSAC.pll, CUSTOM.pll, and OPM.pll libraries. If you still receive message for some other missing libraries, then transfer those as well. Sometimes the error message for a missing library can be misleading. For example, you might get the error, “Cannot find APPDAYPK.pll,” despite the file being present in the C:\code\forms directory. In such cases, try to open APPDAYPK.pll itself to see the dependent libraries that remain to be transferred to your desktop.

Once all the required objects have been transferred to C:\code\forms, you will be able to open TEMPLATE.fmb without any errors and at that point you can start developing the custom form.

11.2 Steps for Developing Custom Forms in E-Business Suite

Follow these steps to create custom forms in E-Business Suite:

1. **Create TEMPLATE form** Make a copy of TEMPLATE.fmb and rename it to your custom form name. Your form name will begin with XX.
2. **Develop form** Develop your form as per programming guidelines and standards in Oracle Applications Forms Development Guide. Detailed guidelines are available at http://download.oracle.com/docs/cd/B34956_01/current/acrobat/120devg.pdf.

3. **Generate runtime file** Transfer the FMB file to mid-tier and compile it to generate an FMX compiled file.
4. **Register** Register the form with Custom Application.
5. **Create form function** Create a form function attached to the form.
6. **Attach to menu** Attach the form function to the menu.
7. **Assign to responsibility** Assign the menu to the responsibility.



Note: The custom forms based on TEMPLATE.fmb will not run successfully from the Forms Developer tool on your desktop due to the dependencies on Oracle Applications–specific AOL objects and user exits that are not available in Forms Developer. For this reason, the custom form must be generated and tested from the middle-tier machine where the Oracle Applications forms server resides after registering the custom form with AOL and assigning it to a menu and a responsibility.



TASK17: Create a mailing label report as described below:

This denotes the task to be completed by the audience on his own PC. The layout of the output has to be followed as it is. For any confusion, the faculty should be contacted.

<Tasks are being Synched with the Master Case study, Details to be provided soon>



Video11: Script: Making custom forms for Oracle Applications

Screenplay showing how to create a custom form using TEMPLATE.fmb for Oracle Application with example.

This is the last document in this series. For further studies, you may refer to the further studies Appendix.

APPENDIX A: Further Reading

- Oracle10g Forms I: Build Internet Applications, Volume A (Volume Set)
by Sideris Courseware Corporation
Available on TCS Books 24x7 (Ultimatix > Learning & Collaboration > Book 24x7