CHAPTER 5

ITEMS

CHAPTER OBJECTS

In this Chapter, you will learn about:

Nearly all of the interaction between users and your forms will take place through items. In this Chapter, you will learn more about how to create items of various types and how to set their properties. You will create database items for items based on columns in a database and non-database items to initiate some kind of action or to represent data or information that is not based on a column in a database.

Throughout the Exercises that deal with items, you will encounter questions and answers that provide brief tips about GUI design techniques. This is by no means intended to be an exhaustive coverage of the topic. In fact, in many of the Exercises you will be encouraged to ignore layout and aesthetics in favor of concentrating on creating Forms objects. Once you have mastered the basic functions of Oracle Forms, it is recommended that you read the “Designing Visually Effective Applications” section in the Oracle Forms on-line manuals, or purchase a separate book dedicated to the subject of GUI design such as *GUI Design Essentials* by Susan Weinschenk, Pamela Jamar, and Sarah Yeo (John Wiley & Sons, 1997).

go to contents

LAB 5.1

TEXT ITEMS AND DISPLAY ITEMS

LAB OBJECTIVES

After this Lab, you will be able to:

- Create and Define Text Items Without the Wizard
- Create and Define Display Items

In Chapters 1 through 4, you created and defined some text and display items using wizards. In this Lab, you will go a few steps further by exploring their uses and properties in more detail

Display items and text items are fairly similar and share many of the same properties. The biggest difference between the two is that a user can navigate to a text item and change its value. This is not possible with a display item. As its name implies, it merely *displays* information.

Items, especially text and display items, have by far the most properties of all the objects in Forms. It would be impractical and unnecessary to discuss all of them here. Impractical because of the sheer number and unnecessary because so many of the property names are self-explanatory. However, there are several properties in the Functional, Data, and Database property categories that are worth exploring and that you will experiment with in the Exercises.

TEXT ITEMS

Text items are usually database items, meaning they are commonly based on columns in a database. The Data Block Wizard is the easiest tool to use to create these types of items. As you have seen in previous Chapters, it

go to contents

automatically sets properties so that the form knows which database column the text item is based on.

Text items can serve as non-database items as well. That is, they do not always have to be based on a column in a database.

DISPLAY ITEMS

As described in Chapter 1, "Concepts and Objects," display items can be either database items or non-database items. In this Lab's Exercises, the display item you create will be a non-database item. You will use its properties to configure it to perform a calculation. In other situations, you will use display items to display data from other tables.

■ *FOR EXAMPLE:*

Assume you have a block based on the ENROLLMENT table that includes the STUDENT_ID column. Along with the enrollment information, you'd also like to display the student's last name. You could create a display item to hold the last name and use a trigger to fetch the value from the database. This will be covered in Chapter 6, "Triggers & Built-ins."

Keep in mind that not all non-database display items have to display the results of calculations or values fetched from other tables. Often, display items are used to provide simple information to the user such as the time, date, or perhaps the name of the database to which the user is connected.

LAB 5.1 EXERCISES

5.1.1 CREATE AND DEFINE TEXT ITEMS WITHOUT THE WIZARD

The purpose of this Exercise is to manually create a database text item. A second and equally important purpose is for you to become familiar with some of the more common text item properties. Incidentally, almost all of the properties you will explore here also apply to display items.

Use the wizards to quickly create a new form based on the COURSE table. Do not include the COST column in your block. Enforce data integrity should be **unchecked.** Include the audit columns in the block, but do not display them on the canvas.

Choose Form as your layout style. Name both the canvas and its frame COURSE.

go to contents

Use the Object Navigator to create an item in the COURSE block. Name it COST and position it between DESCRIPTION and PREREQUISITE in the COURSE block.

a) Is COST positioned on the COURSE canvas? What one property should you change to place it there? What values has Forms assigned for X Position and Y Position?

b) Instead of manually dragging COST to position it between DESCRIPTION and PREREQUISITE, which of the Layout Editor's functions can you use to position it automatically?

Now that you have created and positioned the text item, begin exploring its properties.

c) Did you have to adjust the Item Type property when you created this item? What does this tell you about the default behavior of creating items in the Object Navigator?

d) How would the Item Type property have been set if you had created COST in the Layout Editor?

Change the COST item's Enabled property to No. Run the form and execute a query.

go to contents

e) How does the appearance of the value in the COST item differ from that of the other items? Can you enter the item via the keyboard or with the mouse? When might you want to use this function?

Exit the form and return to the Form Builder. Change the COST item's Enabled property back to Yes. Stay in the Functional category and look at the Multi Line and Word Wrap properties.

f) Judging from the names of these properties and the description in the hint line, would it be appropriate to set Multi Line to Yes for COST? Which other item in the COURSE block might it apply to and why?

You will need to use the Database Objects node to answer parts of the following questions.

g) What is the data type of the course.cost column? What is the value of the COURSE.COST item's Data Type property? What does this tell you about the default behavior of creating items outside of the wizard?

Change the COST item's Data Type property to Number.

For the next question, you will work with the COURSE_NO item. Select COURSE_NO in the Object Navigator and set its Initial Value property to:

```
:SEQUENCE.COURSE_NO_SEQ.NEXTVAL
```

h) How will this affect the COURSE_NO item?

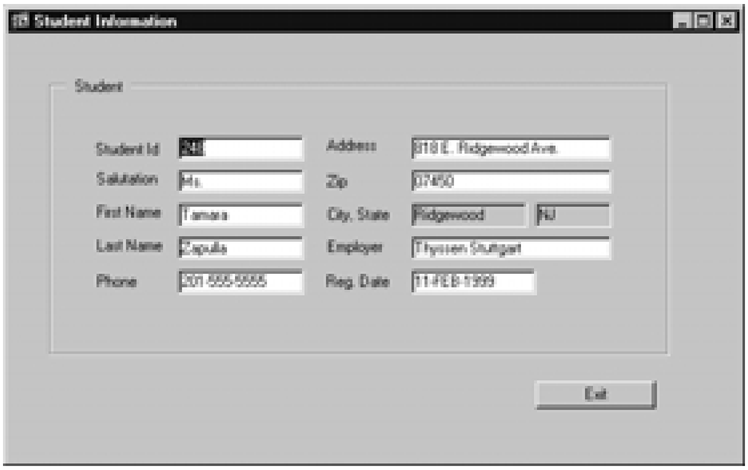
Return to the properties for the `COST` item. Select `Format Mask` in the Property Palette and press the F1 key on your keyboard. Scroll down in the help screen until you see a section titled "`Numbers`".

> **i)** What should you put in the `Format Mask` property to format `COST` so that it is displayed like this: `$1,195`?
>
> ______________________________
>
> ______________________________

> **j)** Is `COST` set to be a database item?
>
> ______________________________
>
> ______________________________

Save this form as `R_COURSE.fmb` as you will be using it again.

> **k)** What are some of the things about the layout of the form in Figure 5.1 that make it attractive and easy to read?
>
> ______________________________
>
> ______________________________



Student Id	[illegible]	Address	818 E. Ridgewood Ave.
Salutation	Ms.	Zip	07450
First Name	Tamara	City, State	Ridgewood NJ
Last Name	Capula	Employer	Thyssen Stuttgart
Phone	201-555-5555	Reg. Date	11-FEB-1999

Figure 5.1 ■ Items from the `STUDENT` block arranged on a canvas.

go to contents

5.1.2 Create and Define Display Items

Use the wizards to quickly create a new master-detail form based on the ENROLLMENT and GRADE tables. Include the audit columns in the blocks, but do not display them on the canvas. For both blocks, Enforce data integrity should be **unchecked.**

Choose Form as the layout style for the ENROLLMENT block.

Choose Tabular as the layout style for the GRADE block and display five records.

Adjust the widths of the items in the GRADE block so that they fit neatly on the canvas.

Name the canvas ENRO_GRAD and the frames ENROLL and GRADE, respectively.

In this Exercise, you will create a display item that displays the average grade for each enrollment.

Use the Layout Editor's Tool Palette to create a display item in the GRADE block and name it GRADE_AVG. Position it below the NUMERIC_GRADE column of items.

a) How many GRADE_AVG items are displayed? Remember, GRADE_AVG must belong to the GRADE block.

b) Which property can you change so that the GRADE_AVG item is displayed only once, but the rest of the items in the GRADE block are displayed five times?

Resize GRADE_AVG to match NUMERIC_GRADE and give it a meaningful prompt.

go to contents

c) Which of the Prompt properties should you change so that the GRADE_AVERAGE prompt is positioned like the prompt for ENROLLMENT.SECTION_ID?

You are going to make GRADE_AVG a calculated item and have it display the average NUMERIC_GRADE.

d) Which category of properties will you work with for GRADE_AVG to make it a calculated item?

You will be using a pre-written function to calculate the average of the grades.

e) How do you think you should set the Calculation Mode and Summary Function properties?

f) Which item in which block will you be summarizing? Set the associated properties accordingly.

g) What should the data type of the GRADE_AVG item be? Why?

Set the GRADE block's Query All Records property to Yes.

go to contents

h) Why do you think you had to set this property?

Run the form again to test your calculation.

LAB 5.1 EXERCISE ANSWERS

5.1.1 ANSWERS

a) Is COST positioned on the COURSE canvas? What one property should you change to place it there? What values has Forms assigned for X Position and Y Position?

Answer: No, COST *is not positioned on the* COURSE *canvas. Change its* Canvas *property to position it on the canvas. Forms has assigned the value 0 for both* X Position *and* Y Position.

For an item to be visible at run-time, it must be assigned to a canvas that is visible in a window. Even if it has values for other physical properties such as Height, Width, and so on, the user will not be able to see it until it is positioned on a canvas.

This is so because it is common to create and use items that are never positioned on canvases and therefore never appear to the user. You should have noticed that the COST item's Canvas property was set to Null before you changed it. When the Canvas property of an item is set to Null, it is referred to as a *null-canvas item*. It is common to use null-canvas items as variables and assign and reference their values using PL/SQL. You can see null-canvas items in the Object Navigator, and you can configure their properties; however, they are not visible in the Layout Editor or at run-time.

Above the Canvas property in the Property Palette, you should have noticed the Visible property. This property must also be set to Yes to make an item visible on the canvas it is assigned to. You did not have to explicitly set this because whenever you create an item, its Visible property defaults to Yes. It is common to programmatically change an item's Visible property to show it and hide it at run-time. Note that when the Visible property is set to No, you will still be able to see it in the Layout Editor. However, when you run the form, it will not be visible to the user.

go to contents

b) Instead of manually dragging COST to position it between DESCRIPTION and PREREQUISITE, which of the Layout Editor's functions can you use to position it automatically?

Answer: You can select the COURSE *frame and click the* Update Layout *button.*

c) Did you have to adjust the Item Type property when you created this item? What does this tell you about the default behavior of creating items in the Object Navigator?

Answer: No. This tells you that the default behavior of the Object Navigator is to create items as text items.

d) How would the Item Type property have been set if you had created COST in the Layout Editor?

Answer: That would depend on which item tool you selected on the Layout Editor's Tool Palette.

In the Tool Palette, you create items using the tools that correspond to their item type. If you are creating a button, you use the Button tool. If you are creating a text item, you use the Text Item tool.

e) How does the appearance of the value in the COST item differ from that of the other items? Can you enter the item via the keyboard or with the mouse? When might you want to use this function?

Answer: The COST *value is grayed out and it cannot be navigated to with the mouse or the keyboard.*

Setting a text item's Enabled property to No will allow you to display values in the item, but prevent the user from editing or updating those values. It will also gray them out, which will set them off from other items on the canvas. This can be useful if you want to display information to a user, but not let them change or update it.

■ *FOR EXAMPLE:*

The STUDENT schema contains SEQUENCES that generate values for columns such as student.student_id and instructor.instructor_id. When you display these sequence-generated values in forms, you will not want to allow the user to insert or update their values. By setting the Enabled property to No, the user will be able to view the value of student_id or instructor_id, but not change it.

Won't a display item provide the same functionality? In a way, yes, in that it will also display information in an item but will prevent the user from accessing it. However, a display item prevents access during Enter Query mode, as well as Normal mode. This means that if `STUDENT_ID` is a display item, a user cannot use Enter Query mode to search for a student whose ID is 101. However, a text item with `Enabled` set to `Yes` is accessible during Enter Query mode. This will allow a user to enter query criteria for the item. Note that there is no `Enabled` property for display items.

f) Judging from the names of these properties and the description in the hint line, would it be appropriate to set `Multi Line` to `Yes` for `COST`? Which other item in the `COURSE` block might it apply to and why?

Answer: No, it would not really be appropriate for `COST`. *It might apply to* `DESCRIPTION`.

As their names imply, the `Multi Line` and `Wrap Style` properties allow you to create items that can display more than one line. Because a `DESCRIPTION` can be rather lengthy, it may be appropriate to set its `Multi Line` property to `Yes`. If you set `Multi Line` to `Yes`, you must remember to manually adjust the `Height` property of the item if you want more than one line to be visible to the user. The `Multi Line` and `Word Wrap` properties are often used when displaying address items.

g) What is the data type of the `course.cost` column? What is the value of the `COURSE.COST` item's `Data Type` property? What does this tell you about the default behavior of creating items outside of the wizard?

Answer: The `course.cost` *column has* `Number` *as its data type. The* `COURSE.COST` *item's* `Data Type` *property is set to* `Char`. *When you create items outside of the wizard, the item does not inherit any properties from the database and sets every item's* `Data Type` *property to* `Char`.

In this case, the mismatch of data types did not prevent the form from running and functioning properly. However, it is wise to adjust the item's `Data Type` property to match the data type of its base column. In Question i, you will work with format masks to format the appearance of the `COST` item. If you had left the `Data Type` as `Char`, you would have encountered problems when trying to create a format mask.

h) How will this affect the `COURSE_NO` item?

Answer: When the user is creating new records, the form will populate the `COURSE_NO` *item with the next value in the sequence.*

In many applications, sequences will exist in the database and will be used to populate column values. Forms can make use of database sequences by setting the `Initial Value` property using the following syntax:

go to contents

```
:SEQUENCE.sequence_name.NEXTVAL
```

In the `STUDENT` schema, there is a sequence called `COURSE_NO_SEQ` that you can use for `COURSE_NO` values. There are also sequences for the `INSTRUCTOR_ID`, `SECTION_ID`, and `STUDENT_ID` columns.

i) What should you put in the `Format Mask` property to format `COST` so that it is displayed like this: `$1,195`?

Answer: You should use `$9,999`.

`Format Mask` is a powerful and flexible property in that it lets you display information in a format that is different from how the information is stored in the database. You can use format masks to format the display of currencies, Social Security numbers, telephone numbers, product codes, dates, character strings, etc. Format masks can also be used to validate how values are entered into an item.

■ FOR EXAMPLE:

Set the `COURSE.CREATED_DATE` item's `Canvas` property to `COURSE`. Do not be concerned with where it is positioned on the canvas. Set the `COURSE.CREATED_DATE` item's `Format Mask` property to `DY-DD-MM-YY`. Set its `Width` property to `90`. Run the form and issue a query.

Note that the date was returned and displayed as indicated in the `Format Mask` property. In the running form, change the `CREATED_DATE` value to `12-MAR-99`. Tab out of the item and look at the running form's hint line. Note that the error message is indicating the proper format mask. So, not only has the format mask affected the display of the item, but it will also prevent users from entering data in invalid formats.

j) Is `COST` set to be a database item?

Answer: Yes it is. The `Database Item` *property is set to* `Yes`.

The `Database Item` property tells Forms that this item is based on a column in the database. Forms will include this item's name in whatever SQL statements it issues to the database. Take the following simple steps to get a feel for how the properties you set for an item can affect how Forms builds queries for blocks.

1) Set `Database Item` to `No`.
2) Run the form and issue a query.

go to contents

What happened? Note that the COST value was not returned. This is because Forms ignored this item when it issued its SELECT statement to the database. It assumed that COST was a non-database item.

1) Set Database Item to Yes.
2) In the Object Navigator, change the name of COST to V_COST.
3) Run the form and issue a query.

What happened? You got an Unable to Perform Query error because Forms included V_COST in the SQL statement it issued to the database. Because V_COST is not a column in the COURSE table, the database returned an error.

1) Set the Column Name property to COST.
2) Run the form and issue a query.

Note that this time the query worked. The item name V_COST was overridden by the item's Column Name property. When you create data items manually, it is wise to set the Column Name property appropriately, even if you name the item after its base column. Note that the other data items in the block that were created by the wizard have this property set.

k) What are some of the things about the layout of the form in Figure 5.1 that make it attractive and easy to read?

Answer: Read the discussion below.

When using an application, it is important that the user can accomplish her tasks quickly and easily without being distracted by the interface. The user should be able to navigate from item to item quickly and smoothly and read and understand the information on the form easily.

■ FOR EXAMPLE:

In Figure 5.1, there is plenty of space between each item so that the form does not appear crowded. The items are sized similarly so that their right-hand edges are nearly flush. If they were all sized differently, the right-hand edge of all the items would not be smooth and would create a distracting, jagged edge.

The font is uniform across all of the items and is rather plain. Fancy fonts with serifs are attractive, but they are rather difficult to read and should not be used in Forms applications.

go to contents

The items are arranged so that navigation starts in the upper left-hand corner with STUDENT_ID. The next navigation item is SALUTATION and from there navigation continues to flow vertically down through the items in the left-hand column. When the user tabs out of the PHONE item, the form will navigate to ADDRESS and continue through the right-hand column using the same vertical flow.

Occasionally, the Layout Wizard will lay items out in two columns when you select the Form style. The form will look similar to the one pictured in Figure 5.1. However, the navigation will go from left to right, from column to column. This can be a bit awkward as the cursor jumps in a jerky fashion from column to column. Therefore, it is sometimes necessary to rearrange items that have been positioned by the wizard so that navigation is smoother.

5.1.2 Answers

a) **How many GRADE_AVG items are displayed? Remember, GRADE_AVG must belong to the GRADE block.**

Answer: Five GRADE_AVG *items are displayed.*

When the Layout Wizard sets the Number of Records Displayed, it does it for all items in the block being created. Then, all of the items in the block inherit this value. Even though you created GRADE_AVG outside the wizard, it is still inheriting this property from the GRADE block.

If only one GRADE_AVG item is displayed, then you have created it in the ENROLL block by accident. What probably happened is that you did not set the value of the Block drop-down properly in the Layout Editor's Utility toolbar. If you created GRADE_AVG in the ENROLL block, delete it and try it again using the proper tools in the Layout Editor.

b) **Which property can you change so that the GRADE_AVG item is displayed only once, but the rest of the items in the GRADE block are displayed five times?**

Answer: You can change the GRADE_AVG *item's* Number of Items Displayed *property to* 1.

As you can see, by setting an individual item's Number of Items Displayed property, you can override the block-level value for that specific item, but you will not affect the other items in the block.

go to contents

c) Which of the Prompt properties should you change so that the `GRADE_AVERAGE` prompt is positioned like the prompt for `ENROLLMENT.SECTION_ID`?

Answer: `Attachment Offset`.

The properties under the Prompt category in the Property Palette allow you to set the position of the prompt relative to its item. You can attach the prompt to any of the item's four edges, set how far the prompt should be from the item, set how the prompt should be aligned to the item, and so on.

At design-time, you can set the values for `Prompt Alignment Offset` and `Prompt Attachment Offset` in the Property Palette and Layout Editor. However, `Prompt Attachment Edge`, `Prompt Alignment`, `Prompt Justification`, and the rest of the properties in the Prompt category can only be changed in the Property Palette.

The beauty of these Prompt properties is that they will not change if you reposition an item. What this means is that if you drag an item from one position on the canvas to another, its prompt will be dragged along with it. Not only will it accompany the item across the canvas, but its position relative to the item will stay the same. This saves you from having to reposition the prompt every time you reposition an item.

d) Which category of properties will you work with for `GRADE_AVG` to make it a calculated item?

Answer: Calculation.

e) How do you think you should set the `Calculation Mode` and `Summary Function` properties?

Answer: `Calculation Mode` *should be set to* `Summary` *and* `Summary Function` *should be set to* `AVG`.

A calculated item gets its value from either an existing summary function, like `AVG` or `SUM`, or from a formula. Summary functions are convenient because like pre-existing database functions, the mathematical expression is already written for you.

In this example, since you wish to calculate the average of the grades for each enrollment, the `Calculation Mode` property should be set to `Summary` and the `Summary Function` property should be set to `AVG`.

go to contents

In another example, you may wish to determine the total cost of all the courses taken by a single student. In that case, you would set `Calculation Mode` to `Formula` and write your own PL/SQL expression in the `Formula` property.

■ FOR EXAMPLE:

Assume you created an item called `NO_OF_ENROLL` that contained the number of students enrolled in a given section. You included `NO_OF_ENROLL` in a `SECTION` block. Now you want to create another item called `SEATS_LEFT` in the `SECTION` block. In this item you want to display the number of seats remaining in the section. You would calculate this by subtracting `SECTION.NO_OF_ENROLL` from `SECTION.CAPACITY`.

In the properties for the `SEATS_LEFT` item, you would set `Calculation Mode` to `Formula` and you would write the following expression in the `Formula` property:

```
:SECTION.CAPACITY - :SECTION.NO_OF_ENROLL
```

At run-time, the results of this formulaic expression would be displayed in the `SEATS_LEFT` item.

f) Which item in which block will you be summarizing? Set the associated properties accordingly.

Answer: You will be summarizing `GRADE.NUMERIC_GRADE`*. Therefore,* `Summarized Block` *should be set to* `Grade` *and* `Summarized Item` *should be set to* `Numeric Grade`.

g) What should the data type of the `GRADE_AVG` item be? Why?

Answer: The `Data Type` *property should be set to* `NUMBER` *since the calculation will produce a number value.*

If you leave `GRADE_AVG` as `CHAR`, the form will return an error.

h) Why do you think you had to set this property?

Answer: So that the average is computed for all of the records in the query's result set.

By default, Forms does not always return all of the records in a result set to the form at once. Each block has a `Query Array Size` property, which determines how many records will be fetched from the database at a time. The default value of this property is set by the `Number of Records Displayed` property. What this means is that by default, the number of records returned to a block is equal to the number of records displayed on

go to contents

the canvas. For instance, in the `GRADE` block, only 5 records are displayed on the canvas, so the `Query Array Size` property is also set to `5`.

What if 10 or 20 records are part of the result set? Forms would not be able to calculate the average correctly because not all of the values would be returned. Therefore, when creating summary items, it is necessary to set the `Query All Records` property to `Yes`. By doing so, the `Query Array Size` property is overridden, all records are returned to the form, and the average is computed accurately.

In this example, it is safe to return all of the records to the form because the result sets are rather small. However, there may be cases in which the result set could be rather large. This would mean that setting `Query All Records` to `Yes` could possibly degrade the performance of the application. In these cases, you would set the properties for the `GRADE` block a bit differently. First, you would set `Query All Records` to `No`. Then, under the Advanced DML category in the Property Palette, you would set the `Pre Compute Summaries` property to `Yes`. This will not return all of the records to the block at once. Instead, the number of records returned to the block will be based on the number set by the `Query Array Size` property. But won't this mean that the average for the calculated item will be incorrect? Not in this case, because Forms will issue a second query to figure out the average. This way, the average is being computed by the database and then returned to the form. The value will be correct and the form will not have to fetch all of the rows from the database.

LAB 5.1 SELF-REVIEW QUESTIONS

In order to test your progress, you should be able to answer the following questions:

1) What is the major difference between a display item and a text item?
- **a)** ____ Text items can display database values, display items cannot
- **b)** ____ Text items are always database items while display items are always non-database items
- **c)** ____ Text items are navigable and editable, display items are not
- **d)** ____ Text items and display items are identical except for their background color

2) Why would you set the `Multi Line` property?
- **a)** ____ To display items on the canvas
- **b)** ____ To display more than one record in a single item
- **c)** ____ To display more than one line of text in a single item
- **d)** ____ None of the above

go to contents

3) The `Format Mask` property allows you to display values in the form in a different format than they are stored in the database.

a) ____ True
b) ____ False

4) Which of the following must be true for an item to be visible?

a) ____ `Enabled` must be set to `Yes`
b) ____ It must be a database item
c) ____ The `Width` property must not exceed the length of the column in the database
d) ____ It must be assigned to a canvas with the `Visible` property set to `Yes`

5) How can you view the properties for a prompt?

a) ____ By selecting the prompt in the Object Navigator
b) ____ By viewing the properties for the prompt's item
c) ____ You can't because prompts don't have properties
d) ____ a & b

6) Which properties can you set to prevent a user from changing the value in an item?

a) ____ Set `Item Type` to `Display Item`
b) ____ Set `Enabled` to `No`.
c) ____ Set `Update Allowed` and `Insert Allowed` to `No`
d) ____ All of the above

Quiz answers appear in Appendix A, Section 5.1.

LAB 5.2

BUTTONS, LIST ITEMS, RADIO GROUPS, AND CHECK BOXES

LAB OBJECTIVES

After this Lab, you will be able to:

- Create Buttons
- Put Simple Code Behind Buttons
- Create List Items
- Create Radio Groups
- Create Check Boxes

The items in this Lab are different from text items and display items in that they don't simply display text to communicate their value. Instead they employ different combinations of text and graphics to display information or provide a function. As you may have noticed, there are many other item types available in Forms such as images, sounds, ActiveX Controls, and so on. In this Lab and in the rest of this book, you will focus on buttons, list items, radio groups, and check boxes. These, along with text and display items, are by far the most common Forms item types. For a complete list of all the different item types, simply look at the `Item Type` property in the Property Palette and refer to the help system for details on how to implement them.

go to contents

BUTTONS

Buttons give users opportunities to make a form do something. This could be something simple like saying "OK," or it could be something more involved like executing a query or even opening another form. Creating and positioning buttons in Forms is easy. What's challenging is writing the code that goes behind them. As you already know, each button should have a `WHEN-BUTTON-PRESSED` trigger associated with it so that it can respond to the Button Pressed event. In the Exercises, you will use the help system to locate Forms built-ins to put behind your buttons.

LIST ITEMS, RADIO GROUPS, AND CHECK BOXES

List items, radio groups, and check boxes look very different, but they are similar to each other in that they present the user with a number of choices. The choices they present are often more understandable representations of the data or information stored behind them.

■ *FOR EXAMPLE:*

The list item in Figure 5.2 shows three choices for the `SECTION` table's `LOCATION` column. The labels are `Lecture Hall One`, `Lecture Hall Two`, and `Lab One`. But, they may not necessarily be this way in the database. The column values could be `L210`, `L500` and `L510`, respectively. The list has been configured so that the user sees values that are more meaningful to them.

What this illustrates is that radio groups, list items, and check boxes allow you to display information in any way you'd like, regardless of how the values are stored in the database. When you configure these items, you define the values you'd like to display, along with how they should be represented in the database.

These three types of items can also handle data that they are not designed to expect.

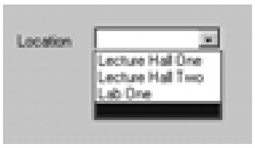


Figure 5.2 ■ The list item elements are not necessarily how the information is stored in the database.

go to contents

■ FOR EXAMPLE:

At design-time, the `Location` list item in Figure 5.2 could be configured to handle the database values L210, L500, and L510. When the user issues a query, the list item would expect that one of these three values would be returned to the form. But, what would happen if the user issued a query and the database returned the value L999? The list item is not expecting this value and would not know what to do with it. Luckily there is a list item property called `Mapping of Other Values` that can be set to handle this situation. Radio groups and check boxes have a similar property. In the Exercises in this Lab, you will explore the `Mapping of Other Values` property in more detail.

LIST ITEMS

List items are exactly what their name implies—a short list of values that the user can choose from.

List items can serve as either database or non-database items. The list item you create in the Exercise will be based on the `COST` column in `COURSE` table. You can also use list items for selection lists that are not bound to columns in the database but instead act as controls. The Form Builder makes use of list items in this way in the Layout Editor's Utility toolbar as shown in Figure 5.3.

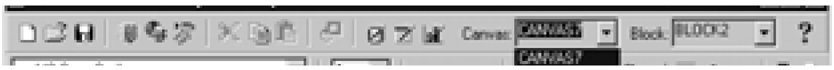


Figure 5.3 ■ List items in the Layout Editor's toolbar.

RADIO GROUPS

Radio groups are logical containers of radio buttons. Radio buttons are the small circles that appear on the screen which the user can click to select values. Radio group selections are mutually exclusive. That is, selection of one radio button deselects the previously selected button.

The radio group you will build in the Exercises will be a database item. However, radio groups can also be non-database items.

■ FOR EXAMPLE:

You often see radio groups used in the Form Builder itself to let the user make choices. Figure 5.4 shows the `New Data Block` dialog with a radio group with two radio buttons.

go to contents

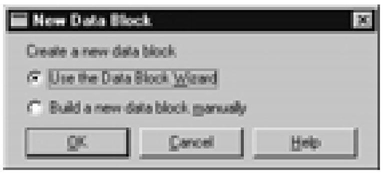
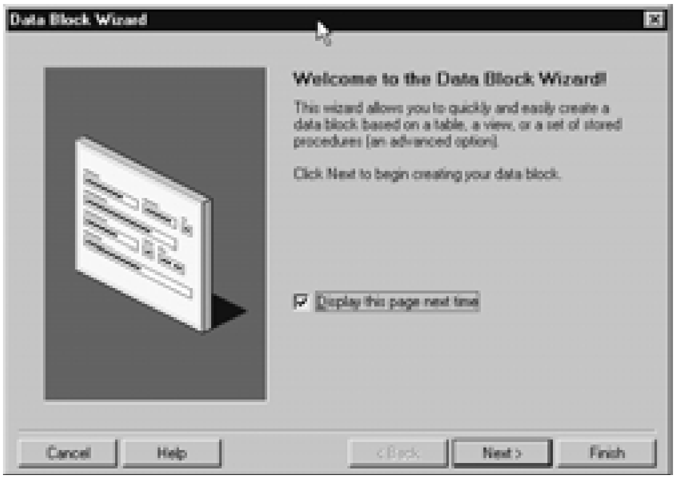


Figure 5.4 ■ The `New Data Block` dialog with a radio group.

In this case, the radio group is acting as a control, much like a button item acts as a control. It gives a user the opportunity to choose how the application should behave. If you were to employ such a radio group in one of your Forms applications, you would have to write triggers to change the behavior of the form depending on which radio button was selected.

CHECK BOXES

Check boxes are useful for storing Yes/No, True/False, and On/Off-type values. Like radio groups and list items, they can also serve as both database items and non-database items. In the Form Builder, there are multiple examples of check boxes as non-database items. Figure 5.5 shows a check box on the welcome page of the Data Block Wizard.


Data Block Wizard
Welcome to the Data Block Wizard!
Display this page next time
Cancel
Help
< Back
Next >
Finish

This wizard allows you to quickly and easily create a data block based on a table, a view, or a set of stored procedures (an advanced option).

Click Next to begin creating your data block.

Figure 5.5 ■ The welcome page of the Data Block Wizard, which includes a check box.

go to contents

LAB 5.2 EXERCISES

5.2.1 CREATE BUTTONS

Open the R_COURSE.fmb form that you created previously.

Create a block manually and name it CONTROL. The form should resemble Figure 5.6, but does not have to match it exactly.

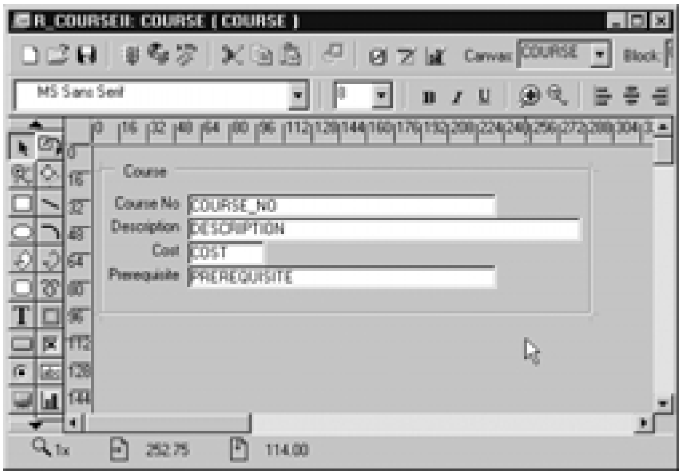
Create three buttons using the Layout Editor's Tool Palette.

a) What should you do when you select from the Tool Palette to make the creation of the buttons easier?

---

b) What should you set on the Layout Editor's toolbar to make sure that these buttons are assigned to the CONTROL block?

Rename the buttons Clear, Save, and Exit.



COURSE_NO
DESCRIPTION
COST
PREREQUISITE

Figure 5.6 ■ Suggested layout for R_COURSE.fmb.

go to contents

c) Edit the properties so that the text on the button matches its name in the Object Navigator. Which property did you use?

Run the form.

d) Which item has the Forms Runtime navigated to (where is the cursor)? Why is Forms doing this?

Exit the running form and return to the Form Builder.

e) What should you do to prevent the button from being navigated to first?

Save `R_COURSE.fmb` as you will need it in the next Exercise.

5.2.2 PUT SIMPLE CODE BEHIND BUTTONS

Open `R_COURSE.fmb` in the Form Builder. Right-click the `Exit` button in either the Object Navigator or Layout Editor and explore the Smart Triggers.

a) Which trigger should you create to respond to the event of a user clicking this button?

Select this trigger.

b) What is the title of the window that has just opened and what should you do in it?

You will need to use a Forms built-in to exit the form. Use the help system to determine which one you need. From the Form Builder's Main Menu, select `Help | Form Builder Help Topics`. Click the `Index` tab and type the word `Exit` into the first field.

c) What is the name of the built-in you should use to exit the form?

Type the code for the built-in into the PL/SQL Editor. Put a semi-colon after the built-in so that it compiles correctly. Click the PL/SQL Editor's `Compile` button. Close the PL/SQL Editor. Run the form and test the `Exit` button.

Create the same trigger for the `Clear` button using the `CLEAR_BLOCK` built-in as the trigger code. Click the PL/SQL Editor's `Compile` button.

The `Save` button on this form is not for saving files, but for saving changes a user has made by inserting or updating records to the database. With this in mind, answer the following questions.

d) What is the command for saving updated or inserted records to the database?

Based on the answer to the previous question, search the help system again.

e) Which built-in should you use to "save" the changes a user has made?

go to contents

Type the code for the built-in into the PL/SQL Editor and click `Compile`. Close the PL/SQL Editor. Run the form and issue a query. Test the `Save` button by making a change to one of the records returned. Test the `Clear` button as well. Save the `R_COURSE.fmb` form for future Exercises.

Read the answers and discussions for Exercises 5.2.1 and 5.2.2 before continuing to the next Exercise.

5.2.3 CREATE LIST ITEMS

Use the `R_COURSE.fmb` form for the following questions. For Question a, consider the following information about the columns in the `COURSE` table:

`course_no` is numeric and a unique identifier of each course.

`course_description` is a textual description of each course. Each course has a different description.

`course_cost` is a numeric cost of each course. One of the STUDENT business rules is that there are only three possible costs for a course.

`course_prerequisite` is a numeric column based on `course_no`. It indicates the course that must be taken as a prerequisite.

a) Which text item(s) in the `COURSE` block would be better displayed as a list item(s)?

Change the `COURSE.COST` item's `Item Type` property to `List Item`. Use the `Freeze/Unfreeze` button you learned about in Chapter 3, "The Development Environment," to compare the Functional properties of `COURSE.COST` to those of `COURSE.COURSE_NO`.

b) What list-related properties have been added to the Functional category of `COURSE.COST`?

go to contents

c) What happens when you click the `Elements in List` property and then click the `More` button?

d) If list elements are what appear to the user, what are list item values used for?

e) Can a list element be different from its list item value? When and why might you want them to be different?

Assume that there are only three possible costs for courses in the `STUDENT` database (`1095`, `1195`, `1595`). Use these values to populate the `List Element` and `List Item Value` fields.

f) Can you format list elements with dollar signs and commas?

g) Which Data property would you set so that a list item would start at 1595 during an insert?

Run the form and query Course Number 100.

h) Can you change the `COST` to 2000?

go to contents

Exit the form and return to the Form Builder. Change the COST item's List Type property to Combo Box. Run the form and query Course Number 100 again.

i) Can you change the COST to 2000 now?

j) Can you change the COST to $2,000 *and* save the change? Why not?

Exit the form and return to the Form Builder. Change the COST item's List Type property to Tlist.

k) Does COST still look like a list item? What does it look like now?

Change the COST item's Height property to 34. Run the form.

l) What has happened to COST? How many of its elements are visible?

m) If there were 15 values for the COST item, would it still be appropriate to use a list item? What if there were 20? 30?

Read the answers and discussions for Exercise 5.2.3 before continuing to the next Exercise.

go to contents

5.2.4 Create Radio Groups

Use the `R_COURSE.fmb` form for the following questions. Before you begin, take the following steps to organize the canvas.

1) Change the `COST` item's `Height` property to `14`.
2) Reposition `COST` after `PREREQUISITE` in the Object Navigator.
3) Update the layout for the `COURSE` frame by using the `Update Layout` button on the Layout Editor.
4) Reposition the buttons if they are in the way.

In this Exercise, you will use the `COST` item again, but this time you will implement it as a radio group item.

Change the `COST` item's `Item Type` property to `Radio Group`.

a) Is `COST` still visible on the `COURSE` canvas? Is it still in the `COURSE` block? Why isn't it still visible on the canvas?

b) What are the nodes under the `COST` item in the Object Navigator?

c) Can you create radio buttons in the Object Navigator? Why might it be easier to create them in the Layout Editor?

Select the `Radio Button` tool from the Layout Editor's Tool Palette and create three radio buttons. Use the pinning feature you learned in Chapter 3, "The Development Environment," to make this easier. Update the layout when you are done. Change the background color of the buttons to match the canvas.

Readjust the position of the `Save`, `Clear`, and `Exit` buttons if they are in the way.

go to contents

d) What else did you have to decide in addition to selecting the position of the radio buttons?

Assume that there are only three possible costs for courses in the STUDENT database (`1095`, `1195`, `1595`). Rename the radio buttons in the Object Navigator.

e) Are you able to rename them `1095`, `1195`, `1595`? Why?

f) Which radio button property should you change so that `$1,095`, `$1,195`, and `$1,595` appear next to each radio button respectively?

g) Which radio button property should you use to indicate its value?

Assume that a query returns a value to the `COST` radio group that is not one of its pre-defined values.

h) What Functional radio group property can handle this situation?

i) Does the `COST` radio group have a property for labeling or titling the group?

go to contents

j) Can you create a graphic object to indicate to the users that this group of radio buttons describes the course's cost? Which ones can you use?

k) If there were 6 values for the `COST` item, would it still be appropriate to use a radio group? What if there were 10? 20?

5.2.5 CREATE CHECK BOXES

Use the wizards to quickly create a new form based on the `Grade_Type_Weight` table. Include the audit columns in the block, but do not display them on the canvas. `Enforce data integrity` should be **unchecked.** Choose `Form` as your layout style. Rename the canvas and its frame `GRADE_TYPE_WEIGHT`.

The purpose of this Exercise is to learn about check boxes, so do not be overly concerned if the layout is unattractive.

Change the `GRADE_TYPE_WEIGHT.DROP_LOWEST` item's property to `Check Box`.

a) Which of the `DROP_LOWEST` item's Functional properties allow you to set the value that will be inserted or updated to the database?

b) If the allowable values for `DROP_LOWEST` are `Y` and `N` (Yes and No), what should you enter for `Value When Checked`? `Value When Unchecked`? Why?

c) What are the three values of the Check Box Mapping of Other Values property? Give a brief description of what you think each means.

Set the DROP_LOWEST item's Check Box Mapping of Other Values property to Not Allowed. Run the form.

d) What was the error you received just before the Forms Runtime started? If you were unable to see the error message, minimize the Forms Runtime and navigate to the Form Builder.

Set the DROP_LOWEST item's Check Box Mapping of Other Values property to Checked and run the form to confirm that the error has been corrected.

Set the DROP_LOWEST item's Width property to 90. Set the DROP_LOWEST item's Label property to Drop Lowest Grade.

e) What is the difference between the DROP_LOWEST item's Prompt property and its Label property?

f) Are there any check box-related triggers? Check the Smart Triggers to find your answer.

LAB 5.2 EXERCISE ANSWERS

5.2.1 ANSWERS

a) What should you do when you select from the Tool Palette to make the creation of the buttons easier?

Answer: Double-click the `Button` *tool to pin it.*

b) What should you set on the Layout Editor's toolbar to make sure that these buttons are assigned to the `CONTROL` block?

Answer: Set the `Block` *list item on the Layout Editor's Utility toolbar to* `CONTROL`.

c) Edit the properties so that the text on the button matches its name in the Object Navigator. Which property did you use?

Answer: You should have used the `Label` *property.*

The `Label` property positions text on the button itself. The button also has a `Prompt` property that, as with other items, will position text somewhere next to the item. You should always use labels for buttons. However, these labels don't always have to be textual; it is possible to put icons on buttons as well.

■ FOR EXAMPLE:

Select the `Save` button in `R_COURSE.fmb` and locate its `Iconic` property. Change `Iconic` to `Yes`, and change `Icon Name` to `Save`. Look at the `Save` button in the Layout Editor and note how the text has been replaced by an icon. If you were to use this icon, you would obviously have to adjust the size and position of the button to make it look better. But for now, simply leave it as it is.

When you installed Oracle Developer, a number of sample icon files were installed along with it. They should be in your `\ORACLE_HOME\tools\devdem60\bin\icon` directory. The Form Builder and Forms Runtime are able to find these icon files because an entry in the Windows Registry points to this directory. For more information about the Registry, see Appendix B.

d) Which item has the Forms Runtime navigated to (where is the cursor)? Why is Forms doing this?

Answer: Forms has navigated to the `Save` *button because it is listed first in the Object Navigator.*

go to contents

Exit the running form and return to the Form Builder.

e) What should you do to prevent the button from being navigated to first?

Answer: You should reposition the `CONTROL` *block after the* `COURSE` *block.*

This will prevent the form from navigating to these buttons before it navigates to the enterable items in the `COURSE` block.

You should also set the `Mouse Navigate` button to `No`. Users will still be able to click the buttons, but the cursor will not rest on a button; it will return to the previous item. If the cursor can rest on a button, it may prevent the user from performing some operations. A form, for instance, will not allow a user to issue a query if the cursor is resting on a button.

5.2.2 Answers

a) Which trigger should you create to respond to the event of a user clicking this button?

Answer: The `WHEN-BUTTON-PRESSED` *trigger.*

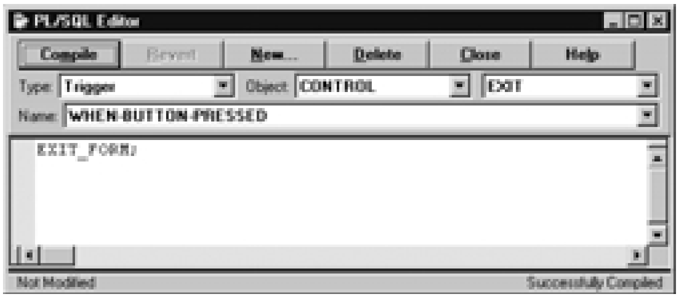
b) What is the title of the window that has just opened and what should you do in it?

Answer: The PL/SQL Editor.

The PL/SQL Editor is the tool you will use to write trigger and procedure code in Oracle Forms. It will be discussed in more detail in Chapter 6, "Triggers & Built-ins."

c) What is the name of the built-in you should use to exit the form?

Answer: You should use the `EXIT_FORM` *built-in.*



Type: Trigger   Object: CONTROL   EXIT
Name: WHEN-BUTTON-PRESSED

```
EXIT_FORM;
```

Figure 5.7 ■ The PL/SQL Editor with the `EXIT_FORM` built-in. Note the semi-colon following the built-in.

go to contents

Forms built-ins, as stated before, will be discussed in Chapter 6. For now, simply type the built-in into the PL/SQL Editor and click the `Compile` button. If you are having problems, refer to Figure 5.7 to see how the code should look in the PL/SQL Editor.

d) What is the command for saving updated or inserted records to the database?

Answer: You commit updated or inserted records to the database.

e) Which built-in should you use to "save" the changes a user has made?

Answer: The `COMMIT_FORM` *built-in.*

Type the code for the built-in into the PL/SQL Editor and click `Compile`. Close the PL/SQL Editor. Run the form and issue a query. Test the `Save` button by making a change to one of the records returned. Test the `Clear` button as well. Save the `R_COURSE.fmb` form for future Exercises.

5.2.3 Answers

a) Which text item(s) in the `COURSE` block would be better displayed as a list item(s)?

Answer: `COURSE.COST` *might be better displayed as a list item because it has only three allowable values.*

b) What list-related properties have been added to the Functional category of `COURSE.COST`?

Answer: `List`, `List Style`, *and* `Mapping of Other Values` *have been added to the Functional category.*

c) What happens when you click the `Elements in List` property and then click the `More` button?

Answer: You are presented with a `List Elements` *dialog box.*

This is the main tool for configuring a list item.

d) If list elements are what appear to the user, what are list item values used for?

Answer: The list item values are how each item in the list is represented in the database.

Each element in a list is represented in two ways: to the user as a list element and to the database as a list item value. Every list element must have a corresponding list item value. The `List Elements` dialog box lets you set both of these values.

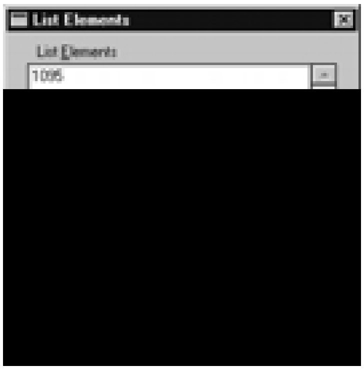


Figure 5.8 ■ The completed `List Elements` dialog box for `COURSE.COST`.

■ *FOR EXAMPLE:*

For the list item `COURSE.COST`, you will have three list elements: `1095`, `1195`, and `1595`. The list elements are what the user will see in the list item at run-time. These list elements will have corresponding list item values of 1095, 1195, and 1595, respectively. The list item values are what will be inserted or updated to the database when the user makes a change. Figure 5.8 shows the completed `List Elements` dialog box for `COURSE.COST`.

Note that in Figure 5.8, the `1195` list element is selected, so its corresponding value appears in the `List Item Value` field. The values in both fields are the same now, but as you move through the Exercises, they will change.

e) Can a list element be different from its list item value? When and why might you want them to be different?

Answer: Yes.

The ability to use list items to represent elements one way and then send and retrieve their values from the database in another is what makes list items convenient and powerful.

■ *FOR EXAMPLE:*

The `GRADE_TYPE` table contains a `GRADE_TYPE_CODE` column. The values in this column are `FI` for final, `QZ` for quiz, `HW` for homework, and so on. If you were to create a list item based on the `GRADE_TYPE_CODE` column, it would make sense to represent the values differently from how they are stored in the database. For example, you would have Final as a list element with FI as its corresponding list item value. Refer to Figure 5.9 to see how this would look in the `List Elements` dialog box.

go to contents

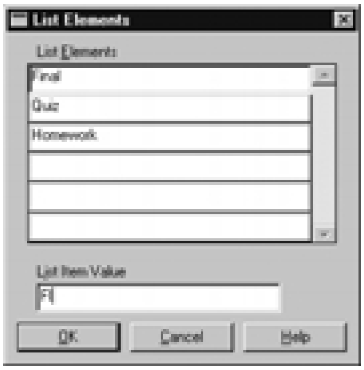


Figure 5.9 ■ List elements with different list item values. Note that `Final` is selected.

The Functional category has the `Mapping of Other Values` property. As discussed in the Lab text, its purpose is to handle values that are returned from the database that don't match any of those entered as list item values. That is, if a value that is not one of the list item values is returned from the database, it will be mapped to the value indicated in the `Mapping of Other Values` property.

■ *FOR EXAMPLE:*

Assume that in the `GRADE_TYPE_CODE` example, `FI`, `QZ`, and `HW` are the only values entered as list item values. Also assume that the `Mapping of Other Values` property is set to `QZ`. What would appear in the list item if the user queried a record in the `GRADE_TYPE` table that had `TEST` as the value for `GRADE_TYPE_CODE`? `TEST` does not match any of the list item values. But, since `Mapping of Other Values` was set to `QZ`, the list item would display `QZ`'s corresponding list element. Then, when this record is resaved to the database, the column will be updated with the `QZ` value.

f) Can you format list elements with dollar signs and commas?

Answer: Yes you can.

This is the same as changing FI to Final in the answer discussion for Question e. The list element does not have to match its list item value and can contain any combination of characters.

g) Which Data property would you set so that a list item would start at 1595 during an insert?

Answer: You would set the `Initial Value` *property to* `1595`.

If the user is creating a new record, the value in the list item will initially be set to `1595`. The value in the `Initial Value` property must be one of the list item values that were entered in the `List Elements` dialog.

h) Can you change the `COST` to 2000?

Answer: No you can't. There is no way to add values that are not on the list.

i) Can you change the `COST` to 2000 now?

Answer: Yes you can.

There are three types of list items: poplists, Tlists, and combo boxes. In Questions a through g, you worked with poplists, which are the default type. They restrict the user from selecting only from the list elements that are displayed.

A combo box is nearly identical to a poplist in look and functionality except that it allows users to enter values that are not defined on the list. For example, in this case you were able to change the `COST` to 2000, even though it was one of the list elements.

j) Can you change the `COST` to $2,000 *and* save the change? Why not?

Answer: No, because `COST` *is a numeric column in the database and will not accept the dollar sign.*

k) Does `COST` still look like a list item? What does it look like now?

Answer: No, it looks like a regular text item with a miniature scrollbar.

l) What has happened to `COST`? How many of its elements are visible?

Answer: All of its elements are visible in a list.

m) If there were 15 values for the `COST` item, would it still be appropriate to use a list item? What if there were 20? 30?

Answer: Yes, it would still be appropriate with 15 values, but not 20 or 30.

A good rule of thumb is to limit the number of values in a list item to 15. If the list grows much larger than 15, you should consider using a List of Values object, which you will learn about in Chapter 7, "LOVs and Alerts." For more information on the design concerns for list items, refer to the *Oracle Developer 6.0 On-line Manuals* in the section titled "Designing Effective GUI Applications."

5.2.4 ANSWERS

a) Is COST still visible on the COURSE canvas? Is it still in the COURSE block? Why isn't it still visible on the canvas?

Answer: No, it is not visible on the COURSE *canvas. Yes, it is still in the* COURSE *block.*

A radio group is a logical container of radio buttons. Therefore, the group itself is not represented on the canvas because it has no physical properties. For this radio group, you will create radio buttons that have physical properties and are visible on the canvas. The radio buttons serve as a visual representation of the choices in the radio group. In this Exercise, you have already created a radio group called COURSE.COST. Soon you will create three radio buttons to represent the values 1095, 1195, and 1595.

b) What are the nodes under the COST item in the Object Navigator?

Answer: The Triggers *and* Radio Buttons *nodes are under the* COST *item.*

The Radio Button node is positioned here in the Object Navigator's hierarchy so that the Form Builder will know which radio buttons are logically contained by which radio groups.

c) Can you create radio buttons in the Object Navigator? Why might it be easier to create them in the Layout Editor?

Answer: Yes, you can create them in the Object Navigator.

However, it would be easier to create them in the Layout Editor so that you can position them as you create them.

d) What else did you have to decide in addition to selecting the position of the radio buttons?

Answer: You had to select which radio group they would be assigned to by using the Radio Groups *dialog.*

In this case, you only had one radio group created so the choice was simple. If you had created more than one radio group in this form, they would have been listed here. Also note that you could have created a new radio group using this dialog.

e) Are you able to rename them 1095, 1195, 1595? Why?

Answer: No, because Forms will not accept numbers as names of objects.

go to contents

You must prefix the numbers with something like `COST_` or `RB_` if you would like the numbers to be included in a radio button's name.

f) Which radio button property should you change so that `$1,095`, `$1,195`, and `$1,595` appear next to each radio button respectively?

Answer: You should use the `Label` *property.*

Radio buttons are similar to button items in that they have both `Label` and `Prompt` properties. Again, it is customary to use `Label` for radio buttons rather than `Prompt`.

g) Which radio button property should you use to indicate its value?

Answer: You should use the `Radio Button Value` *property.*

h) What Functional radio group property can handle this situation?

Answer: The `Mapping of Other Values` *property.*

If the database returns a value that does not match one of the values for one of the radio buttons, the form will return an error. However, if the `Mapping of Other Values` property is set, the unmatched value will be handled by the form.

■ FOR EXAMPLE:

Assume that the radio button values are set to `1095`, `1195`, and `1595`. Also assume that the `Mapping of Other Values` property is set to `1095`. If the database returns a value of `9999` to the form, then the `1095` radio button will be selected as is indicated in the `Mapping of Other Values` property. When the user commits changes to this record, the `COST` column will be updated to `1095`.

i) Does the `COST` radio group have a property for labeling or titling the group?

ANSWER: No, it does not.

j) Can you create a graphic object to indicate to the users that this group of radio buttons describes the course's cost? Which ones can you use?

Answer: You can use text or a frame.

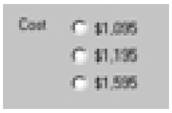


Figure 5.10 ■ The `Cost` radio group with a textual label.

go to contents

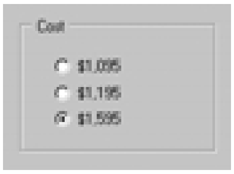


Figure 5.11 ■ The `Cost` radio group with a titled frame.

You can use the Tool Palette to create a textual label for the radio group as in Figure 5.10, or you can create a frame with a title around the radio group as in Figure 5.11.

In this example, the textual label probably makes more sense since the inclusion of a frame under the rest of the `COURSE` block's items would look strange.

k) If there were 6 values for the `COST` item, would it still be appropriate to use a radio group? What if there were 10? 20?

Answer: Yes, it would still be appropriate with 6 values, but not with 10 or 20.

A good rule of thumb is to have no more than six radio buttons in a radio group. If there are going to be more than six, you might want to consider using a list item, which provides similar functionality.

Radio groups are convenient because all of their choices are always visible. There is no need to display or click a radio group to view its choices like there is with a list item or a list of values. But this comes with a cost in that radio groups require a lot of real estate on the canvas. You must consider the space available on the form and the number of other items you wish to display before determining whether or not a radio group is appropriate.

5.2.5 ANSWERS

a) Which of the `DROP_LOWEST` item's Functional properties allow you to set the value that will be inserted or updated to the database?

Answer: The `Value When Checked` *and* `Value When Unchecked` *properties.*

The check box has two states, checked and unchecked, that correspond to the values Yes/No, True/False, and On/Off. In the checked state, the user will see a small check mark in the check box. In the unchecked state, the check box will be empty.

b) If the allowable values for `DROP_LOWEST` are `Y` and `N` (Yes and No), what should you enter for `Value When Checked`? `Value When Unchecked`? Why?

go to contents

Answer: `Y` *for* `Value When Checked`, `N` *for* `Value When Unchecked`.

A Forms check box is analogous to the check boxes you have seen on paper-based forms. On paper-based forms, you often see check boxes for things like "Add me to your mailing list," "Insured?" "Criminal Record?", and so on, which require that you check the box if you meet the condition stated. The behavior is the same in Forms. The `Value When Checked` property typically corresponds to the positive or affirmative state, while `Value When Unchecked` corresponds to the negative state. So in this case, by checking `DROP_LOWEST`, the user is indicating "Yes, the lowest grade for this type should be dropped." Forms will insert a `Y` into the `DROP_LOWEST` column for this record.

c) What are the three values of the `Check Box Mapping of Other Values` property? Give a brief description of what you think each means.

Answer: `Checked`, `Unchecked`, *and* `Not Allowed`.

The `Check Box Mapping of Other Values` is similar to the `Mapping of Other Values` function in list items and radio groups. It is used to determine how to handle those values returned from the database that do not match either of the values you have assigned to the check box in the `Value When Checked` or `Value When Unchecked` properties.

■ *FOR EXAMPLE:*

Assume that for the `DROP_LOWEST` check box, the `Value When Checked` property and `Value When Unchecked` property are set to `Y` and `N` respectively. Also assume that a user using a different application has inserted an "`M`" for Maybe into the `DROP_LOWEST` column for one of the records in the `GRADE_TYPE_WEIGHT` table. When a user queries this record, the form will behave in the following ways depending on how the `Check Box Mapping of Other Values` property is set:

1) `Checked`—The check box will be set to the checked state when `M` is returned.
2) `Unchecked`—The check box will be set to the unchecked state when `M` is returned.
3) `Not Allowed`—The form will not be able to retrieve the record. An FRM-40301 error will be returned and appear with a message in the hint line.

In Situations 1 and 2, the form will accept the `M` value and let the user continue processing the record even though it does not match either of the values specified in the `Value When Checked` or `Value When`

Unchecked properties. When the record is saved, the DROP_LOWEST column will be updated with the value in the Check Box Mapping of Other Values property. In Situation 3, the form will reject the M value and not allow the user to process the record.

d) What was the error you received just before the Forms Runtime started? If you were unable to see the error message, minimize the Forms Runtime and navigate to the Form Builder.

Answer: The message reads "FRM-30188: No initial value given, and other values are not allowed (item GRADE_TYPE_WEIGHT .DROP_LOWEST)."

If you decide to set Check Box Mapping of Other Values to Not Allowed, then you must specify an Initial Value for the check box. This Initial Value must match one of the values specified in the Value When Checked or Value When Unchecked properties.

e) What is the difference between the DROP_LOWEST item's Prompt property and its Label property?

Answer: The Prompt *property's value appears to the left of the item, while the* Label *property's value appears to the right.*

As with button items and radio buttons, it is customary to use the Label property rather than the Prompt property.

f) Are there any check box-related triggers? Check the Smart Triggers to find your answer.

Answer: Yes, there is a WHEN-CHECKBOX-CHANGED *trigger.*

The following is optional. To test the behavior of the check box as it responds to a WHEN-CHECKBOX-CHANGED trigger, take the following steps:

1) Right-click on the DROP_LOWEST item.
2) Navigate to Smart Triggers and select the WHEN-CHECKBOX-CHANGED trigger.
3) The PL/SQL Editor will open. Enter the following code:

```
IF GRADE_TYPE_WEIGHT.DROP_LOWEST = 'Y' THEN
    MESSAGE('The lowest grade will be dropped');
ELSE
    MESSAGE('The lowest grade will not be dropped');
END IF;
```

go to contents

4) Run the form and issue a query.
5) Toggle `DROP_LOWEST` and watch the Forms Runtime's status line.

Lab 5.2 Self-Review Questions

In order to test your progress, you should be able to answer the following questions:

1) How can you communicate the purpose of a button?
a) ____ With a label
b) ____ With a prompt
c) ____ With an icon
d) ____ a & c

2) How do you make a radio group visible?
a) ____ You set the `X Position` and `Y Position` properties for the group
b) ____ You create radio buttons for the group and make them visible
c) ____ You use the Layout Wizard to position the group
d) ____ None of the above

3) Which type of list item will allow the user to enter values that are not displayed in the list?
a) ____ Combo box
b) ____ Poplist
c) ____ Tlist
d) ____ List of values

4) Radio group selections are mutually exclusive.
a) ____ True
b) ____ False

5) Which of the following items can have triggers associated with them?
a) ____ Radio groups
b) ____ List items
c) ____ Check boxes
d) ____ All of the above

6) Which of the following is true about the `Mapping of Other Values` property for a list item?
a) ____ It will handle the values that are returned to the form but not specified in the list item values
b) ____ It will prevent the user from entering her own items to the list
c) ____ It must be set to one of the values that are specified as list item values
d) ____ a & c

Quiz answers appear in Appendix A, Section 5.2.

go to contents

CHAPTER 5

TEST YOUR THINKING

1) Many Windows applications provide "bubble help" or Tool Tips for fields and buttons. How can you do the same in Forms? Add Tool Tips to all the items in the R_COURSE.fmb form. Can you add Tool Tips to buttons just as easily? What is the difference between a hint and a Tool Tip?

2) Use the wizards to create a block based on the GRADE_TYPE_WEIGHT table. Set GRADE_TYPE_CODE to be a list item. Use SQL*Plus to determine all of the distinct values for GRADE_TYPE and create one list element for each. Make sure the list elements make sense to the user and are not just simple codes. For example, if the value in the database is FI, then the list element should be Final.

3) Use SQL*Plus to query the system tables to find out all of the sequences that are defined for the STUDENT schema. Go through the forms R_STUDENT, R_INSTRUCTOR, R_CRSESECT.fmb, and R_STUDENRL.fmb to see which items could be set using sequences. Adjust the appropriate property so that these items are populated by a value from the sequence when a user tries to create a new record. NOTE: Do not do this for items that belong to blocks that cannot be queried.

   Go through the forms listed above, but this time, adjust the appropriate property so that all date fields are populated with the current date when a user tries to create a record. Search the help system if you need help.

go to contents