Unit 6.01 The ListBox Assignment

A list box is used to display a list of items. A scroll bar automatically appears in the list box if the number of list items exceeds the height of the list box.

A ListBox control has the properties:

* **Name** identifies a control for the programmer. It is good programming style to begin ListBox object name with lst.
* **Items** contains the buttons that is clicked to display the String Collection Editor dialog box where a set of strings is typed.
* **Sorted** is set to True to display the list items in alphabetical order. This property is available only in the Design window.
* **SelectedItem** is the selected list item. This property is available only at run time.
* **SelectedIndex** is the index of the selected list item. This property is available only at run time.

The items in a list box have an index value, with 0 being the index of the first item, 1 the index of the next, and so on. An index value of -1 means that none of the list items are selected.

A SelectedIndexChanged event procedure is sometimes coded for a list box. This procedure executes when a list item is clicked. The specific item clicked is determined with the SelectedItem or the SelectedIndex properties. The Select….Case statement belwo ises the SelectedItem property to determine the selected item:

Select Case Me.lstCourseLevels.SelectedItem

Case “Undergraduate”

decCreditHour = 75

Case ‘Graduate’

decCreditHour = 145

Case “Thesis and Dissertation”

decCreditHour = 150

End Select

The ListBox control class contains methods that can be used to display output at run time. Up to this point, labels have primarily been used to display output. However, a list box is a good choice for output that should be displayed as one item after another. The height of the list box is of little concern because a scroll bar is automatically displayed when the list gets longer than the box.

The Items.Add() mothod is used to add an item to a list at run time and takes the form:

lstControl.ItemsAdd(item)

1stControl is the name of the list Box object and Item is a value or string. The item is added to the end of the list, or in the proper position of the list is sorted. The following code demonstrates Items.Add():

For intNum = 2 To 20 Step 2

Me.1stByTwos.ItemsAdd(intNum & “x5=”&intNum\*5)

Next intNum

Note that the & was used for concatenation. The vbTab constant can also be used to form strings with consistent spacing.

The Items.Remove() method is used to delete a specified item form the list and takes the form:

lstControl.Items.Remove(item)

lstControl is the name of the list box object and Item is a value or string.

The Items.Clear() method is used to delete the contents of the list box and takes the form:

lstControl.Items.Clear()

lstControl is the name of the list box object.

**ListBox Application Assignment**

**I**n this assignment you will create a Tuition Calculator application.

1. **Create A New Project**
   1. Create a new Window application named Tuition Calculator.
2. **Create The Interface**
   1. Refer to the form below to add, position, and size objects. User the table below to set properties.

Tuition Calculator

Course Level

Calculate

Undergraduate

Graduate

Thesis and Dissertation

|  |  |  |
| --- | --- | --- |
| Object | Name | Text |
| Form1 |  | Tuition Calculator |
| Label1 | lblCourseLevelList | Course Level: |
| ListBox1 | lstCourseLevels |  |
| Label2 | lblTuition | Empty |
| Button1 | btnCalculate | Calculate |

* 1. Click the list box to select it and then click the Items property in the Properties window. Click the button to display a dialog box. Type the following items, pressing Enter after each except the last:

Undergraduate

Graduate

Thesis and Dissertation

1. **Write the Application Code**
   1. Display the code window.
   2. Add comments that include your name and today’s date.
   3. Create a btnCalculate\_Click event procedure and then add the statements:

Const decUnderGraduatePerHour As Decimal = 75

Const decGraduatePerHour As Decimal = 145

Const decThesisPerHour As Decimal = 160

Dim decTuition As Decimal

Select Case Me.lstCourseLevels.SelectedItem

Case “Undergraduate”

decTuition = decUndergraduatePerHour

Case “Thesis and Dissertation”

decTuition = decThesisPerHour

End Select

Me. lblTuition.Text = “Tuition is “&Format(decTuition, “Currency”) &\_”per credit hour”

* 1. Create a lstCourseLevels\_SelectedIndexChanged event procedure and then add the following statement to clear the label when a different list item is clicked:

Me.lblTuition.Text = Nothing

1. **Run the Application**
   1. Save the modified Tuition Calculator project and then run the application. Test the application by clicking each of the list items and then Calculate.