C863 Stage 1, Visual Basic
Certificate IV in Information Technology
ListBoxes – Drag and Drop

Drag and Drop
Implementing a drag-and-drop operation in the .NET Framework is accomplished by a short sequence of events. Typically it begins in a MouseDown event of one control and always ends with a DragDrop event of another. We will create an application to demonstrate this. This application will allow the user to drag the items from one ListBox to another.

Application
1. Create a new application adding the following controls:

Object	Name	Text
Button	btnLoadListBox	Load ListBox
ListBox	lstListNames1	
ListBox	lstListNames2	

2. On the Click event of the btnLoadListBox Button, add the all the names from a text file. Either create your own file or get a copy from the T Drive.

Now Let us get to what we want to achieve.

The first step in making drag and drop work is specifying whether or not a control will accept a drop. By default, all controls will reject such an act and do not respond to any attempt by the user to drop something onto them.

3. Set the AllowDrop property of the second ListBox to True

Object	Property	Value
lstListNames1	AllowDrop	True

The second step is to invoke the drag-and-drop operation. This is typically done in the MouseDown event of the control containing the date you want to drag (although you're not restricted to it). The DoDragDrop method is used to start the drag-and-drop operation. This method defines the data that will be dragged and the type of dragging that will be allowed, it returns a value from the DragDropEffects enumeration that represents the final effect that was performed during the drag-and-drop operation. In our application, we want to drag the text/name of the selected ListBox item, and we want to permit both a move and a copy of the date to occur.

Here is a list of the DragDropEffects Enumerations"

Member name	Description
All	The data is copied, removed from the drag source, and scrolled in the drop target.
Copy	The data is copied to the drop target.
Link	The data from the drag source is linked to the drop target.
Move	The data from the drag source is moved to the drop target.
None	The drop target does not accept the data.
Scroll	Scrolling is about to start or is currently occurring in the drop target.

4. On the MouseDown event of the lstNames1 ListBox put the following code:

```
Dim DragDropResult As DragDropEffects = DragDropEffects.None
If e.Button = Windows.Forms.MouseButtons.Left Then
  DragDropResult = lstNames1.DoDragDrop(lstNames1.Items(lstNames1.SelectedIndex), _
     DragDropEffects.Move Or DragDropEffects.Copy)
  'Here we are leaving room to add code checking the result for DragDropResult,
  'in particular if it is a DragDropEffects.Move result
End If
```

We will come back to the code and fill in the rest, for now the method DoDragDrop is getting us started.

The third step involves the recipient of the data, which is the lstNames2 ListBox. Here there are two events that are important to monitor: DragEnter and DragDrop.

The **DragEnter** event is raised when the user first drags the mouse cursor over the control during a drag-and-drop operation. The DragEnter event has a parameter of type DragEventArgs that contains two properties of interest, that is the Effect property and the KeyState properties.

The **Effect** property allows you to set the display of the for drop icon for the user to indicate if a move or a copy will occur when the mouse Button is released.
The **KeyState** property allows you to determine the state of the Crtl, Alt and Shift keys as well as the state of the mouse Buttons. As you know, it is a Windows standard that when both a move or a copy can occur a user is to indicate the copy action by holding down the Crtl key. Therefore, in this event, we will check the KeyState property and use it to determine how to set the Effect property.

5. On the DragEnter event of the lstNames2 ListBox put the following code:

```
If e.KeyState = 9 Then        'Control key has been pressed
  e.Effect = DragDropEffects.Copy
Else
  e.Effect = DragDropEffects.Move
End If
```

Note that we could also use the DragOver event if we wanted to, but it will fire continuously as the mouse moves over the target control.

The fourth step in the operation occurs when the user lets go of the mouse Button to drop the date at its destination. This is captured by the DragDrop event. The parameter contains a property holding the data that is being dragged. Now it is a simple process of placing it into the recipient control.

6. On the DragDrop event of the lstNames2 ListBox put the following code:

```
lstNames2.Items.Add(e.Data.GetData(DataFormats.Text))
```

The last step is to delete the item from the lstNames1 ListBox if the drag and drop was a move. Here we will add the code to the MouseDown event of the lstNames1 ListBox. That is once the DragDrop has occurred and it is a DragDropEffects.Move then delete the selected index.

7. On the MouseDown event of the lstNames1 ListBox replace the following code:

```
'Here we are leaving room to add code checking the result for DragDropResult,
  'in particular if it is a DragDropEffects.Move result
```

With this code

```
If DragDropResult = DragDropEffects.Move Then
  lstNames1.Items.RemoveAt(lstNames1.SelectedIndex)
End If
```

8. **Run**, **test** and **understand** your application.
Make sure you drag an item from lstNames1 ListBox. Also try copying by holding down the control key when you do it.