Advanced Architectures in LabVIEW™ Exercises

Course Software Version 2011
February 2012 Edition
Part Number 325183B-01

Worldwide Technical Support and Product Information

ni.com

Worldwide Offices

Visit ni.com/niglobal to access the branch office Web sites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 683 0100

For further support information, refer to the *Additional Information and Resources* appendix. To comment on National Instruments documentation, refer to the National Instruments Web site at ni.com/info and enter the Info Code feedback.

Contents

4

Advanced User Interface Techniques

Exercise 4-1 Creating an XControl (Optional)

Goal

To create an XControl that functions as a scalable radio control button.

Scenario

One benefit of XControls is that the architect can blend the functionality of several controls into one. In this exercise, you modify a Listbox control to have the look and feel of a radio button. Unlike the static radio button, this XControl scales dynamically while the program runs. The benefit is that the operator can see all of the items that are available (unlike a ring control) and which item is currently selected (unlike a plain Listbox). Moreover, this is a control that you can utilize in any application.

Design

Create and save the XControl with the names `X Listbox.xctl`, `X Listbox State.ctl`, `X Listbox Facade.vi`, and `X Listbox Init.vi`.

Implementation

1. Create the XControl and save the control and the abilities.
   - ❑ Open the `<Exercises>\Advanced Architectures in LabVIEW\X Controls\X Listbox\X Listbox.lvproj` directory.
   - ❑ In the `<Exercises>\Advanced Architectures in LabVIEW\X Controls\X Listbox\` folder, create a new subfolder named `Abilities`.
   - ❑ In the Project Explorer, right-click **My Computer** and select **New»XControl**.

Note You can also create an XControl from the template browser.

   - ❑ Right-click the XControl and select **Save»Save**.

- ❑ When prompted, click **Yes** to save the new unsaved files of the XControl.
- ❑ Save each of the following XControl files as follows:

Table 4-1. XControl File Names and Locations

XControl File	New Name	Folder
Facade	`X Listbox Facade.vi`	`...\Abilities`
State	`X Listbox State.ctl`	`...\Abilities`
Data	`X Listbox Data.ctl`	`...\Abilities`
Init	`X Listbox Init.vi`	`...\Abilities`
XControl	`X Listbox.xctl`	`...\X Listbox`

Note An XControl is a special kind of project library. If you must rename or move folders, do so within the XControl project. Renaming or moving files in Windows Explorer breaks the XControl.

End of Exercise 4-1

Exercise 4-2 Modifying X Listbox Abilities (Optional)

Goal

To modify the Data, State, and Init abilities of the X Listbox control you created in Exercise 4-1.

Design

Modify X Listbox Data.ctl

- This component specifies the data type of the XControl, in this case, the string data type.

Modify X Listbox State.ctl

- This component specifies any information other than the Data of an XControl that affects the appearance of the control. In this exercise, the component includes an I32 labeled as `# Items` and an array of strings labeled as `Names`.

Modify X Listbox Init.vi

- LabVIEW calls the Init ability when the XControl is first placed on a front panel or when a VI that contains the XControl is loaded into memory. You can use this ability to initialize the display state before the XControl is displayed. In this exercise, you rearrange the front panel to display only the Default Control State and Default Indicator State.

Implementation

Modify the Data, State, and Init abilities of the XControl.

1. Modify the Data ability. This component specifies the data type of the XControl, in this case, the string data type.
   - ❑ Open `X Listbox Data.ctl`.
   - ❑ Delete the control.
   - ❑ Add a string control.
   - ❑ Label the string control as `Item`.

Your control should resemble Figure 4-1.

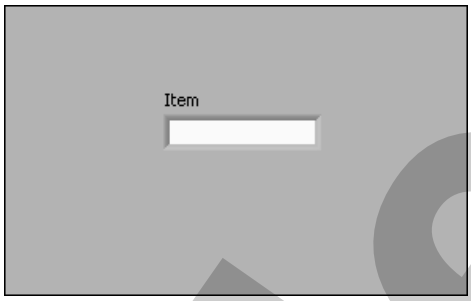

Figure 4-1. X Listbox.xctl

❑ Save and close the control.

2. Modify the State ability. This component specifies any information other than the Data of an XControl that affects the appearance of the control.

   ❑ Open `X Listbox State.ctl`

   ❑ Relabel the numeric control inside the cluster to `# Items`.

   ❑ Set the representation to I32.

   ❑ Add an array of strings.

   ❑ Label it `Names`.

   Your control should resemble Figure 4-2.

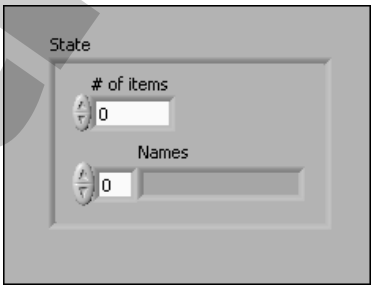

Figure 4-2. X Listbox State Control

   ❑ Save and close the control.

3. Modify the Init ability. LabVIEW calls the Init ability when the XControl is first placed on a front panel or when a VI that contains the XControl is loaded into memory.

   ❑ Open `X Listbox Init.vi`.

- ❑ Scroll down the front panel until you find the Default Indicator State and the Default Control State.

Tip Some controls on the front panel window are locked to prevent accidental deletion, which would destroy the VI. You can unlock and move the controls if you want.

- ❑ Move the controls and resize the front panel to display only the items shown in Figure 4-3.

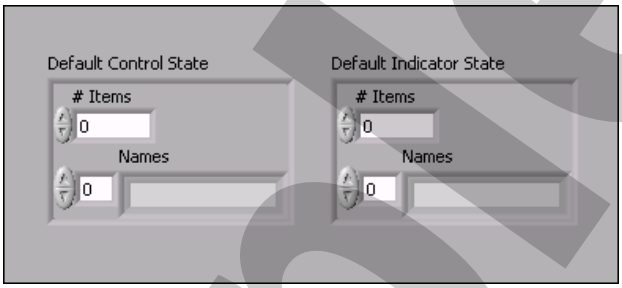

Figure 4-3.

- ❑ Save and close the VI.

End of Exercise 4-2

Exercise 4-3 Creating X Listbox Properties and Methods (Optional)

Goal

To create XControl properties and methods.

Scenario

You can create properties for an XControl in which one attribute or element of the display state is modified. In such cases, the property you create has one parameter associated with it. You use methods in cases when no parameters are needed (as in the Delete All method) or when more than one parameter is needed. Additionally, invoking a method implies that an action is taken, such as adding an item and deleting an item.

Design

- Create the Delete All Items method.
- Create the Delete Item from Listbox method.
- Create the Add Item to Listbox method.

Implementation

In this exercise, the data type of the X Listbox control is a string, specifically the item name of the active row. Users of the XControl must be able to add an item, delete an item, and delete all items. When complete, the front panel for the test VI resembles Figure 4-4.

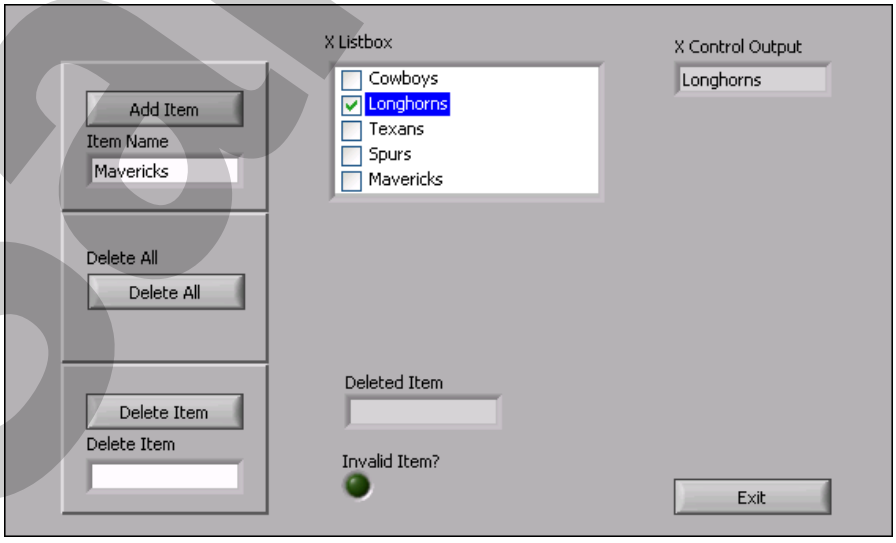

Figure 4-4. TEST X Listbox Front Panel

Remember that the X Listbox State contains the elements that you modify with the methods you create. When the parent VI that holds the XControl calls a method or property, the corresponding Facade VI executes the Display State Change event. After you create these methods, proceed to Exercise 4-4 to modify the Display State Change event case to handle the data modified by the methods.

1. Create the Delete All Items method.

   - ❑ Right-click `X Listbox.xctl` and select **New»Method**.
   - ❑ Right-click the new method and select **Save**.
   - ❑ Save the VI as `Delete All Items.vi` in the `<Exercises>\Advanced Architectures in LabVIEW\X Controls\X Listbox\Methods` directory.
   - ❑ Right-click the `Delete All Items.vi` and select **Configure Method**.
   - ❑ View the settings. No changes must be made for this method because it does not have any inputs or outputs.
   - ❑ Click **OK** to close the Configure Method dialog box.
   - ❑ Open the block diagram of `Delete All Items.vi`.
   - ❑ Build the block diagram as shown in Figure 4-5.

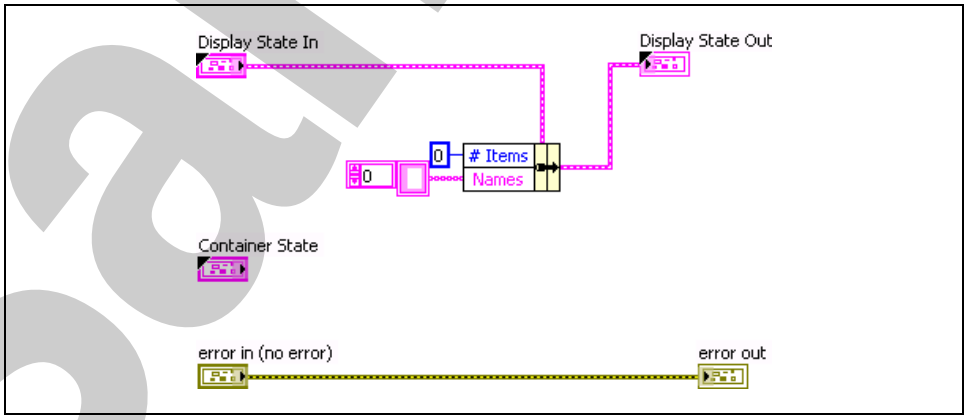

Figure 4-5. Delete All Items VI Block Diagram

   - ❑ Save and Close the VI.
   - ❑ Save `X Listbox.xctl`.

2. Create the Delete Item From Listbox method.

   ❑ Right-click `X Listbox.xctl` and select **New»Method**.

   ❑ Right-click the new method and select **Save**.

   ❑ Save the VI as `Delete Item From Listbox.vi` in the `<Exercises>\Advanced Architectures in LabVIEW\X Controls\X Listbox\Methods` directory.

   ❑ Open the block diagram of `Delete Item From Listbox.vi`.

   ❑ Build the diagram of the VI as shown in Figure 4-6.

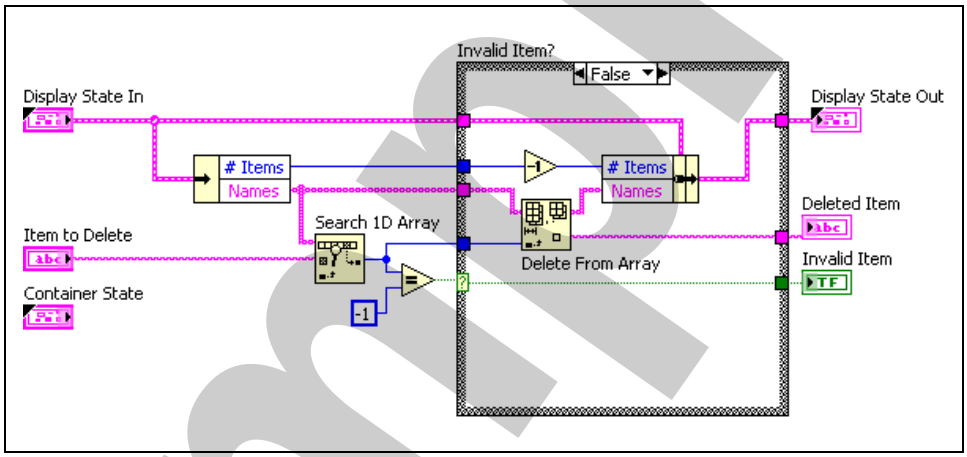

Figure 4-6. Delete Item From Listbox VI Block Diagram - False Case

❑ Wire the True case as shown in Figure 4-7

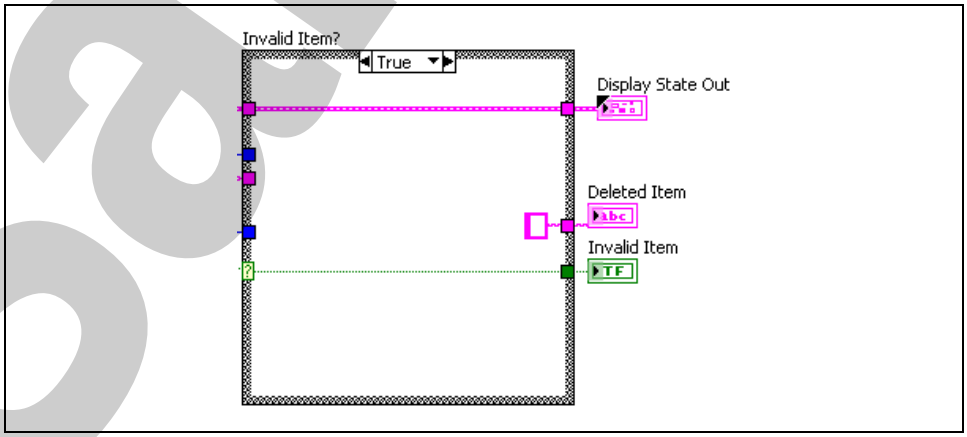

Figure 4-7. Delete Item From Listbox VI Block Diagram - True Case

- ❑ Wire **Item to Delete**, **Deleted Item**, and **Invalid Item?** to appropriate connectors on the connector pane.
- ❑ In the project explorer, right-click the `Delete Item From Listbox.vi` and select **Configure Method**.
- ❑ Add and configure the parameters for the input and each output, as shown in Figure 4-8.

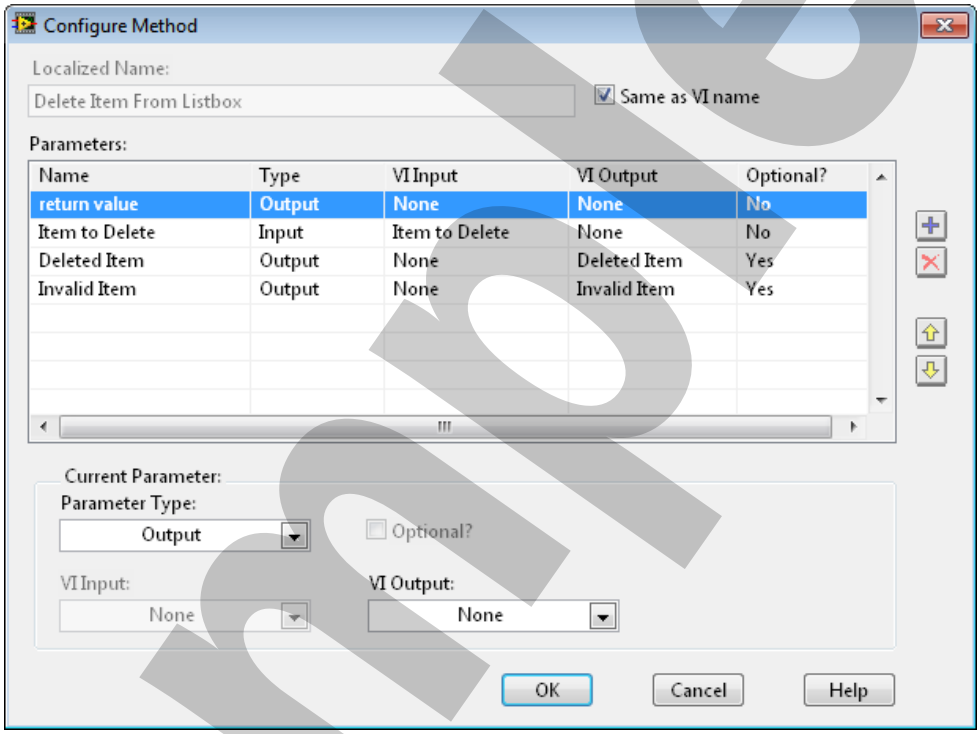

Figure 4-8. Configure Method Dialog for Delete Item From Listbox VI

Note If you did not define connector pane terminals, the above items will not be available. Each item added in the Configure method is a parameter for the XControl method.

- ❑ Select **OK** to close the Configure Method dialog box.
- ❑ Save and close the VI.
- ❑ Save `X Listbox.xctl`.

3. Create the Add Item To Listbox method.

- ❑ Right-click `X Listbox.xctl` and select **New»Method**.
- ❑ Right-click the new method and select **Save**.

- ❑ Save the VI as `Add Item To Listbox.vi` in the `<Exercises>\Advanced Architectures in LabVIEW\X Control\X Listbox\Methods` directory.
- ❑ Open the block diagram of `Add Item To Listbox.vi`.
- ❑ Add a string control to the front panel. Rename the control as `Item Name`.
- ❑ Build the diagram of the VI as shown in Figure 4-9.

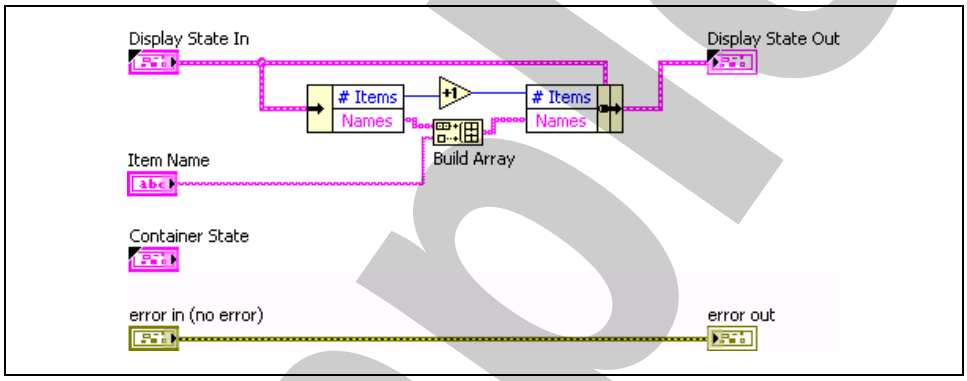

Figure 4-9. Add Item to Listbox VI Block Diagram

- ❑ Wire **Item Name** to an appropriate connector on the connector pane.
- ❑ Right-click the `Add Item To Listbox.vi` and select **Configure Method**.

❑ Add a parameter for the input, as shown in Figure 4-11.

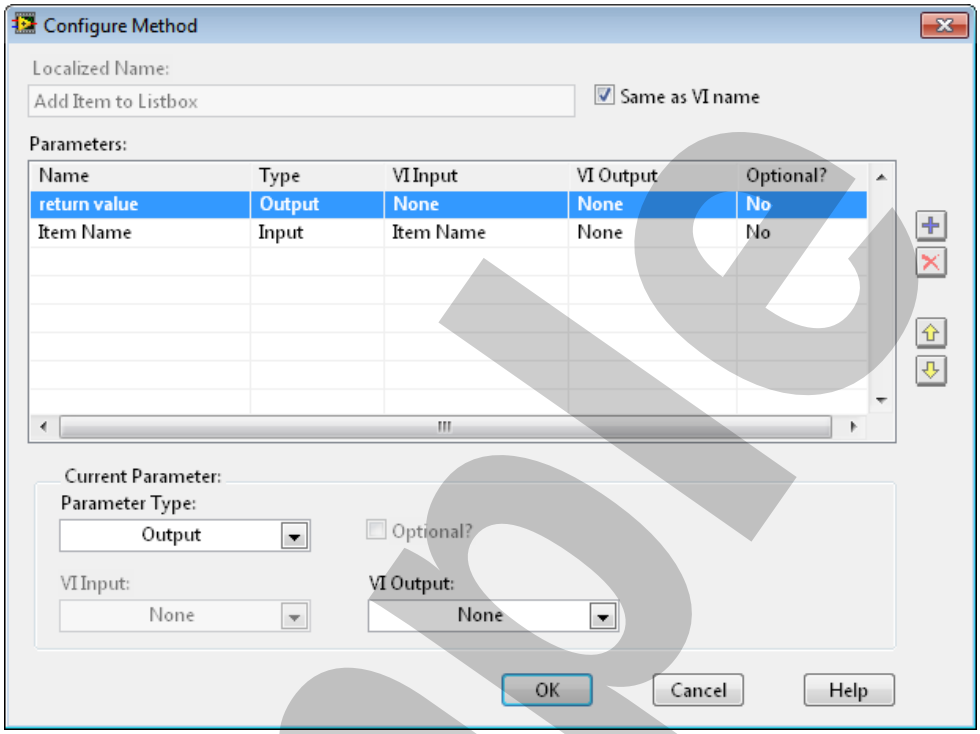

Figure 4-10. Configure Method Dialog for Add Item To Listbox VI

❑ Click **OK** to close the Configure Method dialog box.

❑ Save and close the VI.

❑ Save `X Listbox.xctl`.

End of Exercise 4-3

Exercise 4-4 Creating the X Listbox Facade VI (Optional)

Goal

To create the Facade VI.

Design

Create the front panel window

- The front panel window should include a Listbox that is sized to include many items. The Fit Control to Pane option should be enabled.

Implement the Display State Change event

- This section should update the properties of the X Listbox so that new items are added and others are deleted. The `X Listbox Update for Add and Delete.vi` can be used here.

Implement the Exec State Change event

- The number of rows and the item names of the listbox should be updated based on the display state data.

Create a Listbox: Mouse Down event

- The item symbols should be updated depending upon the active listbox item. The `X Listbox Update Symbols for Select.vi` can be used here.

Organize the X Control library.

Test the VI using the TEST X Listbox VI that has been provided.

Implementation

1. Modify the Facade front panel.
   - ❑ Open `X Listbox Facade.vi`.
   - ❑ Delete the comment on the front panel window.
   - ❑ Add a listbox to the front panel window.
   - ❑ Right-click the listbox control and select **Visible Items»Label** to hide the label.

Note The XControl has its own label when you drop it onto a front panel.

- ❑ Right-click the Listbox control and select **Visible Items»Symbols** to make them visible. This gives the Listbox a radio button feel.
- ❑ Right-click the Listbox control and deselect **Visible Items»Vertical Scrollbar** to remove the scrollbar.
- ❑ Size the Listbox control and the front panel window so that the listbox control takes up the entire window and exactly matches the window borders, as shown in Figure 4-11.
- ❑ Right-click and select **Fit Control to Pane**.

Figure 4-11. Listbox Control Window

2. Modify the Display State Change event.

Note The Display State Change event is generated on the Facade VI when the display state of the XControl changes. This can be in response to a property or method call, or from the Init ability. This event is not generated when other Facade events set the State Changed? element of action. At any given time, the appearance and behavior of the XControl should depend only on the display state, and therefore the Display State Change event must apply any changes in the state to the appearance or behavior of the XControl.

❑ Complete the Display State Change event, as shown in Figure 4-12, to update the appearance of the X Listbox XControl when its properties and methods modify the Display State.

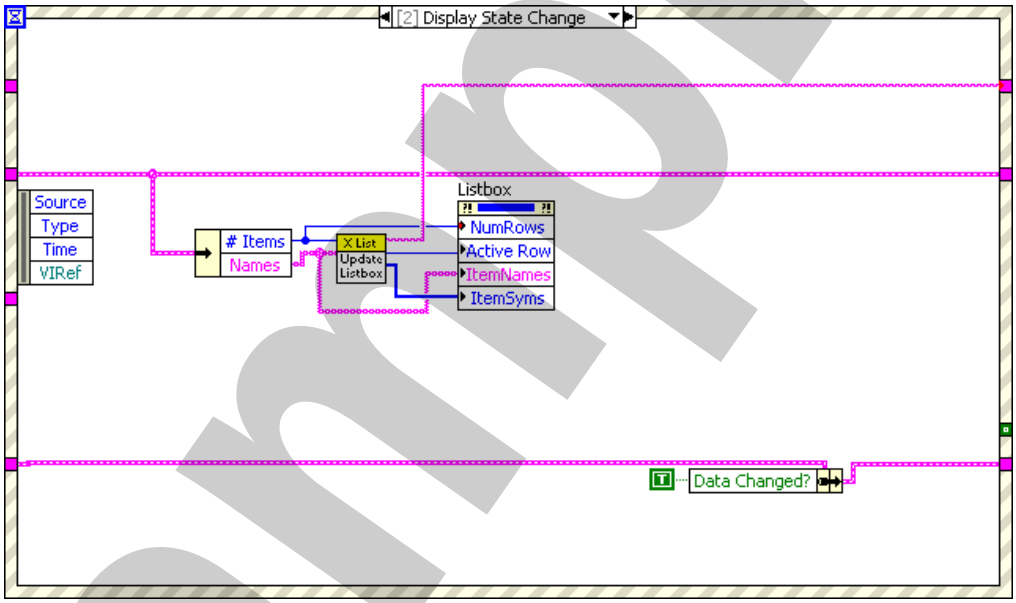

Figure 4-12. Display State Change Event Case

Note The `X Listbox Update for Add and Delete.vi` is located in the SubVIs directory of the X Listbox Control Project.

The number of the event case does not relate to the functionality of the event structure; it specifies the order of the event case when you view it.

3. Modify the Exec State Change case.

Note The Exec State Change event is generated on the Facade VI when a VI containing the XControl changes from edit mode to run mode, or vice versa. You can use this event to modify the control for any differences between its edit-time and run-time behavior.

❑ Complete the Exec State Change event, as shown in Figure 4-13.

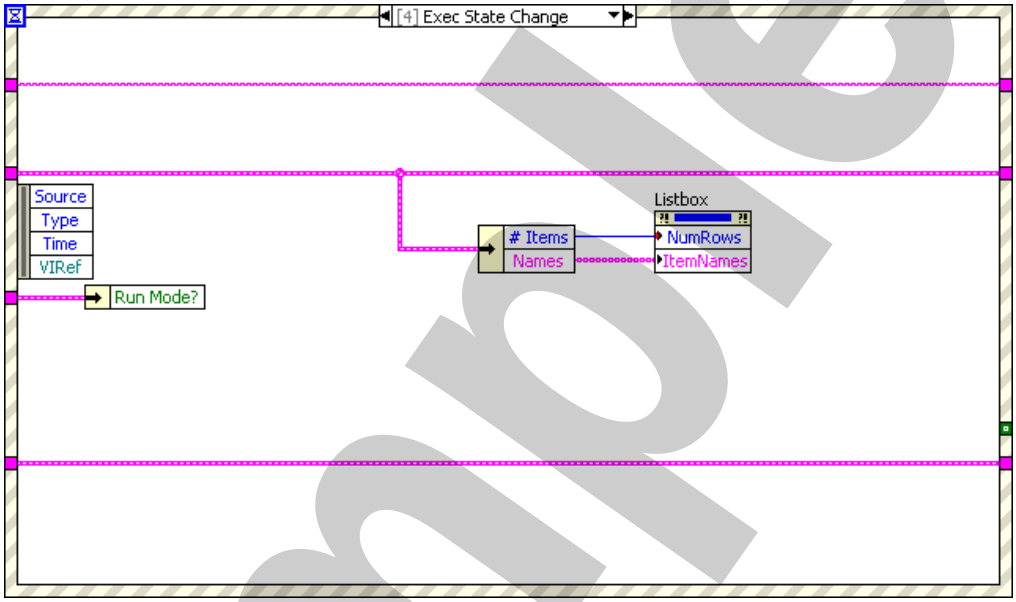

Figure 4-13. Exec State Change Event Case

4. Add the code for a mouse down event on the listbox.

Remember that you can add any event for the controls on the Facade of your XControl.

The XControl template adds only four new event cases to those that are already available for controls and VIs.

❑ Add a `"Listbox":Mouse Down` event.

❑ Complete the block diagram, as shown in Figure 4-14, to update the appearance of the X Listbox XControl when the user selects a row using the mouse.

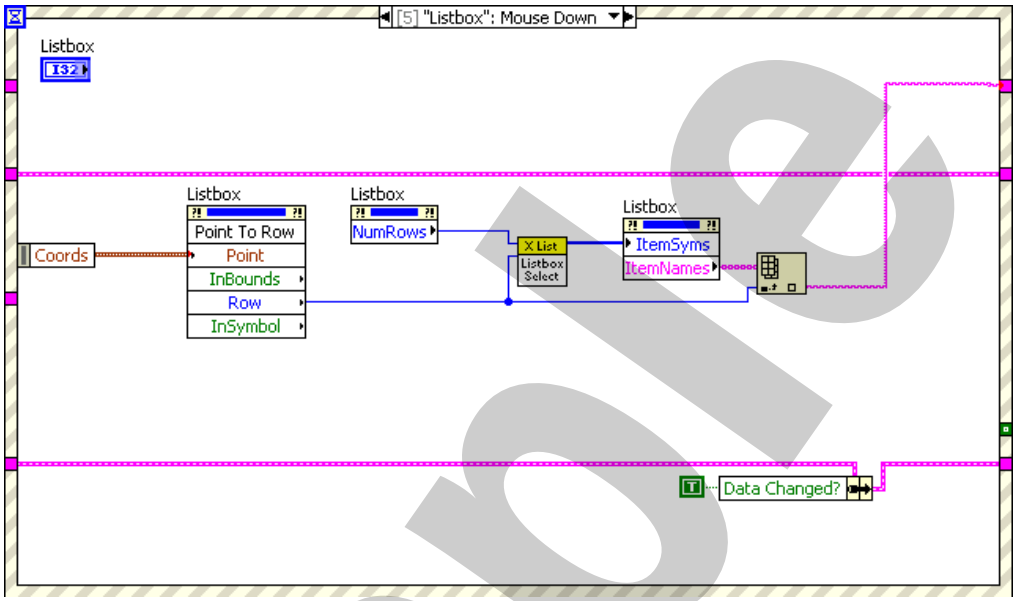

Figure 4-14. "Listbox": Mouse Down Event Case

Note The `X Listbox Update Symbols for Select.vi` is located in the SubVIs folder in the X Listbox Control.lvproj.

5. Save and close the Facade VI.

6. Organize the X Listbox library.

   ❑ Save the `X Listbox.xctl`.

   ❑ In the Project Explorer, add two virtual folders under the `X Listbox.xctl` item: `Abilities` and `Methods`.

   ❑ Drag the `X Listbox Data.ctl`, `X Listbox State.ctl`, `X Listbox Facade.vi` and `X Listbox Init.vi` into the `Abilities` virtual folder.

   ❑ Drag the `Delete All Items.vi`, `Delete Item from Listbox.vi`, and `Add Item to Listbox.vi` into the `Methods` virtual folder.

   ❑ Drag the `SubVIs` virtual folder under the `X Listbox.xctl` item.

❑ Select **File»Save All**. Your project should resemble Figure 4-15.

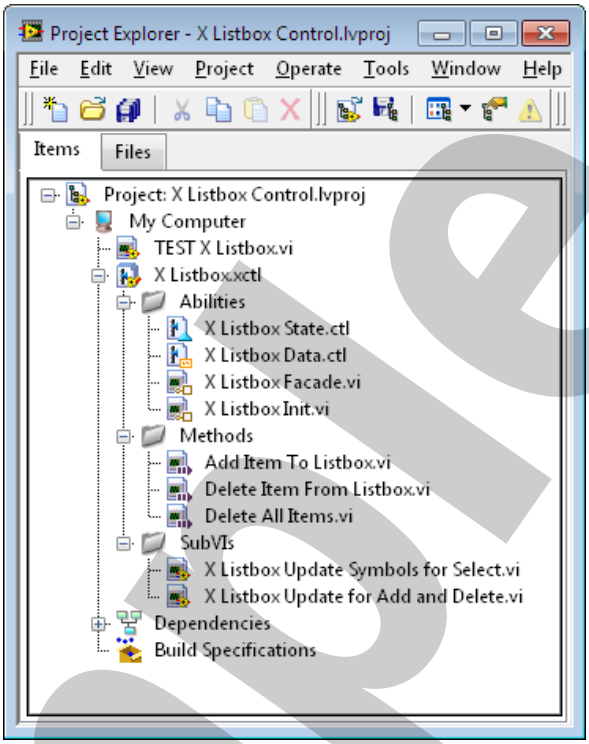

Figure 4-15. X Listbox Control Project

❑ Close the XControl.

7. Test the XControl.

   The X Listbox Control project already contains the Test VI stub.

   ❑ Open `TEST X Listbox.vi`.

   ❑ Drag the `X Listbox.xctl` from the project onto the front panel.

   ❑ Open the block diagram.

❑ Wire the X Listbox control to the X Control Output control and edit the event handled by this case, as shown in Figure 4-16.

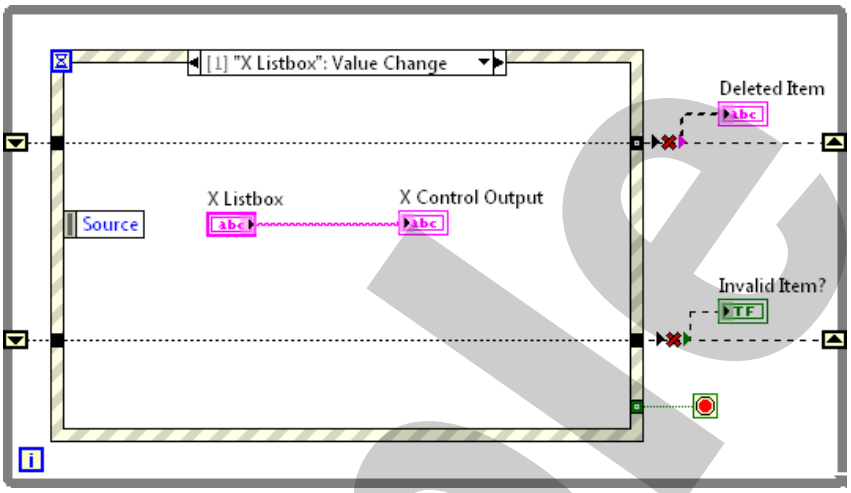

Figure 4-16. X Control Output Control

Note Wires on the block diagram appear broken until you wire the `"Delete Item": Value Change` case.

❑ Create the methods from the X Control.

– Right-click the X Control terminal and select **Create»Invoke Node»Delete All Items** to create the method. Add to the corresponding event, as shown in Figure 4-17.

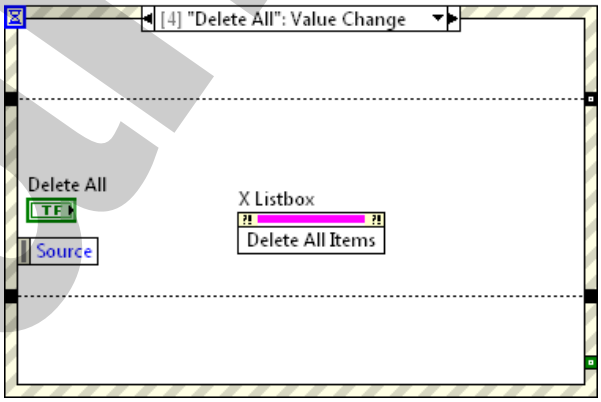

Figure 4-17. X Control Delete All Items Event

– Right-click the X Control terminal and select **Create»Invoke Node»Delete Item from Listbox** to create the method. Add to the corresponding event, as shown in Figure 4-18.

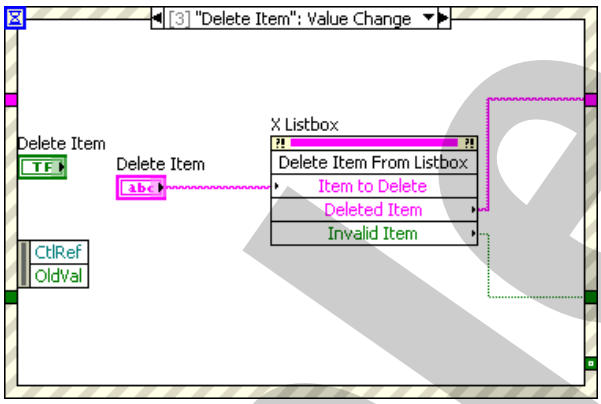

Figure 4-18. X Control Delete Item From List Box Event

– Right-click the X Control terminal and select **Create»Invoke Node»Add Item to Listbox** to create the method. Add to the corresponding event, as shown in FigureX Control Add Item to Listbox Event 4-19.

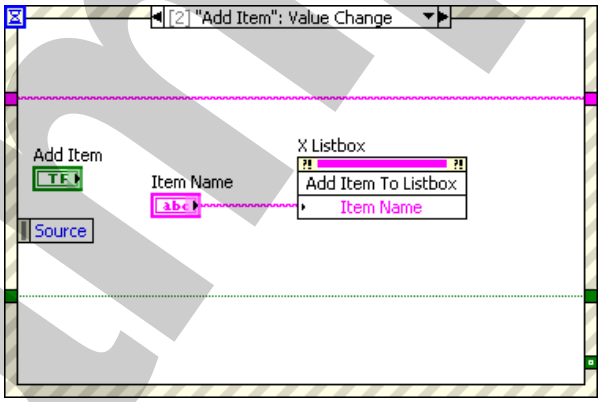

Figure 4-19. X Control Add Item to Listbox Event

❑ Save the VI.

8. Test the VI.

❑ Add items to the listbox.

❑ Delete items from the listbox.

❑ Delete all items from the listbox.

❑ Stop the VI with items still in the listbox.

- ❑ Run the VI again to ensure items remain.
- ❑ Save and close the VI.

Note In order for state data to be saved with a parent VI, a Convert for Save ability must be added to the XControl. No modifications need to be made.

- ❑ Save and close the project.

Challenges

- ❑ As it is currently coded, the X Listbox operates in the following manner. When a user adds or deletes an item, the last item becomes the active item. This may not be the cleanest solution when the user deletes an item in the middle of the list. Modify the X Listbox so that upon deleting an item, the active row remains the same as the row that was just deleted. You must modify the `X Listbox State.ctl` as well as the `X Listbox Facade.ctl`.
- ❑ Modify the `X Listbox Init.vi` so that it reads an `*.ini` to populate the initial values of the item names.
- ❑ Modify the appearance of the check boxes to be radio buttons. Hint: this will involve copying an image to the symbol table.
- ❑ Modify Add Item to Listbox to check for duplicate names.
- ❑ Add an `Add Items to Listbox.vi` to add an array of names at one time.

End of Exercise 4-4

Notes

Notes