XOJO

Desktop Tutorial

Xojo, Inc.

Chapter 1

Introduction

Welcome to Xojo, the easiest way to create cross-platform desktop and web applications.

Section 1

About the Desktop Tutorial

STARTING XOJO

1. **Download the installer for your operating system from: http://www.xojo.com/download**
2. **Run the installer.**
3. **Run Xojo.**
4. **In the Project Chooser, select Desktop and click OK.**

This Desktop Tutorial is for people who are new to programming and new to Xojo. It is an introduction to the Xojo desktop development environment and will lead you through the development of a real desktop application.

It should take you about an hour to complete this tutorial.

Note: If you have experience with other programming languages, you'll also want to check out the User Guide and Language Reference.

information. Use of a term in this book should not be regarded as affecting the validity of any trademark, registered trademark, or service mark. Xojo, Inc. is not associated with any product or vendor mentioned in this book.

Section 2

Presentation Conventions

The Tutorial uses screen snapshots taken from the Windows, OS X and Linux versions of Xojo. The interface design and feature set are identical on all platforms, so the differences between platforms are cosmetic and have to do with the differences between the Windows, OS X, and Linux graphical user interfaces.

- **Bold type** is used to emphasize the first time a new term is used and to highlight important concepts. In addition, titles of books, such as *Xojo User Guide*, are italicized.

- When you are instructed to choose an item from one of the menus, you will see something like “choose File ↠ New Project”. This is equivalent to “choose New Project from the File menu.”

- Keyboard shortcuts consist of a sequence of keys that should be pressed in the order they are listed. On Windows and Linux, the Ctrl key is the modifier; on OS X, the ⌘ (Command) key is the modifier. For example, when you see the shortcut “Ctrl+O” or “⌘-O”, it means to hold down the Control key on a Windows or Linux computer and then press the “O” key or hold down the ⌘ key on OS X and the press the “O” key. You release the modifier key only after you press the shortcut key.

- Something that you are supposed to type is quoted, such as “GoButton”.

- Some steps ask you to enter lines of code into the Code Editor. They appear in a shaded box:

```
ShowURL(SelectedURL.Text)
```

When you enter code, please observe these guidelines:

- Type each printed line on a separate line in the Code Editor. Don’t try to fit two or more printed lines into the same line or split a long line into two or more lines.

- Don’t add extra spaces where no spaces are indicated in the printed code.

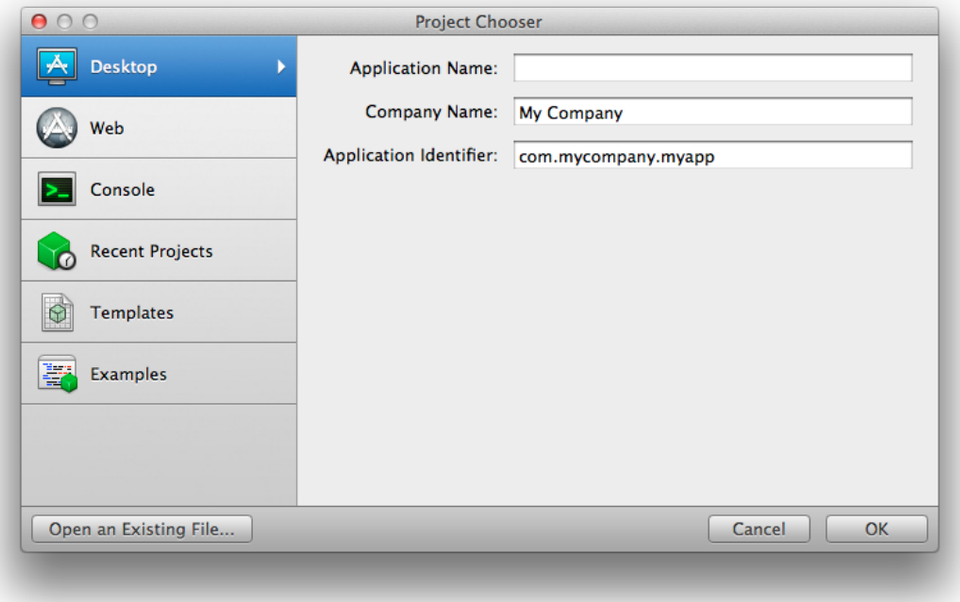
- Of course, you can copy and paste the code as well.

Whenever you run your application, Xojo first checks your code for spelling and syntax errors. If this checking turns up an error, an error pane appears at the bottom of the main window for you to review.

Getting Started

If you haven't done so already, now is the time to start Xojo.

Figure 1.1 The Project Chooser Window



Double-click the Xojo application icon to start Xojo. After it finishes loading, the Project Chooser window appears.

Xojo lets you build three different types of applications (Desktop, Web and Console). For this Tutorial, you are building a desktop application, so click on Desktop.

You should now see three fields to specify: Application Name, Company Name and Application Identifier.

Application Name is the name of your application. This will be the filename of the actual application file that gets created.

Company Name is the name of your company. You may choose to leave this blank.

Application Identifier is a unique identifier for this application. It will automatically populate using what you enter for the Application and Company Names, but you can also change it to whatever you want.

Enter "SimpleBrowser" as the Application Name. You can leave Company Name as it is or change it.

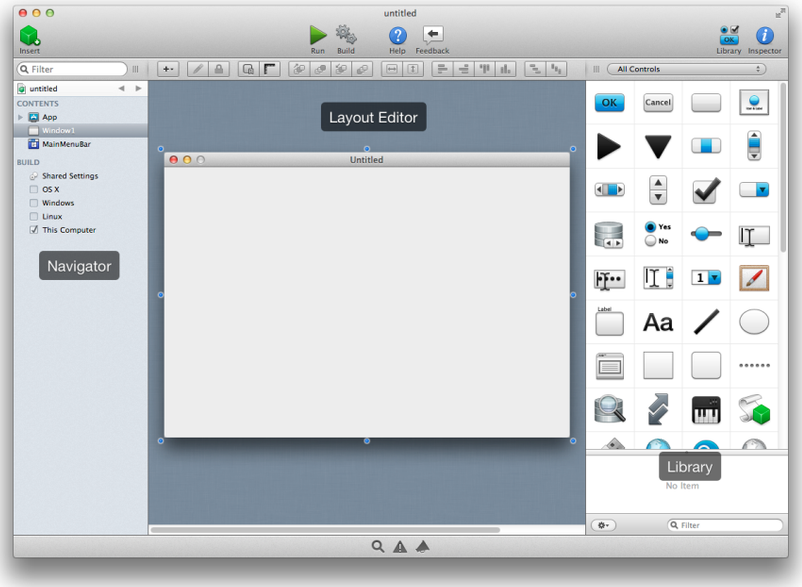
Click **OK** to open the main Xojo window (called the Workspace), where you will begin designing your application.

Section 4

Workspace

Xojo opens the Workspace with the default window for your application selected in the **Navigator** and displayed in the **Layout Editor**.

Figure 1.2 The Xojo Main Window



Navigator: The area on the top left shows you all the items in your project. By default you can see Window1 (which is selected), the App object and the MainMenuBar object. You use the Navigator to navigate within your project.

Layout Editor: The center area is the Layout Editor. You use the Layout Editor to design the user interface for the windows in your application. It shows the window and previews how it looks when the application runs. In this illustration, the window is blank because you haven't yet added any user interface controls from the **Library**.

Library: The area on the right is the Library and shows the controls and interface elements that you can add to a window or to the project. You design the window by dragging controls from the Library to the window. You can also add a control to the window by double-clicking it.
You can change how the controls display in the Library by clicking the small gear icon  and choosing a different setting.

Note: If the Library is not visible, click the Library button on the toolbar to show it.

Inspector: Not shown in the above illustration is the Inspector, which allows you to see and change the properties for the

selected control. This area of the Main window is shared with the Library. You can show the Inspector by clicking the Inspector button on the toolbar. The Inspector shows information about the selected item in the Navigator or Editor. The contents of the Inspector changes as you click on different items. You change an Inspector value by entering a new value in the field to the right of the field label.

About the Tutorial Application

In this tutorial you will build an application that manages URLs and email addresses. Typically a URL is prefixed with “http://” to indicate that it is an address of a web page to display.

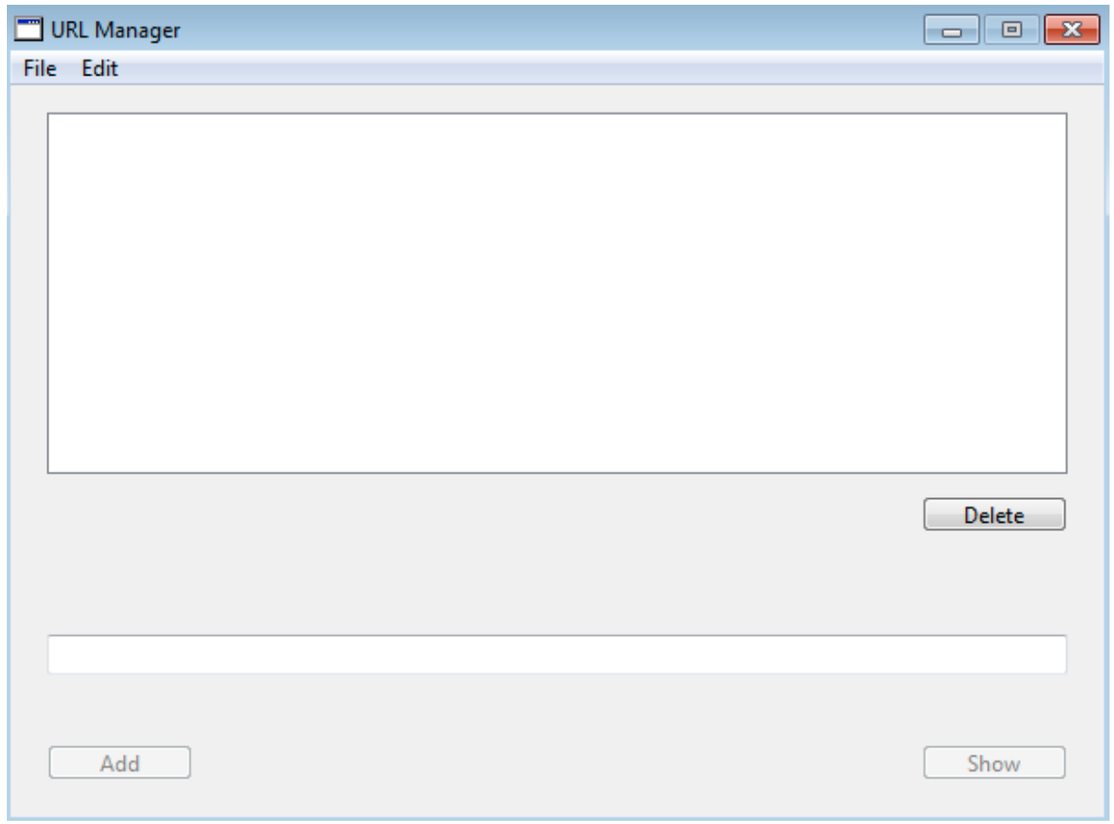
A URL can also have a prefix of “mailto:” to send an email using the default email software.

URL Manager

For the URL Manager application, you enter a URL in the Text Field and click the Show button to display it in your default web browser. Click the Add button to save it to the List Box. To display a URL from the List Box, select it and click Show.

To remove a URL, select it from the List Box and click Delete.

Figure 1.3 URL Manager Application



URL Manager uses three types of controls:

List Box: A control that holds scrollable lists. It can hold both single and multiple-column lists and scroll horizontally and vertically.

Text Field: A control that holds a single line of text.

Button: A standard button. It is most often used to initiate an action.

Chapter 2

Designing the User Interface

Now you will lay out the interface for URL Manager.

URL List

Adding the URL List

You should have Xojo open and at the Window Layout Editor. If not, please refer to Chapter 1, Sections 3 and 4.

You are now going to add a List Box to the window. The List Box is used for storing the URLs.

1. Add the List Box:

   In the Library, click on the List Box and drag it to the top-left corner of the Layout Editor.

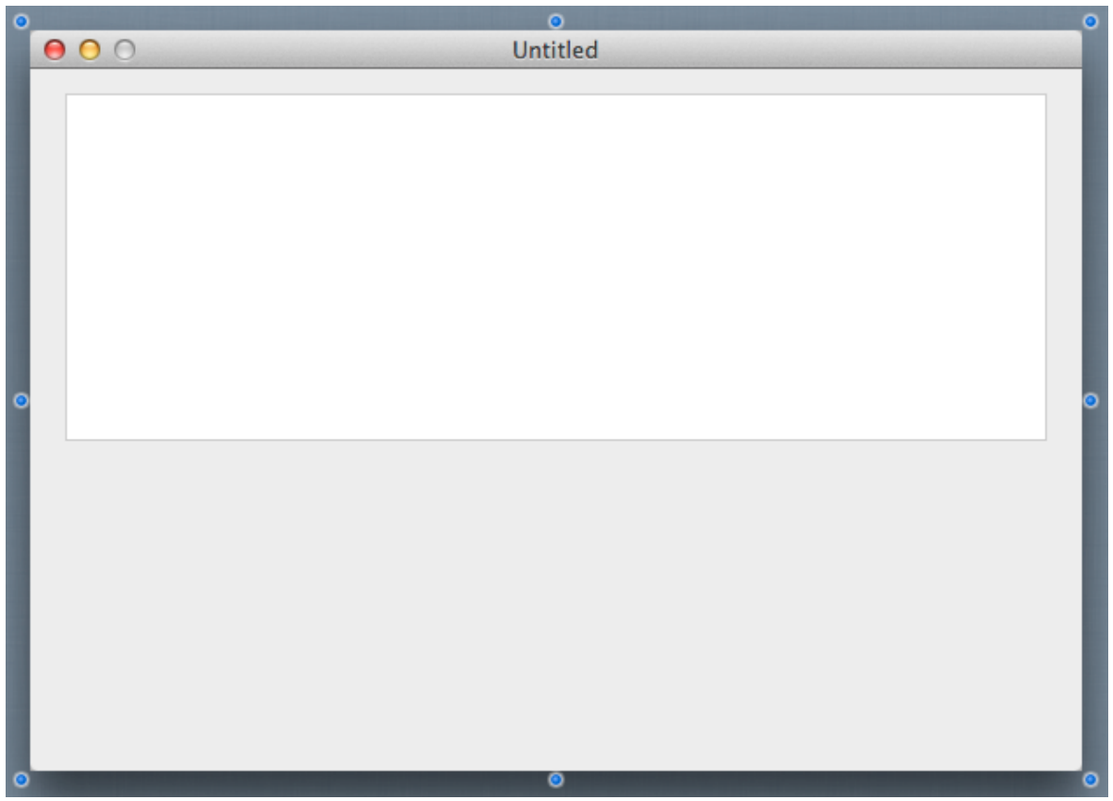
   As you get close to the edges of the window, you will see alignment indicators that help you position the control. Drop the List Box when you are happy with its position on the window.

2. Resize the List Box:

   Click on the List Box so that the resizing handles appear. Grab the handle in the bottom-right corner and drag it to enlarge the Listbox to fill the top 2/3 of the window.

3. Your window should now look like this:

Figure 2.1 Window Layout with List Box



Buttons

Adding the Buttons

Now you will add the Delete button to the window. The delete button will be used to remove URLs from the Listbox.

1. Add the Delete Button:

   In the Library, click on the Generic Button control (located in the Buttons group) and drag it to the window below the lower-right corner of the List Box.

   Use the alignment indicators to help you position the button so that it lines up with the right edge of the List Box

2. Add the Add Button:

   In the Library, click on the Generic Button control and drag it to the window near the bottom-left corner.

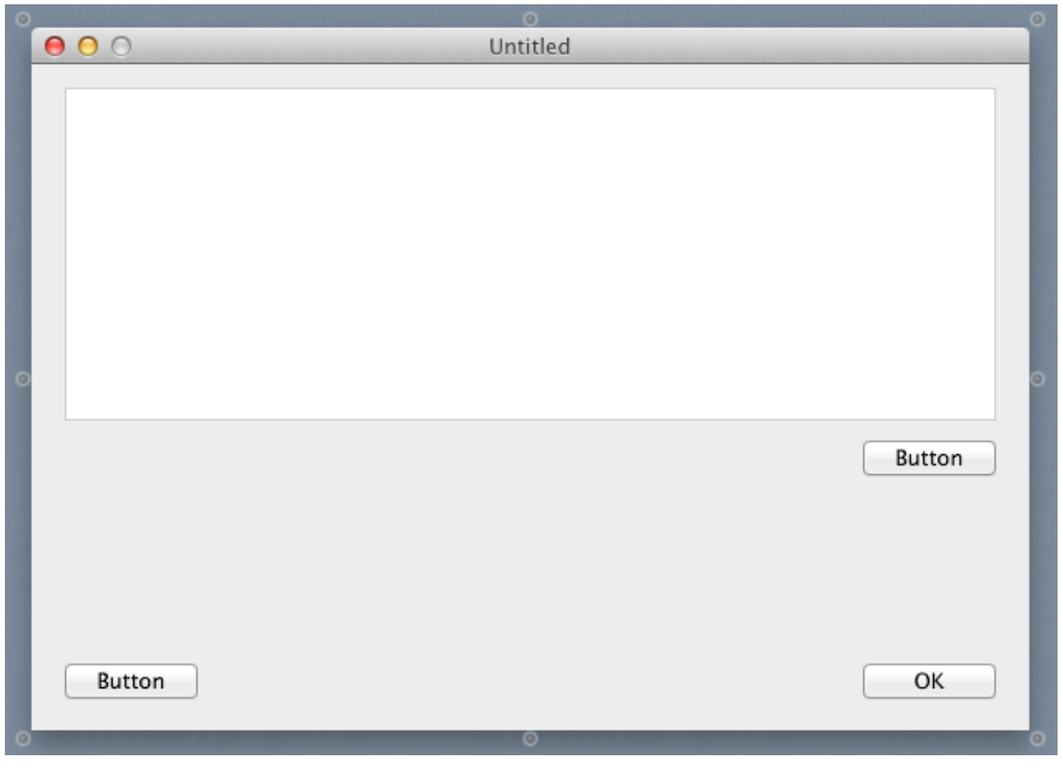
   Again, take advantage of the alignment indicators to help you position the button.

3. Add the Show Button:

   In the Library, click on the Default Button control and drag it to the window near the bottom-right corner.

4. Your window now looks like this:

Figure 2.2 Window Layout with Buttons



Text Field

Add the URL Field

The URL field is where the user types the URL to add to the list.

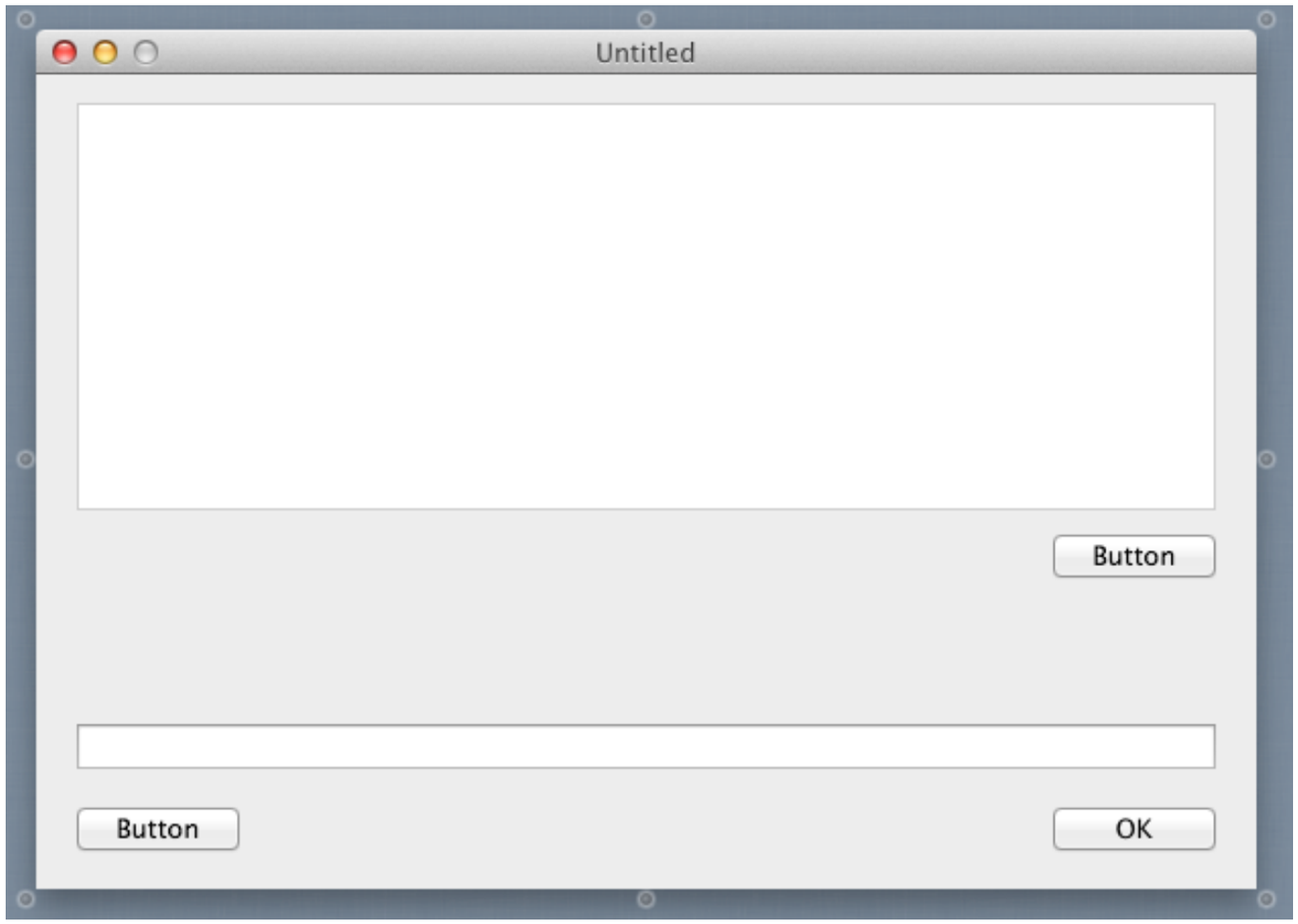
1. In the Library, click on Text Field and drag it to the window so that it is between the delete and add buttons.

2. Resize the URL field. Select the rightmost drag handle and drag the Text Field so that it is the same width as the List Box.

   Use the alignment indicators as guides to help you line everything up correctly.

3. Your window now looks like this:

Figure 2.3 Window Layout with Text Field



Properties

What is a Property?

A **property** is a value of a class, such as a control on a Window. Changing property values allows you to change the behavior of the class.

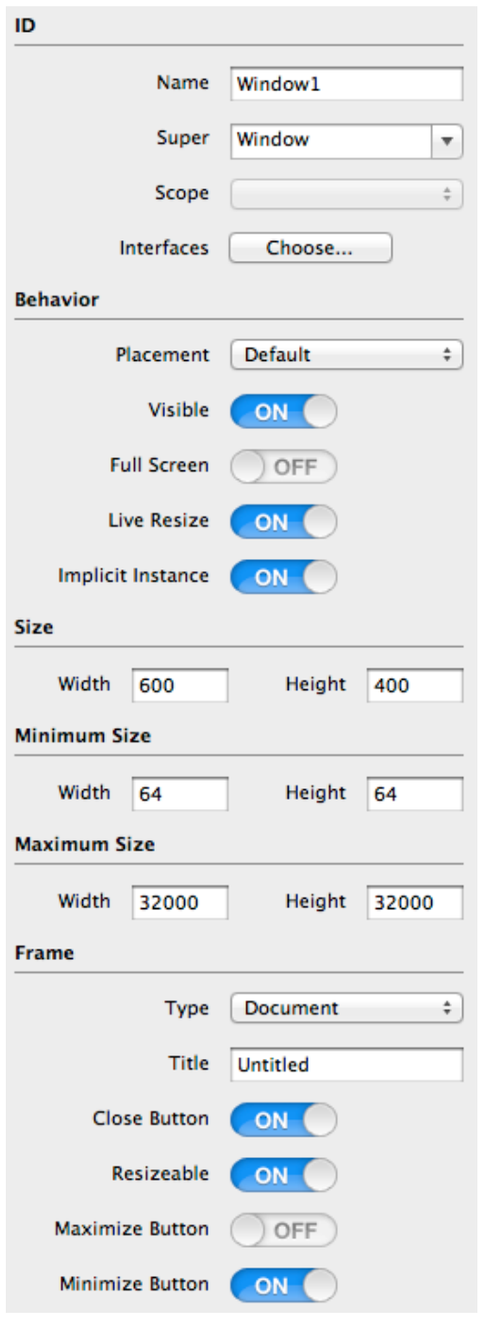
For this project, you want to change various properties for the window and its controls. Some of the things you need to do are:

- Rename all controls (and the window) so that they describe their behavior and are easy to refer to in code.
- Add a Caption to the Buttons.
- Set Locking properties so that the controls resize properly when the window is resized.

Inspector

The **Inspector** is used to change the window and control properties. It shares the same area on the right of the main window as the Library. In order to show the Inspector, click the Inspector button on the toolbar.

Figure 2.4 Window Properties in the Inspector



You can switch between the Library and Inspector using ⌘-L on OS X (Control-K on Windows and Linux) and ⌘-I on OS X (Ctrl-I on Windows and Linux).

Window Properties

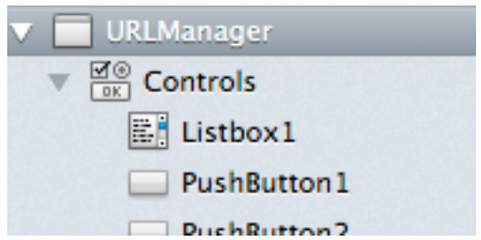
If you haven't already, display the Inspector by clicking the Inspector button on the toolbar.

You need to change the **Name** and **Title** properties:

1. First, in the Layout Editor, click on the title bar of the window to select it. The Inspector now shows the properties for the window.

2. In the Name field (located in the ID group), change the name from "Window1" to "URLManagerWindow". Press *Return* to see the name change in the Navigator.

Figure 2.6 Name Change in Navigator



3. In the Title field (located in the Frame group), change the name from "Untitled" to "URL Manager". Press *Return* and you will see the name change in the title bar of the window.

Figure 2.5 Changing the Title Bar for the Window

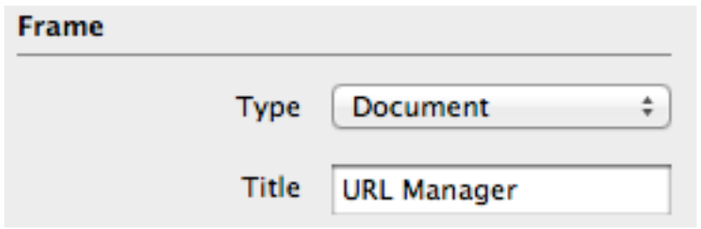
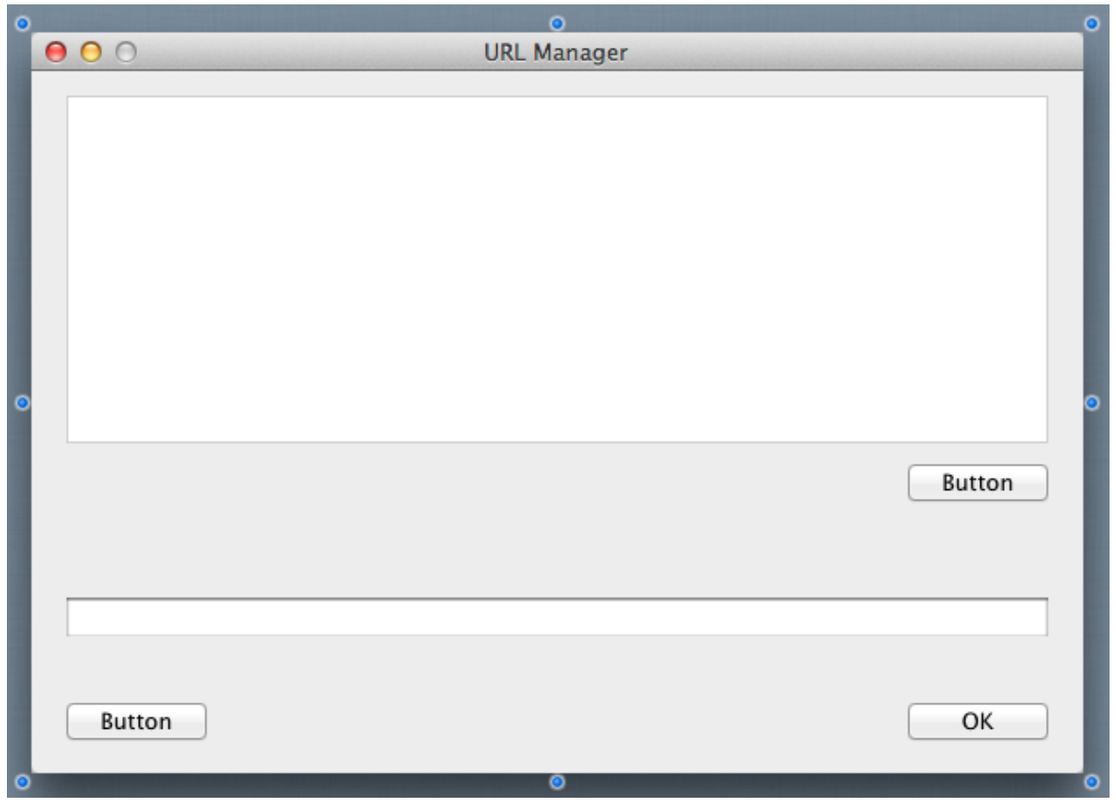


Figure 2.7 Window Layout with Updated Title Bar



Section 6

List Box Properties

The Listbox is where the URLs that your user enters are stored. You need to change the following properties: **Name** and **Locking.**

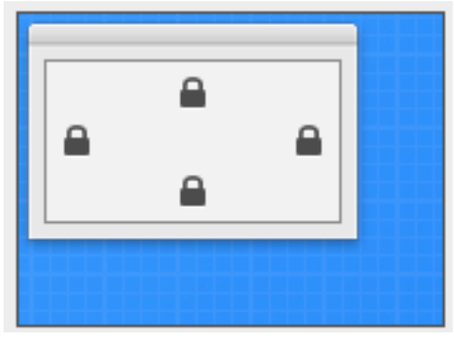
First, in the Layout Editor, click on the Listbox to select it. The Inspector now shows the properties for Listbox.

1. In the Name field (located in the ID group), change the name from “Listbox1” to “URLList”. Press *Return* to see the name change in the Navigator.
2. Now you need to make changes to the locking so that the Listbox gets larger or smaller as the window resizes.

   In the Locking group look at the image that shows the window with small locked icons for the top and left and small unlocked icons for bottom and right.

Click the locks so that top, left, bottom and right are all locked.

Figure 2.8 Locking for URLList



Button Properties

The three buttons on the Window are used to perform actions. You need to change the following properties for each button: **Name, Caption** and **Locking.**

Delete Button

The Delete button is used to remove URLs from the URLList.

1. First, in the Layout Editor, click on the Delete button to select it (this is the button directly below the Listbox). The Inspector now shows the properties for PushButton.

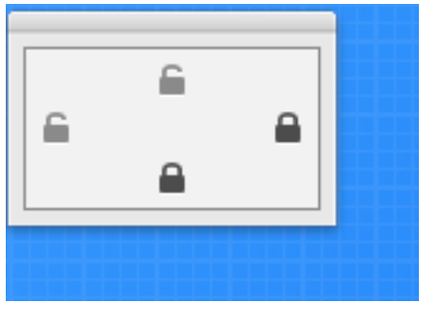
2. In the Name field of the Inspector (located in the ID group), change the name from “PushButton1” to “DeleteButton”. Press *Return* to see the name change in the Navigator.

3. In the Caption field (located in the Appearance group), change the name from “Button” to “Delete”. Press *Return* to see the name change on the button in the window.

4. Now you need to make changes to the locking so that the Delete button stays on the right side of the window when the window resizes.

In the Locking group look at the image that shows the window with small locked icons for the top and left and small unlocked icons for bottom and right. Click the locks so that right and bottom are locked and left and top are unlocked.

Figure 2.9 Locking for DeleteButton



Add Button

The Add Button is used to add the URL entered in the TextField to the URLList.

1. In the Layout Editor, click on the Add button to select it (this is the button on the far left of the window below the TextField). The Inspector now shows the properties for PushButton.

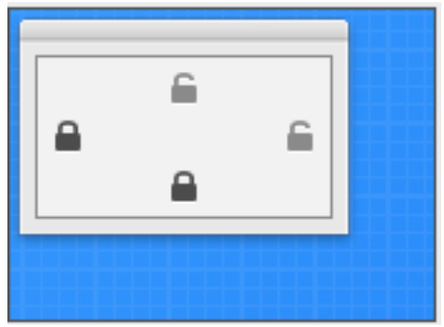
2. In the Name field (located in the ID group), change the name from “PushButton2” to “AddButton”. Press *Return* to see the name change in the Navigator.

3. In the Caption field (located in the Appearance group), change the name from “Button” to “Add”. Press *Return* to see the name change on the button in the window.

4. Now you need to check the locking so that the Add button stays on the bottom of the window when the window resizes.

   In the Locking group look at the image that shows the window with small locked icons for the top and left and small unlocked icons for bottom and right. Click the locks so that left and bottom are locked and top and right are unlocked.

Figure 2.10 Locking for AddButton



Show Button

The Show button is used to open your web browser and show the URL that is in the TextField.

1. In the Layout Editor, click on the Show button to select it (this is the button directly below the TextField on the right). The Inspector now shows the properties for PushButton.

2. In the Name field (located in the ID group), change the name from “PushButton3” to “ShowButton”. Press *Return* to see the name change in the Navigator.

3. In the Caption field (located in the Appearance group), change the name from “OK” to “Show”. Press *Return* to see the name change on the button in the window.

4. Now you need to make changes to the locking so that the Complete button stays on the right side of the window when the window resizes.

   In the Locking group look at the image that shows the window with small locked icons for the top and left and small unlocked icons for bottom and right. Click the locks so that right and bottom are locked and left and top are unlocked.

Figure 2.11 Locking for ShowButton

Text Field Properties

The Text Field is where your user will type the URLs to add to the list. It is also where URLs display when they are clicked in the list. You need to change the following properties: **Name** and **Locking.**

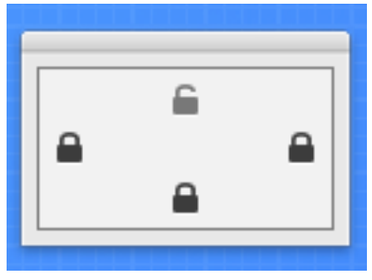
1. First, in the Layout Editor, click on the Text Field to select it. The Inspector now shows the properties for TextField.

2. In the Name field (located in the ID group), change the name from “TextField1” to “URLField”. Press *Return* to see the name change in the Navigator.

3. Now you need to make changes to the locking so that the TextField gets larger or smaller when the window resizes.

   In the Locking group look at the image that shows the window with small locked icons for the top and left and small unlocked icons for bottom and right. Click the locks so that left, bottom and right are locked and top is unlocked.

Figure 2.12
Locking for URLField



Testing the Project

Saving Your Project

You should save your work periodically and always before running your project.

1. Save the project by choosing File ➙ Save.

2. Name the project “QuickStartDesktop” and click Save.

Running Your Project

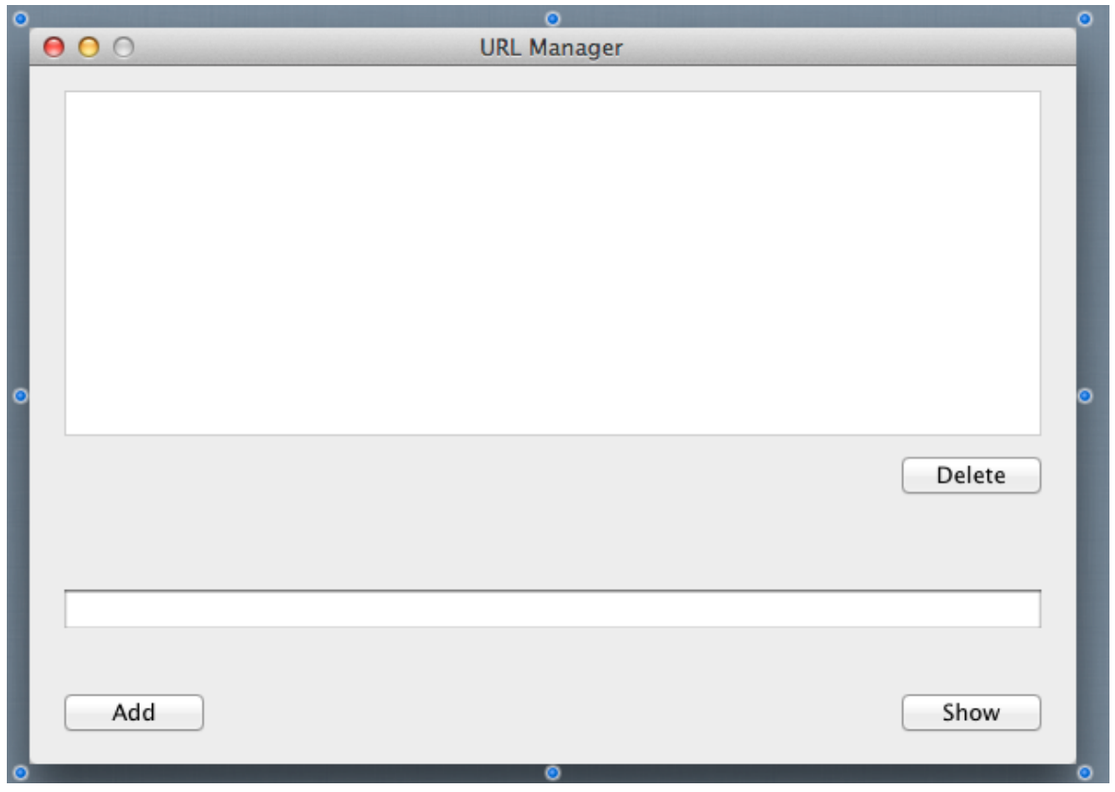
Now you can test your finished application:

Your user interface layout is now complete, so it’s time to try it out. Click the Run button in the toolbar to run the project.

When URL Manager is displayed, you can interact with the buttons by clicking on them, you can type in the TextField and you can resize the window to see the buttons reposition themselves.

But URL Manager doesn’t yet do anything. For that you need to add some code, which is the subject of the next chapter.

Figure 2.13 Completed URL Manager Window Layout



Chapter 3

Adding Code

The final step in creating URL Manager is to add the code.

Show Button

Adding Code to the Show Button

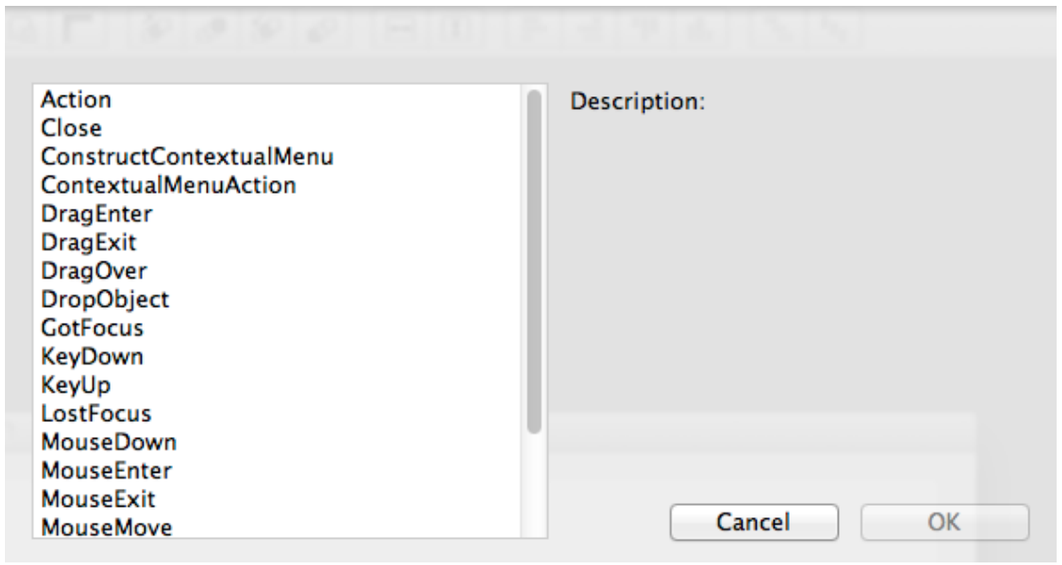
The first control to update is the Show button. When the user clicks this button, the user's default web browser should open to the URL displayed in the URL field.

Follow these steps to add the code:

1. On the window, double-click the **ShowButton** control, labelled "Show".

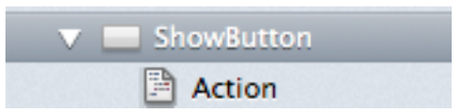
   The **Add Event Handler** dialog appears. When a user clicks on a PushButton, code in its **Action** event handler is run. This means you want to add your code to the Action event handler, so select Action from the Event Handler list and click OK.

Figure 3.1 Add Event Handler Dialog



Notice the Navigator updates to show the Action event underneath the ShowButton control and the **Code Editor** displays.

Figure 3.2 The Action Event Handler for ShowButton



2. The command to open a URL in the user's default browser is **ShowURL**. To test out this button, make it so that it opens the same web page each time:

```
ShowURL("http://www.wikipedia.org")
```

   ShowURL is a **global method** because it is not attached to any particular object. It can be called from anywhere in the application.

3. Run the app and click the Show button. Your default web browser should open to the Wikipedia home page. This is not terribly useful, so next you need to modify the code so that it uses the URL that the user has entered in the URL field. Quit the application to go back to editing the Window (choose File ↠ Exit on Windows or Linux (or URLManager.debug ↠ Quit on OS X).

4. Return to the Code Editor for the ShowButton Action event and remove the code that you added previously.

   Now you need to get the URL that was typed into the URL field. You might think you could get the URL just be referring to the name of the field, URLField. That is close, but not quite what you want.

   What you instead need is a property of URLField. When you need to refer to a property of an object, you use the name of the object, followed by a dot, followed by the name of the property. In other words, you use this syntax: **ObjectName.PropertyName**. This is something called “dot” notation and is commonly used in object-oriented programming.

   In this case the object is **URLField** and the property you want is **Text** (use the Language Reference to find out about all the properties available to TextFields).

   Now you can add the code:

   ```
   ShowURL(URLField.Text)
   ```

   (If you are having trouble entering the text, be sure you actually quit the running URL Manager application you were running in the previous test.)

5. Save the project by choosing File ↠ Save.

6. Run the app and enter a URL in the URL Field. Try something different this time, such as “http://www.xojoblog.com”, and click the Show button.
   Your default web browser should open the web page.

   Quit the application to go back to the Editor (choose File ↠ Exit on Windows or Linux (or URLManager.debug ↠ Quit on OS X).

Add Button

Adding Code to the Add Button

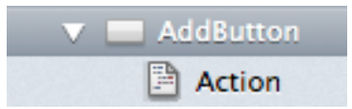
The Add button is used to add URLs to the list. The code you add to the button needs to take what was typed in the URL field and add it as a new row to the list.

Follow these steps to add the code:

1. On the window, double-click the **AddButton** control, labelled "Add".

   The **Add Event Handler** window appears. As you learned with ShowButton, you want to use the **Action** event handler to have your code run when the user clicks on a button. Select Action from the Event Handler list and click OK.

   Figure 3.3 Action Event Handler for AddButton

   Notice the Navigator updates to show the Action event underneath the AddButton control and the code editor displays.

2. To add a row to a Listbox, you use the AddRow method. You already know how to get the text in the URL field. Combine the two to get this code:

   ```
   URLList.AddRow(URLField.Text)
   ```

   As you have seen before, objects can have properties. And as you now see with URLList, objects can also have methods. AddRow is one of many methods available to Listboxes.

3. Save the project by choosing File ↠ Save.

4. Run the app to test it out. Type URLs in the URL field and click the Add button to see them appear in the URL list. Quit the application to go back to the Editor (choose File ↠ Exit on Windows or Linux (or URLManager.debug ↠ Quit on OS X).

Delete Button

Adding Code to the Delete Button

The Delete button is used to remove URLs from the list. The code you add to the button needs to determine the selected row in the list and remove it from the list.

Follow these steps to add the code:

1. On the window, double-click the **DeleteButton** control, labelled “Delete”.

   The **Add Event Handler** window appears. As you learned with the other buttons, you want to use the **Action** event handler to have your code run when the user clicks on a button. Select Action from the Event Handler list and click OK.

   Notice the Navigator updates to show the Action event underneath the DeleteButton control and the code editor displays.

2. To remove a row from the Listbox, you first need to determine what row (if any) is selected.

   In a Listbox, the currently selected row is contained in the ListIndex property.

3. Use the Listbox method **RemoveRow** to remove a row from the Listbox. You pass RemoveRow the row number to remove as a parameter. So your code looks like this:

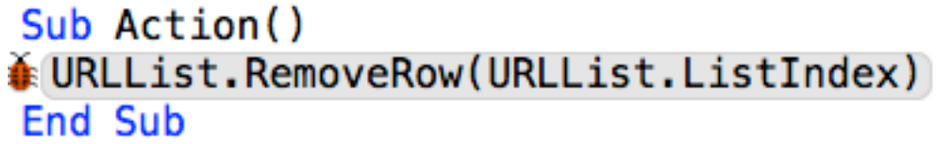
```
URLList.RemoveRow(URLList.ListIndex)
```

4. Save the project by choosing File ⇥ Save.

5. Run the app to test it out. Type URLs in the URL field and click the Add button to see them appear in the URL list. Now click on a URL in the URL list and click the Delete button. The URL is removed from the list.

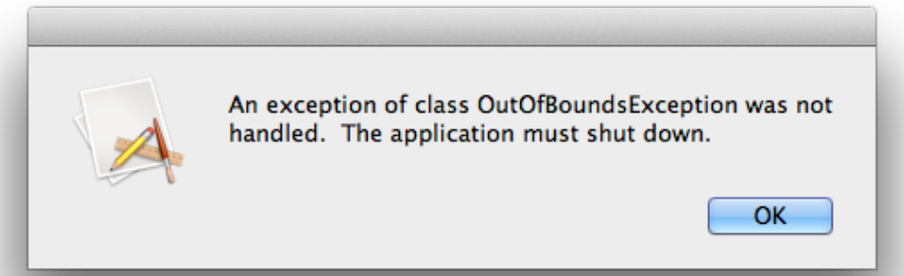
   But there is a bug in your application. What happens if someone clicks the Delete button but has not selected a row in the list? Try it. Your app will switch back to the Editor with a line of code highlighted. Your code crashed with an OutOfBoundsException and you are now in the debugger.

Figure 3.5 A Runtime Error in the Debugger



The error occurred because you attempted to remove a row that does not exist. When no row is selected in the Listbox, the ListIndex property returns -1. If you click the Resume button in the debugger toolbar, you will see the actual error message.

Figure 3.4 Runtime Error Message



6. Quit the application to go back to the Editor by pressing the “Stop” button in the Debugger toolbar.

7. Now you can add code to prevent the error. Essentially, you do not want to call the RemoveRow method if a row is not selected.

   The code looks like this:

```
If URLList.ListIndex >= 0 Then
  URLList.RemoveRow(URLList.ListIndex)
End If
```

   This code verifies that a row is selected by checking the ListIndex property to ensure that it contains a valid row. If it does, then the row is removed.

8. Save the project by choosing File ↠ Save.

9. Run the project again and click the Delete button without selecting a row in the URL list. No more crash!

List Box

Adding Code to the Listbox

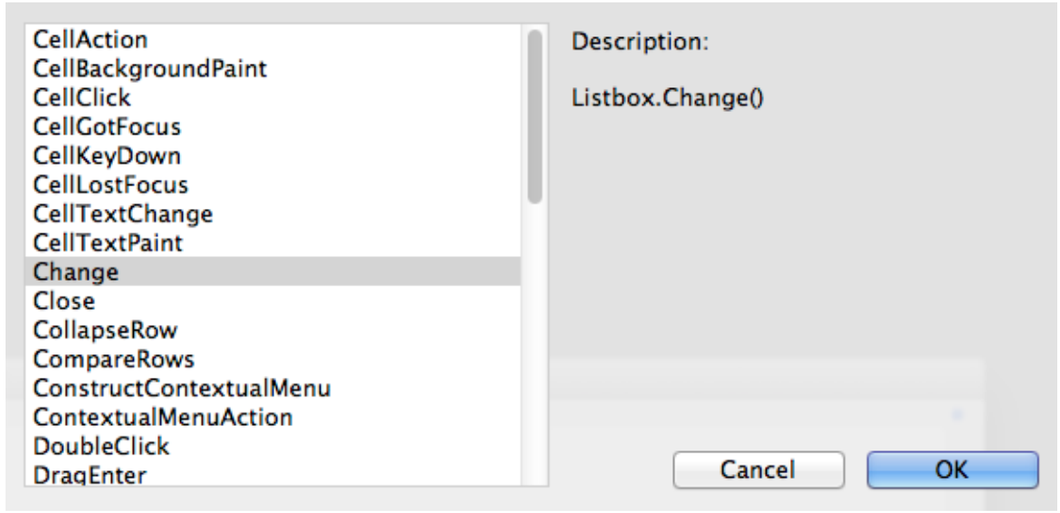
The last control to address is the Listbox. All URLs that are added are displayed in the Listbox. When the user clicks on a URL in the Listbox, your app should display the URL in the TextField.

Follow these steps to add the code:

1. On the window, double-click the **URLList** control.

The **Add Event Handler** window appears. A different list of event handlers appears (compared to what you saw with the PushButtons). You want to update the URL in the TextField when the user clicks on a row. It just so happens that the **Change** event handler for is called whenever the user selects a new row.
Select Change from the Event Handler list and click OK.
Notice the Navigator on the left updates to show the Change event underneath the URLList control and the code editor displays.

Figure 3.6 Adding a Change Event Handler



2. The text of the selected row in the Listbox is stored in the Text property. And you already know that the TextField.Text property contains the text that the user types in the field. You can also assign a value to this property to put text in the field programmatically.
   So your code looks like this:

```
URLField.Text = Me.Text
```

   Notice that you are using Me.Text to get the text from the

selected row in the Listbox. You could have also used URLList.Text. Why the difference?

Because you are in the Change event handler of the Listbox, you can refer to the Listbox methods and properties using Me. This allows you to rename the Listbox control without having to go back and change your code to use the new name of the Listbox.

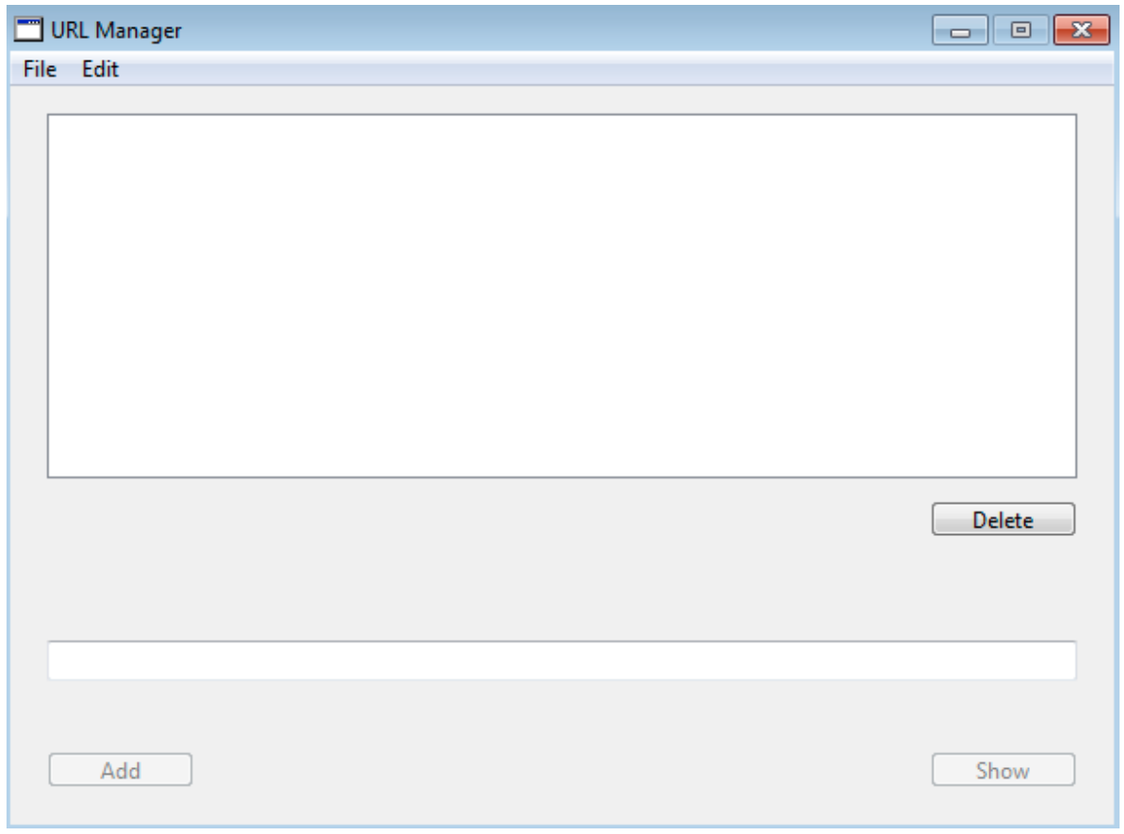
Save the project by choosing File ↠ Save.

Run the app to test it out. Type URLs in the URL field and click the Add button to see them appear in the URL list. After you have a few in the list, click on them and note that the URL appears in the URL field.

Quit the application to go back to the Editor (choose File ↠ Exit on Windows or Linux (or URLManager.debug ↠ Quit on OS X).

That's it. Your application is now complete.

Figure 3.7 URL Manager on Windows



Chapter 4

Next Steps

Now that you have a completed app, it's time to test it and maybe make a few enhancements.

Testing URL Manager

You Still Have to Test

Just because you have finished coding your application, doesn't mean you are finished. A good developer always thoroughly tests their applications to look for possible problems.

You already found and fixed one problem (clicking Delete when no row was selected). Do you think there are other problems to fix?

Run your application and play around with it a bit. Make a note of things you want to change. In the next section, you will add some improvements to URL Manager.

Improvements

Button Usage

Did you notice that there are times when the buttons in URL Manager probably should not do their action? For example, the Show button should not try to show a URL if one is not entered in the URL field. Also, you are not going to want to add a URL to the list if nothing has been entered in the URL field.

There are several ways to accomplish this, but one way is to disable the buttons when they should not be used.

Follow these steps to add this improvement:

1. On the window, select ShowButton, labelled “Show”. In the Inspector, turn the Enabled property (in the Appearance group) to Off.

Figure 4.1 Enabled Property

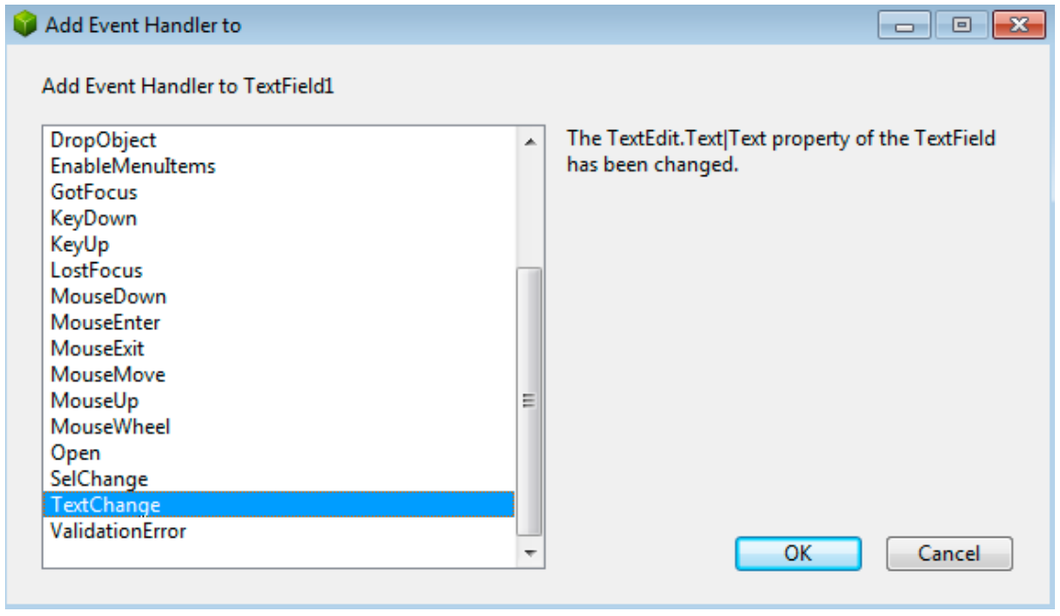
2. On the window, select AddButton, labelled “Add”. In the Inspector, turn the Enabled property (in the Appearance group) to Off.

3. Now you will add code to enable those buttons when there is text in the URL Field.

   On the window, double-click the **URLField** control.

   The **Add Event Handler** window appears. Here you are seeing yet another list of event handlers. Every control type has its own specific list of event handlers. In this case, we

Figure 4.2 Event Handlers for Text Field



want to disable AddButton and ShowButton when there is no text in the URL field. The **TextChange** event is called whenever the text in the URL field is changed, either by the user typing or by your code changing the Text property. Select TextChange from the Event Handler list and click OK.

Notice the Navigator updates to show the TextChange event underneath the URLField control and the code editor displays.

4. You want to add this code:

```
If Me.Text <> "" Then
  ShowButton.Enabled = True
  AddButton.Enabled = True
Else
  ShowButton.Enabled = False
  AddButton.Enabled = False
End If
```

This code checks the Text property of the Text Field (Me.Text) to see if anything is there. If there is text there, then both ShowButton and AddButton are enabled by setting their Enabled property to True. If there is no text, then both buttons are disabled by setting their Enabled property to False.

5. Save the project by choosing File ↠ Save.

Run the app to test it out. Notice that the Add and Show buttons are initially disabled. But try typing some text in the URL field. Those buttons immediately become enabled. And if you remove the text from the URL field, the buttons again become disabled.

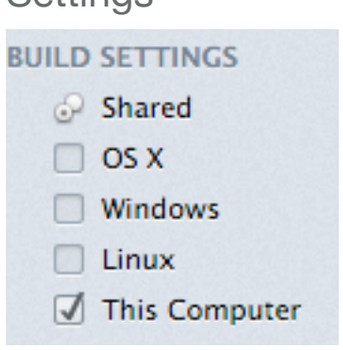
Building a Standalone App

Sharing Your Application

Now that you have created this fine application, you probably want to share it with the world. To do so, you need to build a standalone application.

Xojo lets you create desktop applications for OS X, Windows and Linux. The first thing you want to do is to decide which platforms you wish to build. You do this using the BUILD section of the Navigator.

Figure 4.3 Build Settings



First, check the box next to the platforms you want to build. By default, “This Computer” is checked so that you will at least create a build for the platform you are currently using.

Shared Settings has settings that are shared by all platforms. To see settings specific to each platform, click on the platform name.

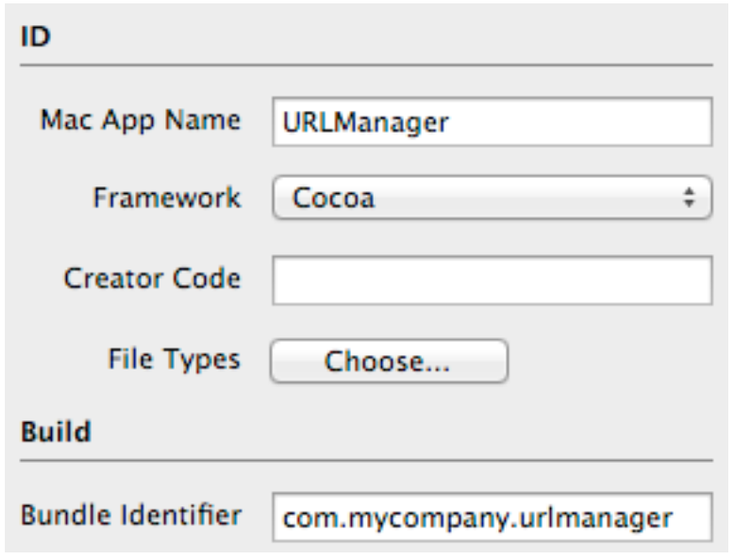
Shared Settings

The Shared Build Settings contain the version information and settings to use the build folder and include function names (for debugging purposes).

OS X

The OS X Build Settings allow you to specify the name for the Mac application, the architecture (either Carbon or Cocoa), the Creator Code, File Types and Bundle Identifier.

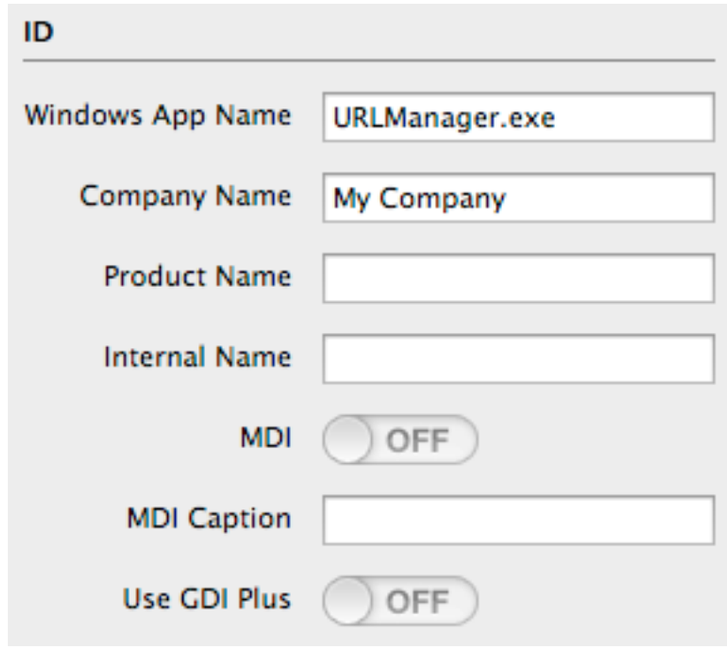
Figure 4.4 OS X Build Settings



Windows

The Windows Build Settings allow you to set the name for the Windows application, the Company, Product and Internal Name (which appear on the file properties for the application), whether you are using MDI (Multiple Document Interface) and if so, what the caption is. Lastly you can enable GDI Plus (enhanced Graphics Device Interface).

Figure 4.5 Windows Build Settings



Linux

The Linux Build settings only let you change the application name for Linux.

This Computer

The This Computer section displays the build settings for the platform you are currently using. For example, if you are using OS X, then This Computer shows you the OS X build settings.

Building Your Application

Once you have checked the platforms you want and adjusted the build settings you are ready to build your application. To do so, click the Build button in the toolbar (or choose Project ↠ Build Application from the menu). Xojo will create a standalone application for each selected platform.

Testing Your Built Application

In the folder containing your project, you will now see a folder called “Builds - TutorialDesktop.xojo_binary_project” and inside this folder will be folders for the builds for each platform.

Navigate to the build for your current platform and double-click the file to run it.

Section 4

All Done!

Congratulations

You have successfully completed the Desktop Tutorial and now have a fully functional application.

To continue on your journey of learning Xojo, you should next move on to the User Guide, which covers Xojo in its entirety.

You will also want to refer to the Language Reference, which covers the specifics of language elements, classes and other details of Xojo.