APPENDIX A

Introducing Xcode 4

When learning about any Apple product, it's important to keep in mind that Apple likes to change things (sometimes quite dramatically). This book was written based on Xcode 3.2 tools, which have looked very much the same for the past decade (like the versions of Xcode that preceded them, and the NeXT tools before those). In mid-2010, Apple released the first "developer preview" of Xcode 4, a release that, to borrow a phrase from Steve Jobs, "changes everything."

Xcode 4 pulls all of Xcode and Interface Builder into a single window. No more floating windows with tools and inspectors; instead, everything is consolidated into a single iTunes-like view, as shown in Figure A.1.

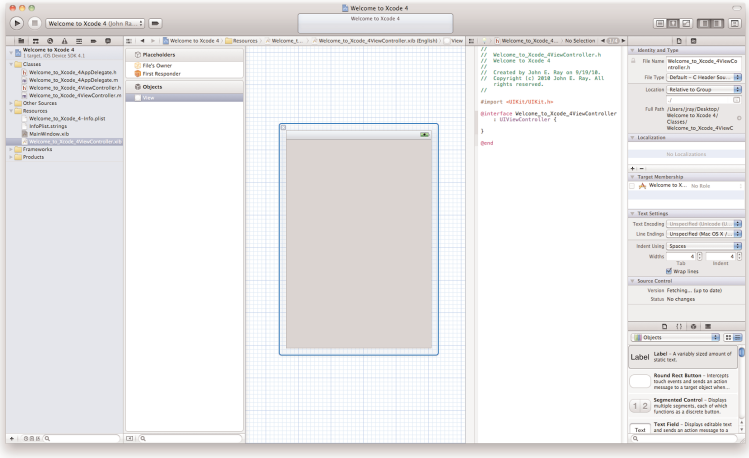


FIGURE A.1 Xcode 4 consolidates Xcode and Interface Builder (and all their associated windows) into a single view.

The bad news is that if you're used to using Xcode 3.2, Xcode 4 is a bit of a shock. The good news is that (almost) everything you know and love about the earlier Xcode is still present; you just have to know where to look. The best news? If you're already programming for iOS, you're through the tough stuff! You don't have to relearn iOS development; you just need to learn where Apple hid all of your tools!

In preview form, Xcode 4 still has many rough edges and features that feel incomplete, so it's difficult to predict when it will become Apple's primary development platform. Even after the release of Xcode 4.0, Xcode 3.2 will likely remain the platform of choice for many developers because of its maturity and stability.

I recommend that, as a new developer, you begin coding in Xcode 3.2 to avoid any surprises that come with adopting just-released software. On the other hand, I fully understand the desire to play with the latest and greatest toys from Apple, so I've created this appendix to help you get started in Xcode 4.

Let's walk through the tasks you need to perform when working on your iOS projects.

Installing Xcode

You install the developer tools exactly as in the past. Download the latest Xcode 4 release from http://developer.apple.com/ios and run the installer package. Unlike the 3.2.x versions, however, Xcode 4 will be installed in a folder called Xcode4 at the root of your hard drive. If you have an earlier version installed, it can coexist with Xcode 4 with no problem, as shown in Figure A.2.

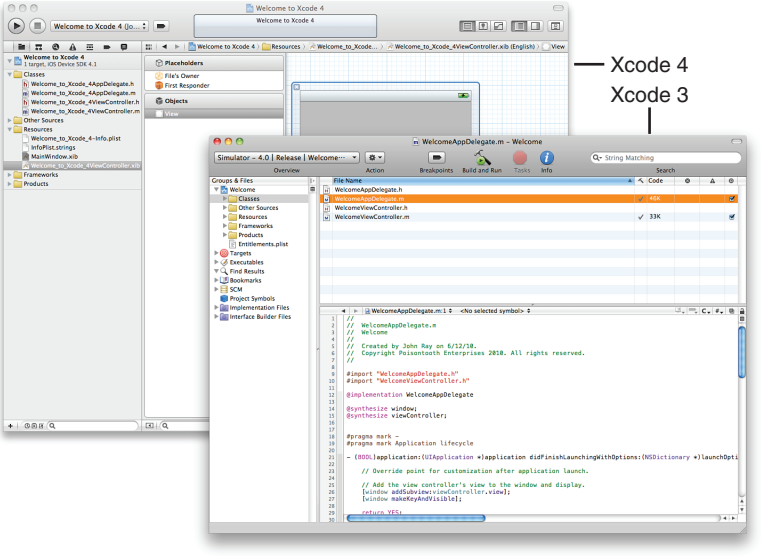


FIGURE A.2
If you install Xcode 4, it can coexist with earlier versions of the developer tools.

As with 3.2.x, the developer applications themselves are located in the Applications folder within the main installation directory (/Xcode4/Applications). Your main interest, obviously, will be in the Xcode application, which wraps up the functionality of Xcode and Interface Builder into a single integrated tool.

Creating a New Project

To create a new project, start Xcode and either use the shortcut on the Xcode welcome screen or choose File, New Project from the menu bar to begin a new iOS project. Choose Application from the iOS category on the left, and then pick from one of the available templates on the right, as shown in Figure A.3.

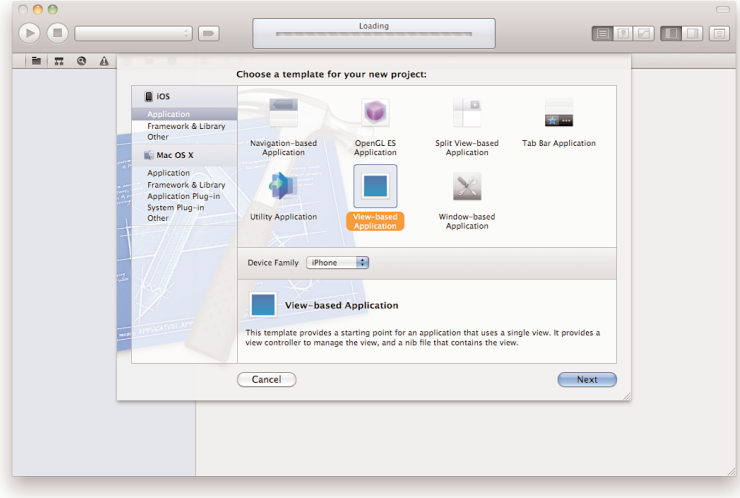


FIGURE A.3
Create a new iOS project within Xcode 4.

One small difference is that Xcode 4 refers to the deployment platform (iPhone, iPad) as the Device Family rather than the Product. Choose the appropriate device, and then click Next to proceed.

In a new step, Xcode prompts you for the product name and a company identifier. These are used to automatically set the bundle identifier for your application. Recall that the bundle identifier uniquely identifies your product on the App Store. Click Next to continue.

You are prompted for a project name. Then, after saving, you are presented with the Xcode 4 project workspace. The Xcode 4 workspace consists of four main areas, as shown in Figure A.4.

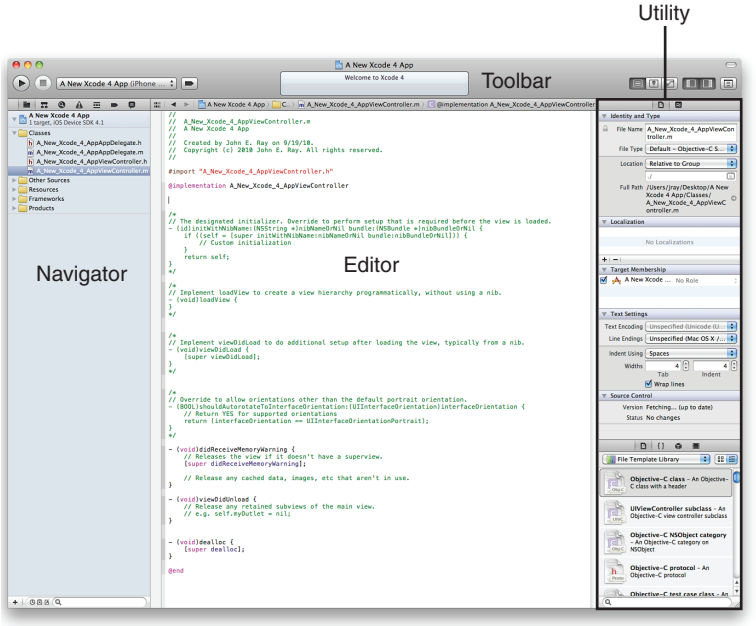


FIGURE A.4
Find your way around the new Xcode interface.

- **Toolbar:** Displays project status and provides easy access to common functions
- **Navigator:** Manages files and groups
- **Editor:** Edits project content
- **Utility:** Consolidates the Xcode and Interface Builder inspectors, help, and libraries

Did you Know?

A fifth area, the "debugger," appears below the editor when needed, but I tend to consider this to be an extension of the code editor.

You'll want to familiarize yourself with all the features offered within each of these areas because all will come into play during your development.

Navigating Your Code

After you've created or loaded a project, the Xcode workspace displays the project's files and groups in the upper-left corner, in an area called the navigator. The con-

tents of a file selected in the navigator display in the center of the Xcode window, as shown in Figure A.5. Although not all the same groups present in Xcode 3.2 are visible in the project navigator, everything you need to get started *is*.

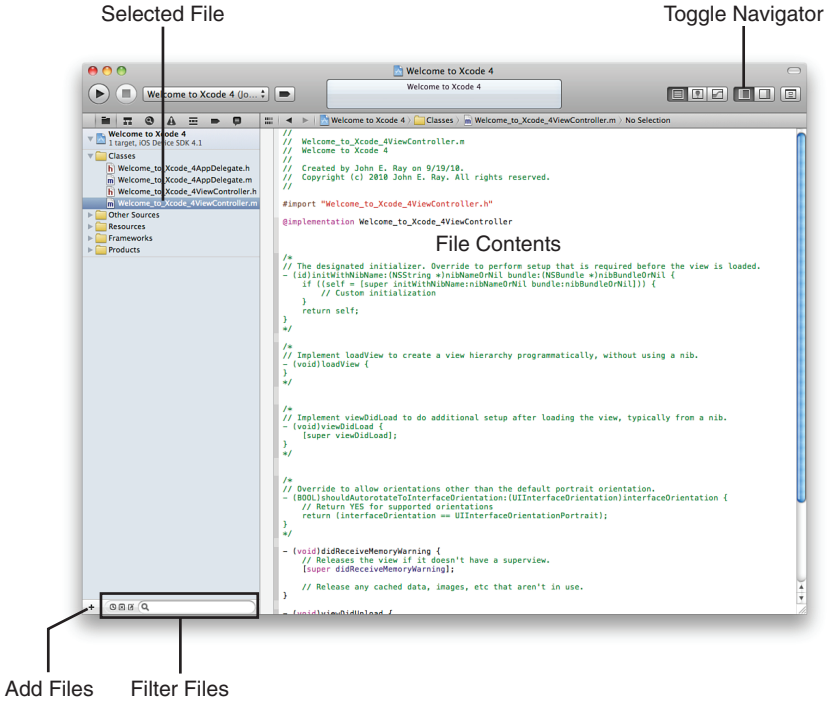


FIGURE A.5 Xcode displays files, file groups, and the contents of the files—just like you're used to.

Did you Know?

Xcode 4 eliminates smart groups. Instead, you can use the search field and icons at the bottom of the project navigator area to filter based on filename, recent edits, source control status, and save status.

In general, the editing works identically to what you've experienced in Xcode 3.2, but some enhancements make your development even easier. Code completion, for example, works even better—recognizing things such as interface definitions (try typing **`@interface`**) and providing structure for the file. The most notable enhancement, however, is the assistant editing mode.

Did you Know?

If you want to remove the navigator temporarily, you can drag its right edge to resize it, or just click the third button from the right on the Xcode toolbar. This button toggles the visibility of the navigator off and on.

Using the Assistant Editing Mode

The assistant editing mode displays related "counterpart" files side by side with the file you are editing. In other words, if you're editing a .m implementation file, the corresponding .h interface file will be available beside it, as shown in Figure A.6. Editing the interface file? The implementation file is automatically selected and made available.

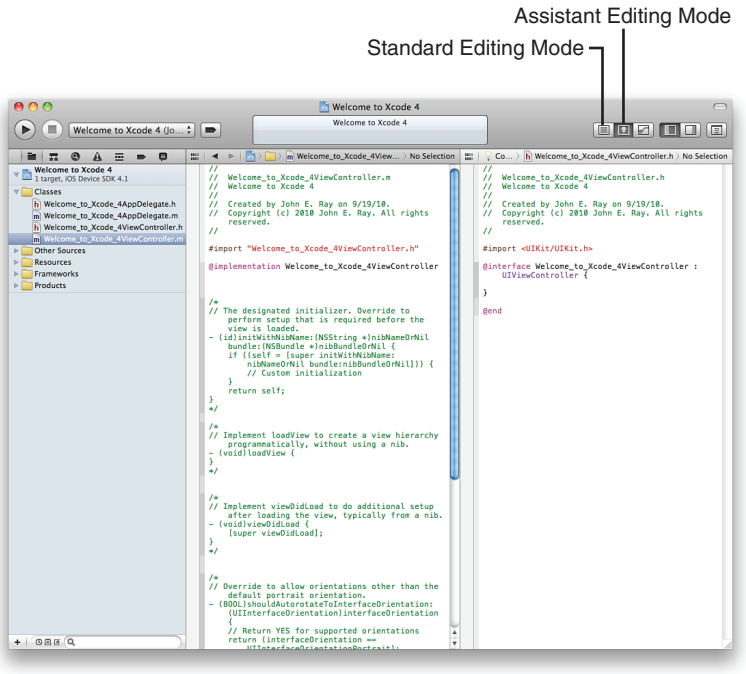


FIGURE A.6
The assistant editing mode makes your file's counterparts immediately available.

You can toggle between the standard editing mode and the assistant mode using the two leftmost icons in the upper-right corner of the Xcode window.

Adding Files

To add new files (classes, for example) to the project, you can follow the same process that you did in Xcode 3.2 (choose File, New File from the menu bar), you can click the + button at the bottom of the navigator, or you can open and drag and drop from the File Template Library. To show the File Template Library, click the second icon from the right in the top right of the Xcode toolbar. This hides and shows the Xcode utility area.

When the utility area is open, you'll notice that near the bottom is a pane that displays a set of four icons. This is the Library pane, and the icons represent the File Template Library, Code Snippet Library, Object Library, and Media Library. Click the icon for the File Template Library and your screen should resemble Figure A.7.

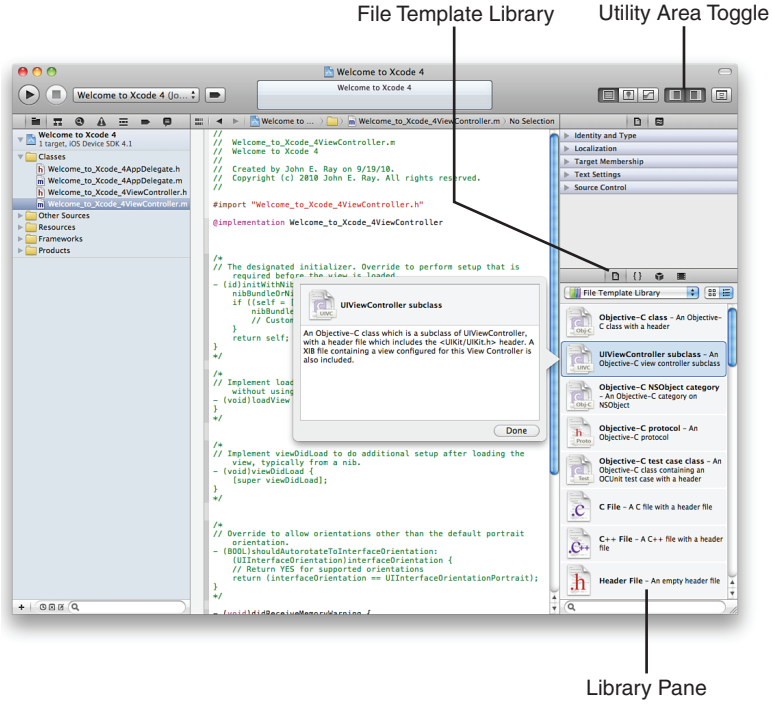


FIGURE A.7
Use the File Template Library to quickly add new files to your project.

To use the templates, drag an icon from the File Template Library list into the project navigator. You are prompted for a filename for your new class, as shown in Figure A.8.

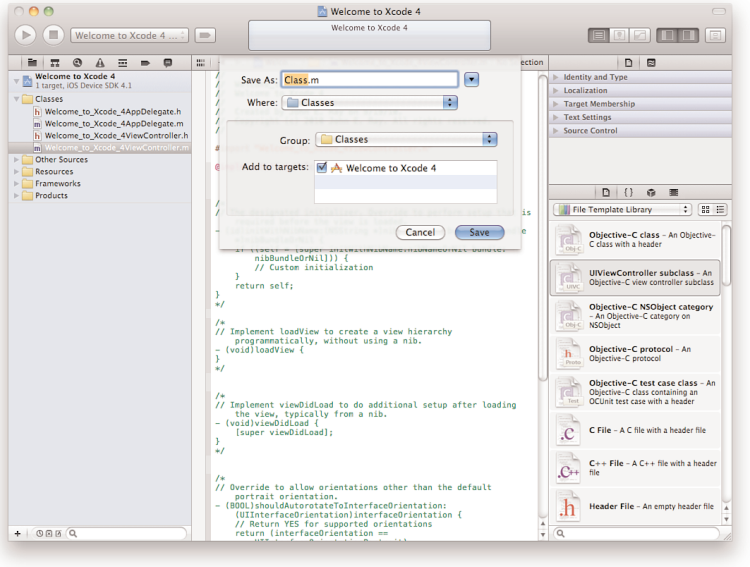


FIGURE A.8
Drag and drop to create new classes.

When appropriate, Xcode even creates the interface and XIB files to accompany your new class.

Adding Frameworks

As you build more complex projects, you will need to include more iOS frameworks in your applications. In Xcode 3.2, you easily added frameworks much like you would add a new file. In Xcode 4, this process has been made a bit more obscure. To add an existing framework to your project, select the main project icon within the Xcode navigator, and then the icon for your application within the Targets section to the right of the navigator. Finally, click the Build Phases tab at the top of the editor area. Your screen should now resemble Figure A.9.

Make sure the Link Binary with Libraries section is expanded, and then click the + button at the bottom of the section. Finally, you are prompted to choose the framework you want to use. Select it, and then click Add, as shown in Figure A.10.

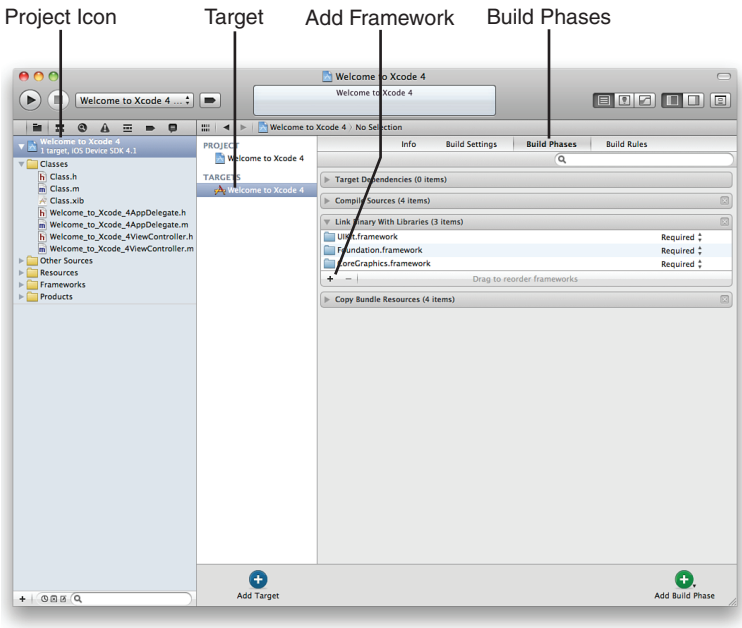


FIGURE A.9
Navigate to the build phases for your project.

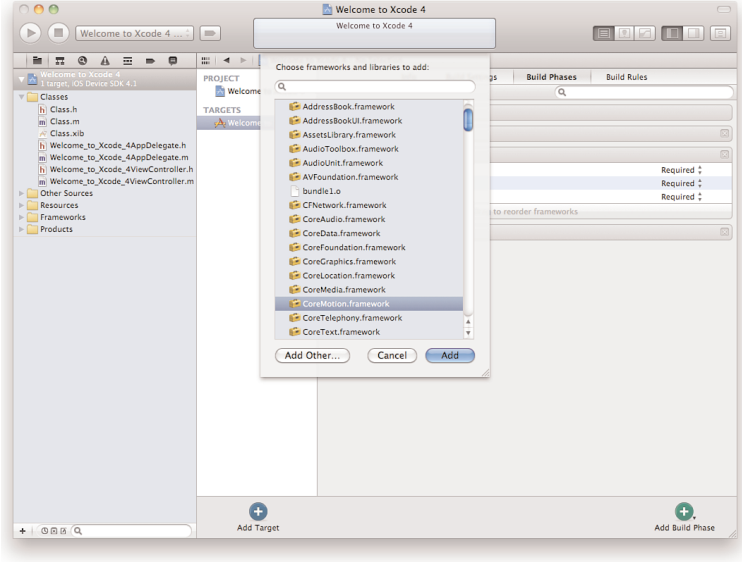


FIGURE A.10
Choose the framework you want to add.

After adding the framework, you'll notice that a new framework icon appears in the project navigator. You can drag this into the Frameworks group for safekeeping.

Setting Project Options and Icons

You can still edit your project plist file to make many of the standard configuration changes that you've grown accustomed to in Xcode 3.2. In addition, Apple has "prettified" many of the common settings and made them available with an Info section for the project. To access this area, follow the same steps as if you were adding a new framework (select the project icon, then the target icon), but then click the Info tab, as shown in Figure A.11.

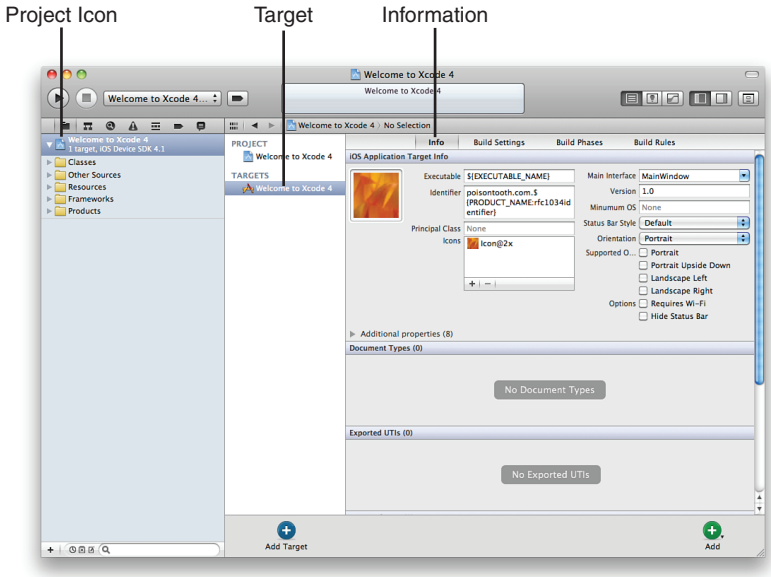


FIGURE A.11
Access common project settings in the Info area.

Using these options, you can configure many of the most frequently needed settings for your project. You can even add your application icons by simply adding them as resources and then clicking + under Icons to choose which you want to use.

Getting Help

One Xcode feature I've always loved is how easy it is to get help—and the help system is even better in 4.0. You can still use all the same help tools you know and love from Xcode 3.2; specifically, you can access the full, searchable, developer documentation by choosing Help, Developer Documentation from the menu. You can also Option-click keywords in your code to display a pop-up Quick Help window that contains context-aware information about what you are working on.

In Xcode 4, you can use the utility area to toggle on and off a display of Quick Help information that will remain visible while you are editing. To do this, click the second button from the right in the toolbar to display the utility area, if it isn't already visible. Next, click the second icon from the right at the top of the utility area (two wavy lines in a black square, called Symbols). This opens the Quick Help pane, which displays information about the objects you're using, as you use them, as shown in Figure A.12.

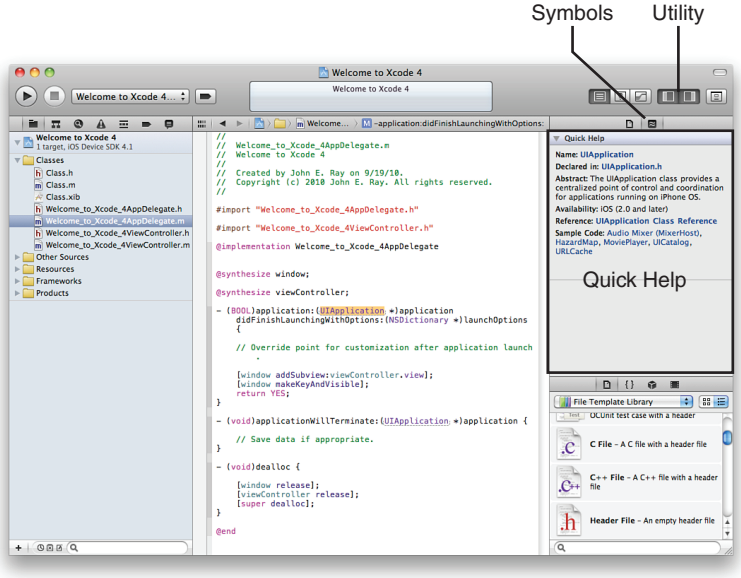


FIGURE A.12
Use the utility area's Quick Help pane to display context-sensitive help information.

Building Interfaces

The biggest and best new feature of Xcode 4 is Interface Builder integration. More precisely, Interface Builder is gone; you can complete a project from start to finish entirely in Xcode 4, without having to switch between applications. To see this in action, all you have to do is click an XIB file in your project. The editor area transforms into an interface editor, as shown in Figure A.13.

After the initial delight of seeing the interface appear directly in Xcode, you'll probably start asking where the heck did they put everything. That's a great question. Everything you're used to using—the Document window, inspectors, and Object Library—are all there; they're just hiding.

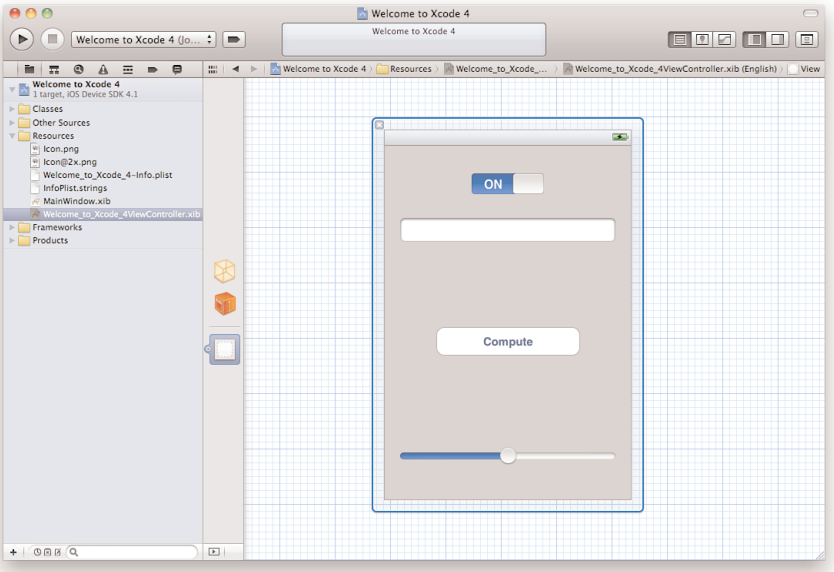


FIGURE A.13
The features of Interface Builder are now integrated into Xcode 4!

Accessing XIB File Objects

The objects that typically would be represented in the Interface Builder Document window are now located between the navigator and the editor. By default, only icon representations of the objects are visible, but this is easily corrected by clicking the arrow at the bottom of the section, as shown in Figure A.14.

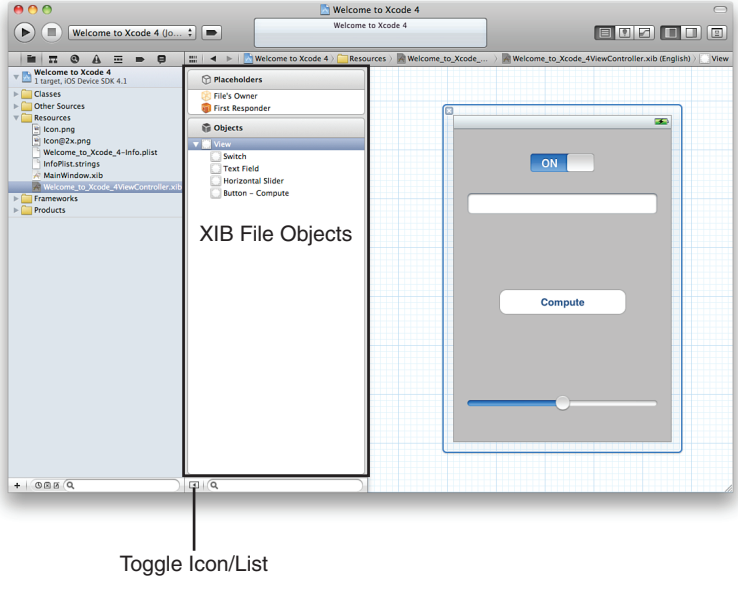


FIGURE A.14
The Document window is replaced by a list of objects between the navigator and the editor.

You can interact with these items the same way you would in the Interface Builder Document window. You can Control-drag to or from them to make connections and expand view hierarchies to see their contents. Of course, this isn't going to do you much good unless you can actually add items to interface.

Using the Object Library

To access the interface Object Library, open the utility area, and then click the cube icon at the top of Library pane, as shown in Figure A.15. Now things should be looking familiar.

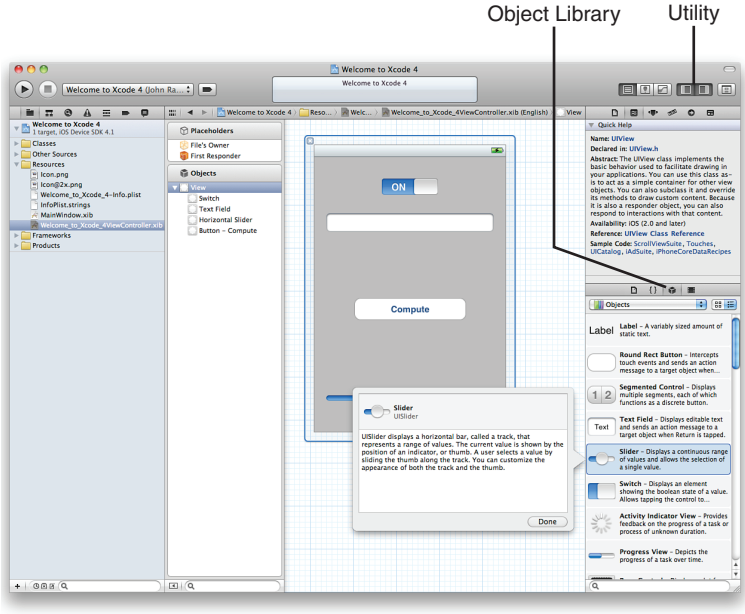


FIGURE A.15 Access the Object Library from the utility area.

You can drag items from the library directly into your interface design, just as you could in Interface Builder. You can also use the two buttons near the top of the pane to switch between list and icon views, and the search field at the bottom to quickly focus on a specific interface object.

> You can now drag objects into the editor area outside your main view. This can be helpful for preparing objects before adding them to your view or for experimenting with different interfaces. The objects *are* in your XIB file and will be instantiated, but will not be loaded as part of the main view.

Did you Know?

Accessing Inspectors

The last missing piece to the Interface Builder puzzle is the location of the inspectors. You've grown accustomed to accessing the attributes, size, connections, and identity inspectors throughout the projects in this book. In Xcode 4, all of these are located in the utility area. To find them, first make sure the utility area is visible, and then look to the last four icons at the top of the utility area. These icons represent the four missing inspectors: Attributes, Size, Connections, and Identity, as shown in Figure A.16

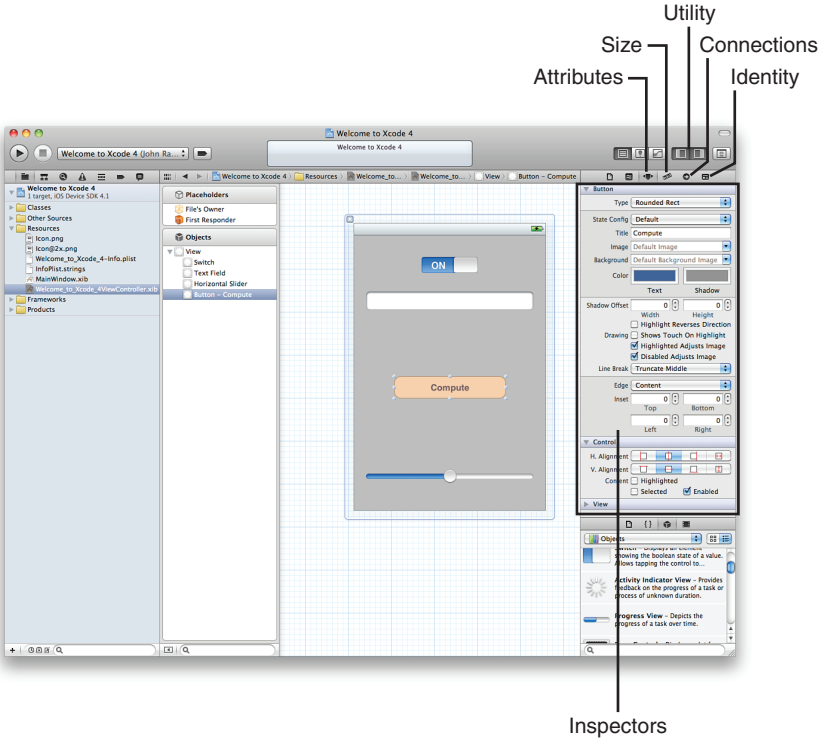


FIGURE A.16
Use the icons at the top of the utility area to switch between the different interface object inspectors.

By the Way

Hey, Those Icons Weren't There a Second Ago!

Nope, they weren't. The utility area changes based on the type of file you are editing. If you are editing code, you'll see an inspector that shows information about the file itself. Only in the interface-editing mode are the interface object inspectors visible.

You can use the inspectors exactly as you did in Interface Builder; little has changed in terms of their content, their location is just updated.

Connecting to Outlets and Actions

You already have enough information to connect to objects and outlets using the techniques of the older Interface Builder application, but now that Xcode 4 integrates interface building and coding, you can do a couple of cool things that previously weren't possible.

First, you can connect objects directly to outlets and actions in code. To do this, you must first define your outlets and actions in an interface (.h) file. You're used to this; you've done it dozens of times throughout the book. After you've properly set up your file, open the corresponding XIB file in the editor, and then switch to the assistant editing mode. The editor refreshes to display the .h file to the right of the interface design. Now, as counterintuitive as it might seem, you can Control-drag from one of the objects in the XIB to an `IBOutlet` or `IBAction` definition in the code. The line highlights, as shown in Figure A.17. Release the mouse button and you've just made a connection! You can do the same thing from the Connections Inspector to target a specific action.

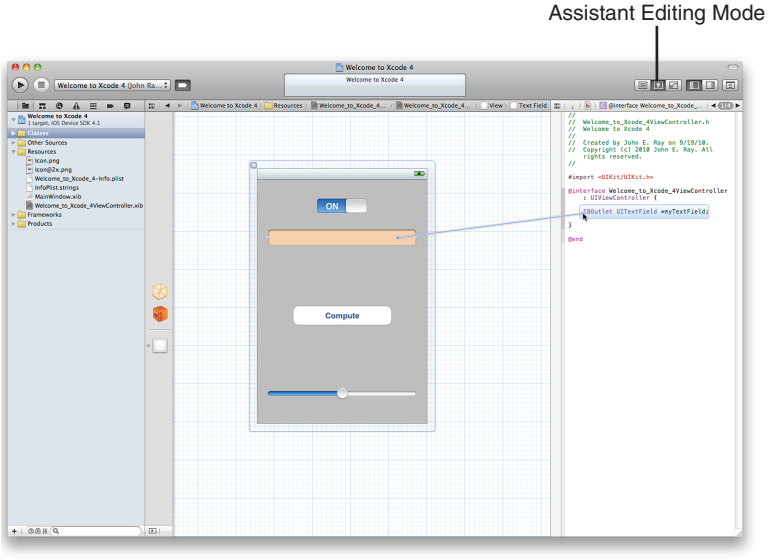
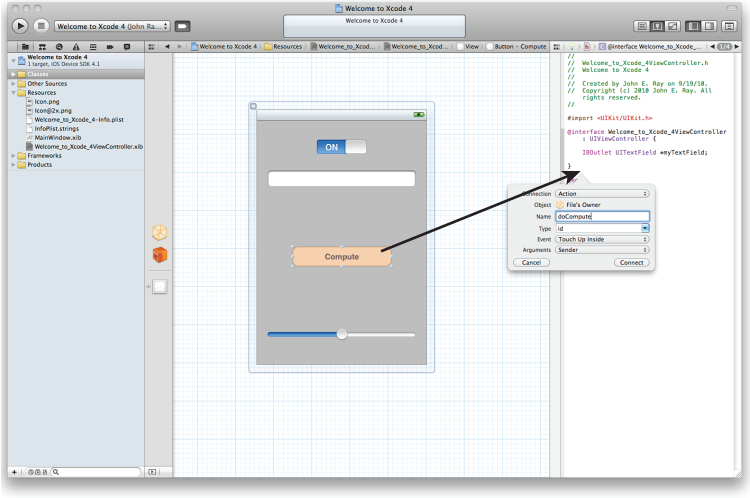


FIGURE A.17
Connect directly to outlets and actions in your code.

The second new trick is that you can use your interface to actually create code for you! Xcode 4 will help write and connect your outlets and actions all at once, just by clicking and dragging. To use this technique, you don't need to define anything ahead of time. Just open the XIB in the editor and switch to the assistant editing mode. Next, Control-drag from an object in the interface to the location in the code where it should be inserted. When you release your mouse button, you'll be prompted for the type of connection to make (outlet or action) and given the opportunity to name it, as shown in Figure A.18.

FIGURE A.18
Write code without typing!



When finished configuring the addition, click Connect, and the code is added to your interface (.h) file and the connections are automatically built in your XIB file.

Watch Out!

It is still unclear whether Xcode 4 will add additional supporting code beyond just outlets and actions. For example, in the initial preview releases, an outlet that you add in this manner will not have corresponding `@property` and `@synthesize` lines added, nor will it be released in the `dealloc` method. Personally, I recommend continuing to add outlets and actions by hand so that you have complete control over the process and know exactly what is happening behind the scenes.

Building an Application

Building (compiling) an application works similarly to how it did in Xcode 3.2. Use the drop-down menu at the top of the window to choose whether to target the simulator or an actual device, and then click the Run button to build and run the application.

Using the Issues View

If you encounter any errors during the compilation, they'll be denoted with the same warning and error symbols that you're familiar with from earlier versions of Xcode. One difference, however, is that the navigator will switch to an issues view, as shown in Figure A.19. You can access the issues view mode manually by clicking the fourth icon at the top of the navigator (the warning symbol). Click each line in the issue list to jump to the corresponding problem in the code.

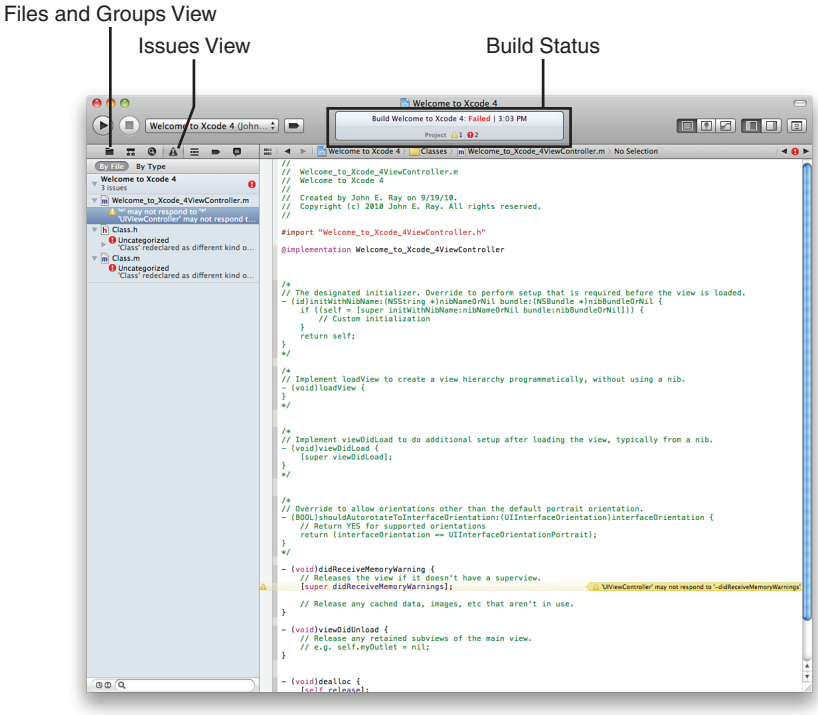


FIGURE A.19
Use the issues list to display problems found during a build.

To return to the standard navigator view of files and groups, click the folder icon at the top left of the navigator.

Preparing a Device for Testing

We'll conclude our exploration of Xcode 4 with one of the handiest new features that Apple has made available. Remember in Hour 1, "Preparing Your System and iPhone for Development," you spent 15 to 20 minutes creating a development profile on Apple's site, downloading certificates, and all that fun stuff? Xcode 4 makes it simple to get a device provisioned for testing fast and without needing to visit the iOS developer portal.

To prepare a device for testing, first plug in the device and open the Xcode organizer by choosing Window, Organizer within Xcode. When the Organizer window opens, click the Devices button at the top, and then choose your new device from the left side of the window. When the device is chosen, you'll notice an Add to Portal button at the bottom. Clicking this button prompts you for your iOS developer portal login. Enter your information, and then click Log In, as shown in Figure A.20.

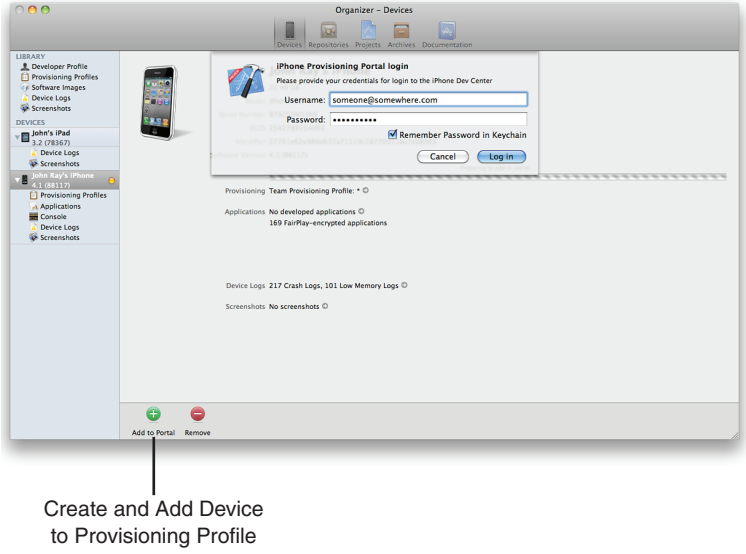


FIGURE A.20
The Organizer can quickly provision a device for development.

This simple step will add your iOS device to the Apple portal and create a wildcard provisioning profile called Team Provisioning Profile and install it on your device. When it's complete, you can immediately use it for development!

Making the Leap

The purpose of *Sams Teach Yourself iPhone Application Development in 24 Hours* isn't to be an end-all guide to Xcode. The Xcode application suite has many more features than can easily be conveyed in a few hours. Instead, I want to give you the tools you need to get real work done. When you're familiar with the basics, everything else will fall into place. The same goes for this appendix and Xcode 4. Many, many features are offered in Apple's latest incarnation of the development environment, and I encourage you to explore the tool to find additional features that might aid your development workflow. Although it's beyond the scope of this book, Xcode 4 "workspaces" (File, New, New Workspace) provide a way to collect groups of projects together. Doing so can prove very helpful, especially if they share classes.

The Xcode documentation (located under the Help menu) will provide you with most anything you could ever need to know about the application features. Apple has even included an Xcode 4 Transition Guide (also located under the Help menu), which aims to ease the transition from Xcode 3.2 to Xcode 4. I highly recommend that you review all this documentation and work through a few small projects from this book in Xcode 4. There appears to be no immediate rush to force developers into Xcode 4, so make the transition at your own pace. The most important thing is to learn the development concepts, not to spend all of your time focusing on (or fighting with) the tools.