Jiva DeVoe

Cocoa Touch™ for iPhone® OS 3

Developer Reference

www.wileydevreference.com

Cocoa Touch™ for iPhone® OS 3

Cocoa Touch™ for iPhone® OS 3

Jiva DeVoe

WILEY

Wiley Publishing, Inc.

Cocoa Touch™ for iPhone® OS 3

Published by
Wiley Publishing, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Published by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

ISBN: 978-0-470-48107-3

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (877) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Library of Congress Control Number: 2009937274

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

*For my wife, Dawn, and my children, Robert, Alex, and Izzy.
You are, and always will be, the most important
people in the world to me.*

About the Author

Jiva DeVoe has been writing software for nearly 25 years, starting with his Commodore VIC-20 using BASIC and assembly language, and gradually working his way through C, C++, Python, Ruby, Java, and finally, Objective-C. In 2001, he founded Random Ideas, LLC, a software company dedicated to building great applications for the Mac. When the iPhone SDK was announced, he was honored to be selected as one of the earliest developers to have access to the SDK, and using it, he developed several applications that were available in the iTunes App Store when it launched on July 11, 2008. Since that time, his iPhone applications have received several awards — including being chosen as among the Top 100 apps and games in the App Store — and been featured as Apple Staff Picks and in Apple advertisements. Today, Jiva continues to work full time for his company, developing great iPhone and Mac applications. He lives with his wife, three children, and two basset hounds in the desert of Arizona.

Credits

Acquisitions Editor
Aaron Black

Executive Editor
Jody Lefevere

Project Editor
Martin V. Minner

Technical Editor
Dallas Brown

Copy Editor
Marylouise Wiack

Editorial Director
Robyn Siesky

Editorial Manager
Cricket Krengel

Business Manager
Amy Knies

Senior Marketing Manager
Sandy Smith

Vice President and Executive Group Publisher
Richard Swadley

Vice President and Executive Publisher
Barry Pruett

Project Coordinator
Katie Crocker

Graphics and Production Specialists
Andrea Hornberger
Jennifer Mayberry
Mark Pinto

Quality Control Technician
Rebecca Denoncour

Proofreading
Christine Sabooni

Indexing
BIM Indexing & Proofreading

Media Development Project Manager
Laura Moss

Media Development Assistant Project Manager
Jenny Swisher

Contents

Preface

My goal in writing this book was to provide a comprehensive toolkit for both new and experienced iPhone developers. Its focus is intended to be primarily on the new technologies of iPhone OS 3, but it includes enough general iPhone development material that even a new developer to the platform will receive a great deal of benefit from reading it.

Writing a book about Cocoa Touch programming is an incredible challenge. It's very difficult to judge the technical capability of the typical reader of a book like this. Does the reader already know Objective-C? Has he or she already been developing for Mac OS X? These are the first questions that I had to ask myself when I began to work on this project.

In the end, I decided that with the development of iPhone OS 3, it was a unique opportunity where the new and experienced reader intersected and therefore, I thought it was an ideal time to write a book that would be useful to them both.

So I decided that this book would contain some introductory material, but that it would primarily focus on the new features of iPhone OS 3. In this way, it provides an excellent bridge for learning the technologies of the new operating system, as well as an introduction to general iPhone development.

As a reader, you are expected to already be somewhat familiar with Objective-C. Additionally, you should be somewhat familiar with either Mac OS X, or iPhone development, though you need not be an expert.

If you're completely new to the platform, I suggest picking up a book specifically on Objective-C to complement this book. There are several such books on the market, but I recommend one that approaches Objective-C from the point of view of learning the core language itself, rather than one that mixes an introduction to Objective-C with other topics. After you've worked through enough of that book to feel comfortable with the syntax of the language, you can begin this book at Chapter 1.

If you've already been developing for Mac OS X, but haven't done any iPhone development, you probably already know Objective-C and are familiar with many of the tools used in iPhone development. However, while iPhone development is similar to Mac OS X development, it's not exactly the same. So I suggest that you skip Chapter 1, and start reading Chapter 2, where you build a basic iPhone app from scratch.

Finally, if you're already an experienced iPhone developer, you already know all the basics involved in iPhone development; you're just here for the iPhone OS 3 material. I suggest you start at Chapter 5. All the chapters were written with the new and updated iPhone OS 3 API in mind, and so even if you are already familiar with UIViewController and friends, you will find new material there.

In these ways, this book provides an excellent extension to your library if you already have books on any of these three subjects.

With regard to conventions used within this book, I've tried to be reasonably consistent, and also tried to generally err on the side of Apple conventions when prudent. The only notable exception has been in my use of the term "method" to indicate functions on instances and classes. Apple generally prefers the term message. This is in part due to the influence of Smalltalk on Objective-C.

Also with regard to method calling conventions, Objective-C is known for being particularly verbose in its method names. As a result, some abbreviation conventions have been adopted when writing about particular methods. I have chosen to follow two standards. The first, I use when the usual use of the method in question is in implementing your own, or in overriding it in a subclass. This is most commonly the case when dealing with delegate methods. In these cases, because it's helpful to know the entire signature of the method, in order to write your own implementation, I have chosen to include the entire method signature. So, for example, these will be written as `-(BOOL)foo:(NSString *)bar withBaz:(NSString *)baz`. As you can see, in this example, we have a method called `foo`, which takes two parameters of type `NSString *`, one called `bar` and a second called `baz`, and which returns a `BOOL` value.

Alternatively, in the case where your typical use will be only to use the method in your own code, since Xcode generally automatically completes the types of the parameters for you, I have chosen to use the abbreviated form of writing the method signatures. In the case of the previous method, that means it's written `foo:withBaz:`. Notice that wherever a parameter is inserted, a colon holds the place of the parameter.

When referring to keyboard shortcuts, I opted to use the term Command key or the ⌘ symbol to indicate keyboard shortcuts using the key directly to the left of the space bar on most Apple keyboards. You may also know this as the "Apple" key, as until only a few years ago, it included a small Apple logo on it. Additionally, the key next to the Command key has been called the Option key and the key next to that, the Control key. These should all be consistent with Apple documentation conventions.

When discussing the use of menus in Xcode, I've used the technique of separating the nested menu items using arrow notation. So, to describe the New File sub item of the File menu, it will be written as File ➪ New File.

Finally, with regard to sample code, in chapters where I have instructed you to build specific full projects, I have generally tried to include full listings for the code. In cases where I have not, you can always download the projects, complete with artwork and other supporting files, from the book's Web site, located at `http://www.wiley.com/go/cocoatouchdevref`. There are also chapters where it didn't really make sense to create a full project to demonstrate a technology. In these cases, the code listings are snippets that you can use as a basis for your own code. Because these snippets don't comprise fully functional projects, there will not be example projects for them on the Web site.

I hope that you find this book as enjoyable an experience to read as I had writing it. To me, the mark of a good technical book is that it doesn't sit on my shelf. It holds a place of honor by my desk because I keep returning to it, time and again. I hope that this book holds such prestige in your library, and that it becomes a dog-eared, cover-torn, page-scribbled-on reference that remains useful to you for years to come.

Jiva DeVoe

`book@random-ideas.net`

Acknowledgments

Writing this book has been one of the most challenging and exciting projects I have done in my career, but I could not have done it without the aid and support of some specific individuals whom I would like to thank.

First, I'd like to thank my friend and technical editor, Dallas Brown, of HashBang Industries, who took up the challenge of correcting my mistakes and keeping me honest. Your time spent, and your thoughtful comments were excellent.

Along that same line, I'd also like to thank my friend, Brad Miller, of Cynical Peak Software, who also provided welcome criticism and an extra set of eyes on several chapters, and who always seems to be awake and online at the same times I am.

For pressing forward with the book, even with a tight schedule, I'd like to thank all the folks at John Wiley & Sons. I look forward to working with you on new projects in the future.

For teaching me to marvel at the wonders of technology and encouraging me to pursue my dreams in computers, I'm thankful to my father, Robert A. DeVoe.

Many thanks to my children, who have endured these several months of my working late without complaint. You have earned your trip to Disneyland! It is for you that I do everything I do.

Finally, and most importantly, I'd like to thank my wife, for her unerring support, not just in this project, but in all my work. Without her, this book could not have been finished. You lift me when my spirits are low and tired, and inspire me to keep reaching for new accomplishments and goals. I can't thank you enough.

I

Getting Started with the iPhone

In This Part

1 Getting to Know Your Tools

They say that when a craftsman finds a tool that he loves, over time it becomes an extension of him. He learns its idiosyncrasies inside and out, backwards and forwards, and this leads to a relationship that transcends simple use of the tool and instead becomes more involved. The tool becomes an extension of his hand, an extension that he can guide with an almost extrasensory vision.

You are fortunate then, that Apple provides you with an excellent set of free tools for developing software for the iPhone. They are tools that have evolved over the last 15 years of Objective-C development, first on NeXT computers, then on Mac OS X, and now for iPhone.

In this chapter, you will take a brief look at these tools and learn where you can find more information about them. They are incredibly powerful tools that seem to be unique in software development, both for their ability to provide enough power to enable incredibly complex software systems to be developed, and also because they seem to know just when to stay out of your way and simply provide a great text-editing environment for you to write code in.

Unfortunately, a comprehensive description of every last feature of these applications is beyond the scope of this book, and so I won't be delving into them in great detail. My main goal here is simply to introduce you to the tools so that you're familiar with them.

1 **In This Chapter**

Becoming familiar with Xcode and Interface Builder

Introducing Xcode

The cornerstone of iPhone software development is the Xcode integrated development environment, or IDE. Xcode originated on NeXT Step computers as Project Builder. Over the years, it has gone through many revisions to finally arrive at the version that is available to you today. It uses GCC as its underlying compiler technology and provides many sophisticated features found in modern IDEs today, such as code completion, re-factoring, and sophisticated code navigation. Interestingly, it also has one of the best cross-platform compiling capabilities of any modern IDE. With it, you can compile for Intel, PowerPC, iPhone OS, or even (with third-party tools) Microsoft Windows. You can do all of this simply by configuring targets in the IDE.

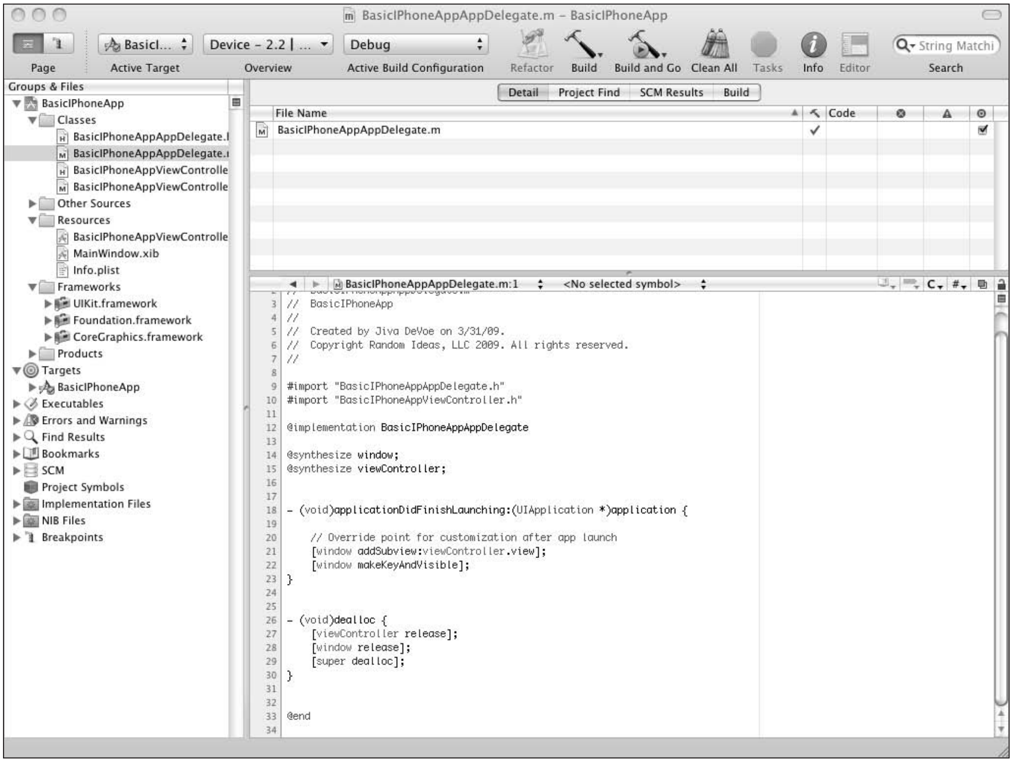
NOTE

To download Xcode, all you need to do is sign up for a free developer account on the Apple Web site. You can do this at `http://developer.apple.com/iphone`.

Figure 1.1 shows the main Xcode window. In it, you can see the left panel, which shows the file organization view. From here, you can drag and drop files into your projects or organize them by groups. It also provides the ability to organize your files by Smart groups, which are built using search queries and can be useful for looking for particular files in your projects.

Figure 1.1

The Xcode interface



```
//  BasicIPhoneApp
//
//  Created by Jiva DeVoe on 3/31/09.
//  Copyright Random Ideas, LLC 2009. All rights reserved.
//

#import "BasicIPhoneAppAppDelegate.h"
#import "BasicIPhoneAppViewController.h"

@implementation BasicIPhoneAppAppDelegate

@synthesize window;
@synthesize viewController;


- (void)applicationDidFinishLaunching:(UIApplication *)application {

    // Override point for customization after app launch
    [window addSubview:viewController.view];
    [window makeKeyAndVisible];
}


- (void)dealloc {
    [viewController release];
    [window release];
    [super dealloc];
}


@end
```

On the right side, you can see the main editing window. It is in this window that you will do the majority of your programming work. At the top of the text-editing window, you can see some drop-down menus that enable you to quickly jump to any recently opened files. The second drop-down menu from the left enables you to quickly jump to any method in the current file. Holding down the ⌘ key and using the right and left arrow keys enables you to quickly navigate backwards and forwards through the file history. Additionally, holding down the Option and ⌘ keys together and pressing the up arrow key enables you to quickly swap between the implementation and header files for the currently active compilation unit.

The Xcode editor is quite sophisticated and can be configured with a variety of shortcuts and hot-keys that make your editing much easier and faster. For example, using the Option key and the right and left arrow keys enables you to quickly jump from word to word in your code. Holding down the ⌘ key and using the left and right arrow keys enables you to quickly jump to the front or beginning of the current line.

It's a good idea to learn the intricacies of the Xcode editor inside and out, because it is such a fundamental tool to everything that you will be doing as an iPhone OS developer.

Introducing Interface Builder

The second major component of the Xcode programming environment is the graphical user interface (GUI) builder called Interface Builder, shown in Figure 1.2. It is this application that you will use to draw your GUI for your application and connect your buttons to actions in your code.

Many developers coming to iPhone development from other environments are sometimes confused by Interface Builder because it doesn't generate any code. However, this is an asset and not a liability. IDEs that simply generate code tend to be more difficult to work with over time, as the code that they generate becomes out of sync with the user interface. Interface Builder uses more of a metadata style approach. This means that you tell it that you want to instantiate an object of a given type, and when your nib is loaded, it goes and finds the class for that type, instantiates it, and attaches the outlets and actions that you have configured to the appropriate places. It does not serialize actual instances of your objects, nor does it generate code that is compiled.

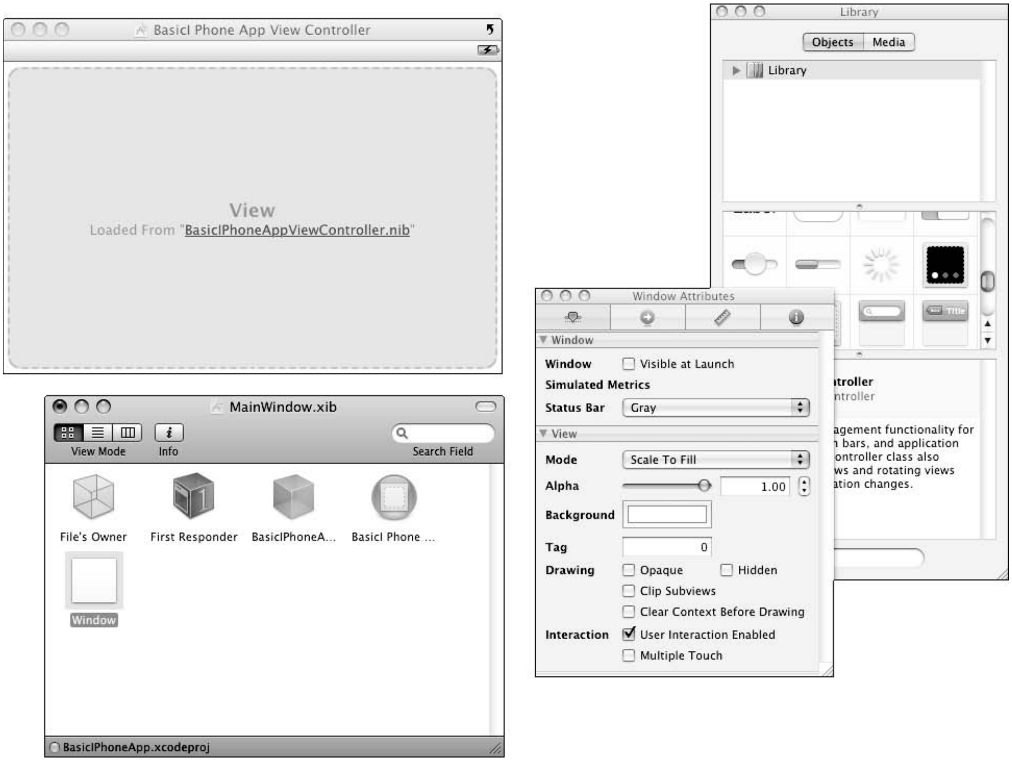
NOTE

The files that Interface Builder saves are referred to as nib files. This stands for NeXT Interface Builder. The file extension of the nib files used with the iPhone is .xib. This is to differentiate them as containing XML versus the older, original NeXT format.

I will talk about Interface Builder in a bit more detail in Part II. For now, the important thing to know is that you can start Interface Builder either separately by simply launching it or by double-clicking any of the `.xib` files in your project.

Figure 1.2

Interface Builder



▼ Window

Window ☐ Visible at Launch

Simulated Metrics

Status Bar: Gray

▼ View

Mode: Scale To Fill

Alpha: 1.00

Background

Tag: 0

Drawing ☐ Opaque ☐ Hidden

☐ Clip Subviews

☐ Clear Context Before Drawing

Interaction ☑ User Interaction Enabled

☐ Multiple Touch

Summary

In this chapter, I introduced you to Xcode and Interface Builder. In the upcoming chapters, you will use these tools in much more depth, but I wanted to simply give you a brief overview of what they look like so that when you use them in the future you will be familiar with them. I encourage you to read through the documentation on them very carefully and try to become as intimately familiar with them as you can.