More great iOS 6 SDK APIs coverage, depth and insight

More iOS 6 Development

Further Explorations of the iOS SDK

David Mark | Alex Horovitz | Kevin Kim | Jeff LaMarche

Apress®

More iOS6 Development

Further Explorations of the iOS SDK

Alex Horovitz
Kevin Kim
Jeff LaMarche
David Mark

Apress®

More iOS6 Development

ISBN 978-1-4302-3807-2

ISBN 978-1-4302-3808-9 (eBook)

President and Publisher: Paul Manning
Lead Editor: Steve Anglin
Technical Reviewer: Nick Waynik
Editorial Board: Steve Anglin, Mark Beckner, Ewan Buckingham, Gary Cornell, Louise Corrigan, Morgan Ertel, Jonathan Gennick, Jonathan Hassell, Robert Hutchinson, Michelle Lowman, James Markham, Matthew Moodie, Jeff Olson, Jeffrey Pepper, Douglas Pundick, Ben Renow-Clarke, Dominic Shakeshaft, Gwenan Spearing, Matt Wade, Tom Welsh
Coordinating Editor: Brigid Duffy
Copy Editor: Mary Behr
Compositor: SPi Global
Indexer: SPi Global
Artist: SPi Global
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a Delaware corporation.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at www.apress.com/bulk-sales.

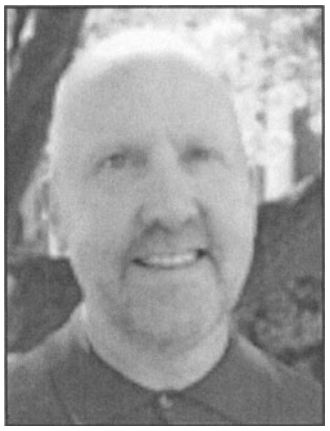
Any source code or other supplementary materials referenced by the author in this text is available to readers at www.apress.com. For detailed information about how to locate your book's source code, go to www.apress.com/source-code.

To Annie, for all her love and support.

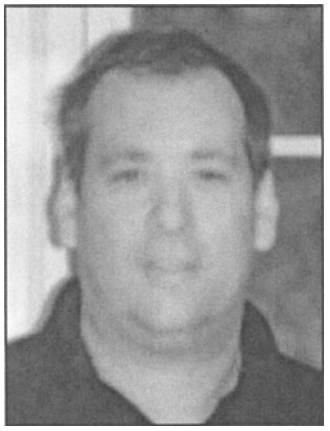
Contents at a Glance

Contents

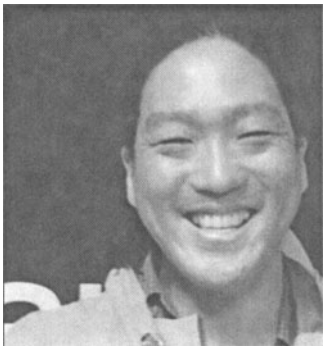
About the Authors



Dave Mark is a longtime Mac developer and author who has written a number of books on Mac and iOS development, including *Beginning iPhone 4 Development* (Apress, 2011), *More iPhone 3 Development* (Apress, 2010), *Learn C on the Mac* (Apress, 2008), *Ultimate Mac Programming* (Wiley, 1995), and The *Macintosh Programming Primer* series (Addison-Wesley, 1992). Dave was one of the founders of MartianCraft, an iOS and Android development house. Dave loves the water and spends as much time as possible on it, in it, or near it. He lives with his wife and three children in Virginia.

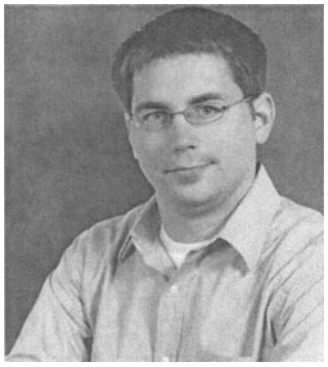


Jeff LaMarche is a Mac and iOS developer with more than 20 years of programming experience. Jeff has written a number of iOS and Mac development books, including *Beginning iPhone 4 Development* (Apress, 2011), *More iPhone 3 Development* (Apress, 2010), and *Learn Cocoa on the Mac* (Apress, 2010). Jeff is a principal at MartianCraft, an iOS and Android development house. He has written about Cocoa and Objective-C for MacTech Magazine, as well as articles for Apple's developer web site. Jeff also writes about iOS development for his widely read blog at `www.iphonedevelopment.blogspot.com`.



Kevin Kim is a co-founder and developer at AppOrchard LLC, a Tipping Point Partners company focused on sustainable iOS development. A graduate of Carnegie Mellon University, he was first exposed to the NeXTStep computer (the ancestor of today's iPhone) as a programmer at the Pittsburgh Supercomputing Center and has been hooked ever since. His career has spanned finance, government, biotech, and technology, including Apple where he managed the Apple Enterprise Services team for the New York metro area. Kevin was also a co-author of *Pro iOS 5 Tools* (Apress, 2011). He currently resides in the Alphabet City section of New York City with his wife and a clowder of rescued cats.

About the Technical Reviewer



Nick Waynik has been working in the IT field for over 13 years and has done everything from network administration to web development. He started writing iOS apps when the SDK was first released. Since then he has gone on to start his own business focusing on iOS development. He loves spending his free time with his wife, Allison, and son, Preston; sometimes he even plays golf. He blogs at `nickwaynik.com` and can be found on Twitter as @n_dubbs.

Acknowledgments

Writing a book like this one is more than the effort of us, the authors. Even though our names are on the cover, it is the result of the hard work of many people.

First, I would like to thank Dave Mark and Jeff LaMarche for writing the first version of the book, and for giving me a solid foundation from which to build and expand upon.

I would like to thank the staff at Apress for making sure this book was completed as close to schedule as possible. Brigid Duffy provided the guidance and oversight to make sure I completed this book. Tom Welsh made sure I stayed on topic and kept things clear. Mary Behr made the manuscript look beautiful. I'd also like to thank Brandon Levesque for making sure people knew this book was coming.

Thanks to the technical reviewers Nick Waynick and Mark Dalrymple for making sure the code I wrote actually works. Any mistakes that still exist are mine.

Thanks to my friends and colleagues at AppOrchard for their patience through the last several months of curmudgeonly behavior and for helping me make this project successful.

A great deal of thanks to my wife, Annie, for making sure I worked on this book when I would have rather been watching baseball or playing guitar. Thanks to my cats, PK, Manny, and Leela, for wanting to be fed when I needed a break. An extra thanks goes to Manny for being the subject of many of the examples in this book.

Finally, thanks to you, the reader, for buying this book. We like to think of programming as a scientific discipline, but, at times it feels more like a black art. If this book helps you on your journey of understanding iOS programming, then it is all worthwhile.

Chapter 1

Here We Go Round Again

So, you're still creating iPhone applications, huh? Great! iOS and the App Store have enjoyed tremendous success, fundamentally changing the way mobile applications are delivered and completely changing what people expect from their mobile devices. Since the first release of the iOS Software Development Kit (SDK) way back in March 2008, Apple has been busily adding new functionality and improving what was already there. It's no less exciting a platform than it was back when it was first introduced. In fact, in many ways, it's more exciting, because Apple keeps expanding the amount of functionality available to third-party developers like us.

Since the last release of this book, *More iPhone 3 Development* (Apress 2010), Apple has released a number of frameworks, tools, and services. These include, but aren't limited to

- **Core frameworks**: Core Motion, Core Telephony, Core Media, Core View, Core MIDI, Core Image, and Core Bluetooth
- **Utility frameworks**: Event Kit, Quick Look Framework, Assets Library, Image I/O, Printing, AirPlay, Accounts and Social Frameworks, Pass Kit
- **Services and their frameworks**: iAds, Game Center, iCloud, Newsstand
- **Developer-centric enhancements**: Blocks, Grand Central Dispatch (GCD), Weak Linking Support, Automatic Reference Counting (ARC), Storyboards, Collection Views, UI State Preservation, Auto Layout, UIAutomation

and many more...

Obviously, there are too many changes to cover completely in a single book. But we'll try our best to make you comfortable with the ones that you'll most likely need to know.

What This Book Is

This book is a guide to help you continue down the path to creating better iOS applications. In *Beginning iOS 6 Development* (Apress, 2012), the goal was to get you past the initial learning curve and to help you get your arms around the fundamentals of building your first iOS applications. In

this book, we're assuming you already know the basics. So, in addition to showing you how to use several of the new iOS APIs, we're also going to weave in some more advanced techniques that you'll need as your iOS development efforts grow in size and complexity.

In *Beginning iOS 6 Development*, every chapter was self-contained, each presenting its own unique project or set of projects. We'll be using a similar approach in the second half of this book, but in Chapters 2 through 8, we'll focus on a single, evolving Core Data application. Each chapter will cover a specific area of Core Data functionality as we expand the application. We'll also be strongly emphasizing techniques that will keep your application from becoming unwieldy and hard to manage as it gets larger.

What You Need To Know

This book assumes that you already have some programming knowledge and that you have a basic understanding of the iOS SDK, either because you've worked through *Beginning iOS 6 Development* or because you've gained a similar foundation from other sources. We assume that you've experimented a little with the SDK, perhaps written a small program or two on your own, and have a general feel for Xcode. You might want to quickly review Chapter 2 of *Beginning iOS Development*.

COMPLETELY NEW TO IOS?

If you are completely new to iOS development, there are other books you probably should read before this one. If you don't already understand the basics of programming and the syntax of the C language, you should check out *Learn C on the Mac for OS X and iOS* by David Mark and James Bucanek (Apress, 2012), which is a comprehensive introduction to the C language for Macintosh programmers (`www.apress.com/9781430245339`).

If you already understand C but don't have any experience programming with objects, check out *Learn Objective-C on the Mac* (Apress, 2012), an excellent and approachable introduction to Objective-C by Mac programming experts Scott Knaster, Wagar Malik, and Mark Dalrymple (`www.apress.com/9781430218159`).

Next, navigate over to the Apple iPhone Development Center and download a copy of *The Objective-C 2.0 Programming Language*, a very detailed and extensive description of the language and a great reference guide at *`http://developer.apple.com/library/ios/#documentation/Cocoa/Conceptual/ObjectiveC/Introduction/introObjectiveC.html`*.

Once you have a firm handle on Objective-C, you need to master the fundamentals of the iOS SDK. For that, you should check out the prequel to this book, *Beginning iOS 6 Development: Exploring the iOS SDK* by David Mark, Jack Nutting, Jeff LaMarche, and Fredrik Olsson (Apress 2011, `www.apress.com/9781430245124`).

What You Need Before You Can Begin

Before you can write software for iOS devices, you need a few things. For starters, you need an Intel-based Macintosh running Lion (Mac OS X 10.7 or later). Any Macintosh computer—laptop or desktop—that has been released since 2008 should work just fine, but make sure your machine is Intel-based and is capable of running Lion.

This may seem obvious, but you'll also need an iPhone (3GS or later), iPod touch (3rd generator or later), or an iPad (iPad 2 or later). While much of your code can be tested using the iPhone/iPad simulator, not all programs will run in the simulator. And you'll want to thoroughly test any application you create on an actual device before you ever consider releasing it to the public.

Finally, you'll need to sign up to become a Registered iOS Developer. If you're already a Registered iOS Developer, go ahead and download the latest and greatest iPhone development tools, and skip ahead to the next section.

If you're new to Apple's Registered iOS Developer programs, navigate to `http://developer.apple.com/ios/`, which will bring you to a page similar to that shown in Figure 1-1. Just below the iOS Dev Center banner, on the right side of the page, you'll find links labeled Log in and Register. Click the Register link. On the page that appears, click the Continue button. Follow the sequence of instructions to use your existing Apple ID or create a new one.

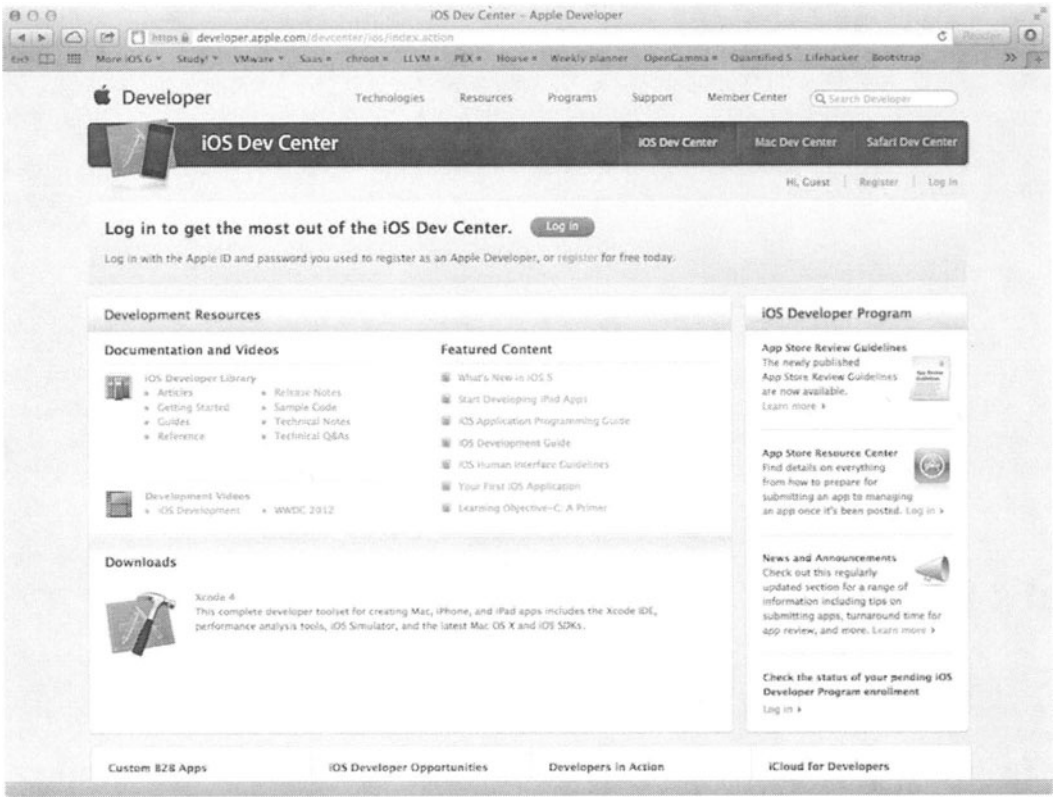


Figure 1-1. Apple's iOS Dev Center web site

At some point, as you register, you'll be given a choice of several paths, all of which will lead you to the SDK download page. The three choices are free, commercial, and enterprise. All three options give you access to the iOS SDK and Xcode, Apple's integrated development environment (IDE). Xcode includes tools for creating and debugging source code, compiling applications, and performance-tuning the applications you've written. Please note that although you get at Xcode through the developer site, your Xcode distribution will be made available to you via the App Store.

The free option is, as its name implies, free. It lets you develop iOS apps that run on a software-only simulator but does not allow you to download those apps to your iPhone, iPod touch, or iPad, nor sell your apps on Apple's App Store. In addition, some programs in this book will run only on your device, not in the simulator, which means you will not be able to run them if you choose the free solution. That said, the free solution is a fine place to start if you don't mind learning without doing for those programs that won't run in the simulator.

The other two options are to sign up for an iOS Developer Program, either the Standard (commercial) Program or the Enterprise Program. The Standard Program costs $99. It provides a host of development tools and resources, technical support, distribution of your application via Apple's App Store, and, most important, the ability to test and debug your code on an iPhone rather than just in the simulator. The Enterprise Program, which costs $299, is designed for companies developing proprietary, in-house applications for the iPhone, iPod touch, and iPad. For more details on these two programs, check out `http://developer.apple.com/programs/`.

> **Note** If you are going to sign up for the Standard or Enterprise Program, you should go do it right now. It can take a while to get approved, and you'll need that approval to be able to run applications on your iPhone. Don't worry, though—the projects in the early chapters of this book will run just fine on the iPhone simulator.

Because iOS devices are connected mobile devices that utilize a third party's wireless infrastructure, Apple has placed far more restrictions on iOS developers than it ever has on Macintosh developers, who are able to write and distribute programs with absolutely no oversight or approval from Apple. Apple is not doing this to be mean, but rather to minimize the chances of people distributing malicious or poorly written programs that could degrade performance on the shared network. It may seem like a lot of hoops to jump through, but Apple has gone through quite an effort to make the process as painless as possible.

What's In this Book

As we said earlier, Chapters 2 through 7 of this book focus on Core Data, Apple's primary persistence framework. The rest of the chapters cover specific areas of functionality that are either new with iOS SDK or were simply too advanced to include in *Beginning iOS 6 Development*.

Here is a very brief overview of the chapters that follow:

Chapter 2, The Anatomy of Core Data: In this chapter, we'll introduce you to Core Data. You'll learn why Core Data is a vital part of your iPhone development arsenal. We'll dissect a simple Core Data application and show you how all the individual parts of a Core Data-backed application fit together.

Chapter 3, A Super Start: *Adding, Displaying, and Deleting Data*: Once you have a firm grasp on Core Data's terminology and architecture, you'll learn how to do some basic tasks, including inserting, searching for, and retrieving data.

Chapter 4, The Devil in the Detail View: In this chapter, you'll learn how to let your users edit and change the data stored by Core Data. We'll explore techniques for building generic, reusable views so you can leverage the same code to present different types of data.

Chapter 5, Preparing for Change: *Migrations and Versioning*: Here, we'll look at Apple tools that you can use to change your application's data model, while still allowing your users to continue using their data from previous versions of your application.

Chapter 6, Custom Managed Objects: To really unlock the power of Core Data, you can subclass the class used to represent specific instances of data. In this chapter, you'll learn how to use custom managed objects and see some benefits of doing so.

Chapter 7, Relationships, Fetched Properties, and Expressions: In this final chapter on Core Data, you'll learn about some mechanisms that allow you to expand your applications in powerful ways. You'll refactor the application you built in the previous chapters so that you don't need to add new classes as you expand your data model.

Chapter 8, iCloud Storage: The iCloud Storage APIs are among the coolest features of iOS. The iCloud APIs will let your apps store documents and key-value data in iCloud. iCloud will wirelessly push documents to a user's device automatically and update the documents when changed on any device—automatically. You'll enhance your Core Data application to store information on iCloud.

Chapter 9, Peer-to-Peer Over Bluetooth Using GameKit: The GameKit framework makes it easy to create programs that communicate over Bluetooth, such as multiplayer games for the iPhone and iPod touch. You'll explore GameKit by building a simple two-player game.

Chapter 10, CoreLocation and MapKit: This chapter explores another great new piece of functionality added to the iOS SDK: an enhanced CoreLocation. This framework now includes support for both forward and reverse geocoding location data. You will be able to convert back and forth between a set of map coordinates and information about the street, city, country (and so on) at that coordinate. Plus, you'll explore how all this interoperates with enhanced MapKit.

Chapter 11, Messaging: Mail, Social, and iMessage: Your ability to get your message out has gone beyond e-mail. In this chapter, we'll take you through the core options of Mail, the Social Framework, and iMessage and you'll see how to leverage each appropriately.

Chapter 12, Media Library Access and Playback: It's now possible to programmatically get access to your users' complete library of audio tracks stored on their iPhone or iPod touch. In this chapter, you'll look at the various techniques used to find, retrieve, and play music and other audio tracks.

Chapter 13, Locking it Down: iOS Security: In this chapter, you'll be taking a look at the Security framework (Security.framework), which provides a standard set of security-related services for iOS applications. In addition to the basic interfaces of this framework, you will utilize some additions for managing credentials that are not specified by standards but that are required by many applications.

Chapter 14, Keeping Your Interface Responsive: Long-running programming tasks can easily bog down the iOS user interface. In this chapter, you'll take a look at implementing advanced Storyboarding techniques so that your application remains responsive.

Chapter 15, Unit Testing, Debugging, and Instruments: No program is ever perfect. Bugs and defects are a natural part of the programming process. In this chapter, you'll learn various techniques for preventing, finding, and fixing bugs in iOS SDK programs.

Chapter 16, The Road Goes Ever On...: Sadly, every journey must come to an end. We'll wrap up this book with fond farewells and some resources we hope you'll find useful.

As we said in *Beginning iOS 6 Development*, iOS is an incredible computing platform, an ever-expanding frontier for your development pleasure. In this book, we're going to take you further down the iPhone development road, digging deeper into the SDK, touching on new and, in some cases, more advanced topics.

Read the book and be sure to build the projects yourself—don't just copy them from the archive and run them once or twice. You'll learn most by doing. Make sure you understand what you did, and why, before moving on to the next project. Don't be afraid to make changes to the code. Experiment, tweak the code, observe the results. Rinse and repeat.

Got your iOS SDK installed? Turn the page, put on some iTunes, and let's go. Your continuing journey awaits.

Chapter 2

Core Data: What, Why, and How

Core Data is a framework and set of tools that allow you to save (or persist) your application's data to an iOS device's file system automatically. Core Data is an implementation of something called object-relational mapping (ORM). This is just a fancy way of saying that Core Data allows you to interact with your Objective-C objects without having to worry about how the data from those objects is stored and retrieved from persistent data stores such as relational database (such as SQLite) or into a flat file.

Core Data can seem like magic when you first start using it. Core Data objects are, for the most part, handled just like plain old objects, and they seem to know how to retrieve and save themselves automagically. You won't create SQL strings or make file management calls, ever. Core Data insulates you from some complex and difficult programming tasks, which is great for you. By using Core Data, you can develop applications with complex data models much, much faster than you could using straight SQLite, object archiving, or flat files.

Technologies that hide complexity the way Core Data does can encourage "voodoo programming," that most dangerous of programming practices where you include code in your application that you don't necessarily understand. Sometimes that mystery code arrives in the form of a project template. Or, perhaps you download a utilities library that does a task for you that you just don't have the time or expertise to do for yourself. That voodoo code does what you need it to do, and you don't have the time or inclination to step through it and figure it out, so it just sits there, working its magic... until it breaks. As a general rule, if you find yourself with code in your own application that you don't fully understand, it's a sign you should go do a little research, or at least find a more experienced peer to help you get a handle on your mystery code.

The point is that Core Data is one of those complex technologies that can easily turn into a source of mystery code that will make its way into many of your projects. Although you don't need to know exactly how Core Data accomplishes everything it does, you should invest some time and effort into understanding the overall Core Data architecture.

This chapter starts with a brief history of Core Data and then it dives into a Core Data application. By building a Core Data application with Xcode, you'll find it much easier to understand the more complex Core Data projects you'll find in the following chapters.

A Brief History of Core Data

Core Data has been around for quite some time, but it became available on iOS with the release of iPhone SDK 3.0. Core Data was originally introduced with Mac OS X 10.4 (Tiger), but some of the DNA in Core Data actually goes back about 15 years to a NeXT framework called Enterprise Objects Framework (EOF), which was part of the toolset that shipped with NeXT's WebObjects web application server.

EOF was designed to work with remote data sources, and it was a pretty revolutionary tool when it first came out. Although there are now many good ORM tools for almost every language, when WebObjects was in its infancy, most web applications were written to use handcrafted SQL or file system calls to persist their data. Back then, writing web applications was incredibly time- and labor-intensive. WebObjects, in part because of EOF, cut the development time needed to create complex web applications by an order of magnitude.

In addition to being part of WebObjects, EOF was also used by NeXTSTEP, which was the predecessor to Cocoa. When Apple bought NeXT, the Apple developers used many of the concepts from EOF to develop Core Data. Core Data does for desktop applications what EOF had previously done for web applications: it dramatically increases developer productivity by removing the need to write file system code or interact with an embedded database.

Let's start building your Core Data application.

Creating a Core Data Application

Fire up Xcode and create a new Xcode project. There are many ways to do this. When you start Xcode, you may get the Xcode startup window (Figure 2-1). You can just click the area titled "Create a New Xcode Project." Or you can select **File ➤ New ➤ Project**. Or you can use the keyboard shortcut ⇧⌘N. Whatever floats your boat. Going forward, we're going to mention the options available in the Xcode window or the menu options, but we won't use the keyboard shortcut. If you know and prefer the keyboard shortcuts, feel free to use them. Let's get back to building your app.

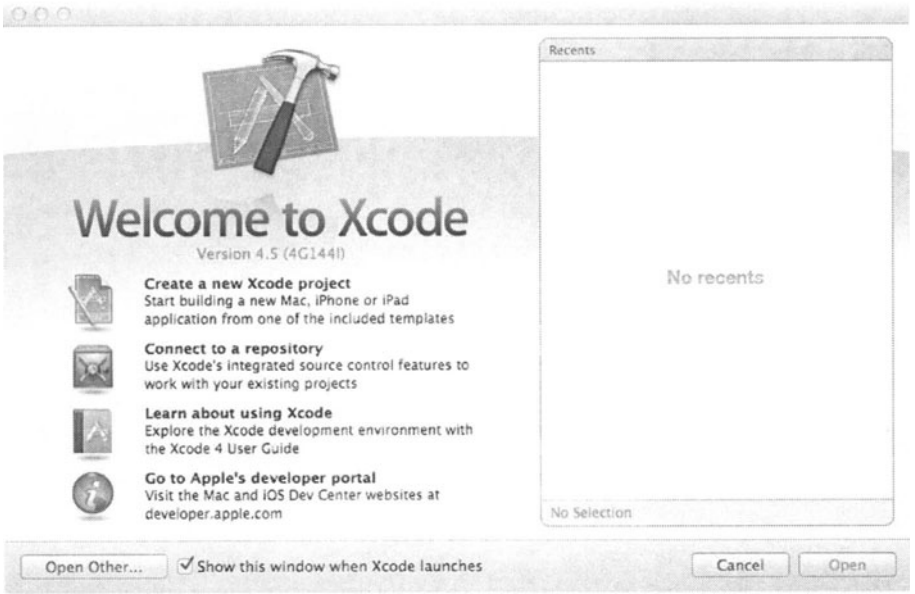


Figure 2-1. Xcode startup window

Xcode will open a project workspace and display the Project Template sheet (Figure 2-2). On the left are the possible template headings: iOS and OS X. Each heading has a bunch of template groups. Select the Application template group under the iOS heading, and then select Master-Detail Application template on the right. On the bottom right, there's a short description of the template. Click the Next button to move the next sheet.

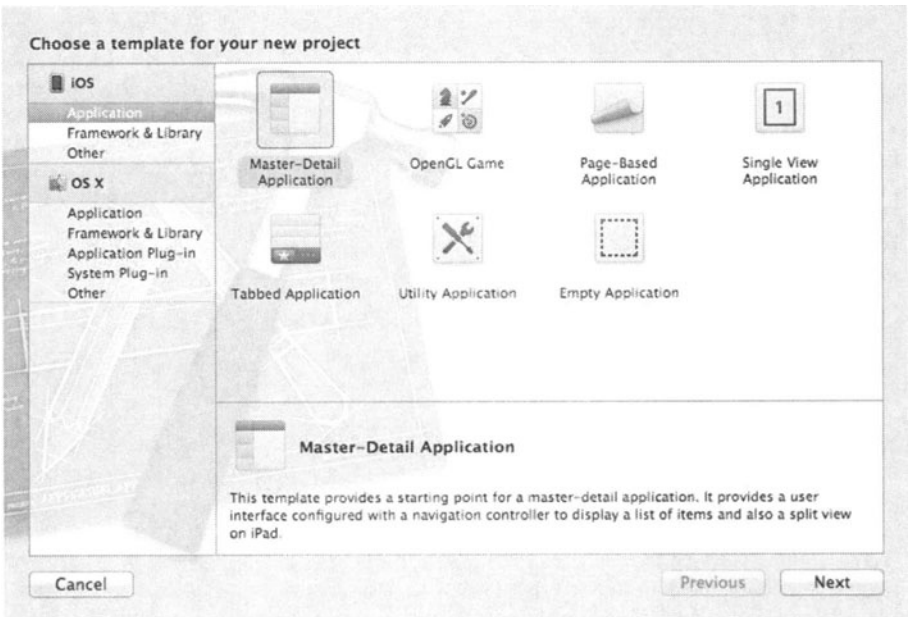


Figure 2-2. Project Template sheet