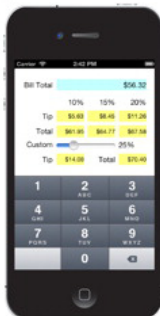
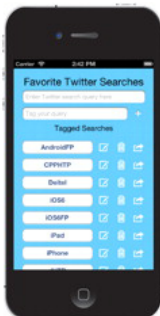


DEITEL® DEVELOPER SERIES

Dive Into® iOS® 6

3 FULLY
DEVELOPED
APPS

An App-Driven Approach



A Brief Introduction for
NEW iOS DEVELOPERS

PAUL DEITEL • HARVEY DEITEL • ABBEY DEITEL

FREE SAMPLE CHAPTER



SHARE WITH OTHERS

DIVE INTO[®] IOS 6
AN APP-DRIVEN APPROACH
DEITEL[®] DEVELOPER SERIES

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

The publisher offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales, which may include electronic versions and/or custom covers and content particular to your business, training goals, marketing focus, and branding interests. For more information, please contact:

U. S. Corporate and Government Sales
(800) 382-3419
corpsales@pearsontechgroup.com

For sales outside the U. S., please contact:

International Sales
international@pearsoned.com

Visit us on the Web: informit.com/ph

© 2012 Pearson Education, Inc.

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, One Lake Street, Upper Saddle River, New Jersey 07458, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-13336569-6

ISBN-10: 0-13-336569-7

DIVE INTO[®] IOS 6

AN APP-DRIVEN APPROACH

DEITEL[®] DEVELOPER SERIES



Trademarks

DEITEL, the double-thumbs-up bug and DIVE INTO are registered trademarks of Deitel and Associates, Inc.

Apple, iOS, iPhone, iPad, iPod touch, Xcode, Objective-C and Cocoa are registered trademarks of Apple, Inc.

Throughout this book, trademarks are used. Rather than put a trademark symbol in every occurrence of a trademarked name, we state that we are using the names in an editorial fashion only and to the benefit of the trademark owner, with no intention of infringement of the trademark.

In memory of Steve Jobs

Paul, Harvey and Abbey Deitel

Contents

Preface **x**

Before You Begin **xvii**

1 Introduction to iOS App Development **I**

Introduction to iOS and iOS 6, Cocoa Touch® Frameworks, Xcode® Integrated Development Environment, the iOS Simulator, the Objective-C Programming Language and Object Technology; Downloading Apps from the App Store; Test-Driving the SpotOn Game App in the iPhone and iPad Simulators

1.1	Introduction	2
1.2	iPhone and iPad Sales	3
1.3	iPhone and iPad in Business	4
1.4	iOS Device Features	5
1.5	iOS	11
1.6	iOS 6	14
1.7	Downloading Apps from the App Store	17
1.8	Objective-C Programming Language	19
1.9	Cocoa Touch and iOS Frameworks	19
1.10	Xcode Toolset	24
1.11	Object Technology: A Quick Refresher	26
1.12	Test-Driving the SpotOn Game App in the iPhone and iPad Simulators	29
1.13	iOS Developer Documentation	31
1.14	Wrap-Up	32

2 App Store and App Business Issues **33**

Setting up an iOS Developer Program Profile; the iOS Human Interface Guidelines; Characteristics of Great Apps; App Store Submission and Ad Hoc distribution; Pricing, Monetization, In-App Purchase of Virtual Goods and the iAd Network; Marketing and Advertising

2.1	Introduction	34
2.2	iOS Developer Program: Setting Up Your Profile for Testing and Submitting Apps	34
2.2.1	Setting Up Your Development Team	35
2.2.2	Provisioning a Device for App Testing	37

2.2.3	Creating an Ad Hoc Provisioning Profile for Beta Testing	39
2.2.4	Creating Explicit App IDs	41
2.3	iOS Human Interface Guidelines	41
2.4	What Makes a Great App?	42
2.5	Preparing Your App for Submission through iTunes Connect	44
2.6	Pricing Your App: Free or Fee	47
2.7	Managing Your Apps with iTunes Connect	50
2.8	Information You'll Need for iTunes Connect	50
2.9	Creating the App Record in iTunes Connect	52
2.10	Submitting Your App for Approval and Shipping Your Approved App	53
2.11	Monetizing Your Apps	54
2.11.1	Using In-App Purchase to Sell Virtual Goods	55
2.11.2	iAd Network	56
2.11.3	Developing Custom Apps for Organizations	56
2.11.4	Ultimate Payoff: Equitizing Your App-Development Business	57
2.12	Marketing Your App	58
2.13	Other App Platforms	63
2.14	Tools for Multiple-Platform App Development	64
2.15	Wrap-Up	64
3	Welcome App	65
	<i>Dive-Into[®] Xcode: Introducing Visual GUI Design with Cocoa Touch, Interface Builder, Storyboarding and Auto Layout, Universal Apps, Accessibility, Internationalization</i>	
3.1	Introduction	66
3.2	Technologies Overview	66
3.3	Creating a Universal App Project with Xcode	67
3.4	Storyboarding the Welcome App's GUI	74
3.4.1	Configuring the App for Portrait and Landscape Orientations	74
3.4.2	Designing the iPhone GUI	75
3.4.3	Designing the iPad GUI	82
3.5	Running the Welcome App	83
3.6	Making Your App Accessible	85
3.7	Internationalizing Your App	87
3.8	Wrap-Up	90
4	Tip Calculator App	92
	<i>Introducing Objective-C, Text Fields, Sliders, Outlets, Actions, Event Handling, NSDecimalNumber, NSNumberFormatter, Automatic Reference Counting (ARC), Blocks and Grand Central Dispatch (GCD)</i>	
4.1	Introduction	93
4.2	Test-Driving the Tip Calculator App	94
4.3	Technologies Overview	94

4.4	Building the App's GUI	99
4.4.1	Creating the Project	99
4.4.2	Adding the GUI Components	101
4.5	Creating Outlets with Interface Builder	107
4.6	Creating Actions with Interface Builder	111
4.7	Class <code>TCViewController</code>	112
4.7.1	Class <code>TCViewController</code> 's Header	112
4.7.2	Class <code>TCViewController</code> 's Implementation File	113
4.8	Wrap-Up	122

5 Favorite Twitter® Searches App 124

Social Framework Sharing, iCloud Key–Value Storage, Collections, Buttons, Scroll Views, Web Views, Alert Dialogs, Storyboard Segues, Programmatic Auto Layout, and Programmatic Accessibility Strings and Localized Strings

5.1	Introduction	125
5.2	Test-Driving the Favorite Twitter Searches App	126
5.3	Technologies Overview	132
5.4	Building the App's GUI	137
5.4.1	Creating the Project	137
5.4.2	Main View	139
5.4.3	Flipside View	145
5.5	Class <code>FTSMainViewController</code>	146
5.5.1	<code>FTSMainViewController</code> Header	146
5.5.2	<code>FTSMainViewController</code> Implementation	147
5.6	Class <code>FTSFlipsideViewController</code>	169
5.6.1	<code>FTSFlipsideViewController</code> Header	169
5.6.2	<code>FTSFlipsideViewController</code> Implementation	170
5.7	Testing the App on iOS Devices	173
5.8	Accessibility and Internationalization	175
5.8.1	Accessibility Strings for Dynamically Generated GUI Components	175
5.8.2	Localized Strings for Dynamically Generated GUI Components	176
5.9	Wrap-Up	178

Index 180

Preface

Welcome to the world of iOS 6® app development with the iOS Software Development Kit (SDK) 6, the Cocoa Touch® frameworks, Xcode® 4.5 development tools and the latest versions and idioms of the Objective-C® programming language.

iOS device sales are growing exponentially, creating enormous opportunities for iPhone®, iPod touch® and iPad® app developers. This short five-chapter e-book quickly gets you started developing great apps and going through the detailed process of getting them published on Apple's App Store. The three programming chapters include a *visually designed* app with no programming and two *Objective-C based* apps presented using our signature **app-driven approach**—each technology is discussed in the context of a fully developed iOS app, complete with a test drive, technologies overview, syntax coloring, code walkthroughs and sample live outputs. The e-book's source code is available at www.deitel.com/books/DiveIntoiOS6/.

Dive Into® iOS 6: An App-Driven Approach was fun to write! The book's apps were carefully designed to introduce you to a few key iOS features and frameworks to get you started building apps quickly. You'll begin with a test-drive of our **SpotOn** game app (from our forthcoming print book *iOS 6 for Programmers: An App-Driven Approach*) in Chapter 1, then build your first app in Chapter 3. Chapter 2, App Store and App Business Issues, walks you through what makes a great app, the app submission process, marketing your apps and more.

Copyright Notice and Code License

All of the code and iOS apps in the e-book are copyrighted by Deitel & Associates, Inc. The sample programs in the book are licensed under a Creative Commons Attribution 3.0 Unported License (creativecommons.org/licenses/by/3.0/), with the exception that they may not be reused in any way in educational tutorials and textbooks, whether in print or digital format. You're welcome to use the apps in the book as shells for your own apps, building on their existing functionality. If you have any questions, contact us at deitel@deitel.com.

Intended Audience

We assume that you're comfortable with Mac OS X, as you'll need to work on a Mac to develop iOS apps. We also assume that you're a programmer with significant experience working in a C-based object-oriented language such as Java, C#, C++ or Objective-C. If you don't know Objective-C, you should still be able to read the source code and the associated discussions and pick up what you'll need to master the book's apps.

Key Features

Here are some of this Dive-Into® e-book's key features:

App-Driven Approach. You'll learn some key iOS 6 programming technologies in the context of three complete, working iOS 6 apps. Each of the three app chapters presents one app—we discuss what the app does, show screen shots, test-drive it and overview the technologies and the architecture you'll use to build it. Then we build the app, present the complete code and do a detailed code walkthrough. Along the way, we discuss the programming concepts and demonstrate the functionality of the iOS APIs (application programming interfaces) that we use. Figure 1 lists the three apps in the e-book and the key technologies we introduce in each.

Dive Into® iOS 6: An App-Driven Approach Apps

Chapter 3, Welcome App

Dive-Into® Xcode: Introducing Visual GUI Design with Cocoa Touch, Interface Builder, Storyboarding and Auto Layout, Universal Apps, Accessibility, Internationalization

Chapter 4, Tip Calculator App

Introducing Objective-C, Text Fields, Sliders, Outlets, Actions, Event Handling, NSDecimalNumber, NSNumberFormatter, Automatic Reference Counting (ARC), Blocks and Grand Central Dispatch (GCD)

Chapter 5, Favorite Twitter® Searches App

Social Framework Sharing, iCloud Key-Value Storage, Collections, Buttons, Scroll Views, Web Views, Alert Dialogs, Storyboard Segues, Programmatic Auto Layout, and Programmatic Accessibility Strings and Localized Strings

Fig. 1 | Dive Into® iOS 6: An App-Driven Approach apps and the technologies they introduce.

Objective-C. This book is *not* an Objective-C tutorial, but it teaches a good portion of this object-oriented programming language in the context of iOS 6 app development.

Cocoa Touch Frameworks. Cocoa Touch is the set of frameworks and the runtime environment for iOS. Throughout the e-book, we use many of the Cocoa Touch features and frameworks.

iOS SDK 6. We cover several of the great new features in iOS SDK 6.

Xcode 4.5. Apple's Xcode 4.5 integrated development environment (IDE) and its associated tools for Mac OS X, combined with the iOS 6 SDK, provide everything you need to develop and test iOS 6 apps.

Uploading Apps to the App Store. In Chapter 2, App Store and App Business Issues, we walk you through the process of obtaining development certificates, creating provisioning profiles, submitting your apps to the App Store for approval, criteria for approval, deciding whether your app should be free or fee based, where to go to set up your iTunes Connect account to receive payments and track your app sales, marketing your app and more.

Features

Syntax Coloring. For readability, we syntax color the code, similar to Xcode's use of syntax coloring. Our syntax-coloring conventions are as follows:

comments appear in green
 keywords appear in dark blue
 constants and literal values appear in light blue
 all other code appears in black

Code Highlighting. We emphasize the key code segments in each program by enclosing them in yellow rectangles.

Using Fonts for Emphasis. We use various font conventions:

- Defining occurrences of key terms appear in **bold maroon** for easy reference.
- On-screen IDE components appear in **bold Helvetica** (e.g., the **File** menu).
- Program source code appears in Lucida (e.g., `int x = 5;`).

In this book you'll create GUIs using a combination of visual programming (drag and drop) and writing code. We use different fonts when we refer to GUI components as they appear in the Xcode IDE versus GUI components as they appear in program source code:

- When dragging and dropping GUI components onto a new GUI, we refer to the component using its name as it appears in the Xcode library (e.g., **Text Field** or **Scroll View**).
- When we refer to a GUI component that we create in a program, we place its variable name and class name in a Lucida font—e.g., “myButton” or “UIButton.”

Using the > Character. We use the > character to indicate selecting a menu item from a menu. For example, we use the notation **File > New** to indicate that you should select the **New** menu item from the **File** menu.

Source Code. All of the source-code examples are available for download from:

www.deitel.com/books/DiveIntoiOS6/

Documentation. All of the manuals that you'll need to develop apps for the current iOS release are available at developer.apple.com/iOS/.

Chapter Objectives. Each chapter begins with a list of objectives.

Figures. Abundant tables, source code listings and iOS screen shots are included.

Index. We include an extensive index for reference. The page number of the defining occurrence of each key term in the e-book is highlighted in the index in **bold maroon**.

Our Forthcoming Book—iOS 6 for Programmers: An App-Driven Approach, 2/e

This brief, five-chapter e-book is based on the first several chapters from our forthcoming print book, *iOS 6 for Programmers: An App-Driven Approach, 2/e*, which is expected to be published by Pearson in 2013. The print book will include additional chapters based on fully developed iOS 6 apps (Fig. 2). As we complete each chapter, it will be available to *paid* subscribers of Safari Books Online (www.safaribooksonline.com)—an online library that provides subscription access to thousands of e-books, videos and other e-learning resources. All of the source code for Deitel books and *LiveLessons* video products is available for download at www.deitel.com.

Apps		
Flag Quiz Game	Cannon Game	SpotOn Game
Doodlz	Address Book	Route Tracker
Slideshow	Weather Viewer	Voice Recorder
3D Art	HTML5 Favorite Twitter® Searches	

Fig. 2 | Some of the fully developed apps likely to appear in the forthcoming *iOS 6 for Programmers: An App-Driven Approach, 2/e* (subject to change).

Join the Deitel & Associates, Inc. Social Networking Communities

To receive updates on this and other Deitel publications, new and updated apps, online Resource Centers, instructor-led onsite training courses, partner offers and more, join the Deitel social networking communities—Facebook (www.deitel.com/DeitelFan), Twitter (@deitel), Google+ ([gplus.to/deitel](https://plus.google.com/+deitel)) and LinkedIn (bit.ly/DeitelLinkedIn)—and register for the free *Deitel® Buzz Online* e-mail newsletter at:

www.deitel.com/newsletter/subscribe.html

Contacting the Authors

We'd sincerely appreciate your comments, criticisms, corrections and suggestions for improvement. Please address all questions and other correspondence to:

deitel@deitel.com

We'll respond promptly, and post corrections and clarifications on:

www.deitel.com/books/DiveIntoiOS6/

and on Facebook, Twitter, Google+, LinkedIn and the *Deitel® Buzz Online*.

Visit www.deitel.com to:

- Receive information on our Dive Into® Series instructor-led programming language training courses offered at customer sites worldwide
- Download code examples
- Check out the growing list of programming Resource Centers
- Receive updates for this e-book, subscribe to the free *Deitel® Buzz Online* e-mail newsletter at www.deitel.com/newsletter/subscribe.html

Acknowledgments

Thanks to Barbara Deitel for long hours devoted to this project—she created all of our iOS 6 Resource Centers, and patiently researched hundreds of technical details. We would also like to thank to Eric Kern, who co-authored our book, *iPhone for Programmers: An App-Driven Approach*, and who made significant contributions to early drafts of this e-book and our forthcoming print book, *iOS 6 for Programmers: An App-Driven Approach, 2/e*.

We're fortunate to have worked on this project with the talented and dedicated team of publishing professionals at Prentice Hall/Pearson. We appreciate the extraordinary efforts and mentorship of Mark L. Taub, Editor-in-Chief of Pearson Technology Group. Olivia Basegio recruited and managed the review team. Chuti Prasertsith designed the e-book's cover. John Fuller managed the e-book's production and Stephane Nakib managed the marketing program.

Reviewers

We wish to acknowledge the efforts of our reviewers. Adhering to a tight time schedule, they scrutinized the manuscript and the source code for the apps and provided constructive suggestions for improving the accuracy and completeness of the presentation:

- Marcantonio Magnarapa, Chief Mobile Officer, www.bemyeye.com
- Scott Gustafson, Owner/Developer, Garlic Software LLC
- Firoze Lafeer, Master Developer, Capital One Labs
- Cory Bohon, Indie Developer at CocoaApp.com and Writer at Mac|Life
- Dan Lingman, Partner, www.nogotogames.com
- Nik Saers, iOS Developer, SAERS

Some of the reviewers' pre-publication comments appear at

www.deitel.com/books/DiveIntoiOS6/

We sincerely hope you enjoy reading *Dive Into[®] iOS 6: An App-Driven Approach* as much as we enjoyed writing it!

Paul Deitel

Harvey Deitel

Abbey Deitel

About the Authors

Paul J. Deitel, CEO and Chief Technical Officer of Deitel & Associates, Inc., is a graduate of MIT, where he studied Information Technology. Through Deitel & Associates, Inc., he has delivered hundreds of professional training courses on many of today's most widely used programming languages to industry clients, including Cisco, IBM, Siemens, Sun Microsystems (now Oracle), Dell, Lucent Technologies, Fidelity, NASA at the Kennedy Space Center, the National Severe Storm Laboratory, White Sands Missile Range, Rogue Wave Software, Boeing, SunGard Higher Education, Stratus, BMI, Cambridge Technology Partners, One Wave, Hyperion Software, Adra Systems, Entergy, CableData Systems, Nortel Networks, Puma, iRobot, Invensys and many more. He and his co-author, Dr. Harvey M. Deitel, are the world's best-selling programming-language professional book, textbook and video authors.

Dr. Harvey M. Deitel, Chairman and Chief Strategy Officer of Deitel & Associates, Inc., has over 51 years of experience in the computer field. Dr. Deitel earned B.S. and M.S. degrees from MIT and a Ph.D. from Boston University. He has extensive college teaching experience, including earning tenure and serving as the Chairman of the Computer Sci-

ence Department at Boston College before founding Deitel & Associates, Inc., with his son, Paul J. Deitel. He and Paul are the co-authors of dozens of books and *LiveLessons* video packages and they are developing many more. The Deitels' publications have earned international recognition, with translations in Japanese, German, Russian, Chinese, Spanish, Korean, French, Italian, Portuguese, Greek, Polish, Urdu and Turkish. Dr. Deitel has delivered hundreds of professional programming seminars to major corporations, academic institutions, government organizations and the military.

Abbey Deitel, President and Chief Marketing Officer of Deitel & Associates, Inc., is a graduate of Carnegie Mellon University where she received a B.S. in Industrial Management. Abbey has been managing the business operations of Deitel & Associates, Inc. for 15 years. She has contributed to numerous Deitel & Associates publications and, together with Paul and Harvey, is the co-author of *iPhone for Programmers: An App-Driven Approach*, *Android for Programmers: An App-Driven Approach*, *Android How to Program* and *Internet & World Wide Web How to Program, 5/e*.

Corporate Training from Deitel & Associates, Inc.

Deitel & Associates, Inc., founded by Paul Deitel and Harvey Deitel, is an internationally recognized authoring, corporate training and software development organization specializing in iOS and Android app development, computer programming languages, object technology and Internet and web software technology. The company offers instructor-led training courses delivered at client sites worldwide on major programming languages and platforms, such as Objective-C and iOS app development, Android app development, Java™, C, C++, Visual C++®, C#, Visual Basic®, Python®, object technology, Internet and web programming, and a growing list of additional programming and software development courses. The company's clients include many of the world's largest corporations, government agencies, branches of the military, and academic institutions.

Through its 37-year publishing partnership with Prentice Hall/Pearson, Deitel & Associates, Inc., publishes leading-edge programming professional books, college textbooks, and *LiveLessons* web-based video courses. Deitel & Associates, Inc. and the authors can be reached at:

deitel@deitel.com

To learn more about Deitel's *Dive Into*® Series Corporate Training curriculum, visit:

www.deitel.com/training/

To request a proposal for on-site, instructor-led training at your company or organization, e-mail deitel@deitel.com.

Individuals wishing to purchase Deitel books and *LiveLessons* web-based video training courses can do so through www.deitel.com, www.informit.com, www.amazon.com, www.bn.com, Barnes and Noble book stores and other retail booksellers. Bulk orders by corporations, the government, the military and academic institutions should be placed directly with Pearson. For more information, visit www.pearson.com.

Before You Begin

This section contains information and instructions you should review to ensure that your Mac is set up properly for use with this book. We'll post updates (if any) to this Before You Begin section on the book's website:

www.deitel.com/books/DiveIntoiOS6

Font and Naming Conventions

We use fonts to distinguish between on-screen components (such as menu names and menu items) and Objective-C code or commands. Our convention is to show on-screen components in a sans-serif bold **Helvetica** font (for example, **File** menu) and to show program source code and commands that you'd execute in a Terminal window in a sans-serif Lucida font (for example, `@interface`).

System Requirements

To develop iOS 6 apps you need a Mac running Mac OS X 10.7.4 (Lion) or higher—some features require Mac OS X 10.8 (Mountain Lion) or higher. You'll also need Xcode 4.5, which you can download and install from the Mac App Store. When you open Xcode for the first time, it will download and install additional features required for development. For the latest information about Xcode, visit

developer.apple.com/xcode

Apple iOS Developer Account

You should register for a free Apple iOS developer account at:

developer.apple.com/devcenter/ios/index.action

This will give you access to the latest released version of the SDK, documentation and code examples. There is also a *paid* developer program that lets you download SDK betas, upload your finished apps to the App Store and load your apps directly onto an iOS device for testing. We demonstrate the apps in this book using the iOS simulator that comes with the iOS SDK; however, the app in Chapter 5 uses some features that are available *only* on actual iOS devices. Testing apps on an iOS device *requires* a *paid* Apple iOS developer account.

Obtaining the Code Examples

The examples for *Dive Into iOS 6: An App-Driven Approach* are available for download at

www.deitel.com/books/DiveIntoiOS6/

If you're not already registered at our website, go to www.deitel.com and click the **Register** link below our logo in the upper-left corner of the page. Enter your information. There's no charge to register, and we *do not* share your information with anyone. We send you only account-management e-mails unless you register separately for our free, double-opt-in *Deitel*[®] *Buzz Online* e-mail newsletter at

```
www.deitel.com/newsletter/subscribe.html
```

After registering for our website, you'll receive a confirmation e-mail with your verification code. *You'll need this code to sign in at www.deitel.com for the first time.* Configure your e-mail client to allow e-mails from www.deitel.com to ensure that the confirmation e-mail is not filtered as junk mail.

Next, visit www.deitel.com and sign in using the **Login** link below our logo in the upper-left corner of the page. Go to www.deitel.com/books/DiveIntoiOS/. Click the **Examples** link to download the ZIP file to your computer. Double click the ZIP file to unzip the archive. We assume that you extract the example code in your Documents folder.

Configuring Xcode to Display Line Numbers

Many programmers find it helpful to display line numbers in the code editor. To do so:

1. Open Xcode and select **Preferences...** from the **Xcode** menu.
2. Select the **Text Editing** tab, then ensure that the **Editing** subtab is selected.
3. Check the **Line Numbers** checkbox.

Configuring Xcode's Code Indentation Options

We use three space indents in our code. To configure your own indentation preferences:

1. Open Xcode and select **Preferences...** from the **Xcode** menu.
2. Select the **Text Editing** tab, then ensure that the **Indentation** subtab is selected.
3. Specify your indentation preferences.

You're now ready to begin developing iOS apps with *Dive Into iOS 6: An App-Driven Approach*. We hope you enjoy the book!

Introduction to iOS App Development



Objectives

In this chapter you'll be introduced to:

- The iOS Developer Programs.
- Features of the iPhone and iPad.
- The iOS operating system.
- Key software for iPhone and iPad app development, including the Xcode® integrated development environment, the iOS simulator, the Objective-C programming language and the Cocoa® Touch frameworks.
- A review of object-technology concepts.
- Test-driving a game-playing app that runs on the iOS simulator.
- Key Apple publications for iOS developers.

1.1 Introduction	1.10 Xcode Toolset
1.2 iPhone and iPad Sales	1.11 Object Technology: A Quick Refresher
1.3 iPhone and iPad in Business	1.12 Test-Driving the SpotOn Game App in the iPhone and iPad Simulators
1.4 iOS Device Features	1.13 iOS Developer Documentation
1.5 iOS	1.14 Wrap-Up
1.6 iOS 6	
1.7 Downloading Apps from the App Store	
1.8 Objective-C Programming Language	
1.9 Cocoa Touch and iOS Frameworks	

1.1 Introduction

Welcome to iPhone and iPad app development! We hope that working with this brief *Dive Into iOS 6: An App-Driven Approach* e-book will be an informative, challenging, entertaining and rewarding experience for you.

This e-book is geared toward experienced programmers who have worked in a C-based object-oriented language such as C++, Java™, C# or Objective-C®. If you don't specifically know object-oriented programming using Apple's Objective-C and the Cocoa® Touch frameworks, you should be able to absorb a good amount of it by running the book's iPhone and iPad apps and studying the feature presentations and detailed code walkthroughs.

App-Driven Approach

We use an **app-driven approach**—new features are discussed in the context of complete working iPhone or iPad apps, with one app per chapter. Some of our apps are built as **universal apps** so they can execute on iPhone, iPad and iPod touch devices. For each app, we start by describing the app, then having you test-drive it. Next, we briefly overview the key **Xcode**® (integrated development environment), Objective-C and Cocoa Touch technologies we use to implement the app. For apps that require it, we walk through designing the GUI *visually*. Then we provide the complete source-code listing using *line numbers*, *syntax coloring* (to mimic the notion of syntax coloring used in the Xcode IDE—these colors are customizable) and *code highlighting* to emphasize the key portions of the code. We also show one or more screen shots of the running app. Then we do a detailed code walkthrough, emphasizing the new programming concepts introduced in the app. You can download the source code for all of the book's apps from www.deitel.com/books/DiveIntoiOS6/.

Downloading the Software and Becoming a Registered Apple Developer

You can download Xcode—which includes the software for building iOS apps—for free from the Mac App Store. You should also become a registered Apple iOS developer at

developer.apple.com/programs/register/

This will give you to access free downloads plus documentation, how-to videos, coding guidelines and more. As a registered iOS developer, you'll be able to build and run your iOS apps on your Mac computer using the iOS simulator. To load apps onto iOS devices for testing and to submit your apps to Apple's App Store, you must join Apple's *fee-based iOS Developer Program* at

developer.apple.com/programs/ios/

This program allows you to access the latest iOS Software Development Kit (iOS SDK) betas and features, and it includes technical support. Organizations may register for the **iOS Developer Enterprise Program**

`developer.apple.com/programs/ios/enterprise/`

which allows developers to deploy proprietary iOS apps to employees within an organization. Colleges and universities interested in offering iOS app-development courses can apply to the **iOS Developer University Program** (`developer.apple.com/iphone/program/university.html`). Qualifying schools receive free access to all the developer tools and resources. Students can share their apps with each other and test their apps on iOS devices.

1.2 iPhone and iPad Sales

iOS device sales are growing exponentially, creating enormous opportunities for iPhone and iPad app developers. A report by mobile analytics firm Flurry revealed that approximately 7 out of every 10 apps developed are for iOS devices.¹ According to a recent comScore study, the iPhone currently has over 30 percent of the smartphone market share.² Here are some iPhone sales statistics by model:

- *First-generation iPhone*: The first-generation iPhone was released in June 2007 and was an instant blockbuster success. Sales have grown significantly with each new version. According to Apple, 6.1 million first-generation iPhones were sold in the initial five quarters of availability.³
- *iPhone 3G*: The second-generation *iPhone 3G* included GPS and was released in July 2008; it sold 6.9 million units in the first quarter alone.
- *iPhone 3GS*: The faster *iPhone 3GS* included a compass; it was launched in June 2009 and sold 5.2 million in its first month of availability.
- *iPhone 4*: The *iPhone 4*, launched in June 2010, sold over 3 million units in its first three weeks.
- *iPhone 4S*: The *iPhone 4S*, released in October 2011, sold over 4 million in its first three days!⁴ Apple sold 35.1 million iPhones during the first three months of 2012, helping the company to nearly *double* its profits from the previous quarter.⁵
- *The new iPhone*: Between sales of the current iPhone model and the new iPhone, analysts expect Apple will sell 110 million iPhones in 2012.⁶

Sales of the *iPad* are equally impressive, and the overall market for tablet devices is expected to grow rapidly. Forrester research predicts that global tablet sales will rise from 56 million in 2011 to 375 million in 2016.⁷ Here are some sales statistics by iPad model:

1. blog.flurry.com/bid/85911/App-Developers-Signal-Apple-Allegiance-Ahead-of-WWDC-and-Google-I-O.
2. www.comscore.com/Press_Events/Press_Releases/2012/5/comScore_Reports_March_2012_U.S._Mobile_Subscriber_Market_Share.
3. www.apple.com/pr/library/2009/07/21results.html.
4. www.apple.com/pr/library/2011/10/17iPhone-4S-First-Weekend-Sales-Top-Four-Million.html.
5. money.cnn.com/2012/04/25/technology/apple-supplier-stocks/index.htm.
6. www.cultofmac.com/113453/the-demand-for-the-iphone-5-is-absolutely-unprecedented/.

4 Chapter 1 Introduction to iOS App Development

- *First-generation iPad:* The iPad, launched in April 2010, sold 3 million units in its first 80 days of availability⁸ and a total of over 40 million worldwide by September 2011.⁹
- *iPad 2:* The thinner, lighter and faster iPad 2 was launched in March 2011 and sold one million units in just the first weekend of availability. By the end of 2011, the iPad accounted for 58 percent of worldwide tablet market share.¹⁰
- *The New iPad:* The New iPad (the third-generation iPad) went on sale in March 2012; 3 million of these devices were sold in just three days.¹¹ Overall iPad sales in the first quarter of 2012 reached 11.8 million units—a 151-percent increase over same quarter the previous year.
- *iPad Mini:* The iPad Mini—expected to be available in October 2012—will compete both in size and price (estimated to be \$299) with smaller tablets such as Amazon’s Kindle Fire, Google’s Nexus 7, Samsung’s Galaxy Tab 2 7.0 and more. The iPad Mini is expected to feature a 7.85-inch, 1024-by-768 pixel display¹² and to run iPad apps without adjustment—developers would not need to support a separate resolution.¹³

1.3 iPhone and iPad in Business

The iPhone quickly gained popularity among consumers, but other devices—particularly the BlackBerry—dominated the business smartphone marketplace. That’s changing rapidly. The iPhone is gaining market share (and the market itself continues to grow), while BlackBerry market share is waning. Meanwhile, Apple dominates in the enterprise tablet market. In the first quarter of 2012, the iPad accounted for 97 percent of enterprise tablet activations, compared for just 2.7 for Android tablets.¹⁴ Figure 1.1 lists some popular iOS business apps.

App	Description
Dropbox	Access and share documents, photos and videos.
Pages	Word processing.
Numbers	Spreadsheets.

Fig. 1.1 | A few iOS business apps. (Part I of 2.)

7. blogs.forrester.com/frank_gillett/12-04-23-why_tablets_will_become_our_primary_computing_device?cm_mmc=RSS-_-IT-_-71-_-blog_154.
8. www.ipadinsider.com/tag/ipad-sales-figures/.
9. www.statista.com/statistics/180656/sales-of-tablets-and-ipads-in-the-us-until-2012/.
10. finance.yahoo.com/news/why-google-android-tablet-market-185500797.html.
11. www.apple.com/pr/library/2012/03/19New-iPad-Tops-Three-Million.html.
12. www.eweek.com/c/a/Mobile-and-Wireless/iPad-Mini-Ready-to-Battle-Other-7Inch-Tablets-From-Google-Amazon-470562/?k=EWKNLEDP07112012A.
13. www.gottabemobile.com/2012/07/05/ipad-mini-release-date-in-october-in-according-to-wsj-bloomberg/.
14. www.computerworld.com/s/article/9226624/97_of_enterprise_tablet_users_got_an_ipad_in_Q1_survey_finds.

App	Description
Keynote	Presentations.
Jump Desktop	Control your desktop remotely from your iOS device.
EasySign	Sign business documents (e.g., contracts) securely.
FTP Client Pro	Read and edit text documents (.pdf, .doc, .xls and more).
QlikView	Track sales, customer trends and other key metrics.
Jeppesen Mobile TC	Tool that enables pilots to view airport diagrams and airport approach, arrival and departure procedures.
OsiriX	Tool that allows doctors to view radiology images on an iOS device.
Quickoffice® Pro	Create, edit and share Microsoft Office files.
Quick Adsense	Track Google Adsense earnings by day, month, year, etc.
WorldCard Mobile	Business-card reader that scans a business card and adds the information into your Contacts .
Smartbidnet	Bid-management tool for the construction industry.

Fig. 1.1 | A few iOS business apps. (Part 2 of 2.)

The iPad in particular has enormous potential in government, the military, industry and education. Here are a few examples:

- Restaurants are using iPads to replace printed menus, making it possible for customers to see pictures of the menu items, read detailed descriptions and place and customize their orders (for an example, visit us.menupad.com).
- Sales professionals are using iPads loaded with product catalogs, customer databases and more, and placing orders in real time from their clients' locations.
- In 2012, the U.S. Air Force Air Mobility Command awarded a \$9.36 million contract to purchase up to 18,000 iPads to be used in cargo aircraft.¹⁵
- In December of 2011, American Airlines became the first airline to be fully approved by the FAA to use iPads during all phases of flight.¹⁶
- Numerous organizations are replacing their paper manuals with electronic manuals on iPads that workers can reference easily.
- Doctors are using iPads to collect patient data, share electronic medical records among care providers, assess risk and mortality of medical procedures, provide patients with information, call in prescriptions to pharmacies and more.

1.4 iOS Device Features

The iPhone is uncomplicated and easy to use (Fig. 1.2). On top it has a headset jack and a *Sleep/Awake* button—used to lock and unlock the iPhone and to power it on and off. On

15. www.appleinsider.com/articles/12/03/02/us_air_force_awards_9m_contract_for_up_to_18000_ipads.html.

16. www.slashgear.com/american-airlines-gets-first-ipad-for-cockpit-approval-by-faa-13202062/.



Fig. 1.2 | iPhone hardware.

the left side of the iPhone are the *Ring/Silent* switch and the *Volume* buttons. On the right side is the SIM card tray. On the bottom are the microphone, dock connector (to plug-in a USB cable to charge or sync the device) and speaker. On the front of the phone at the bottom is the *Home* button—used to activate Siri, exit apps, wake the phone, return to the home screen, display apps that are running in the background and go to the **Spotlight** search. The iPhone 4 and higher include both front- and rear-facing cameras.

The iPad has similar controls, though instead of the *Ring/Silent* switch, there's a screen rotation lock button (Fig. 1.3)—though this can be configured as a *Ring/Silent* switch with a simple change in the device's **Settings** app. The iPad 2 and higher includes a microphone and front- and rear-facing cameras.

Multi-Touch Screen

The iPhone wraps the functionality of a mobile phone, Internet client, iPod music player, gaming console, digital camera and more into a handheld smartphone with a full-color **Multi-Touch**[®] screen. With the touch of your fingers, you can easily navigate between your phone, apps, your iTunes[®] music, web browsing and more. The screen can display a keyboard for typing e-mails and text messages, and for entering data in apps. You can zoom in and out on photos, videos and web pages. You can scroll up and down or side to side by just swiping your finger across the screen.

Gestures

Apple's Multi-Touch screen allows you to control the device with **gestures** involving one touch or multiple simultaneous touches (Fig. 1.4).



Fig. 1.3 | iPad hardware—all but the Home button are on the side or back of the device.

Gesture	Action	Used to
Tap	Tap the screen once	Open an app, click a button.
Double Tap	Tap the screen twice	Select text to cut, copy and paste.
Touch and Hold	Touch the screen and hold your finger in position	Move the cursor in e-mail and SMS messages, move app icons, and so on.
Drag	Touch and drag your finger across the screen	Move a slider left and right or up and down, move around to different areas on a map or web page.
Swipe	Touch the screen, then move your finger in the swipe direction and release	Flip item-by-item through a series, such as photos or music album covers. A swipe automatically stops at the next item.
Flick	Touch and quickly flick your finger across the screen in the direction you'd like to move	Scroll through a Table View (e.g., Contacts) or a Picker View (e.g. dates and times in the Calendar). A flick, unlike a swipe, does not have a specific stop point.
Pinch	Using two fingers, touch and pinch your fingers together, or spread them apart	Zoom out and in on the screen (for example, enlarging text and pictures).

Fig. 1.4 | iPhone and iPad gestures.

Built-In Apps

iOS 6 comes with several built-in apps, including **Phone**, **Contacts**, **Mail**, **Music**, **Safari** and more (Fig. 1.5). To access any app, simply touch its icon. iPhone and iPad devices now include the **Find My iPhone** app. The app helps you find your device if it's lost or stolen. You must first set up iCloud on the device by going to **Settings**. If you misplace your device, log in to Apple's **iCloud** from any computer at www.icloud.com/find. You can view a map with the device's approximate location, have the device play a sound to help you locate it, or display a message to help the person who finds your device return it to you. If you're unable to find your iPhone, the **Remote Wipe** feature restores the device to the factory settings (removing all personal data), thus protecting the privacy of your information.



Fig. 1.5 | iOS 6 built-in apps—varies by device.

Retina Display

The **Retina display** on the iPhone 4 features a 326-pixels-per-inch *high-resolution screen*—more than *four times* the pixel resolution and contrast ratio on previous iPhone models. The third-generation iPad tablet features a 2048-by-1536-pixel-resolution screen. The pixel density is so high that the human eye cannot distinguish the individual pixels. Graphics, images and videos are crisp, clear and bright with smooth edges, even when you zoom in. The Retina display uses **in-plane switching (IPS)** technology, which allows you to view the screen clearly at virtually any angle.

Sensors

The iPhone and iPad include several sensors.

- The **accelerometer** allows the device to respond to up/down, left/right and forward/backward acceleration. For example, you can rotate the device from *portrait* to *landscape* (vertical to horizontal) to change the orientation of pictures, e-mails, web pages and more. You can also use the accelerometer to control games by shaking or tilting the device. You can shake the device to “shuffle” randomly to a different song in your music library, or turn the device sideways to display a **landscape keyboard** for easier typing (Fig. 1.6). A drawing app might use the accelerometer to allow the user to erase the current drawing by shaking the device.

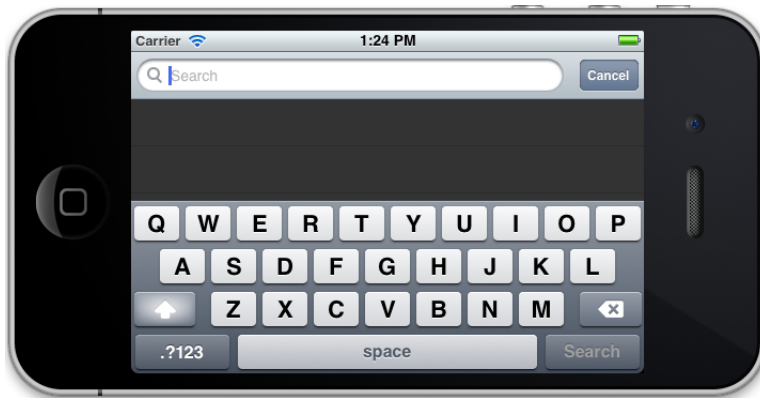


Fig. 1.6 | Landscape keyboard.

- The **gyroscope** (available on the iPhone 4, iPad 2 and higher devices) works with the accelerometer, making the devices more responsive and sensitive to motion by allowing apps to detect the device’s rotation around the *x*-, *y*- and *z*-axes (left/right, up/down and forward/backward, respectively). The gyroscope helps the Camera app stabilize images for better pictures and video, helps improve game controllers and more.¹⁷
- The digital **compass** (included on iPhone 3GS and higher and on the iPad) allows you to orient maps to point in the direction the device is facing.

17. www.zdnet.com/blog/apple/inside-the-iphone-4s-vibrational-gyroscope/7410.

- The **ambient light sensor** determines the amount of light around the device and adjusts the screen's brightness to preserve the battery.
- The iPhone **proximity sensor** determines whether the device is near your face (e.g., when you're on a phone call). The screen turns off when the iPhone is held close to your face and turns back on when the device is moved away from your face. This sensor is not included on the iPad or iPod touch.
- The iPad (second generation and higher) **magnetic sensor** determines whether a smart cover is open or closed, so the screen can be turned on or off, respectively.
- The **GPS sensor** supplies global-positioning satellite data for location-based and mapping apps.

iSight Camera

The iSight 8-megapixel rear-facing camera includes an illumination sensor, an LED (light-emitting diode) flash and zoom capabilities. The zero shutter lag allows you to capture photos without a delay. The camera also has face detection—it determines whether one or more faces are in the frame, then focuses on the most prominent face and balances the exposure across all of the faces (up to a maximum of 10). The rear-facing camera allows you to record high-definition (HD) video. You can edit the videos directly in the **Camera** app or purchase the iMovie app or Avid Studio iPad app (for sale through the App Store) for more sophisticated editing capabilities. You can easily share photos and videos via e-mail, MMS, YouTube® or your **Photos** app.

Bluetooth

You can connect compatible Bluetooth stereo headphones and other accessories to your device. Also, **Internet tethering** enables users in some countries to connect to a Wi-Fi or 3G network on their laptop by using their iOS device as a modem (connected to their laptop via Bluetooth, Wi-Fi or USB cable). iOS now also provides support on recent iPhones and iPads for Bluetooth Low Energy devices, such as heart-rate monitors.

Accessibility

iOS 3.x and higher includes **accessibility** features to help vision-, hearing- and physically impaired users. **VoiceOver** is a gesture-based screen-reader program available in numerous languages. It lets vision-impaired users interact with objects on the screen and understand their context. For example, vision-impaired users can touch the screen to hear a description of the item they touch, then drag their finger to hear descriptions of the surrounding content. VoiceOver is also used with the keyboard to speak each character touched, or each complete word. Starting with iOS 6, VoiceOver is integrated with **Maps**. Users can select a spoken language temporarily without changing the system settings.¹⁸ The voice-recognition capabilities allow you to use voice commands to access features on the phone, such as making phone calls and playing music. Vision-impaired users can also pair their device with a Bluetooth-enabled refreshable braille display.

Users with low vision can change their device display to **Large Text** for readability or **White on Black** for higher contrast, or use **Zoom** to magnify the screen 100–500 percent (including the home screen, all apps, etc.). To magnify the screen, double tap with three fin-

18. www.apple.com/iphone/features/accessibility.html.

gers and drag up to zoom in or down to zoom back out. To turn on **Zoom**, **White on Black** and other accessibility features on the device, go to **Settings > General > Accessibility**.

For hearing-impaired users, iOS has closed-captioning capabilities, MMS texting, visible and vibrating alerts **FaceTime** video calling (available on iPhone 4 and higher and the new iPad running iOS 6) and more. For physically impaired users, **AssistiveTouch** enables entry of Multi-Touch gestures with one finger or a stylus (sold separately). Also, Siri—the personal digital assistant available on iPhone 4S and higher and the new iPad running iOS 6—enables voice entry of numerous commands.

Check out the overview of accessibility features at www.apple.com/accessibility/. To view the *Accessibility Programming Guide for iOS*, visit

```
developer.apple.com/library/ios/#documentation/UserExperience/
Conceptual/iPhoneAccessibility/Introduction/Introduction.html
```

1.5 iOS

In this section we provide a brief history and feature summary of the various versions of the iOS mobile operating system. Originally designed for the iPhone, iOS now also runs on iPod touch, iPad and Apple TV. It's a *proprietary* operating system tightly controlled by Apple and available only on Apple's devices. Google's Android operating system is open source and available for use on third-party devices. iOS does use various open-source libraries. For information on this, visit:

```
opensource.apple.com
```

The iPhone Operating System

The iPhone operating system (later renamed iPhone OS, then iOS) was released in June 2007 along with the first-generation iPhone. The operating system included the **iPod** (media player), **Messages** (for SMS text messaging), **Calendar**, **Camera**, **Photos**, **Maps** and a few other default apps.

iPhone OS 2: Introducing Third-Party Apps and the App Store

iPhone OS 2 and the iPhone 3G—released in July 2008—introduced third-party apps. With the iPhone SDK, developers could create apps for the iPhone and iPod touch. Using the built-in frameworks, developers could build apps that access some of the core functionality of the phone, such as **Contacts**, **SMS** and more. The App Store was launched as a marketplace where users could download free and for-sale apps.

iPhone OS 3

iPhone OS 3.0 was released in June 2009 and introduced many new features, including

- the ability to cut, copy and paste text within and between apps
- landscape keyboard
- recording voice memos using the built-in microphone
- multimedia messaging to send photos and videos via the **Messages** app
- Spotlight search for locating e-mail, contacts, calendars, notes and music in your iPod library
- iTunes access directly from an iPhone

- broader language support—30 spoken languages
- peer-to-peer Bluetooth connectivity for transferring data between phones

iOS 4

iOS 4 and the iPhone 4 were released in June 2010. Figure 1.7 lists some key features of iOS 4. One notable new feature for users was *multitasking* which allowed multiple apps to run simultaneously. iOS 4 also added several developer frameworks for integrating some of the core functionality of the device into your apps. For example, the Event Kit framework is used to access events in the **Calendar** app and the Core Motion framework replaced and enhanced earlier iOS capabilities for reading a device’s motion data from sensors such as the accelerometer, gyroscope and magnetometer. iOS 4 also added Grand Central Dispatch (GCD), which provided a new *asynchronous programming* model that was more efficient than the traditional multithreading model provided in earlier iOS versions. For a complete list of iOS 4 API additions, visit developer.apple.com/library/ios/release-notes/General/iPhone40APIDiffs/.

Feature	Description
Multitasking	For certain app types (e.g., GPS and Audio), you can run multiple apps simultaneously and switch between them without losing data.
FaceTime	Takes advantage of the front- and rear-facing cameras, allowing you to make <i>video calls</i> on the phone. Select a contact from Contacts and tap the FaceTime button, or if you’re already on a call, tap the FaceTime button to switch to a video call. An invitation to join the video call appears on your contact’s device screen. If the invitation is accepted, the video call starts immediately.
iAd	The mobile advertising platform allows you to monetize your apps with in-app banner advertising. Many in-app ads when clicked will open the advertiser’s website in a web browser, taking the user out of your app. iAd opens the ads—full-screen video and interactive ad content—within your app; when done viewing the content, users can close an ad and continue using the app. Apple handles all ad sales and delivers them to the users’ devices. Developers who implement iAd in their apps receive 70 percent of iAd revenue. At the time of this writing, iAd was available in France, Germany, Italy, Japan, Spain, the U.S. and the U.K.
Apple Push Notification	Allows apps to receive notifications, even when they aren’t running. The service can be used, for example, to notify the user when a new version of your app is available for download or to send news and messages to users.
High Dynamic Range (HDR) Photos	Allows you to capture the best exposure for your photos. To create an HDR photo, three photos are taken in rapid succession at varying exposures—low, normal and high. The three photos are then merged using an algorithm that maps the tones across the three images into a single image with optimized tones throughout. The final HDR photo and the original photo are both saved.

Fig. 1.7 | Key iOS 4 features (developer.apple.com/library/ios/#releasenotes/General/WhatsNewIniPhoneOS/Introduction/Introduction.html). (Part 1 of 2.)

Feature	Description
Game Center	The Game Center APIs allow you to create social, multiplayer games. Users can play against friends or find other opponents worldwide, track their scores and compare scores with those of other players.
iTunes TV Show Rentals	Rent commercial-free TV shows for \$0.99 per episode.
iTunes Ping	Ping is a social network for music discovery. Built into iTunes 10, Ping allows users to see what their friends are listening to, follow their favorite artists, see a customized Top 10 list and more.
Folders	Organize apps into folders by dragging and dropping one app icon on top of another.
Improved e-mail	Receive e-mails from multiple accounts in a single inbox, organize messages by threads, check spelling, search your messages and more.
iBooks	Download e-books from the iBooks store to read on an iPhone, iPad or iPod touch.
Create playlists	Create customized music playlists directly on the device.
Spell Checking	New spell-checking functionality works in Mail , Notes , Messages and more.
Wireless Keyboard Support	Pair your device with a wireless Bluetooth keyboard.
iPad Support	iPad support started with iOS 3.2.

Fig. 1.7 | Key iOS 4 features (developer.apple.com/library/ios/#releasenotes/General/WhatsNewIniPhoneOS/Introduction/Introduction.html). (Part 2 of 2.)

iOS 5 Features and Enhancements

iOS 5.x includes several features and enhancements for users and developers, including over a thousand new APIs and tools (Fig. 1.8). For a detailed list, see developer.apple.com/library/ios/#releasenotes/General/iOS50APIDiff/.

Feature	Description
iCloud	iCloud allows users to store data such as music, photos and videos, documents and e-mail virtually (“in the cloud”) and then pushes the data to all of their iOS devices. iCloud Storage APIs allow you to create apps that write and store users’ data in the cloud. That data can then be accessed and modified by users from any of their iOS or Mac devices without transferring files or syncing devices.
Game Center	As of iOS 5, you can post pictures to your Game Center profile and track your overall scores. You can play against people you know or find recommended opponents based on the games you play.

Fig. 1.8 | iOS 5.x user features (www.apple.com/ios/features.html). (Part 1 of 2.)

Feature	Description
Notification Center	Places text, e-mail, voice mail, friend requests, stock prices, weather and other notifications in one place. To access the Notification Center on a device, swipe downward from the top of the screen.
Reminders	Create to-do lists that automatically sync with the Calendar , Mail and iCloud . Location-based alerts remind you to complete an item on the list.
Newsstand App	Places users' newspaper and magazine apps in one folder. When new subscriptions are released, they're automatically loaded into the Newsstand app. The Newsstand Kit and Store Kit frameworks allow you to create apps that <i>push</i> (i.e., automatically send) magazine and newspaper content to the app users' devices.
Camera	Quickly access the Camera app from the Lock screen and press the volume-up button to take a photo. Enable Photo Stream in iCloud to automatically download photos to your other iOS devices.
Twitter integration	Users can tweet directly from Camera , Photos , YouTube , Safari or Maps , and store friends' Twitter usernames in Contacts . The iOS Twitter account API allows you to integrate Twitter into your apps.
Safari browser	Improved performance plus new features such as tabbed browsing on the iPad and a Reading List that allows you to save web pages to read later on any of your iOS devices connected to iCloud .
PC Free	Wirelessly activate and update iOS devices via Wi-Fi without connecting directly to a computer.
AirPrint	Print wirelessly from apps on an iOS device to printers that support AirPrint . For a list of printers from Apple and other manufacturers, see support.apple.com/kb/ht4356 .
Accessibility	Features include an LED flash and custom vibration settings that allow users to see or feel incoming calls, support for Bluetooth-enabled braille displays, audible alerts, speak selection to read highlighted text and more.
Mail	New formatting capabilities include italic, bold, underlined and indented text. You can also flag messages, add and delete folders, search within the body of a message and more. The free e-mail account for iCloud users syncs across iOS devices.
Siri	Available on iPhone 4S and higher, the Siri personal assistant allows you to use your voice to perform numerous tasks on the device. You can tell Siri to make a phone call, dictate and send SMS and e-mail messages, schedule events and appointments on your calendar, perform a web search, find a location on a map, check the weather and more.

Fig. 1.8 | iOS 5.x user features (www.apple.com/ios/features.html). (Part 2 of 2.)

1.6 iOS 6

iOS 6, announced at the Apple World Wide Developer Conference (WWDC) 2012, includes approximately 200 new features. Figure 1.9 summarizes some of the key updates and enhancements.

Feature	Description
Game Center and the Game Kit framework	<p>Game Center and the Game Kit framework include several new and updated features:</p> <ul style="list-style-type: none"> • <i>Challenges</i>, which allow users to invite their friends to beat an achievement (when a player meets a goal) or a score. • Simultaneous submission of multiple achievements. • Achievement, leaderboard and friend request view controllers are now included in a tab in the Game Center view controller. • Increased control over <i>local-player authentication</i>. • <i>Player timeout support</i>. You create a list of players; when a player takes a timeout, the next player in the list is asked to take a turn. • Improved support for <i>matchmaking</i>, allowing you to match players to other players programmatically. Players can then send and receive match invitations. • Support for <i>players' display names</i>. • Determining which player has the <i>best connection</i> to Game Center.
Social framework	<p>Replaces the Twitter framework from iOS 5. The Social framework allows you to build apps that access the user's social media accounts—including Facebook, Twitter and Sina Weibo (China's most popular social media site)—to post status updates and images.</p>
Maps	<p>Additions and enhancements to the Maps app and Map Kit framework include:</p> <ul style="list-style-type: none"> • Improved ability to <i>launch the Maps app from within your app</i> to display directions or points of interest. • Apps that provide directions can be registered as routing apps. This allows other apps—including Maps—to use the directions. Also, the Maps app can direct users to the routing apps in the App Store. • New interfaces allow apps that do not offer routing information to query Maps for directions and points of interest.
Pass Kit	<p>Passes are a digital replacement for tickets (e.g., concert or show tickets, airline boarding passes), membership cards, coupons, etc.—basically items that are normally printed and used or redeemed physically (not online). Passes include information for the user about the pass (e.g., event details, coupon description, etc.) and, if necessary, a bar code or other data to validate the pass for redemption. Users can manage their passes in the Passbook app. <i>Your</i> web service creates the pass and delivers it to the user either through your app, e-mail or Safari.</p>

Fig. 1.9 | Key new iOS 6 features for developers. (Part 1 of 2.)

Feature	Description
In-App Purchase	Store content available for in-app purchase on Apple's servers rather than hosting it yourself. Also, users can purchase iTunes content (e.g., other apps, music and books) from within your app.
iAd	New banner sizes designed for display in iPad apps.
Reminders	Your apps can create and access reminders that appear in the user's Reminders app. The reminders can be set for a given time or triggered when the user enters or exits a specified location.
Collection views	Customize the layout of your data, include animated content, easily create and manage cells and views, and insert, move and delete items in batches.
Auto Layout	Set guidelines for laying out user-interface elements.
State Preservation	Apps can save and restore the user interface to the state it was in when the user last used the app.

Fig. 1.9 | Key new iOS 6 features for developers. (Part 2 of 2.)

iOS 6 User Features

iOS 6 includes several updates and new features for users. Figure 1.10 list some of the key new user features.

Feature	Description
Siri	<p>Siri—which is available on recent iPhones and iPads—can now:</p> <ul style="list-style-type: none"> • Provide sports scores, batting averages, player stats and team standings. • Return results from Rotten Tomatoes (movie information), Yelp! (business listings and reviews) and OpenTable (restaurant reservations). • Launch apps. • Post status updates on Facebook and tweets on Twitter. • Read turn-by-turn directions. • Be integrated into cars via the <i>Eyes Free</i> feature, enabling drivers to ask Siri for directions, change the radio station and more. Vehicle manufactures planning support for this include Audi, BMW, General Motors, Mercedes-Benz and Toyota.
FaceTime	<p>FaceTime is now available over cellular and can be used to make video calls across iPhone, iPad and Mac devices. iMessage works similarly.</p>

Fig. 1.10 | Key new iOS 6 features for users (www.apple.com/ios/ios6). (Part 1 of 2.)

Feature	Description
Facebook integration	<p>Facebook has been integrated into iOS 6, allowing users to perform the following tasks:</p> <ul style="list-style-type: none"> • Tap from Notification Center to post a status update. • View Facebook friend details in Contacts and events in the Calendar. • Post photos from within Photos or Camera, game scores from Game Center and location from Maps.
Passbook	<p>Stores the users tickets, boarding passes, coupons and loyalty cards in one place. Passbook's time- and location-based services display passes as they're needed and the barcodes can be scanned directly from the iOS device. For example, coupons, loyalty cards and gift certificates are displayed when the user enters the related business, and boarding passes are displayed when the user arrives at the airport.</p>
Maps	<p>New Maps features include:</p> <ul style="list-style-type: none"> • Turn-by-turn navigation, real-time traffic updates and an estimated time of arrival (ETA). • Siri can access Maps, help users find a location and speak the directions. • Users can rotate and tilt the iOS device to change the map view. • Flyover provides a high-definition aerial view of metropolitan areas. • Local search with over 100 million business listings, according to Apple's Scott Forstall at WWDC 2012.
Photo sharing	<p>Users can send pictures from the Photos app to friends who are using iCloud on an iOS 6 device or a Mac running OS X Mountain Lion. The shared pictures appear in their friends' Photos app on iOS devices or iPhoto on the Mac. Friends can also view the shared pictures on the web and on Apple TV. Users and their friends can add comments to the photos.</p>
Phone	<p>The Do Not Disturb setting allows users to block all calls or allow only certain callers to get through. Users can quickly respond to incoming calls with text messages and set reminders to call back.</p>

Fig. 1.10 | Key new iOS 6 features for users (www.apple.com/ios/ios6). (Part 2 of 2.)

1.7 Downloading Apps from the App Store

At the time of this writing, there were over 650,000 apps in the App Store—of which 225,000 were iPad apps.¹⁹ The number of apps available is growing rapidly. Figure 1.11

19. mashable.com/2012/06/11/wwdc-2012-app-store-stats/.

lists some popular ones. You can download apps directly onto your device through Apple’s **App Store**, or download apps through iTunes. iCloud automatically (and wirelessly) pushes the downloaded apps to your other iOS-compatible devices that share the same iCloud account—you need not connect the devices. You can also **sync** the device with iTunes wirelessly or by using a USB cable to connect the device to a computer. Syncing allows you to back up your information (contacts, apps and their data, music, photos, videos, and so on) and download new information onto the device. The App Store notifies you when updates to your downloaded apps are available.

Category	Sample apps
Books	NOOK by Barnes & Noble, Kindle, Audible, Goodreads
Business	Dragon Dictation, TurboScan, Adobe Reader, Job Search, Square
Catalogs	Motoscooting, Course Monkey, Stanford Continuing Studies, SkyMall
Education	Starwalk, iTunes U, Graphing Calculator, Learn Spanish, NASA App
Entertainment	Lock My Screen™, AgingBooth, Fandango®, i.TV, FatBooth, Hulu Plus
Finance	Bloomberg for iPad, PayPal™, Ace Budget, BillTracker
Games	Angry Birds, Cooking Academy, Fruit Ninja, Jaws™, Skee-Ball
Healthcare and Fitness	iFitness, Lose It!, Fast Food Calorie Counter, Pedometer, Couch to 5K, WebMD Mobile, RunKeeper Free, Nike BOOM, iMapMyRun
Lifestyle	How to Cook Everything, AroundMe, CraigsPro Free, eBay Mobile
Medical	Epocrates, Medscape, iStethoscope Pro, BMI Tool, Pocket Lab Values
Music	Shazam Encore, I Am T-Pain, Pandora Radio, VEVO, Hitmaker
Navigation	Waze, MapQuest® Navigator, Trailhead, MotionX™ GPS Drive
News	CNN, NYTimes, USA Today, WSJ, Flipboard Yahoo!®
Photography	Instagram, Viddy, iMovie, iPhoto, Camera+, Photobucket, Infinicam
Productivity	Dropbox, Nubi Do, iTranslate, GoodReader, Electronic Toolbox
Reference	Bing, Google® Search, Dictionary.com, Wikipedia Mobile, Ancestry
Social Networking	Facebook®, Pinterest, Twitter, Skype™, LinkedIn®, WhatsApp Messenger
Sports	ESPN® ScoreCenter, NFL Live Football, Bike Repair, MLB.com At Bat
Travel	FlightTrack, GasBuddy, Google Earth, Yelp®, Wikihood Plus, Kayak
Utilities	Battery Boost Magic, iHandyLevel Free, RedLaser Barcode Reader
Weather	The Weather Channel®, WeatherBug®, ZYRTEC® AllergyCast

Fig. 1.11 | Popular iPhone and iPad apps in the App Store.

Visit www.apple.com/iphone/apps-for-iphone/ to check out Apple’s featured apps. Some are *free* and some are *fee based*. Developers set the prices for their apps sold through the App Store and receive 70 percent of the revenue. Many app developers offer free versions of their apps as a marketing strategy, so users can download them and see whether they like them, then purchase more feature-rich versions. We discuss this so-called “lite” strategy in more detail in Section 2.6.

1.8 Objective-C Programming Language

Apple was founded in 1976 by Steve Jobs and Steve Wozniak and quickly became the leader in personal computing. In 1979, Jobs and several Apple employees visited Xerox PARC (Palo Alto Research Center) to learn about Xerox's desktop computer that featured a graphical user interface. That GUI inspired the Apple Lisa personal computer (designed for business customers) and, more notably, the Apple Macintosh. Steve Jobs left Apple in 1985 and founded NeXT Inc. to develop computers primarily for use in colleges.

The **Objective-C** programming language, created by Brad Cox and Tom Love at StepStone in the early 1980s, added capabilities for object-oriented programming (OOP) to the C programming language. In 1988, NeXT licensed Objective-C from StepStone and developed an Objective-C compiler and libraries which were used as the platform for the NeXTSTEP operating system's user interface and Interface Builder—used to construct graphical user interfaces (we discuss Interface Builder in more detail in Section 1.10). Apple's Mac OS X is a descendant of NeXTSTEP—a Unix-based operating system. According to a July 2012 report by TIOBE, Objective-C is now the third most popular programming language behind C and Java due to the popularity of the iPhone and iPad.²⁰ An article in eWeek does a nice job summarizing the strengths of Objective-C.²¹

Over the past few years, Apple has added many new features to Objective-C. Our will look different from code you will see or might have seen in older books, blog posts, tutorials, etc., because this book is for iOS 6 and uses modern Objective-C syntax and a features of Apple's LLVM compiler.

1.9 Cocoa Touch and iOS Frameworks

Objective-C is object oriented and has access to the **Cocoa frameworks** (powerful class libraries of prebuilt components), enabling you to develop apps quickly. **Cocoa Touch** is the version of Cocoa for iOS devices.

Cocoa also evolved from projects at NeXT. OpenStep was developed at NeXT as an object-oriented programming API to be used in developing an operating system. After Apple acquired NeXT, the OpenStep operating system evolved into Rhapsody, and many of the base libraries became the Yellow Box API. Rhapsody and Yellow Box eventually evolved into OS X and Cocoa, respectively.

Cocoa Touch programming in Objective-C is *event driven*—in this book, you'll write apps that respond to timer firings and user-initiated events such as touches and keystrokes. In addition to directly programming portions of your Objective-C apps, you'll also use Interface Builder in Xcode to conveniently drag and drop predefined objects such as buttons and textboxes into place on your screen, label and resize them, and connect them to your code. With Xcode, you can create, run, test and debug iOS apps quickly and conveniently.

Several Cocoa Touch and iOS frameworks allow you to conveniently access iOS features and incorporate them into your apps (Figs. 1.12–1.15). They're written mainly in Objective-C (though some are written in C), and are accessible to Objective-C programs. The frameworks help you create apps which adhere to the iOS's unique look and feel.

20. www.tiobe.com/index.php/content/paperinfo/tpci/index.html.

21. www.eweek.com/c/a/Application-Development/ObjectiveC-Is-Kicking-Butt-in-the-Programming-World-813076/.

Framework	Description
Address Book UI	Interface for displaying contact information from the user's Address Book.
Core Animation	Create dynamic animations with smooth transitions and other effects.
Core Image	Process images and videos. The Core Image APIs include class <code>CIImage</code> , which can be used to perform basic facial recognition in images. With this class, you can determine the left- and right-eye positions and the mouth position, and whether these features are included in the image.
Core Text	APIs for text layout and handling fonts.
Core Motion	Used to receive and process accelerometer and gyroscope data.
Game Kit	Voice and Bluetooth networking capabilities for games and other apps.
Map Kit	Add maps and satellite images to location-based apps. Annotate maps, identify areas on a map using overlays and more.
Message UI	Create e-mail messages from within an app. Create and send SMS messages from within an app.
Quartz 2D	A 2D graphics API that allows you to create graphics such as gradients, transformations and Bézier curves.
UIKit	Classes for creating and managing a user interface, including event handling, drawing, windows, views and Multi-Touch interface controls.
WebKit	A high-level Mac OS X framework—based on the open-source web browser engine—that's used in apps to render HTML, store cookies, authenticate users and more.

Fig. 1.12 | Cocoa Touch layer frameworks for building graphical, event-driven apps. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks).

Framework	Description
Accelerate	C programming APIs for performing digital signal processing, image processing and complex vector and matrix math.
Assets Library	Framework for accessing the user's media library including photos and videos uploaded onto the device or stored in the user's Photos app. Also allows your app to save new photos and videos to the user's photo albums.
Audio Toolbox	Interface for audio recording and playback of streamed audio and alerts.
Audio Unit	Interface for using the iPhone OS audio processing plug-ins.

Fig. 1.13 | Media Layer frameworks for adding audio, video, graphics and animation to your apps. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks). (Part 1 of 2.)

Framework	Description
AV Foundation	Interface for audio recording and playback (similar to the Audio Toolbox). Includes media asset management and editing, video capture and playback, track management, metadata management for media, stereophonic panning, sound synchronization and an Objective-C interface for determining the format, sample rate and number of channels for sound files. Also includes classes for playing a sequence of media objects, reading samples from media files and writing samples to a data file.
Core Audio	Framework for declaring data types and constants used by other Core Audio interfaces.
Core Graphics	API for drawing, rendering images, color management, gradients, coordinate-space transformations and handling PDF documents.
Core Media	Framework for creating, playing and managing audio and video.
Core Video	C-based APIs for video playback, editing and processing.
iAd	In-app advertising framework used to place full-screen or banner advertisements within an app for monetization.
Media Library	Access the music library on the device from within your apps.
Media Player	Finds and plays audio and video files within an app.
OpenAL	OpenAL is an open-source library included in iOS 5 for three-dimensional sound. You can learn more about OpenAL at connect.creative-labs.com/openal/default.aspx .
OpenGL ES	Supports integration with the Core Animation layer and UIKit views. Subset of the OpenGL API for 2D and 3D drawing on embedded systems.
Quartz Core	Framework for image and video processing, and animation using Core Animation technology.
Quick Look	If your app downloads files from the network or other sources, this framework can be used to display previews of files even if they're in formats not directly supported by your app (e.g., Microsoft Office documents).

Fig. 1.13 | Media Layer frameworks for adding audio, video, graphics and animation to your apps. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks). (Part 2 of 2.)

Framework	Description
Address Book	Framework for accessing the user's Address Book contacts.

Fig. 1.14 | iOS Core Services layer frameworks. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks). (Part 1 of 2.)

Framework	Description
Bonjour	Allows you to automatically locate systems and services on a local network, such as iTunes, iChat and more.
BSD Sockets	Networking APIs.
Core Data	Framework for performing tasks related to object life-cycle and object graph management.
Core Foundation	Library of programming interfaces that allow frameworks and libraries to share code and data. Also supports internationalization.
Core Location	Used to determine the location and orientation of an iPhone, then configure and schedule the delivery of location-based events.
Core Telephony	Used to get information about the user's mobile service provider.
Event Kit	Allows your apps to access data in the user's Calendar app and to add and edit events in the Calendar app.
Foundation	Includes <code>NSObject</code> (used to define object behavior), plus tools for creating graphical, event-driven apps. Also includes design patterns and features for making your apps more efficient.
Event Kit UI	View controller classes for editing, creating and displaying calendar events from within your apps.
Image IO	C-based framework for reading and writing image-file formats and accessing image metadata. Also handles color management.
Mobile Core Services	Includes standard types and constants.
SQLite	A lightweight relational database you can embed in your apps.
Store Kit	In-app purchase support for processing transactions.
System Configuration	Determines network availability and state on an iPhone.

Fig. 1.14 | iOS Core Services layer frameworks. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks). (Part 2 of 2.)

Framework	Description
CFNetwork	Framework using network protocols in apps to perform tasks including working with HTTP and authenticating HTTP and HTTPS servers, working with FTP servers, creating encrypted connections and more.
External Accessory	Allows the iPhone to interact with third-party authorized accessories connected via Bluetooth or the dock connector.

Fig. 1.15 | Core OS layer frameworks for accessing the iOS kernel. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks). (Part 1 of 2.)

Framework	Description
Security	Framework for securing data used in an app.
System	BSD operating system and POSIX API functions.

Fig. 1.15 | Core OS layer frameworks for accessing the iOS kernel. (developer.apple.com/library/ios/navigation/index.html#section=Frameworks). (Part 2 of 2.)

Web services are software components stored on one computer that can be accessed by an application (or other software component) on another computer over a network. With web services, you can create **mashups**, which enable you to rapidly develop apps by combining the complementary web services of several organizations and possibly other forms of information feeds. One of the first mashups was www.housingmaps.com, which combined the real-estate listings provided by www.craigslist.org with the mapping capabilities of Google Maps to offer maps that show the locations of apartments for rent in a given area. Figure 1.17 lists some popular web services.

Web services source	How it's used
Google Maps	Mapping services
Facebook	Social networking
Foursquare	Mobile check-in
LinkedIn	Social networking for business
YouTube	Video search
Twitter	Microblogging
Groupon	Social commerce
Netflix	Movie rentals
eBay	Internet auctions
Wikipedia	Collaborative encyclopedia
PayPal	Payments
Last.fm	Internet radio
Amazon eCommerce	Shopping for books and more
Salesforce.com	Customer relationship management (CRM)
Skype	Internet telephony
Microsoft Bing	Search
Flickr	Photo sharing
Zillow	Real-estate pricing
Yahoo Search	Search
WeatherBug	Weather

Fig. 1.16 | Some popular web services (www.programmableweb.com/apis/directory/1?sort=mashups).

Figure 1.17 lists directories where you'll find information about many of the most popular web services.

Directory	URL
ProgrammableWeb	www.programmableweb.com
Webmashup.com	www.webmashup.com/
Webapi.org	www.webapi.org/webapi-directory/
Google Code API Directory	code.google.com/apis/gdata/docs/directory.html
APIfinder	www.apifinder.com/

Fig. 1.17 | Web services directories.

1.10 Xcode Toolset

The Xcode 4 toolset, bundled with all Mac OS X versions since v10.5, is faster and easier to use than previous versions (Fig. 1.18). It's available for free through the Mac App Store (you must be a registered Apple Developer to download beta versions of Xcode and iOS from developer.apple.com/ios). The toolset includes the Xcode IDE, Interface Builder, support for the Objective-C 2.0 language, the Instruments tool (used to improve performance) and more.

Feature	Description
Auto Layout	By default, new Cocoa projects use the Auto Layout feature in Interface Builder.
Xcode.app	Xcode is now an easy-to-install app bundle available through the Mac App Store. Xcode.app supports iOS 6. To access the tools in Xcode, go to Xcode > Open Developer Tool menu.
Single Window	Development can now be done using one editing window. You can navigate between files, debugging data and more using the links on the left side of the window. Use the jump bar at the top of the editor window to switch between files. You can still edit a file in its own window by double-clicking the filename in the Project navigator (Chapter 3).
Storyboarding	Using Interface Builder, you can create a storyboard that graphically maps the paths a user can take through your app, including each screen, transitions and the app's controls.
Xcode Assistant	When you work in Xcode's editor with two panes, Xcode Assistant anticipates other files that you might need to look at. For example, if you're working on a class's implementation, Xcode Assistant displays the corresponding header file, or if a new class you're defining inherits from a superclass, Xcode Assistant displays the superclass.

Fig. 1.18 | Some Xcode 4.x features (developer.apple.com/library/mac/#documentation/DeveloperTools/Conceptual/WhatsNewXcode/00-Introduction/Introduction.html). (Part 1 of 2.)

Feature	Description
Integrated Interface Builder	Interface Builder is a visual GUI design tool. GUI components can be dragged and dropped into place to form simple GUIs without any coding. Interface Builder files use the <code>.xib</code> extension, but earlier versions used <code>.nib</code> —short for NeXT Interface Builder. For this reason, Interface Builder <code>.xib</code> files are commonly referred to as “nib files.” Interface Builder is now integrated with Xcode—it’s no longer a separate application. To open the Interface Builder editor in Xcode, select the <code>.nib</code> (or <code>.xib</code>) file in the project. The utilities on the right include controls, UI objects, etc. Simply drag and drop controls onto the user interface for your iOS app. You’ll learn more about Interface Builder in Chapter 3.
Automatic Reference Counting (ARC)	The improved memory management reduces the opportunity for memory leaks and crashes.
LLVM Compiler	A fast, open-source compiler that’s fully integrated into the Xcode IDE. Features syntax highlighting, code completion and more. As of Xcode 4, the LLVM compiler is the default—previously, it was GNU.
Fix-it	The LLVM Fix-it feature flags mistakes in your code and suggests corrections as you type—you don’t need to build the app first.
Version Editor	View multiple versions of your source code, side by side so you can easily compare them, view a log of past events and more.
Location Simulation	You can now select from a list of locations in the simulator to run location-based apps that use Core Location.
LLDB Debugger	Includes a faster, more efficient multicore debugging engine.
New Instruments	The new Instruments interface makes it easier to test your app, monitor memory allocation, track graphics performance with OpenGL ES, track the interaction of system processes and more.

Fig. 1.18 | Some Xcode 4.x features (developer.apple.com/library/mac/#documentation/DeveloperTools/Conceptual/WhatsNewXcode/00-Introduction/Introduction.html). (Part 2 of 2.)

Xcode Integrated Development Environment (IDE)

The Xcode integrated development environment (IDE) supports many programming languages including Java, C++, C, Python and Objective-C. iOS’s primary programming language is Objective-C, but C and C++ can also be used for iOS development. It includes a code editor with support for syntax coloring, autoindenting and autocomplete; a debugger and a version-control system. You’ll start using Xcode to develop apps in Chapter 3.

The iOS Simulator

The iOS SDK’s iOS simulator allows you to test iOS apps on your Mac. The simulator displays a realistic iPhone or iPad user-interface window. Not all device capabilities are available in the simulator. For example, the camera—which is commonly used in iOS

apps—does not work in the simulator. You can reproduce on the simulator many of the single-touch and multitouch gestures using your Mac’s keyboard and mouse (Fig. 1.19).

Gesture	Simulator action
Tap	Click the mouse once.
Double Tap	Double click the mouse.
Touch and Hold	Click and hold the mouse.
Drag	Click, hold and drag the mouse.
Swipe	Click and hold the mouse, move the pointer in the swipe direction and release the mouse.
Flick	Click and hold the mouse, move the pointer in the flick direction and quickly release the mouse.
Pinch	Press and hold the <i>Option</i> key. Two circles that simulate the two touches will appear. Move the circles to the start position, click and hold the mouse and drag the circles to the end position.

Fig. 1.19 | Gestures on the iOS simulator (developer.apple.com/library/ios/documentation/Xcode/Conceptual/ios_development_workflow/25-Using_iOS_Simulator/ios_simulator_application.html).

Although the iOS simulator can simulate **orientation changes** (to portrait or landscape mode) and the *shake gesture*, there’s *no* built-in way to simulate accelerometer readings and readings from various other sensors. You can, however, install your app on an iPhone or iPad to test these features—you’ll learn about requirements for installing your app on a device in Section 2.2 and install an app on a device in Section 3.5. You’ll start using the simulator to develop apps in Chapter 3’s **Welcome** app.

As you’ll learn in Section 2.2, only members of Apple’s iOS Developer Program can install apps on a device for testing. If you’re not a paid member, there are third-party apps and libraries that you can use to transmit sensor data from an iOS device to an app running in the iOS simulator, such as those provided by Wavefront Labs (www.wavefrontlabs.com).

1.11 Object Technology: A Quick Refresher

This section presents a general introduction to object-technology concepts. Building software quickly, correctly and economically remains an elusive goal at a time when demands for new and more powerful software are soaring. *Objects*, or more precisely, the *classes* objects come from, are essentially *reusable* software components. There are date objects, time objects, audio objects, video objects, automobile objects, people objects, etc. Almost any *noun* can be reasonably represented as a software object in terms of *attributes* (e.g., name, color and size) and *behaviors* (e.g., calculating, moving and communicating). Software developers are discovering that using a modular, object-oriented design-and-implementation approach can make software-development groups much more productive than they could be with earlier popular techniques like “structured programming”—object-oriented programs are often easier to understand, correct and modify.

Automobile as an Object

To help you understand objects and their contents, let's begin with a simple analogy. Suppose you want to *drive a car and make it go faster by pressing its accelerator pedal*. What must happen before you can do this? Well, before you can drive a car, someone has to *design* it. A car typically begins as engineering drawings, similar to the *blueprints* that describe the design of a house. These drawings include the design for an accelerator pedal. The pedal *hides* from the driver the complex mechanisms that actually make the car go faster, just as the brake pedal hides the mechanisms that slow the car, and the steering wheel “hides” the mechanisms that turn the car. This enables people with little or no knowledge of how engines, braking and steering mechanisms work to drive a car easily.

Just as you cannot cook meals in the kitchen of a blueprint, you cannot drive a car's engineering drawings. Before you can drive a car, it must be *built* from the engineering drawings that describe it. A completed car has an *actual* accelerator pedal to make the car go faster, but even that's not enough—the car won't accelerate on its own (hopefully!), so the driver must *press* the pedal to accelerate the car.

Methods and Classes

Let's use our car example to introduce some key object-oriented programming concepts. Performing a task in a program requires a **method**. The method houses the program statements that actually perform its tasks. The method hides these statements from its user, just as the accelerator pedal of a car hides from the driver the mechanisms of making the car go faster. A program unit called a **class** houses the methods that perform the class's tasks. For example, a class that represents a bank account might contain one method to *deposit* money to an account, another to *withdraw* money from an account and a third to *inquire* what the account's current balance is. A class is similar in concept to a car's engineering drawings, which house the design of an accelerator pedal, steering wheel, and so on.

Instantiation

Just as someone has to *build a car* from its engineering drawings before you can actually *drive* a car, you must *build an object* of a class before a program can *perform* the tasks that the class's methods define. The process of doing this is called *instantiation*. An object is then referred to as an **instance** of its class.

Reuse

Just as a car's engineering drawings can be *reused* many times to build many cars, you can *reuse* a class many times to build many objects. Reuse of existing classes when building new classes and programs saves time and effort. Reuse also helps you build more reliable and effective systems, because existing classes and components often have gone through extensive *testing*, *debugging* and *performance tuning*. Just as the notion of *interchangeable parts* was crucial to the Industrial Revolution, *reusable classes* are crucial to the software revolution that has been spurred by object technology.

Messages and Methods Calls

When you drive a car, pressing its gas pedal sends a *message* to the car to perform a task—that is, to go faster. Similarly, you *send messages to an object*. Each message is a **method call** that tells a method of the object to perform its task. For example, a program might call a particular bank-account object's *deposit* method to increase the account's balance.

Attributes and Instance Variables

A car, besides having capabilities to accomplish tasks, also has *attributes*, such as its color, its number of doors, the amount of gas in its tank, its current speed and its record of total miles driven (i.e., its odometer reading). Like its capabilities, the car’s attributes are represented as part of its design in its engineering diagrams (which, for example, include an odometer and a fuel gauge). As you drive an actual car, these attributes are carried along with the car. Every car maintains its *own* attributes. For example, each car knows how much gas is in its *own* gas tank, but *not* how much is in the tanks of *other* cars.

An object, similarly, has attributes that it carries along as it’s used in a program. These attributes are specified as part of the object’s class. For example, a bank-account object has a *balance attribute* that represents the amount of money in the account. Each bank-account object knows the balance in the account it represents, but *not* the balances of the *other* accounts in the bank. Attributes are specified by the class’s **instance variables**.

Encapsulation

Classes **encapsulate** (i.e., *wrap*) attributes and methods into objects—an object’s attributes and methods are intimately related. Objects may communicate with one another, but they’re normally not allowed to know how other objects are implemented—implementation details are *hidden* within the objects themselves. This **information hiding** is crucial to good software engineering.

Inheritance

A new class of objects can be created quickly and conveniently by **inheritance**—the new class absorbs the characteristics of an existing one, possibly *customizing* them and adding unique characteristics of its own. In our car analogy, a “convertible” certainly *is an* object of the more *general* class “automobile,” but more *specifically*, the roof can be raised or lowered.

Object-Oriented Analysis and Design (OOAD)

How will you create the code for your programs? Perhaps, like many programmers, you’ll simply turn on your computer and start typing. This approach may work for small programs, but what if you were asked to create a software system to control thousands of automated teller machines for a major bank? Or suppose you were asked to work on a team of 1,000 software developers building the next U.S. air traffic control system? For projects so large and complex, you should not simply sit down and start writing programs.

To create the best solutions, you should follow a detailed **analysis** process for determining your project’s **requirements** (i.e., defining *what* the system is supposed to do) and developing a **design** that satisfies them (i.e., deciding *how* the system should do it). Ideally, you’d go through this process and carefully review the design (and have your design reviewed by other software professionals) before writing any code. If this process involves analyzing and designing your system from an object-oriented point of view, it’s called an **object-oriented analysis and design (OOAD) process**. Languages like Objective-C are object oriented. Programming in such a language, called **object-oriented programming (OOP)**, allows you to implement an object-oriented design as a working system.

Design Patterns

Design patterns²² are proven reusable architectures that programmers use to solve recurring problems in object-oriented software development. In iOS app development, design

patterns establish a common design vocabulary among iOS app developers. By adhering to well-known iOS design patterns, you'll be able to shorten your app-design phase and take advantage of the powerful capabilities that the iOS APIs provide.

The notion of design patterns originated in the field of architecture. Architects use a set of established architectural design elements, such as arches and columns, when designing buildings. Designing with arches and columns is a proven strategy for constructing sound buildings—these elements may be viewed as architectural design patterns.

The most common design pattern you'll use in iOS app development is the **Model-View-Controller (MVC)** pattern, which separates app data (contained in the **model**) from graphical presentation components (the **view**) and input-processing logic (the **controller**). Figure 1.20 shows the relationships between components in MVC.

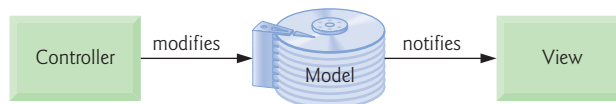


Fig. 1.20 | Model-View-Controller Architecture.

Consider an address book app. When the user adds a new contact's information through the app's GUI, the app's controller stores the contact in a database or file (the model). When the model changes, it notifies the view so that the updated list of contacts can be displayed. As you build iOS apps, you'll use an extensive set of common Cocoa Touch design patterns.

1.12 Test-Driving the SpotOn Game App in the iPhone and iPad Simulators

In this section, you'll run and interact with your first iOS app using both the iPhone and iPad simulators. The **Spot-On** game—a fun app that we created—tests your reflexes by requiring you to touch moving spots before they disappear (Fig. 1.21). The spots shrink as they move, making them harder to touch. The game begins on level one, and you reach each higher level by touching 10 spots. The higher the level, the faster the spots move—making the game increasingly challenging. When you touch a spot, the app makes a popping sound and the spot disappears. Points are awarded for each touched spot (10 times the current level). Accuracy is important—any touch that isn't on a spot decreases the score by 20 times the current level. You begin the game with *three* additional lives, which are displayed in the bottom-left corner of the app. If a spot disappears before you touch it, a flushing sound plays and you lose a life. You gain a life for each new level reached, up to a maximum of *seven* lives. When no additional lives remain and a spot's animation ends without the spot being touched, the game ends (Fig. 1.22).

22. Some books you'll want to consult on design patterns are the seminal “gang of four” book, *Design Patterns: Elements of Reusable Object-Oriented Software*, by Gamma, Helm, Johnson and Vlissides, ©1994, Addison-Wesley, and *Cocoa Design Patterns*, by Buck and Yacktman, ©2010, Addison-Wesley.

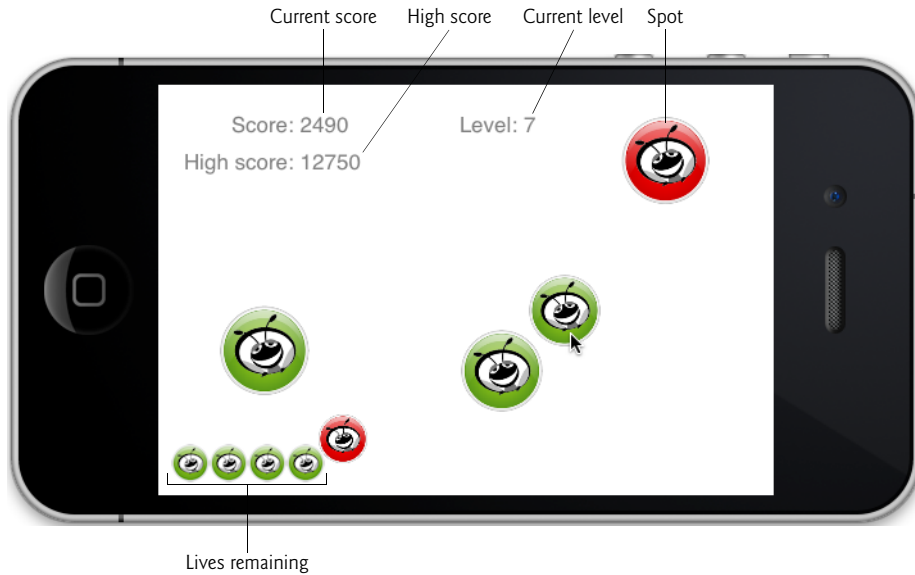


Fig. 1.21 | SpotOn Game app during game play.



Fig. 1.22 | Game Over alert.

Test-Driving the Completed Application Using the iPhone Simulator

The following steps show you how to test-drive the app:

1. *Checking your setup.* Confirm that you've set up your computer properly by reading the Before You Begin section located after the Preface.

2. *Locating the app folder.* Open a **Finder** window and navigate to the Documents/Examples folder or the folder where you saved the chapter's examples.
3. *Opening the SpotOn project.* Open the SpotOn folder, then double click the file name SpotOn.xcodeproj to open the project in Xcode.
4. *Launching the SpotOn game app.* In Xcode, click the **Scheme selector** to the right of the **Run** and **Stop** buttons in the upper-left corner of Xcode (Fig. 1.23), then select **iPhone x Simulator**—where *x* is the version of the iOS SDK you have installed. Next, click the **Run button** to run the app in the simulator. The game begins immediately when the app loads. Touch the moving red and green spots as fast as you can! Don't delay in touching a spot—if the spot disappears before you touch it, you'll lose a life!

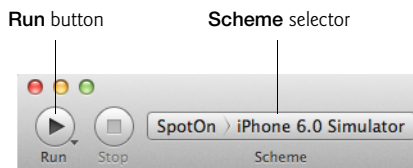


Fig. 1.23 | Run button and **Scheme** selector.

5. *Closing the app.* Close your running app by clicking the *Home* button on the simulator, or by selecting **iOS Simulator > Quit iOS Simulator** from the menu bar.

Test-Driving the Completed Application Using the iPad Simulator

To test-drive the app using the iPad simulator, click the **Scheme** selector, then select **iPad x Simulator**—where *x* is the version of the iOS SDK you have installed. Next, click the **Run** button to run the app in the simulator.

To get a broad sense of the capabilities that you'll learn in this book, check out the test-drives of the book's apps in Chapters 3 and higher.

1.13 iOS Developer Documentation

Figure 1.24 lists some of Apple's online documentation and resources that will help you learn to develop iOS apps. Most of these are available at `developer.apple.com`.

Title	
<i>iOS App Programming Guide</i>	<i>Pass Kit Programming Guide</i>
<i>iOS Human Interface Guidelines</i>	<i>What's New in Xcode</i>
<i>Creating Universal Applications</i>	<i>Xcode 4 User Guide</i>
<i>The Objective-C Programming Language</i>	<i>Cocoa Fundamentals Guide</i>

Fig. 1.24 | Key online documentation (most are at `developer.apple.com`) for iPhone and iPad developers. (Part 1 of 2.)

Title	
<i>Objective-C Runtime Programming Guide</i>	<i>Coding Guidelines for Cocoa</i>
<i>Accessibility Programming Guide for iOS</i>	<i>SDK Compatibility Guide</i>
<i>Game Kit Programming Guide</i>	<i>Sample Code</i>
<i>Social Framework Reference</i>	

Fig. 1.24 | Key online documentation (most are at `developer.apple.com`) for iPhone and iPad developers. (Part 2 of 2.)

1.14 Wrap-Up

This chapter presented an overview of the iPhone and iPad and discussed their functionality. We discussed the rapidly growing sales of iPhone and iPad devices. You learned about the device features. We presented a history of the iOS mobile operating system through the new iOS 6. You learned the various single-touch and multi-touch gestures, and how to perform each on iOS devices and using the simulator. We presented the history of the Objective-C programming language. We introduced the Cocoa Touch and iOS frameworks that enable you to quickly develop iOS apps. You'll use many of these frameworks in this book. We also introduced some key features of the Xcode toolset. We discussed basic object-technology concepts, including classes, objects, attributes and behaviors. You test-drove the **SpotOn** game app. We listed some key titles in the iOS developer documentation.

In Chapter 2, we discuss the business side of app development. You'll see how to prepare your app for submission to the App Store, including making icons and launch images. We'll discuss how to set up an iOS Developer Program profile so you can test your apps on devices and submit them to the App Store. We discuss the characteristics of great apps and the iPhone Human Interface Guidelines to follow when designing your apps. We provide tips for pricing and marketing your app. We also review the benefits of offering your app for free to drive sales of other products, such as a more feature-rich version of the app or premium content. We show how to use iTunes Connect to track app sales, payments and more. We also provide you with a list of some of the other app platforms to which you can port your apps.

Index

Symbols

^ (caret that begins a block) **97**
!= inequality operator **120**
@"" (empty string) **121**
@"string" NSString literal **119**
* in a pointer declaration **114**
* multiplication operator **97**
/ division operator **97**
& (address of) operator **119**
% remainder operator **97**
+ addition operator **97**
- subtraction operator **97**
< less than operator **120**
<= less than or equal to operator **120**
== equality operator **120**
> greater than operator **120**
>= greater than or equal to operator **120**

A

Accelerate framework **20**
accelerometer **9, 12**
Accessibility
 Large Text **10**
 VoiceOver **10**
 White on Black **10**
 Zoom **10**
accessibility **10, 14, 43, 85**
 Accessibility Inspector **175**
 accessibilityHint property of the UIAccessibility protocol **175**
 accessibilityLabel property of the UIAccessibility protocol **175**
 UIAccessibility protocol **87, 175**
Accessibility Inspector **175**

Accessibility Programming Guide for iOS **43**

accessibilityHint property of the UIAccessibility protocol **175**
accessibilityLabel property of the UIAccessibility protocol **175**
accessories **22**
action **96**
 create **111**
 action (event handler) **111**
Ad **36**
Ad Hoc distribution **36**
Ad Hoc Provisioning Profile **39**
add a segue **142**
addConstraint: method of class UIView **135, 158**
addConstraints: method of class UIView **135, 154, 159**
addition **97**
addObserver:selector:name:object: method of class NSNotificationCenter **149**
Address Book framework **21**
Address Book UI framework **20**
addSubview method of class UIView **159**
addTarget:action:forControlEvents: method of a UIControl **154**
admin **35, 36**
AdMob **49**
advertising revenue **49**
Agent (for a development team) **36**
AirPrint **14**
alertView:clickedButtonAtIndex: method **136**

alertView:clickedButtonAtIndex: method of the protocol UIAlertViewDelegate **163**
alignment in a Label **140**
alloc of class NSMutableArray **150**
alloc of class NSMutableDictionary **150**
alternate view **69**
Amazon Mobile app **48**
Ambient light sensor **10**
analysis **28**
API **19**
app xvii
app approval process **34**
app development xvii
App ID
 explicit **173**
app ID **36, 37, 41**
app name **69**
app platforms
 Amazon **63**
 Bing **63**
 BlackBerry **63**
 Chromium **63**
 eBay **63**
 Facebook **63**
 Flickr **63**
 Foursquare **63**
 Google **63**
 LinkedIn **63**
 Pinterest **63**
 Symbian **63**
 Twitter **63**
 Windows Phone **63**
 Yahoo **63**
 YouTube **63**
app record **52**
app review and recommendation sites **60**

- App Store xii, xvii, **18**, 34, 37, 48, 55
 - Books category 18
 - Business category 18
 - Catalogs category 18
 - Education category 18
 - Entertainment category 18
 - Finance category 18
 - Games category 18
 - Healthcare and Fitness category 18
 - Lifestyle category 18
 - Medical category 18
 - Music category 18
 - Navigation category 18
 - News category 18
 - Photography category 18
 - Productivity category 18
 - Reference category 18
 - Social Networking category 18
 - Sports category 18
 - Travel category 18
 - Utilities category 18
 - Weather category 18
- App Store** app 8
- App Store distribution 36
- App Store Marketing and Advertising Guidelines for Developers* 58
- App Store Review Guidelines* 42
- app templates 68
 - Empty Application 69**
 - Master-Detail Application 69**
 - OpenGL Game 69**
 - Page-Based Application 69**
 - Single View Application 69**
 - Tabbed Application 69**
 - Utility Application 69, 132**
- Appcelerator 64
- app-driven approach xii, **2**
- Apple Developer Program roles
 - admin 35, 36
 - team member 35, 36
- Apple Inc. 19
- Apple Macintosh 19
- Apple Push Notification **12**
- ARC (Automatic Reference Counting) 25, **70**
- arithmetic operators 97
- array literal 165
- aspect ratio 78, 160
- Assets Library framework 20
- Assistant** editor (Xcode) **72**, 107, 111, 144, 146
- AssistiveTouch **11**
- atomic property 114
- attribute
 - in the UML 28
 - of a class 26
 - of an object 28
- Attributes** inspector
 - customize a button 141
- Attributes** inspector **78**
- Audio Toolbox framework 20
- Audio Unit framework 20
- auto layout 16, 24, **67**, 80, 81, 87, 99, 101, 102, 103, 144
 - change a constraint in the **Attributes** inspector 142
 - content hugging priority **158**
 - inspect constraints 144
 - priority 158
 - programmatically setting constraints 135
 - vertical bar (|) in a visual format string 160
 - visual format language 154
 - visual format string 159
- auto layout constraint 154
- auto layout constraints 142
- Automatic Reference Counting (ARC) 25, **70**, **114**
- auto-resizing mask 135
- Autoshrink** 104
- auto-synthesize a property 110
- AV Foundation framework 21
- B**
- Background** attribute of a **View** 140
- base class 113
- Base Internationalization 177**
- base internationalization 87
- becomeFirstResponder
 - method of a GUI component **96**, 116
- Before You Begin xvii
- behavior
 - of a class 26
- beta testing **39**
- block **97**, 154
 - ^ **97**
 - syntax 97
- Blocks Programming Topics* 98
- Bluetooth 10, 22
- Bonjour framework 22
- brand awareness 48
- branding apps
 - Amazon Mobile 48
 - Bank of America 48
 - Best Buy 48
 - Epicurious Recipe 49
 - ESPN ScoreCenter 49
 - ING Direct ATM Finder 49
 - NFL Mobile 49
 - Nike Training Club 49
 - NYTimes 49
 - Pocket Agent 49
 - Progressive Insurance 49
 - UPS Mobile 49
 - USA Today 49
 - Wells Fargo Mobile 49
- Breakpoint** navigator **71**
- __bridge keyword 167
- BSD Sockets framework 22
- Built-in Apps 8
 - App Store** 8
 - Calculator** 8
 - Calendar** 8
 - Camera** 8
 - Compass** 8
 - Contacts** 8
 - FaceTime** 8
 - Game Center** 8
 - iTunes** 8
 - Mail** 8
 - Maps** 8
 - Messages (SMS/MMS)** 8
 - Music** 8
 - Newsstand** 8
 - Notes** 8
 - Passbook** 8
 - Phone** 8

Built-in Apps (cont.)

- Photo Booth** 8
- Photos** 8
- Reminders** 8
- Safari** 8
- Settings** 8
- Stocks** 8
- Videos** 8
- Voice Memos** 8
- Weather** 8
- bundle ID search string **41**
- bundle identifier 41, 69
- bundle seed ID **41**
- `buttonWithType:` class method of class `UIButton` **153**

C

- C function 97
- C# xi
- C++ xi
- Calculator** app 8
- Calendar** app 8
- camera 6
- Camera** app 8, 14
- case-insensitive comparison 154
- Catalog Reports 50
- category
 - unnamed 148
- CFNetwork framework 22
- `CFURLCreateStringByAddingPercentEscapes` function **166**
- `CGSize` class **161**
- `CGSizeMake` **161**
- characteristics of great apps 42
- check-in 59
- class **27**
 - header **96**
 - implementation file **96**
 - instance variable **28**
- `Class` class **153**
- class extension **114**, 148
- class hierarchy 113
- class library **19**
- class method 119
- class prefix 70, 99, 138
- `class:` class method of class `NSObject` **153**
- Classes
 - `CGSize` **161**

Classes (cont.)

- `Class` **153**
- `NSArray` **134**
- `NSData` 134
- `NSDate` 134
- `NSDecimalNumber` **96**
- `NSDictionary` **134**
- `NSLayoutConstraint` **154**
- `NSMutableArray` **134**
- `NSMutableDictionary` **134**
- `NSNotificationCenter` **134**
- `NSNumber` 121, 134
- `NSNumberFormatter` **98**, 121
- `NSObject` 113
- `NSString` 134
- `NSUbiquitousKeyValueStore` **134**
- `NSURL` **167**
- `NSUserDefaults` **134**
- `UIActivityIndicatorViewController` **136**
- `UIAlertView` **136**
- `UIButton` **133**
- `UIImageView` **66**, 77
- `UILabel` **66**, **95**
- `UIScreen` **158**
- `UIScrollView` **133**
- `UISlider` **95**
- `UITextField` **95**
- `UIView` **133**
- `UINavigationController` **95**, **115**
- `UIWebView` 127, **133**
- Clock** app 8
- closure 97
- Cocoa Auto Layout Guide 81
- Cocoa Touch xi, xii, 2, **19**, 66, 73, 77
- Cocoa Touch frameworks **19**
 - Accelerate 20
 - Address Book 21
 - Assets Library 20
 - Audio Toolbox 20
 - Audio Unit 20
 - AV Foundation 21
 - Bonjour 22
 - BSD Sockets 22

Cocoa Touch frameworks (cont.)

- CFNetwork 22
- Core Animation 20
- Core Audio 21
- Core Data 22
- Core Foundation 22
- Core Graphics 21
- Core Image 20
- Core Location 22
- Core Media 21
- Core Motion 20
- Core Telephony 22
- Core Text 20
- Core Video 21
- Event Kit 22
- Event Kit UI 22
- External Accessory 22
- Foundation 22
- Game Kit 20
- iAd 21
- Image IO 22
- Map Kit 20
- Media Library 21
- Media Player 21
- Message UI 20
- Mobile Core Services 22
- OpenAL 21
- OpenGL ES 21
- Quartz 2D 20
- Quartz Core 21
- Quick Look 21
- Security 23
- SQLite 22
- Store Kit 22
- System 23
- System Configuration 22
- UIKit 20
- WebKit 20
- Cocoa Touch frameworks
 - Address Book UI 20
- code examples xvii
- code highlighting 2
- code license xi
- code signing **37**
- Code Snippet** library **76**
- code walkthrough 2
- Collection views 16
- Collections
 - `NSArray` **134**
 - `NSDictionary` **134**

- Collections (cont.)
 - NSMutableArray **134**
 - NSMutableDictionary **134**
 - Collections Programming Topics* 134
 - colon (:) in a method name 116, 119
 - Colors dialog 140
 - company identifier **69, 99, 137**
 - compare
 - case insensitive 154
 - compass **9**
 - Compass app 8
 - compilation 95
 - compiler directive 95
 - component 26
 - conform to (implement) a protocol 136
 - Connections inspector **78**
 - constant **148**
 - constraintsWithVisualFormat:options:metrics:views: method of class NSLayoutConstraint **154**
 - constraintWithItem:attribute:relatedBy: toItem:attribute: **158**
 - consumables 56
 - Contacts app 8
 - content hugging priority **158**
 - contentMode property of a UIImageView 160
 - context-sensitive help 72
 - contract information 46
 - Contracts, Tax & Banking Information 50
 - controller (in MVC architecture) 29
 - copy and paste 11
 - Core Animation framework 20
 - Core Audio framework 21
 - Core Data framework 22
 - Core Foundation 22
 - Core Foundation framework 22, 166
 - Core Graphics 21
 - Core Graphics framework 21
 - Core Image framework 20
 - Core Location 22
 - Core Location framework 22
 - Core Media framework 21
 - Core Motion framework 12, 20
 - Core Telephony framework 22
 - Core Text framework 20
 - Core Video framework 21
 - create an action in Interface Builder 111
 - create an outlet in Interface Builder 108
 - cross-platform, mobile-development tools 64
 - currency format 108
 - currency formatting 94
 - customize a button 141
 - customizing auto layout constraints 142
 - cut text 11
- D**
- Debug area (Xcode) 70, 72
 - Debug navigator **71**
 - debugger 25
 - decimalNumberByAdding: method of class NSDecimalNumber **121**
 - decimalNumberByDividingBy: method of class NSDecimalNumber **120, 121**
 - decimalNumberByMultiplyingBy: method of class NSDecimalNumber **121**
 - decimalNumberWithMantissa:exponent:isNegative method of class NSDecimalNumber **120**
 - decimalNumberWithString: method of class NSDecimalNumber **119**
 - default Objective-C compiler 110
 - Default.png launch image 100
 - Default@2x.png launch image 100
 - Default-568h@2x.png launch image 100
 - defaultCenter method of class NSNotificationCenter **149**
 - defaults system 133
 - defaultStore method of class NSUbiquitousKeyValueStore **148**
 - Deitel® *Buzz Online Newsletter* xiv, xviii, 179
 - Deitel® Training (www.deitel.com/training) 179
 - delegate 146
 - delegate protocol 146
 - derived class 113
 - design pattern 28, 137
 - design process **28**
 - destinationViewController method of class UIStoryboardSegue **168**
 - Developer Forums 50
 - Development Certificate **35, 36, 37**
 - development team **35**
 - device family 70, 99, 138
 - device unique identifier 39
 - didReceiveMemoryWarning method of class UIViewController **122**
 - digital certificate **37**
 - disabilities 85
 - dismissViewControllerAnimated:completion: method of class UIViewController **169**
 - dispatch_once function from the Grand Central Dispatch library **98, 118**
 - dispatch_once_t structure type **118**
 - distribution certificate **37**
 - division 97
 - Do Not Disturb phone setting 17
 - dock connector 6, 22
 - documentation
 - Accessibility Programming Guide for iOS* 32, 43
 - App Store Marketing and Advertising Guidelines for Developers* 58

documentation (cont.)

- App Store Review Guidelines* 42
- Blocks Programming Topics* 98
- Cocoa Fundamentals Guide* 31
- Coding Guidelines for Cocoa* 32
- Collections Programming Topics* 134
- Creating Universal Applications* 31
- Game Kit Programming Guide* 32
- iAd Developer's Starter Guide* 56
- iCloud Design Guide* 134
- iOS Application Programming Guide* 31
- iOS Human Interface Guidelines* 31, 34, **41**, 43, 44
- iOS Team Administration Guide* 37
- iTunes Connect Developer Guide* 46, 56
- Objective-C Runtime Programming Guide* 32
- Pass Kit Programming Guide* 31
- Preferences and Settings Programming Guide* 134
- Sample Code* 32
- SDK Compatibility Guide* 32
- Social Framework Reference* 32, 135
- Store Kit Framework Reference* 56
- Store Kit Programming Guide* 56
- The Objective-C Programming Language* 31
- Tools Workflow Guide for iOS* 46
- What's New in Xcode* 31
- Xcode 4 User Guide* 31

dot (.) notation for property access 115

dot notation 152

double tap 26

double-tap gesture 7

drag 26

drag gesture 7

drive sales 48

duplicate existing GUI components 95

dynamic binding 153

dynamically typed **116**

E

earnings 48

Editing Changed event for a **Text Field** 111, 116

Editor area (Xcode) 70, 71

empty string (@"") 121

encapsulation **28**

@end keyword 109, 111, **113**, 115

entitlement **173**

equality 120

equality operator (==) 120

equality operators 120

event handler 111

Event Kit framework 12, 22

Event Kit UI framework 22

event-handling method 96

Events

- Editing Changed** event for a **Text Field** 111, 116
- Value Changed** event for a **Slider** 111, 116

events **19**

Examples xviii

explicit App ID **41**, 173

External Accessory framework 22

Eyes Free 16

F

face detection 10

Facebook 15, **59**, 129

- Deitel page 59, 179
- integration 17
- sharing sheet 129

FaceTime 8, 12, 16

facial recognition 20

factory settings **8**

Favorite Twitter Searches app xii

fee-based app 18

file in the **Project** navigator 73

File inspector **72**

File Template library **76**

Financial Reports 50

financial transaction 55

Find My iPhone 8

Finder window **31**

Fix-it 25

flick 26

flick gesture 7

flipside view 127, **132**, 138

Form Factor button 99, 144

Foundation 22

Foundation Framework **119**, 134

Foundation framework 22

Frameworks

- Core Foundation 166
- Core Motion 12
- Event Kit 12
- Foundation **119**
- Store Kit 55
- UIKit **77**, **95**, 113

free app 18, 47

Free Applications contract **46**

freemium app monetization model 48

function, C 97

G

Game Center 15, 37, 40, 41

Game Center app 8, 13

Game Kit framework 15, 20

- local-player authentication 15
- matchmaking 15
- player display name 15
- player timeout 15

Game Kit Programming Guide 32

games 43

generic pointer **116**

gesture **6**, 26

- double tap 7
- drag 7
- flick 7
- pinch 7
- swipe 7
- tap 7
- touch and hold 7

- Google Maps **23**
- GPS sensor 10
- Grand Central Dispatch (GCD)
 - 12, **98**, 118
 - `dispatch_once` function **98**, 118
 - `dispatch_once_t` structure type **118**
- Graphical User Interface (GUI) **19**
- greater than (>) 120
- greater than or equal to (>=) 120
- group in the **Project** navigator 73
- GUI (Graphical User Interface) **19**
- GUI component
 - programmatically create 134
- GUI Components
 - Image View** **66**, 133
 - Label** **66**, 101, 102, 133
 - naming convention 107
 - Round Rect Button** **133**
 - Scroll View** **133**
 - Slider** 93, 133
 - View** **133**
- GUI components
 - Web View** **133**
- guide lines 77
- gyroscope **9**, 12

- H**
- .h file name extension (header) 96
- hashtag 60
- HDR (High Dynamic Range)
 - Photos 12
- header **95**, 112, 146, 169
 - .h file name extension 96
 - of a class **96**
- Headers
 - <UIKit/UIKit.h> **95**
- headset jack 5
- hearing impaired **10**
- height constraint 155
- HIG (Human Interface Guidelines) 77
- High Dynamic Range (HDR)
 - Photos 12
- Home* button 6

- horizontal layout constraint 155
- horizontal space auto layout constraint 142
- Human Interface Guidelines (HIG) 77, 99

- I**
- i-NewsWire 61
- iAd 12, 16, 49, **56**
- iAd Certification and Submission Checklist* 56
- iAd Developer's Starter Guide* 56
- iAd for Developers **56**
- iAd framework 21
- iAd Network **56**
- IBAction 135, 139
- IBAction macro **96**, 116
- IBOutlet 139
- IBOutlet macro **95**
- ibtool 90
- iCloud **8**, **13**, 14, 37, 40, 41, **126**, 134
 - account 126
 - entitlements 173
 - iOS Simulator 131
 - key-value pair store **134**
 - notification 150
 - NSUbiquitousKeyValueStore **134**
 - NSUbiquitousKeyValueStoreDidChangeExternallyNotification **150**
 - provisioning profile 173, 174
 - sync data across devices 126
- iCloud Design Guide* 134
- iCloud Storage APIs **13**
- iCloud-enabled provisioning profile 131, 173
- icon **44**
- id generic pointer type **116**
- IDE (integrated development environment) xii, 25
- Identifier** attribute of a segue 142
- Identity** inspector **78**
- Image** attribute 78
- Image IO framework 22
- Image View** **66**, 77, 78, 133

- imageName: method of class UIImage 159
- implement (conform to) a protocol 136
- implementation file (.m file name extension) **96**, 112, **113**, 146, 169
- @implementation keyword **115**
- #import preprocessor directive **95**
- in-app advertising 47, **49**
- In-App Purchase 16, 34, 37, 40, 41, 47, 55, 56
- inequality 120
- information hiding **28**
- information icon 69
- inheritance **28**, 113, **115**
- inherits **113**, 146
- init of class NSMutableArray **150**
- init of class NSMutableDictionary **150**
- initWithActivityItems:applicationActivities: method of class UIViewController **165**
- initWithTitle:message:delegate:cancelButtonTitle:otherButtonTitles: method of class UIAlertView **163**
- in-plane switching (IPS) **9**
- inspecting the view hierarchy and auto layout constraints 144
- inspector 72, 78
 - Attributes** **78**
 - Connections** **78**
 - File** **72**
 - Identity** **78**
 - Quick Help** **72**
 - Size** **78**
- instance method 115
- instance of a class **27**
- instance variable **28**, 112, 146
- Instruments 25
- integrated development environment (IDE) xii, 25

interface 113
 Interface Builder 19, **25**, 66, **74**
 duplicate existing GUI components 95
 @interface keyword 109, **113**
 international App Stores 44
 internationalization **87**, **175**,
 176
 .strings file **176**
 string resources 176, 178
 Use Base Internationalization 87, 177
 Internet Public Relations
 InternetNewsBureau.com 61
 PRX Builder 61
 Internet public relations resources
 ClickPress 61
 i-Newswire 61
 Marketwire 61
 Mobility PR 61
 openPR 61
 PR Leap 61
 Press Release Writing 61
 PRLog 61
 PRWeb 61
 PRX Builder 61
 Internet tethering **10**
 iOS 11
 iOS 4.x 13
 iOS 6 110, 129
 Social Framework 135
 iOS app templates 68
 iOS defaults system 133
 NSUserDefaults **134**
 iOS Dev Center 58
 iOS Developer Enterprise Program **3**
 iOS Developer Library 81
 iOS Developer Program **2**, 26,
 34, 35
 iOS Developer University Program **3**
 iOS Distribution Certificate **36**
iOS Human Interface Guidelines
 34, **41**, 43, 44
 iOS Paid Applications contract
 46
 iOS Provisioning Portal 35, **39**

iOS SDK 4.x 25
 iOS SDK 6 xii
 iOS simulator 25, 34, 66, **67**, 83
iOS Team Administration Guide
 37
 iOS Team Provisioning Profile
 37
 iOS wildcard app ID **37**
 iPad 3
 iPad 2 4
iPad Deployment Info 74
 iPad Mini 4
 iPad, first generation 4
 iPad, The New 4
 iPhone 3G 3
 iPhone 3GS 3
 iPhone 4 3, 12
 iPhone 4S 3
 iPhone 5 99
 Form Factor button 99, 144
 resolution 135
iPhone for Programmers website
 www.deitel.com/books/
 iPhoneFP/ xi
 iPhone OS 11
 iPhone OS 2 11
 iPhone OS 3 11
 iPhone sales 3
 iPhone SDK 3.x xi
iPhone/iPod Deployment Info
 74
 iPod touch 2
 isKindOfClass: method of
 class NSObject **153**
Issue navigator **71**
 iTunes 6, 11, 18, 56
iTunes app 8
 iTunes Connect **34**, **50**
iTunes Connect Developer Guide
 46, 56
 iTunes Connect Modules 50
 iTunes Ping 13
 itunesconnect.apple.com
 50
J
 Java xi
 Jobs, Steve 19

K

keyboard 6
 how to display 96, 116
 keyboard shortcuts 73
 keychain **38**
 Keychain Access **38**
 Keywords 44
 __bridge 167
 @end 109, 111, **113**, 115
 @implementation **115**
 @interface 109, **113**
 @optional **136**
 @property **95**, 109, 114
 @protocol **136**
 @required **136**
 @synthesize **110**
 @selector 154
 self **115**
 super **115**

L

Label
 Alignment attribute 82
 Font attribute 82
 Lines attribute 82
 Text attribute 82
Label GUI Component **66**, 80,
 133
 lambda function 97
 landscape keyboard **9**, 11
 landscape orientation 74, 144
 language support 12
Large Text accessibility feature
 10
 launch image **45**, 46
 Default.png 100
 Default@2x.png 100
 Default-568h@2x.png
 100
 layout constraint
 horizontal 155
 vertical 155
 leaderboard 15
 length method of class NSString 121
 less than (<) 120
 less than or equal to (<=) 120
 Libraries
 Code Snippet 76
 File Template 76

- Libraries (cont.)
 - Media** 77
 - Object** 77
 - Library window** 77
 - literal
 - NSString 119
 - LLVM Compiler 25
 - LLVM compiler 113
 - loadURL method of class UIWebView 127
 - locale-specific currency string
 - 94, 121
 - locale-specific percentage string
 - 120, 121
 - localization 51, **175, 176**
 - .strings file **176**
 - string resources 176
 - localizedCaseInsensitiveCompare: method of class NSString **154**
 - localizedStringFromNumber:numberStyle: method of class NSNumberFormatter **98, 121**
 - local-player authentication 15
 - locate your iPhone 8
 - location simulation 25
 - lock the iPhone 5
 - Log navigator** 71
- M**
- .m file name extension (implementation file) 96
 - Mac xi
 - Mac OS X xi, 19
 - Macintosh 19
 - macros **95**
 - magnetic sensor **10**
 - magnetometer 12
 - Mail** app 8, 14
 - main thread 114
 - main view 69, **132, 138**
 - mainScreen class method of class UIScreen **158**
 - Manage Users 50
 - Manage Your Applications 50
 - Managing In-App Purchases* 56
 - Map Kit framework 15, 20
 - Maps** app 8, 15
 - market share 4
 - marketing xii
 - Marketwire 61
 - mashup **23**
 - Master-Detail Application** template **69**
 - Media** library 77
 - Media Library framework 21
 - Media Player framework 21
 - memory leaks 73
 - menu name xvii
 - message **96, 111, 116, 119**
 - Message UI framework 20
 - Messages (SMS/MMS)** app 8
 - method **27, 112, 146**
 - method call **27**
 - method definition 115
 - method name 119
 - micro blogging 59
 - microphone 6, **11**
 - Minimum Font Scale** 104
 - mobile advertising 47
 - mobile advertising networks 49, 62
 - AdMob 49, 62
 - AdWhirl 62
 - Flurry 62
 - Google AdSense for Mobile 62
 - Medialets 62
 - Millennial Media 62
 - Nexage 62
 - Smaato 62
 - Mobile Core Services framework 22
 - mobile platforms
 - Android 63
 - Mode** attribute 78
 - Mode** attribute of a UIButton 141
 - model (in MVC architecture) 29
 - Model-View-Controller (MVC) **29, 29, 137**
 - Modern Objective-C
 - array literal 165
 - monetary values 94
 - monetizing apps 34, 49, 62
 - motion data 12
 - Multi-Touch screen 6, 20
 - multiplication 97
 - multitasking 12
 - Music** app 8
 - mutableCopy method of class NSDictionary **134**
 - mutableCopy of class NSDictionary **150**
 - MVC (Model-View-Controller) **29**
- N**
- name 119
 - namespace 99
 - naming convention
 - GUI components 107
 - Navigator** area (Xcode) 70, 72
 - Navigators
 - Breakpoint** 71
 - Debug** 71
 - Issue** 71
 - Log** 71
 - Project** 70, 73
 - Search** 71
 - Symbol** 70
 - nested message 121
 - network activity indicator 172
 - networkActivityIndicatorVisible property of class UIApplication **172**
 - New App ID** button 41
 - Newsstand** app 8, 14
 - Newsstand Kit **14**
 - NeXT Inc. 19
 - NeXT Interface Builder 25
 - NeXTSTEP 19
 - NeXTSTEP operating system 19
 - Nib file 74
 - nib file 25
 - Nike + iPod** app 8
 - nil **114**
 - nil vs. NULL 167
 - nonatomic property attribute **115**
 - nonconsumables 56
 - Notes** app 8
 - Notification Center** 14
 - notifications 134
 - NSNotificationCenter **134**
 - register to receive 148

- NSArray class 134, **134**
 - objectAtIndex: method **152**
 - NSComparisonResult 154
 - NSData class 134
 - NSDate class 134
 - NSDecimalNumber class **96**, 121
 - decimalNumberByAdding: method **121**
 - decimalNumberByDividingBy: method **120**
 - decimalNumberByMultiplyingBy: method **121**
 - decimalNumberWithMantissa:exponent:isNegative method **120**
 - decimalNumberWithString: method **119**
 - NSDictionary class **134**, **150**
 - mutableCopy method **134**, **150**
 - objectForKey: method **152**
 - NSLayoutConstraint class **154**
 - constraintsWithVisualFormat:options:metrics:views: method **154**
 - constraintWithItem:attribute:relatedBy: toItem: attribute: multiplier: constant: **158**
 - NSLocalizedString macro **176**, 177
 - NSMutableArray class **134**, 148
 - alloc method **150**
 - init method **150**
 - sortUsingComparator: method 154
 - NSMutableArray protocol
 - removeObjectAtIndex: method **164**
 - NSMutableDictionary class **134**, 148
 - alloc method **150**
 - init method **150**
 - NSMutableDictionary protocol
 - removeObjectForKey: method **164**
 - NSNotification class 150
 - userInfo method **152**
 - NSNotificationCenter class **134**
 - addObserver:selector:name:object: method **149**
 - defaultCenter method **149**
 - NSNumber class 121, 134
 - NSNumberFormatter class **98**, 121
 - localizedStringFromNumber:numberStyle: method **98**, 121
 - NSNumberFormatterCurrencyStyle constant 121
 - NSNumberFormatterPercentStyle constant 121
 - NSNumberFormatterStyle enum 121
 - NSObject class 113
 - class: method **153**
 - isKindOfClass: method **153**
 - NSObject protocol
 - respondsToSelector: method **172**
 - NSString class 134
 - @*"string"* literal **119**
 - length method 121
 - localizedCaseInsensitiveCompare: method **154**
 - stringByAddingPercentEscapesUsingEncoding: method 166
 - stringByReplacingPercentEscapesUsingEncoding: method **162**
 - NSString literal 119, 121
 - NSUbiquitousKeyValueStore class
 - synchronize method **150**
 - NSUbiquitousKeyValueStore class **134**, 152
 - defaultStore method **148**
 - NSUbiquitousKeyValueStoreChangeReasonKey 152
 - NSUbiquitousKeyValueStoreDidChangeExternallyNotification **150**
 - NSURL class **167**
 - NSUserDefaults class **134**, 148
 - standardUserDefaults method **150**
 - NULL vs. nil 167
 - numeric keypad 93, 96
 - display 115
- ## O
- object 26
 - object (or instance) 28
 - Object** library **77**
 - object messaging 152
 - object-oriented analysis and design (OOAD) **28**
 - object-oriented language **28**
 - object-oriented programming (OOP) **19**, **28**
 - objectAtIndex: method of class NSArray **152**
 - objectForKey: method of class NSDictionary **152**
 - Objective-C xi, xii, 2, **19**
 - @end 109, 111, **113**, 115
 - @implementation **115**
 - @interface 109, **113**
 - @property **95**, 109, 114
 - array literal 165
 - colon (:) in a method name 116, 119
 - default compiler 110
 - method name 119
 - NSString literal 121
 - parameter 116, 119
 - parameter name 116
 - parameter type 116
 - property 95
 - self **115**
 - subclass 113
 - superclass 113

- Objective-C code xvii
- Objective-C command xvii
- Objects** library 140
- Observer design pattern 137, 146
- observer object 137
- on-screen component xvii
- OOAD (object-oriented analysis and design) 28
- OOP (object-oriented programming) **28**
- OpenAL API 21
- OpenGL ES 21, 25
- OpenGL Game** template **69**
- openPR 61
- OpenStep 19
- Operators
 - (subtraction) 97
 - != (inequality) 120
 - * (multiplication) 97
 - / (division) 97
 - % (remainder) 97
 - + (addition) 97
 - < (less than) 120
 - <= (less than or equal to) 120
 - == (equality) 120
 - > (greater than) 120
 - >= (greater than or equal to) 120
 - & (address of) 119
- @optional keyword **136**
- orientation change **26**
- OS X 19
- outlet **96**
 - create 108
- outlet property name 107
- P**
- package 99
- Page-Based Application** template **69**
- parameter 116, 119
- parameter name 116
- parameter type 116
- Parental Controls 51
- Pass Kit 15
- Pass Kit Programming Guide* 31
- Passbook** app 8, 15, 17
- passes **15**
- paste text 11
- payment 56
- PC free device activation and iOS updates 14
- peer-to-peer games 12
- performance issues 73
- performSegueWithIdentifier:sender: method of class UINavigationController **167**
- Phone** app 8
- PhoneGap 64
- Photo Booth** app 8
- photo sharing 59
- photo sharing in iOS 6 17
- photos 6
- Photos** app 8
- pinch 26
- pinch gesture 7
- Placeholder** attribute of a **TextField** 141
- pointer
 - generic **116**
- pointer (*) declaration 114
- pointer to the sender GUI component of an event 116
- points (measurement) 135
- popover **69**
- portrait orientation 74
- power the iPhone 5
- PR Leap 61
- Preferences and Settings Programming Guide* 134
- prepareForSegue:sender: method of class UINavigationController **168**
- preprocessor **95**
- preprocessor directive **95**
- preprocessor macro 95
- presentViewController:animated:completion: method of class UINavigationController **165**
- press release writing 61
- price 18, 48
- price tier 51
- Pricing Matrix** 51
- pricing your app 47
- primary screenshot 46
- priority (auto layout) 158
- privacy **8**
- private interface 96, 114
- PRLog 61
- programmatically create GUI components 134
- programmatically select a component 96, 116
- programmatically setting auto-layout constraints 135
- programming languages
 - Objective-C 19
- project **68**
- project name **69**
- Project** navigator **70**, **73**
- Project Structure** group **73**
- property 112, 146
 - atomic 114
 - auto-synthesize 110
- property (Objective-C) 95
- property attribute
 - nonatomic **115**
 - strong **114**
 - weak **114**
- property attributes 114
- property declaration 109, 114
- @property keyword **95**, 109
- property name 114
- property of a class 114
- property type 114
- protocol **136**, 146
 - conform to 136
 - delegate 146
 - similar to an interface in other programming languages 136
- @protocol keyword **136**
- Protocols
 - UIWebViewDelegate **170**
- provision **37**
- Provisioning Portal, iOS 35
- Provisioning Profile **36**, **37**
- provisioning profile
 - iCloud enabled 173, 174
 - install 174
- proximity sensor **10**
- public interface 96
- public relations 61
- purchasing interface 56
- Push Notification **12**
- push notifications 37, 40, 41

Q

Quartz 2D 20
 Quartz Core framework 21
Quick Help inspector **72**
 Quick Look framework 21

R

rating apps 51
 receiver 116
 recent projects 68
 record for your app in iTunes
 Connect 52
 reference count 114
 register to receive notifications
 148
 registered Apple developer 2
 relational operators 120
 release date 50
 remainder operator, % 97
Reminders app 8, 14, 16
 Remote Wipe **8**
 removeAllObjects method of
 class NSMutableArray **156**
 removeFromSuperview meth-
 od of class UIView **155**
 removeObjectAtIndex:
 method of class NSMutableArray-
 Array **164**
 removeObjectForKey: meth-
 od of class NSMutableDictionary
 164
 @required keyword **136**
 requirements **28**
 resignFirstResponder
 method of class UITextField
 Field **165**
 respondsToSelector: meth-
 od of protocol NSObject **172**
 Retina display **9**, 135
 reusable software components 26
 reuse 27
 review and recommendation
 sites 60
 Rhapsody 19
 RhoMobile 64
 Ring/Silent switch 6
 Round Rect Button 126
Round Rect Button 133
 routing app **15**
Run button (Xcode) **31**

S

Safari app 8, 14
 Sales/Trend Reports 50
 scene **76**
Scheme selector (Xcode) **31**
 screen resolution 135
 screenshot 44
 scroll 6
Scroll View 133
 adding to a scene 142
 SDK beta xvii
 SDK documentation xvii
Search navigator **71**
 search operators (Twitter) 125
 Security framework 23
 segue 138, 140
 add in Interface Builder 142
 Identifier attribute 142
 select a component program-
 matically 96, 116
 selector **150**
 @selector keyword **150**, 154
 self keyword **115**
 to call another method in the
 same class 121
 send a message to an object 27
 sender of an event **116**
 sensor 9
 accelerometer **9**
 Ambient light sensor **10**
 compass **9**
 GPS 10
 gyroscope **9**
 magnetic sensor **10**
 proximity sensor **10**
 setContentHuggingPriori-
 ty:forAxis: method of
 class UIView 158
 setContentSize: method of
 class UIScrollView **161**
Settings app 8
 setTitle:forState: method
 of class UIButton **154**
 setTranslatesAutoresiz-
 ingMaskIntoCon-
 straints: method of class
 UIView **135**
 sharedApplication class
 method of class UIApplica-
 tion 158

sharing options 129
 sharing sheet for Facebook 129
 sheet 69
 SIM card tray 6
 simulator 66, **67**, 83
 Sina Weibo 15, 129
Single View Application tem-
 plate **69**
 Siri 14
 Eyes Free 16
Size inspector **78**
 Sleep/Awake button 5
Slider 93, 133
 Value Changed event 111,
 116
 SMS 129
 Social Framework **15**, 129, 135
Social Framework Reference 32,
 135
 social media sites 58, 59
 social networking 59
 sortByUsingComparator:
 method of class NSMutableArray-
 Array 154
 source code 2
 source-code listing 2
 speaker 6
Spot-On Game app 29
 SQLite 22
Standard editor (Xcode) **71**
 standardUserDefaults
 method of class NSUserDe-
 faults **150**
 State Preservation 16
 statically typed object **116**
 statusBarOrientation prop-
 erty of class UIApplication
 158
 StepStone 19
Stocks app 8
 stopLoading method of class
 UIWebView **172**
 Store Kit **14**
 Store Kit framework 22, 55, 56
Store Kit Framework Reference 56
Store Kit Programming Guide 56
 storyboard **70**
 segue 138
 storyboarding 24, **74**
 string resources 176

- string resources and internationalization 178
 - stringByAddingPercentEscapesUsingEncoding:
 - method of class NSString 166
 - stringByReplacingPercentEscapesUsingEncoding:
 - method of class NSString **162**
 - .strings file **176**
 - strong property attribute **114**
 - subclass 113
 - subject object 137
 - subscription 56
 - subview **155**
 - subviews property of class UIView **155**
 - super keyword **115**
 - super keyword **115**
 - superclass 113
 - superview **155**
 - Supported Device Orientations** 74
 - swipe 6, 26
 - swipe gesture 7
 - Symbol** navigator **70**
 - sync **18**
 - synchronize method of class NSUbiquitousKeyValueStore **150**
 - syntax coloring 2
 - @synthesize keyword **110**
 - System Configuration framework 22
 - System framework 23
- T**
- tab bar 69
 - Tabbed Application** template **69**
 - tap 26
 - tap gesture 7
 - Team Agent 46
 - team member 35, 36
 - template **68**
 - testing xvii
 - Text Field** 101, 102
 - Editing Changed** event 111, 116
 - text property of a UILabel **121**
 - text property of a UITextField **119**
 - Tip Calculator** app xii
 - toll-free bridging 167
 - Tools Workflow Guide for iOS* 46
 - touch and hold 26
 - touch and hold gesture 7
 - tweet 59
 - Twitter 15, **59**, 129
 - @deitel 60, 179
 - hashtag 60
 - tweet 59
 - Twitter account API **14**
 - Twitter framework 15
 - Twitter integration 14
 - Twitter search 125
 - operators 127
- U**
- UIAccessibility protocol **87**, 175
 - accessibilityHint property **175**
 - accessibilityLabel property **175**
 - UIActivityIndicatorViewController class **136**, 165
 - initWithActivityIndicatorItems:applicationActivities: method **165**
 - presentViewController:animated:completion: method **165**
 - UIAlertView class **136**
 - initWithTitle:message:delegate:cancelButtonTitle:otherButtonTitles: method **163**
 - UIAlertViewDelegate protocol
 - alertView:clickedButtonAtIndex: method **163**
 - UIApplication class 158, 172
 - networkActivityIndicatorVisible property **172**
 - sharedApplication class
 - method 158
 - statusBarOrientation property 158
 - UIApplicationMain 158, 172
 - UIButton class **133**, 138
 - buttonWithType: class method **153**
 - Mode** attribute 141
 - setTitle:forState: method **154**
 - UIButtonTypeCustom 159
 - UIButtonTypeCustom 159
 - UIControlEvents enum 154
 - UIControlEventTouchUpInside event **159**
 - UIControlTouchUpInside 154
 - UIImage class
 - imageName: method 159
 - UIImageView class **66**, 77
 - contentMode property 160
 - UIKit Artwork Extractor 138
 - UIKit framework 20, **77**, **95**, 95, 113
 - UILabel **95**
 - UISlider **95**
 - UITextField **95**
 - UIViewController **95**
 - <UIKit/UIKit.h> header **95**
 - UILabel class **66**, **95**
 - text property **121**
 - UIScreen class **158**
 - mainScreen class method **158**
 - UIScrollView class **133**
 - UISlider class **95**
 - value property **120**
 - UINavigationController class
 - destinationViewController method **168**
 - UITextField class **95**
 - resignFirstResponder method **165**
 - text property **119**

UIView class **133**

- addConstraint: method **135, 158**
- addConstraints: method **135, 154, 159**
- addSubview method **159**
- removeFromSuperview method **155**
- setContentHuggingPriority:forAxis: method **158**
- setTranslatesAutoresizingMaskIntoConstraints: method **135**
- subviews property **155**
- viewDidDisappear: method **169**

UIViewController class **95, 115**

- didReceiveMemoryWarning method **122**
- dismissViewControllerAnimated:completion: method **169**
- performSegueWithIdentifier:sender: method **167**
- prepareForSegue:sender: method **168**

UIWebView class **127, 133**

- loadURL method **127**
- stopLoading method **172**

UIWebViewDelegate protocol **170**

- webView:didFailLoadWithError: method **171**
- webView:shouldStartLoadWithRequest:navigationType: method **170**
- webViewDidFinishLoad: method **170**
- webViewDidStartLoad: method **170**

- unique identifier for a device **39**
- unit tests **73**
- universal app **43, 66, 67, 70, 74**
- universal apps **2**
- unlock the iPhone **5**
- unnamed category **148**

upload finished apps xvii

URL encoding **162**

CFURLCreateStringByAddingPercentEscapes function **166**

Use Base Internationalization 87, 177

user interface events **111**

userInfo method of class NSNotification **152**

utilities **44**

Utilities area (Xcode) **70, 72**

Utility Application template **69, 132, 136**

Utility Application Xcode template **137, 138**

V

Value Changed event for a Slider **111, 116**

value property of a UISlider **120**

version editor **25**

Version editor (Xcode) **72**

vertical bar (|) in a visual format string **160**

vertical layout constraint **155**

vertical space auto layout constraint **142**

video **6, 10**

video sharing **59**

Videos app **8**

View 133

view (in MVC) **29**

view controller **95**

view hierarchy **144**

inspect **144**

viewDidDisappear: method of class UIView **169**

viewDidLoad message **96**

viral marketing **58, 59**

viral video **60**

virtual goods **54, 55**

vision impaired **10**

visual format language **154**

visual format string **159**

visual layout string **135**

Voice Memos app **8**

VoiceOver **10, 85**

enable/disable **86**

Volume buttons **6**

Volume Purchasing Program **57**

W

weak property attribute **114**

Weather app **8**

web services **23**

Web View **133, 145**

WebKit framework **20**

webView:didFailLoadWithError: method of protocol UIWebViewDelegate **171**

webView:shouldStartLoadWithRequest:navigationType: method of protocol UIWebViewDelegate **170**

webViewDidFinishLoad: method of protocol UIWebViewDelegate **170**

webViewDidStartLoad: method of protocol UIWebViewDelegate **170**

Weibo **129**

Welcome app xii, **26**

Welcome to Xcode window **67**

White on Black accessibility feature **10**

WidgetPad **64**

width constraint **155**

word-of-mouth marketing **59**

workspace window **70**

Wozniak, Steve **19**

www.apple.com/ios/features.html **13**

www.apple.com/iphone/apps-for-iphone/ **18**

www.bis.doc.gov/licensing/

exportingbasics.htm **52**

www.craigslist.org **23**

www.deitel.com xviii

www.deitel.com/books/DiveIntoiOS xi

www.deitel.com/newsletter/subscribe.html xviii

www.deitel.com/training **179**

www.deitel.com/training **179**

www.deitel.com/training **179**

www.housingmaps.com **23**

X

Xcode xi, 2, 19, 24, 25, **31**, 46
4.5 110
Assistant editor **72**, 107,
111, 144, 146
Debug area 70, 72
Editor area 70, 71
Navigator area 70, 72
Standard editor **71**
Utilities area 70, 72
Utility Application template
137, 138
Version editor **72**
Xcode 4 User Guide 67
Xcode Assistant 24

Xcode Groups
Project Structure **73**
Xcode IDE 66
Xcode navigators
Breakpoint **71**
Debug **71**
Issue **71**
Log **71**
Project **70**, 73
Search **71**
Symbol **70**
Xcode toolbar 72
Xcode Windows
Library 77
Welcome to Xcode **67**

Xcode .app 24
Xerox PARC (Palo Alto Re-
search Center) 19

Y

Yellow Box API 19
YouTube 10

Z

zoom 6
Zoom accessibility feature 10