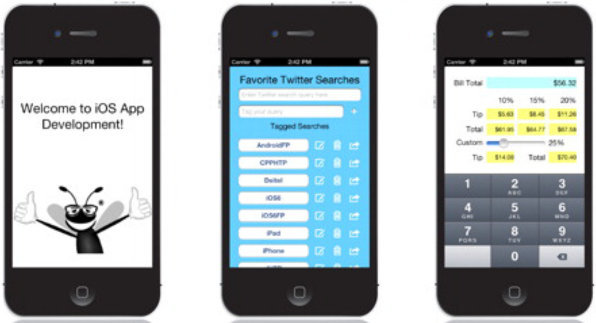
DEITEL® DEVELOPER SERIES

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

ISBN-13: 978-0-13336569-6
ISBN-10: 0-13-336569-7

Dive Into® iOS 6

An App-Driven Approach

Deitel® Developer Series

Paul Deitel
Harvey Deitel
Abbey Deitel
Deitel & Associates, Inc.

PRENTICE
HALL

Upper Saddle River, NJ • Boston • Indianapolis • San Francisco
New York • Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Trademarks

In memory of Steve Jobs

Paul, Harvey and Abbey Deitel

Contents

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

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`) 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 Science 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 `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!

1

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.

Outline

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/`.

- *First-generation iPad*: The iPad, launched in April 2010, sold 3 million units in its first 80 days of availability[8] 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 display[12] 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 1 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/?kc=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/`.

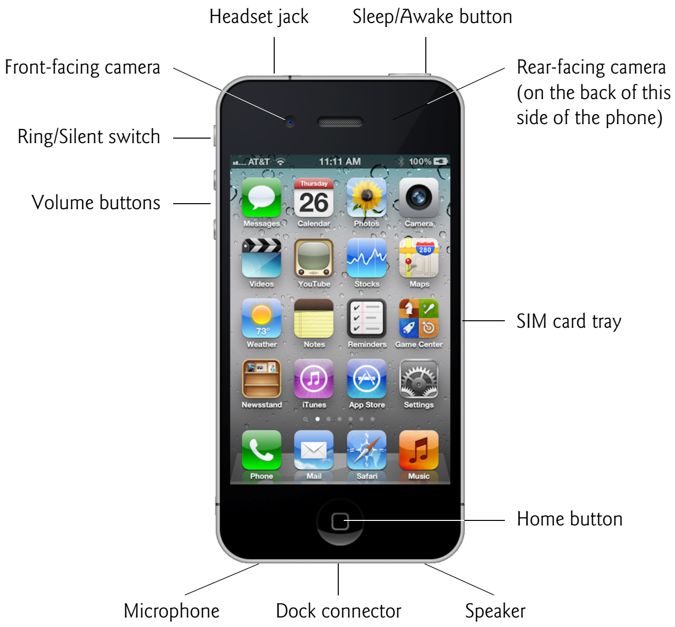


Fig. 1.2 | iPhone hardware.

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).

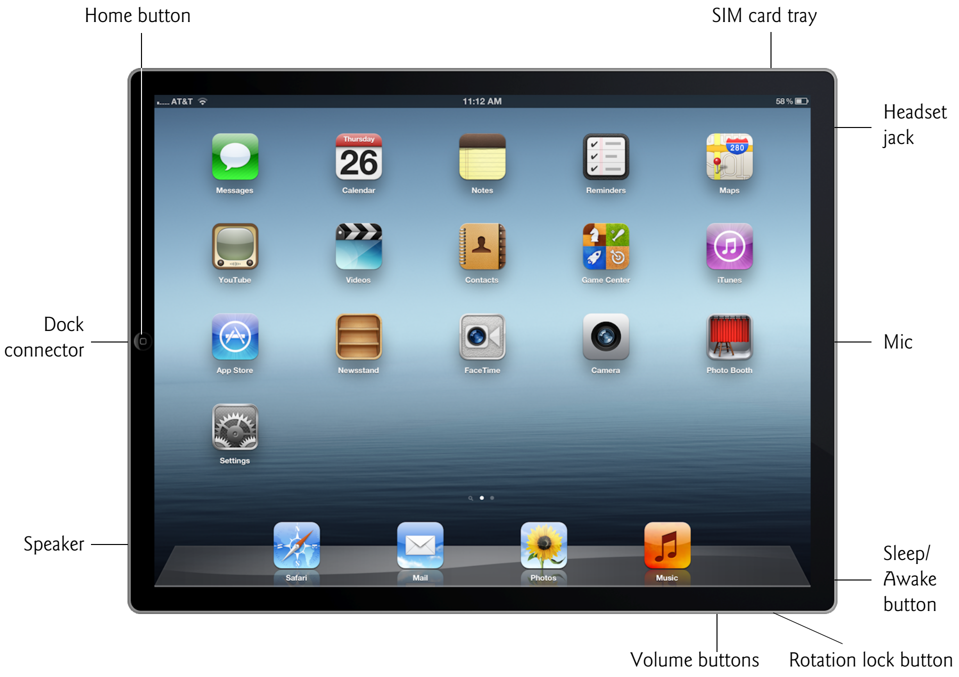


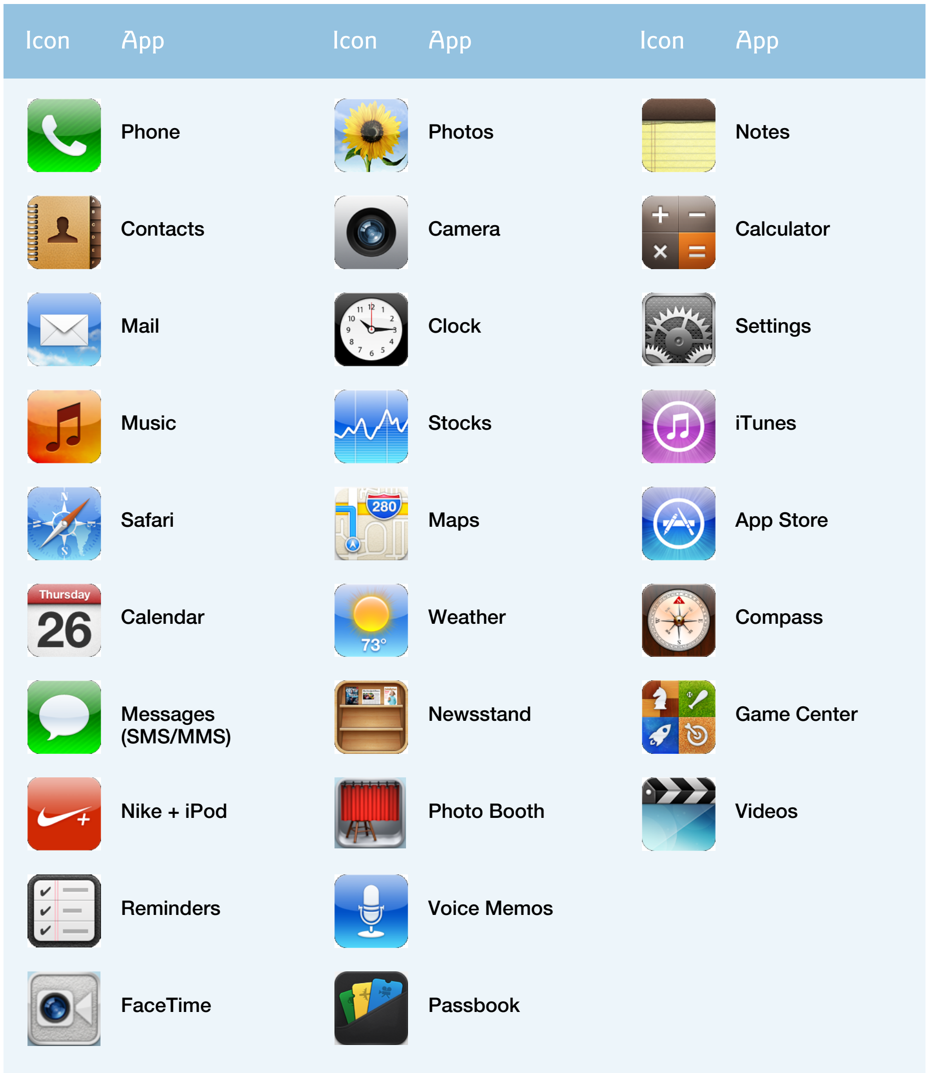
Fig. 1.3 | iPad hardware—all but the Home button are on the side or back of the device.

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.



Icon	App	Icon	App	Icon	App
	Phone		Photos		Notes
	Contacts		Camera		Calculator
	Mail		Clock		Settings
	Music		Stocks		iTunes
	Safari		Maps		App Store
	Calendar		Weather		Compass
	Messages (SMS/MMS)		Newsstand		Game Center
	Nike + iPod		Photo Booth		Videos
	Reminders		Voice Memos		
	FaceTime		Passbook		

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.

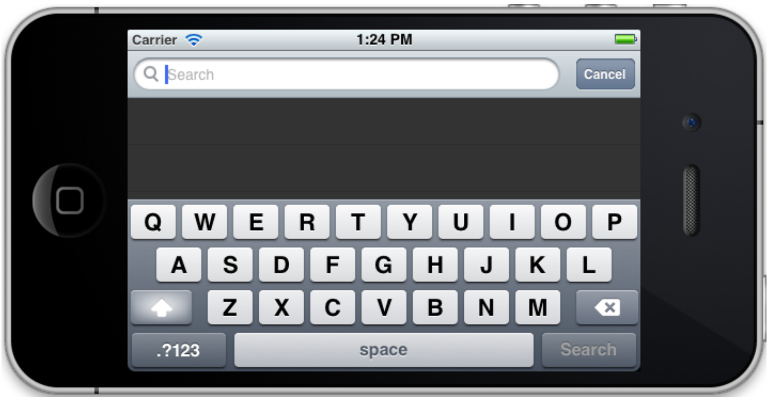


Fig. 1.6 | Landscape keyboard.

- 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/releasenotes/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 *video calls* 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 *push* (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	Game Center and the Game Kit framework include several new and updated features: • *Challenges*, 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 *local-player authentication*. • *Player timeout support*. 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 *matchmaking*, allowing you to match players to other players programmatically. Players can then send and receive match invitations. • Support for *players' display names*. • Determining which player has the *best connection* to Game Center.
Social framework	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.
Maps	Additions and enhancements to the **Maps** app and Map Kit framework include: • Improved ability to *launch the* **Maps** *app from within your app* 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	**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. *Your* web service creates the pass and delivers it to the user either through your app, e-mail or Safari.

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	Siri—which is available on recent iPhones and iPads—can now: • 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 *Eyes Free* 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	**FaceTime** is now available over cellular and can be used to make video calls across iPhone, iPad and Mac devices. **iMessage** works similarly.

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

Feature	Description
Facebook integration	Facebook has been integrated into iOS 6, allowing users to perform the following tasks: • 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	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.
Maps	New **Maps** features include: • 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	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.
Phone	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.

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 `CIFaceFeature`, 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.creativelabs.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 `NSObject` (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 `.xib` extension, but earlier versions used `.nib`—short for NeXT Interface Builder. For this reason, Interface Builder `.xib` 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 `.nib` (or `.xib`) 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 *Option* 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 patterns[22] 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.

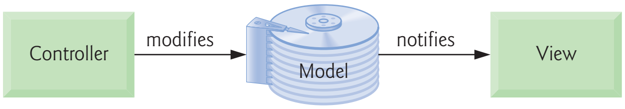


Fig. 1.20 | Model-View-Controller Architecture.

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.

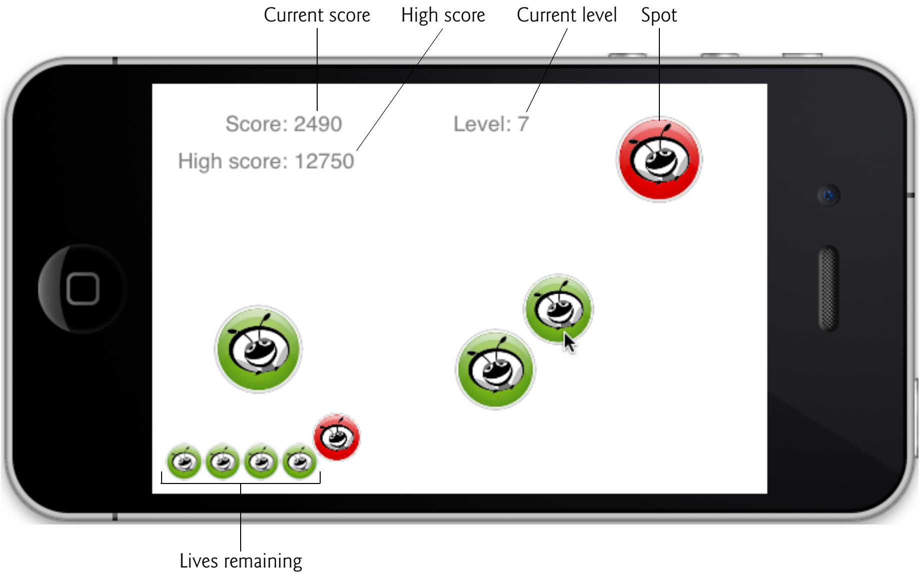


Fig. 1.21 | **SpotOn Game** app during game play.

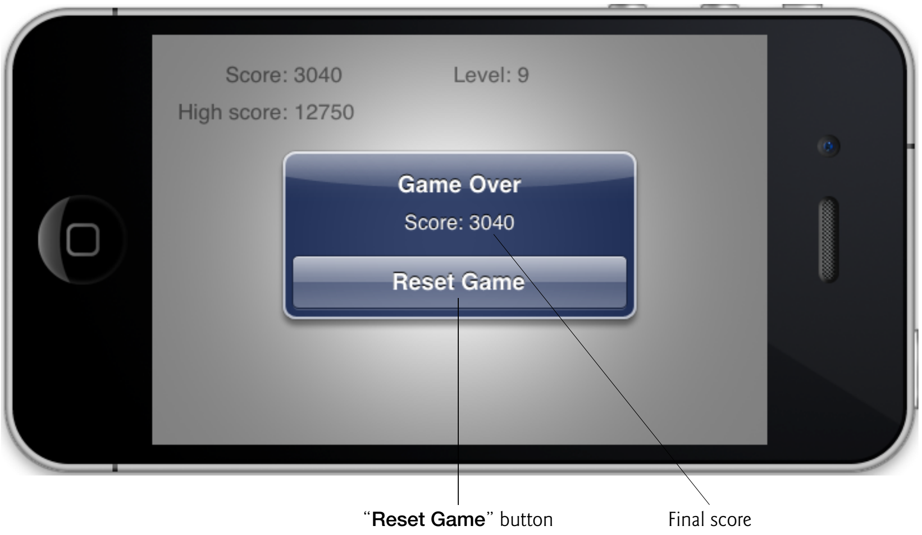


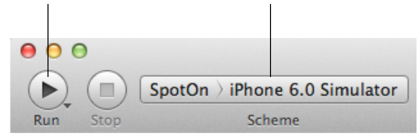
Fig. 1.22 | **Game Over** alert.

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!



Run button

Scheme selector

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

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

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

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