

QUICK & EASY

iPhone PROGRAMMING

A detailed illustration of a man in a top hat and coat, holding a sign that says "MEAP". The man has a beard and is looking thoughtfully to the side with his hand to his chin. He is wearing a dark, textured coat over a blue turtleneck and a blue top hat. The sign is red with white text and is held in front of him. The background is plain white.

MEAP

Bintu Harwani
Amos Bannister

 MANNING



MEAP Edition
Manning Early Access Program
Quick & Easy iPhone Programming version 8

Copyright 2012 Manning Publications

For more information on this and other Manning titles go to
www.manning.com

Table of Contents

Part 1: Introducing iPhone programming

1. Introduction
2. Hello world the iPhone way
3. User interaction under the hood

Part 2: Views and storyboards

4. Working with multiple views
5. The tab control
6. Using tables
7. Dynamic tables and search
8. Other navigation and toolbars

Part 3: Advanced controls

9. Progress and status indicators
10. More user interaction
11. Web and scroll views
12. Storing data

1

Introduction

This chapter covers

- The iPhone's characteristics
- iPhone development fundamentals
- The basics of Objective-C
- Creating programs in Xcode

The iPhone is a revolutionary device from Apple. It has become one of the most popular smart phones of today. It is simple to operate, has a multi-touch screen, can be used for web surfing, e-mails, watching videos, listening to music, and even event scheduling. With the release of the iPhone SDK from Apple, developers around the world are able to develop iPhone applications. This book will guide you in the basics of developing your own applications. It's intended as a launch pad to get you started, rather than an exhaustive reference of everything you can do with the SDK.

You will learn about the different User Interface (UI) controls such as the Label, TextField, Round Rect Button, and more, provided in the SDK and you will see working examples of each of them via individual running applications. You can dive right into any chapter and create a working application from start to finish. The idea is to make you aware of the practical implementation of different UI controls so that you know how to use them in your own applications.

We are going to start our journey by learning about the iPhone's characteristics and its development environment.

1.1 The iPhone and iPhone Programming

Apple's iPod-like mobile phone is an Internet and multimedia enabled smart phone. It combines the main functions of several electronic gadgets like the iPod, a tablet PC, a digital camera and a cell phone in one small hand-held device. It's considered the most revolutionary device in mobile technology and was so highly anticipated that more than 270,000 iPhones were sold in the first 30 hours of its release. When the iPhone 4S went on sale, Apple reported more than 4 million units were sold in the first weekend!

In this section we'll look at some of the common characteristics of the iPhone that have led to its popularity and success, and focus on those that are of particular interest to iPhone application developers.

1.1.1 iPhone Characteristics

The iPhone is a portable device that has a 3.5" 960x640-pixel screen that can be flipped to display in either portrait or landscape mode, 640x960 pixels for portrait or 960x640 for landscape. Its always-on Internet access provides a pleasant, constant Internet experience, keeping the user in touch with the rest of the world. It has several other features:

- 1 GHz ARM CPU
- 512 MB of dynamic RAM (DRAM), and either 16 or 32 GB of flash memory
- Data speeds of up to 7.2 Mbps on compatible 3G networks
- Wi-Fi 802.11 b/g/n connectivity
- Wi-Fi hotspot capability, so you can share your internet connectivity with other devices
- Bluetooth v2.1+ EDR
- Multi-touch capacitive touch screen that responds to the touch of human skin, so there's no need for a stylus or other tool for input. The user accesses the iPhone by just tapping on the screen with one or more fingers.
- CD-quality audio and high frame rate video.
- 5-megapixel camera on the back and 0.3MP (VGA quality) camera on the front for video-calling
- Assisted GPS (A-GPS).
- Battery gives up to 14 hours talk time (on 2G networks, 7 hours on 3G); up to 10 hours Wi-Fi usage; Up to 10 hours video playback; and up to 40 hours audio playback
- The iPhone OS built on Apple's Mac OS X, which is itself built on top of Unix.

The browser used in the iPhone is a mobile version of Apple's Safari. It's a full-fledged browser with access to DOM, CSS, and JavaScript.



Figure 1.1 The Safari browser as it appears on the iPhone

The iPhone has changed the daily lifestyle of many people. Because it's a portable device, several tasks can be done on the iPhone while we are on the move, like surfing the web, playing games, sending and receiving emails, booking tickets, listening to music, playing videos, and shopping online. We can stay in touch with our friends by accessing social networking sites like Facebook and Twitter. There's a built-in camera that can be used to take photos, and a GPS to pinpoint the user's location on a map in real-time. You can also access thousands of iPhone applications available in the App Store. It's set the standard for all the smartphones that have followed it.

That's the brief overview of the iPhone's characteristics and what it gives us. Now it's time to understand the different methods of developing applications for the iPhone.

1.1.2 Two ways to program

iPhone applications can be developed in two ways: One is to develop web applications that can be accessed via mobile Safari and the other to develop native applications using the iPhone SDK. In this section we are going to look at the advantages of these two different methods.

WEB DEVELOPMENT

Web development can be as simple as creating web pages that look good on the iPhone. Web pages can be built using standard web technologies such as HTML, Cascading Style Sheets (CSS), JavaScript, PHP, Ruby on Rails, Python, and other web programming languages. The advantages of doing web development are:

- Easy to develop – When compared to SDK programming, developing and deploying web pages can be much easier and faster.
- Quick delivery of updated, dynamic information – updates on the web server are immediately available to the users, without any waiting for Apple to upload your changes to the App Store.
- Use existing web content – reusing existing content on the web is easy using simple hyperlinks or by incorporating data feeds from other sites using technologies such as RSS and XML.

There are a few drawbacks with web development as a way of creating applications for the iPhone:

- Web applications must be hosted on a web server. This may mean extra costs for you and extra maintenance and security overheads.
- Users must be connected to the internet to be able to access your web application.
- Although the iPhone's Safari web browser supports many modern web standards, it does not support Adobe Flash.
- It is not possible to take advantage of all of the iPhone's advanced features such as the Address Book, accelerometers, or the GPS with web applications.

SDK PROGRAMMING

SDK Programming involves the design of programs that run natively on the iPhone, which means we don't have to use mobile Safari or an external web server to access it. These programs are written in Objective-C, compiled in Xcode, and then deployed through the iPhone App Store as shown in Figure 1.2.

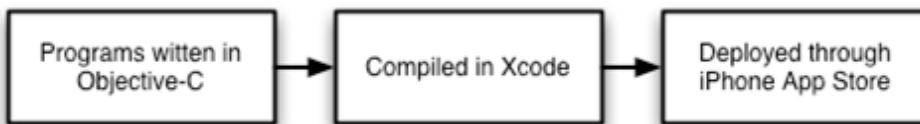


Figure 1.2 Steps used in SDK programming. Xcode is the main link - an IDE that plays a major role in writing and compiling code.

The advantages of SDK programming are:

- SDK programming is object-oriented programming based on the Objective-C language and it allows for sophisticated applications.
- You can import many existing iPhone libraries and directly use their methods to enhance features of an application with minimal effort.
- SDK programming makes it possible to use the advanced capabilities of the iPhone, such as the accelerometer, camera, GPS and multi-touch.
- SDK applications don't require an Internet connection for execution and so they can run when the user is in a remote place (or on an airplane) where Internet access is not available.
- SDK applications can store data locally on the iPhone, which means the user's data is available even when there is no internet connectivity.

SDK programming is essential if you're creating a sophisticated program or suite of programs, or if you need to use functions that are not well supported on the web, such as the address book, the accelerometers, the GPS, and the camera.

Because this book will be focused on programming with iPhone SDK we need to make sure we are all on the same page with the specific terms we'll be using in this chapter and throughout the book. Let's look at these next.

Downloading the iPhone SDK

The iPhone SDK requires an Intel-based Mac running the latest version of Mac OS X. Each version of the SDK has its own operating system version requirements. For the step by step instructions for downloading and installing iPhone SDK refer Appendix A given at the end of the book.

1.1.3 Important Terms

There are a few terms that we need to understand before we move on. We will be using the terms listed in table 1.1 frequently when using iPhone SDK.

Table 1.1 Important terms related to iPhone SDK

Term	Description
Xcode	Apple's IDE (Integrated Development Environment) which comes with the iOS SDK. It's the application used to create iPhone and Mac applications.
Project	All the source code files, libraries, media and other resources required to develop and build an app in Xcode.
iPhone Simulator	A mock iPhone that you can run on the Mac to test iPhone applications
Cocoa	The User Interface (UI) toolkit for Apple devices

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=670>

Framework	This is a collection of libraries, images, and other resources that can be imported into an application.
iOS Dev Center	A repository of many useful documents, videos and tools to help you develop iOS applications.
Apple Developer Key	Required to install applications on iPhone as well as to submit applications on the App Store

Let's look at the above terms in more detail.

XCODE

Xcode is the core of the SDK's integrated development environment. It's the place where you create, edit, compile and maintain projects. It provides all facilities required for developing iPhone applications at one place. We can use it for writing code, compiling as well as executing the code. We can manage our applications via Xcode.

PROJECT

Projects in Xcode contain all the files and other resources required to create an application. The files include the source code, libraries, images and other media, configuration files, and any other resources needed for the application.

IPHONE SIMULATOR

The iPhone simulator is used for running and testing applications before loading onto a real device. It provides an iPhone-like environment with menu options to simulate screen rotations and other basic iPhone events. However, there are several limitations of using the simulator in place of a real iPhone. Several features will not work on iPhone simulator that includes:

- The application can't initiate phone calls.
- The camera and microphone APIs may not function.
- The EDGE or 3G networks will not be accessible.
- The accelerometer API will not be available to the application.

The iPhone simulator doesn't require an Apple developer key to use. You should use the simulator before testing on a real iPhone just to be sure you're not going to harm data on your iPhone (although this is difficult to do).

COCOA TOUCH

Cocoa Touch is a powerful user interface toolkit written in Objective-C. It is an advanced object-oriented framework used for building applications that run on Apple's iOS. Cocoa Touch is based on Cocoa, which is the Mac OS X UI framework. It is a collection of object libraries, a runtime system, and a development environment. Developing graphical user interface (GUI) applications is quite easy with Cocoa.

Cocoa is composed of two object-oriented frameworks: Foundation and UIKit.

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=670>

- The Foundation framework provides objects that provide a “foundation” beneath the user interface; it provides functionality that is required for core application services. It provides classes for handling arrays, strings, and other low level elements.
- The UIKit classes provide functionality for user interface elements seen in an application like text fields and buttons. These classes also handle actions like touch events and keystrokes.

FRAMEWORK

A framework is a collection of resources such as header files, libraries, images, and sounds. A framework usually consists of hundreds of header files, but you don't necessarily see them when you are developing.

IOS DEV CENTER

Apple provides a many useful documents, tools and other resources in the iOS Dev Center. To access the iOS Dev Center you will need to create an Apple Developer account by visiting <http://developer.apple.com/programs/register/> and clicking the Get Started button. While it is free to register as an Apple Developer, the range of resources available will be limited; however you will still find many useful documents and guides even in the free area. In order to be able to test your applications on a real iPhone you will need to subscribe to one of the paid developer programs.

There are three levels of paid membership: Individual Developer, Company Developer and Enterprise Developer. There is little difference between the individual and company developer programs and both cost the same to enroll. The Enterprise developer program is more expensive and is geared more towards large corporations who want to distribute their own apps in-house without using the App Store.

More details on the Apple iOS Developer Programs can be found on Apple's Developer web site at <http://developer.apple.com/programs/ios/>

APPLE DEVELOPER KEY

In order to install applications on an iPhone from Xcode you will need a developer key. Developer keys are only available to members of Apple's paid iOS Developer Programs.

So you now know all the terms that you'll see in the coming pages. Now let's have a brief look at the Objective-C language which you will need to know if you want to make SDK apps.

1.2 Objective-C

At least a little bit of knowledge about Objective-C programming is a must for programming in iPhone SDK. Objective-C tends to be more verbose than many other languages, and it's easy to get lost in its syntax if you're unfamiliar with it; but don't worry, it's actually less imposing than it looks at first.

It's easy to learn the basics and you can be up and running in no time, but mastery will take time. In this section, you are going to learn the concepts of Objective-C programming that will be required to understand the code used in later chapters. We'll look at how to

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=670>

write Objective-C programs using Xcode. Knowledge of an object-oriented programming language such as Java or Smalltalk is really helpful in understanding the concepts of Objective-C but it isn't necessary.

1.2.1 Features of Objective-C

Objective-C is an object-oriented programming language so it has all the features that any other OOP (object-oriented programming) language has including, inheritance, and overriding. It also has the specific following features which makes it ideal for iPhone programming:

- It is a superset of the C language and is easy to learn. You can include and compile C programs within an Objective-C program. This makes it possible to re-use many existing C libraries in your iPhone apps.
- Its syntax is very simple and easy to understand compared to C.
- It has dynamic typing so you can declare variables that can refer to any type of object. The object's type is determined at run-time and allows for some very powerful programming techniques.
- You can send objects messages that aren't defined in the object's interface. Objects can then "capture" undefined messages and modify their behaviour appropriately.
- It is possible to add methods to an existing class without subclassing. For example, you could add some extra string-handling methods to the in-built NSString class without creating a subclass.

It's important to know the file extensions for Objective-C files, since you will be editing various text files that make up the code in an application project. Let's look at these next.

1.2.2 File Extensions for Objective-C

iPhone projects contain many different types of files. The main files you will encounter are listed in Table 1.2.

Table 1.2 Extensions of common file types

Extension	File Type
*.h	Objective-C header file
*.m	Objective-C method implementation file
*.xib	Interface Builder file
*.plist	Property List file

Each class in Objective-C is typically spread across two files. The header file (*.h) contains the definition of the interface for the class (the methods, properties and so on) and the implementation file (*.m) contains the actual code that implements the class.

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=670>

*.xib files contain the user interface definitions. These files are edited using the built-in Interface Builder in Xcode.

*.plist files contain data which you application can use to read and store data.

This has been a quick introduction to Objective-C. Appendix B contains more on Objective-C programming. Now, let's look at the general approach followed when developing iPhone applications.

1.3 The process of iPhone application development

Each time you create a new application from scratch in Xcode you will follow the same basic process:

1. Select project template
2. Define outlets and action methods in the header files
3. Design the views
4. Connect the views' controls to the outlets and action methods
5. Write the code to make the actions do something
6. Build and run the application

Don't worry if some of these steps don't make sense to you, all soon will become clear.

Xcode provides many different types of templates to help you get started. We'll look at templates in more depth in chapter 2.

Every iPhone application consists of a window and one or more views and view controllers. A window represents your application's user interface and takes up the entire screen of your iPhone. A view represents a rectangular area in this window which can take up the entire screen, or may only represent a small part. Views are responsible for drawing what appears on the screen. View controllers handle the user interaction and determine which views are to be displayed.

Views typically contain one or more controls which can display text or images and can respond to user interaction. In order to access the controls placed on a view you need to define particular instance variables in the view controller's header file. Once you have defined the appropriate variables you can connect them to the controls in the view using Interface Builder. Finally you can access the controls' properties and respond to events in the view controller.

Each application you develop has the same basic DNA. Start with a template, add outlets and action methods in the header file(s), design the view(s) visually, connect the outlets and action methods with the controls in the view, write the implementation for the action methods so that they respond to the user, then build the application. Often the build uncovers syntax errors and warnings, so you will have to revisit your code to make sure everything is correct. Once the build is clean you can run the application to see that it

©Manning Publications Co. Please post comments or corrections to the Author Online forum:

<http://www.manning-sandbox.com/forum.jspa?forumID=670>

actually does what you want it to do. You may need to again revisit your code if you have a logic error (meaning that the app does behave as you intended or expected, but still runs).

1.4 Summary

So far we've covered what the iPhone is, what Objective-C is, and how Xcode works. You also know the basic process for designing and developing an iPhone app. With this information you have the background you need to get started creating iPhone apps. And now, since you have an idea of the general approach that's followed in developing iPhone applications, let's apply it in development of a simple Hello World application.