1
Updates on iOS 7

In this chapter, we will find out what's new in iOS 7, starting from new designs, which new APIs and SDKs were presented by Apple with iOS 7, and why you should pick Sprite Kit for game development.

Redesigning the iOS

The new operating system from Apple features overhauled design in almost every element. All buttons, pickers, labels, navigation bars—everything looks and feels different. The concepts that Apple has chosen are simplicity and minimalism. Apple designers have chosen to get rid of the rich textures and gradients that we grew to love for six versions of their mobile operating system.

The new interface is unobtrusive; everything seems to be in its place. Everything that used to draw your attention is now gone, and your content is now in the center of the new design. For example, the following is the screenshot of the iOS 6 calendar followed by its iOS 7 version.

The change to "flat" design was met with enthusiasm by some and not so by others, but all we know is that it is here to stay.

When you are working on your game, you should probably check out the best practices by Apple designers, as they are thought-out, thoroughly tested, and well implemented.

Everything that you need to know about interface design on iOS devices can be found in *Human interface guidelines* by Apple at `https://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehig/`.

On comparing the following two screenshots, you can see the old and new look of the Calendar application. Apple has focused on the user-generated content; you can see that the space for user data is much larger in the iOS 7 version; however, in the old Calendar application, we can see only two lines of our content.

The Calendar application in iOS 6

Buttons have transformed into simple lines of text, without any background or frame, while gradients on the navigation and bottom bar are gone and are replaced by the simple gray background.

The Calendar application on iOS 7

New APIs

The new operating system from Apple features numerous new APIs. Some of them are long overdue and expected, and some are quite surprising. Some of them that are worth mentioning are as follows:

- **Text Kit**: This is an API for laying out text and for fine typography. Text Kit helps you lay out text in the way you like without any headache.
- **Dynamic behaviors for views**: With this, you can now assign physics effects to your views so they can follow each other, or gravity effects can be applied to views with ease.
- **Multitasking enhancements**: With this, applications have the ability to fetch data in the background, something that was available only to certain applications such as Newsstand apps. Now, your game can fetch some amount of data while the user is not playing it, such as daily missions or news.

- **Sprite Kit framework**: This is a new framework for developing 2D games, and features hardware sprites acceleration, simple sound playback support, and physics simulation.
- **Game controller support**: This is a new framework that provides common ground for hardware controllers.

Developing games for iOS 7

In June 2013, Apple announced that the App Store has hit the next milestone—50 billion downloads with more than 14 billion dollars paid to developers all over the world. The ecosystem that only started to exist a few years ago already raked in more money for developers than any other platform.

A major share of this revenue is taken by game developers, ranging from large companies such as EA, Disney, and Rovio to small indie developers that manage to create best-selling applications with small budgets—everyone can find their place under the sun.

The most profitable and most downloaded titles on the App Store are 2D games—Angry Birds, Cut The Rope, and Doodle Jump. Rovio managed to create an empire out of a single title, and now it is selling merchandise, soft drinks, and toys, and all of this came out of a single mobile game (not their first one though, as Angry Birds was their 52nd title!).

Framework for game development

Before iOS 7 (and Sprite Kit), there were various options for frameworks that could be used for game development. Each of them has its own advantages and disadvantages.

If you wanted to make a game before iOS 7, you had only so many options. They are as follows:

- OpenGL ES
- UIKit controls and views
- Third-party libraries (Unity, Unreal 3D, and Cocos2d)

Let us see each of them in detail:

- **OpenGL** is very customizable and open-ended, but it is hard to learn, and you need to know a lot of things just to get an image on screen. It is good if you are an experienced programmer or a company, and you want to write cross-platform solutions. OpenGL offers good performance, but you have to pay with code complexity.
- Next up is **UIKit**, which is the default iOS programming framework. Every element that you see in a regular iOS application, such as buttons, pickers, views, and navigation bars, comes from here. But there are only so many games that can look good with the default interface—some trivia games, maybe some manager games, but that's it. There are benefits to this—your user already knows everything he can do with the interface, gesture controls, and back buttons, and this makes it easier to actually present your idea, but at the same time, UIKit fails to immerse your user into the game; you get the same interface as almost every other application in the App Store. Another big problem with UIKit is performance, or the lack of it. After all, it was not designed for dynamic games, and if you decide to make something complicated in it, you will find the bottleneck pretty fast.
- Another option to consider is third-party libraries. There are dozens of them, and few are very popular among the developers. Unity3D is good, as it offers a cross-platform solution as well as massive numbers of tutorials. The same can be said about Unreal 3D. But these libraries often require you to know completely different programming languages such as C#, C++, or even Lua. It might not be a good choice if you know Objective-C and want to write native applications for the platform, not to mention that the level of complexity of these frameworks is high. You need to learn a lot just before you can have simple sprites moving on screen.
- Another option that you have is the Cocos2d framework. It is somewhat easy, can get you going fast, is open source, and works with Objective-C. But as with any third-party library, it has its disadvantages. It does not support ARC out of the box. It has problems when Apple releases new versions of iOS—so far, every OS release had left Cocos2d code broken in one way or another. You could have the rotation feature stop working altogether, or suddenly some methods may fail to compile with errors. This doesn't really work if all you want is a simple framework for your games.

Knowing about Sprite Kit

Apple presented iOS 7 in September 2013, featuring numerous new features for users and developers. One of them is Sprite Kit—a new framework for game developers.

Sprite Kit is a native SDK from Apple that allows simple entry into game development, without unnecessary overhead and a long learning process. If you are familiar with Cocos2d, you will have an easy time with Sprite Kit, as it is heavily inspired by the former. Sprite Kit provides native rendering and animation infrastructure to work with sprites as well as animations, particle systems, scenes, actions, physics simulation, and video effects.

A sprite is a two-dimensional image or animation integrated into a larger scene. Any image can be a sprite—a character, a tree, or a bullet.

It allows easy work with sprites, the core component of all 2D games. Almost everything you can see on the screen of a 2D game is a sprite.

Benefits of Sprite Kit

Sprite Kit has certain advantages that will help you determine if you want to base your game on it. They are as follows:

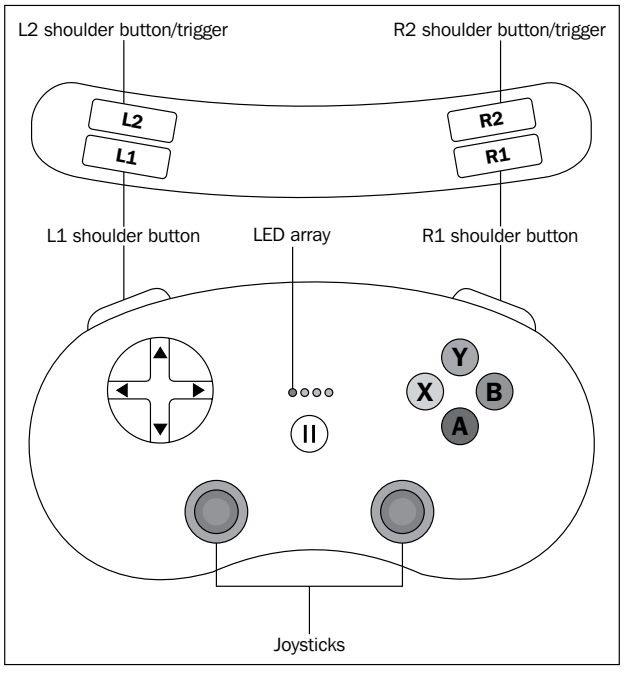
- **Sprite Kit is part of the iOS SDK:** This means that it will be supported by Apple, and everything you write is likely to be future-proof. Your code will not magically stop working (or even compiling!), and things you are getting on screen are guaranteed to stay the same. Everyone who works with third-party libraries is aware of the issues that come with using a non-native SDK. With Sprite Kit, you can forget about installation problems and compatibility problems.
- **Easy-to-use API**: This is developed by some of the best engineers in the world. Everything is logical and works as expected. Clear methods and properties work just as you would expect them to.
- **Built-in tools**: With this, you no longer have to use third-party software to make your texture atlases, assets, or fonts. You just drop in your images and Xcode does everything for you.
- **Built-in physics engine**: This makes your life as a developer much easier. You do not have to pick out one of the third-party physics engines or work on integration of that code into your project—it just works out of the box.

- **Your game will work both on iOS and Mac without much effort:** Sprite Kit supports both Mac and iOS, and all you need to change is controls. You will have touch controls for your iPhone and iPad versions and the mouse and keyboard controls for Mac.

Game controller support

One of the most interesting features of iOS 7 is the native controller support. Some companies such as iCade and others tried working on their own controllers, but this effort has not seen much success. Surely, some games supported it, but the majority of games were left unsupported.

Developers did not feel the need to support such devices, as their install base is small, and return on investment was just not available. But everything changed when Apple decided to roll out support for such controllers. Now we have a native API to work with controllers and all future controllers by different vendors that will work with this API. In the following diagram, you can see an Apple-proposed design for one of the game controllers. As you can see, it offers all the features of a modern controller—two thumb sticks, shoulder buttons, and LEDs.



Game controller for iOS 7

There have been rumors that vendors such as Logitech are already working on such controllers, and you as a developer should probably work on implementing them in your game, as the effort required to make them work is really small, and the satisfaction that your player gets when the game works with his controller is enormous.

The new Game Controller framework allows discovering and connecting compatible game controllers to your iOS device.

Game center renovations

Game center have several new features that will help you with your games. Some of them have been listed as follows:

- **Increased limit of leaderboards per application**: Now you can have up to 100 leaderboards in your game.
- **New feature—exchanges**: This allows your turn-based player to initialize actions even if it is not their turn. Previously, you had to wait for your turn to even chat, and now you can do that on other players' turns if the game supports that.
- **Improved features to prevent cheating**: Cheating, obviously, is never good, especially if your game is competitive and has leaderboards. We all know how such games are infested with hackers, and these new features will certainly help with that.

Summary

In this chapter, we have learned what new exciting features and APIs iOS 7 has to offer us. We have found out what Sprite Kit is and why we should use it for game development and its advantages. We have found out that Apple unified game controllers, and new ones will be available shortly. If you are reading this book, chances are that you are planning to make games for iOS, and Sprite Kit is an excellent choice.

In the next chapter, we will start working on our first Sprite Kit project, a fully featured endless runner game.