



基于全新的 **iOS 11** 进行讲解，演示 **iOS 11** 的新特性  
**Objective-C** 和 **Swift** 双语讲解，全新的 **Swift 4.0** 实例演示  
9 小时的视频讲解，**230** 多个典型实例和两个综合性实例，帮助读者尽快上手开发

# iOS

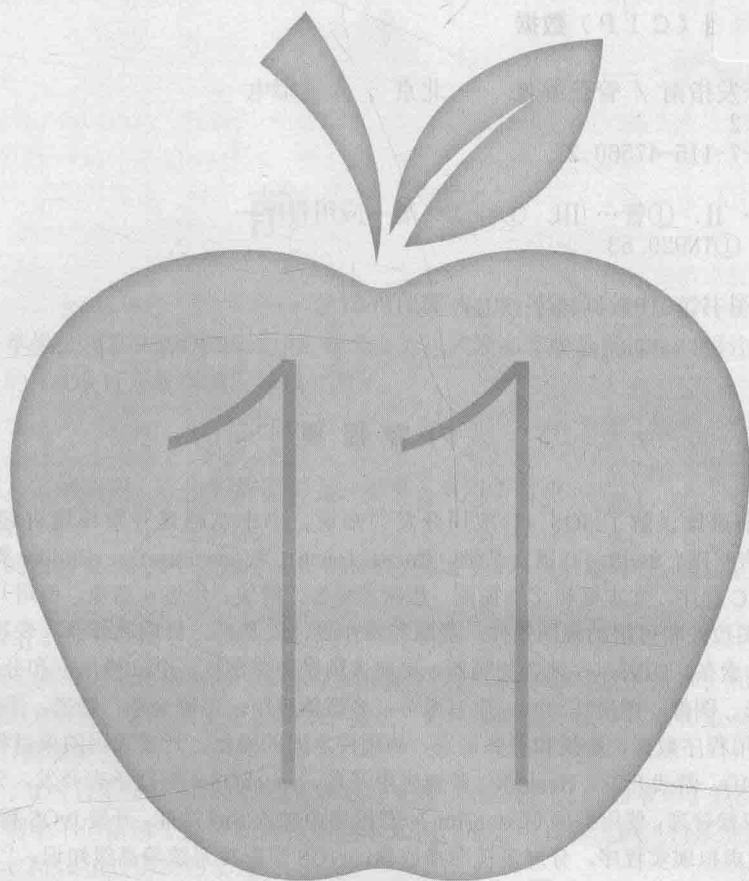


# 开发指南

管蕾 编著



源程序+视频



# iOS 11

## 开发指南

管蕾 编著

人民邮电出版社  
北京

## 图书在版编目 (CIP) 数据

iOS 11 开发指南 / 管蕾编著. — 北京 : 人民邮电出版社, 2018. 2  
ISBN 978-7-115-47560-2

I. ①i… II. ①管… III. ①移动终端—应用程序—程序设计 IV. ①TN929.53

中国版本图书馆CIP数据核字(2017)第318667号

## 内 容 提 要

本书循序渐进地讲解了 iOS 11 应用开发的知识。书中从搭建开发环境讲起,依次讲解了 Objective-C 语言基础, Swift 4.0 语言基础, Cocoa Touch, Xcode Interface Builder 界面开发, 使用 Xcode 编写 MVC 程序, 文本框和文本视图, 按钮和标签, 滑块、步进和图像, 使用开关控件和分段控件, Web 视图控件和可滚动视图控件, 提醒和操作表, 工具栏, 日期选择器, 表视图, 活动指示器, 进度条和检索条, UIView, 视图控制器, 实现多场景和弹出框, iPad 弹出框和分割视图控制器, 界面旋转, 图形、图像、图层和动画, 声音服务, 多媒体应用, 定位处理, 触摸, 手势识别和 Force Touch, 读写应用程序数据, 触摸和手势识别, 和硬件之间的操作, 开发通用的项目程序, 推服务和多线程, Touch ID, 游戏开发, HealthKit 健康应用开发, watchOS 4 智能手表开发, 分屏多任务, 使用 CocoaPods 依赖管理, 使用扩展 (Extension), 在程序中加入 Siri 功能, 开发 tvOS 程序, 开发 Apple Pay 程序, 开发虚拟现实程序, 分屏多视图播放器, tvOS 电影库系统等高级知识。

本书内容全面, 几乎涵盖了 iOS 11 应用开发所需要的主要内容, 适合 iOS 开发初学者和 iOS 程序员学习, 也可以作为相关培训学校和高校相关专业的教学用书。

---

◆ 编 著 管 蕾  
责任编辑 张 涛  
责任印制 焦志炜

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号  
邮编 100164 电子邮件 315@ptpress.com.cn  
网址 <http://www.ptpress.com.cn>  
北京市艺辉印刷有限公司印刷

◆ 开本: 787×1092 1/16

印张: 45.5

字数: 1351 千字

印数: 1-2400 册

2018 年 2 月第 1 版

2018 年 2 月北京第 1 次印刷

---

定价: 118.00 元 (附光盘)

读者服务热线: (010)81055410 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京东工商广登字 20170147 号

# 前言

2017年夏天，苹果公司在WWDC2017开发者大会上正式发布了全新的iOS 11操作系统。2017年年末，经过升级改版的《iOS 11开发指南》与您见面了。

## 本书特色

本书内容丰富，实例全面，在内容的编写上，本书具有以下特色。

### (1) 全新升级。

这次升级更新的内容很多。其中删减了不用的控件，新增了iOS 11的新特性。例如苹果支付Apple Pay和虚拟现实ARKit。

### (2) 全新的Swift 4.0。

本书中的Swift实例采用全新的Swift 4.0编写。Swift 4.0是一款十分稳定的版本，和以前的Swift 1.0、1.1、1.2、2.0、2.2和3.0相比，它的语法更加简洁、高效，更好地解决了以往版本和Xcode的兼容性问题。

### (3) 突出iOS 11的新特性。

本书着重突出了iOS 11系统的新特性，重点剖析了iOS 11的新技术，例如苹果手表的升级和针对iPad产品的升级。在本书中不但讲解了这些新特性的基本知识，而且用具体实例进行了演示。

### (4) Objective-C和Swift双语对照实现。

本书中的实例不仅使用Objective-C语言实现，而且使用了苹果公司推出的Swift 4.0语言。通过本书的学习，读者可以掌握使用Objective-C语言和Swift 4.0语言开发iOS程序的方法。

### (5) 讲解苹果公司力推的新应用技术。

本书内容新颖、全面，讲解了从iOS开始具有或发展起来的新技术。这些新技术是苹果公司所力推的。例如HealthKit、watchOS 4、分屏处理、tvOS和Touch ID，这些内容是市面中同类书籍所没有涉及的。

### (6) 结构合理，易学易用。

从读者的实际需要出发，科学安排知识结构，内容由浅入深，叙述清楚。全书详细地讲解了和iOS开发有关的知识点。读者可以按照本书编排的章节顺序进行学习，也可以根据自己的需求对某一章节进行有针对性学习。书中提供的丰富实例可以帮助读者学以致用。

### (7) 实例典型，实用性强。

本书彻底摒弃枯燥的理论和简单的操作，注重实用性和可操作性。本书正文一共介绍了230多个典型实例和两个综合性实例，并额外赠送了经典实例（赠送实例可从网站[www.toppr.net](http://www.toppr.net)下载），通过实例的实现过程，详细讲解了各个知识点的具体应用方法。

### (8) 内容全面。

无论是搭建开发环境，还是控件接口、网络、多媒体、程序扩展、tvOS和动画，以及游戏应用开发，在本书中都能找到解决问题的方法。

(9) 配套资源丰富，形式多样。视频讲解（共计9小时的视频），以及Objective-C和Swift电子书+PPT教学资源（电子书和PPT通过网站[www.toppr.net](http://www.toppr.net)下载）。

为了帮助初学者更加高效地看懂并掌握本书内容，本书提供了内容全面的配套视频。视频不但讲解了本书中的重要知识点，而且详细讲解并演示了书中的每一个实例。另外，为了方便广大教师的教学工作，特意提供了对应的PPT教学资料。读者可以登录本书售后网站[www.toppr.net](http://www.toppr.net)下载。

## 本书的内容安排

**必备技术部分：**主要讲解了 iOS 开发入门、Xcode 开发环境详解、Objective-C 语言基础、Swift 语言基础、Cocoa Touch 框架、Xcode Interface Builder 界面开发、使用 Xcode 编写 MVC 程序。

**控件实战部分：**讲解了文本框和文本视图，按钮和标签，滑块、步进和图像，开关控件和分段控件，Web 视图控件，可滚动视图控件和翻页控件，提醒和操作表，工具栏，日期选择器，表视图 (UITableView)，活动指示器，进度条和检索条。

**核心技术部分：**讲解了视图控制器，实现多场景和弹出框，UICollectionView 和 UIVisualEffectView 控件，iPad 弹出框和分割视图控制器，界面旋转、大小和全屏处理，图形、图像、图层和动画，多媒体开发，分屏多任务，定位处理。

**典型应用部分：**讲解了读写应用程序数据、触摸、手势识别和 Force Touch、和硬件之间的操作、地址簿、邮件、Twitter 和短消息、开发通用的项目程序、推服务和多线程、Touch ID 详解。

**技术提高部分：**讲解了使用 CocoaPods 依赖管理、使用扩展 (Extension)、游戏开发、watchOS 智能手表开发、HealthKit 健康应用开发、在程序中加入 Siri 功能、开发 tvOS 程序、Apple Pay 和 ARKit。

**综合实战部分：**讲解了两个综合案例，把所学的知识应用起来，如分屏多视图播放器和 tvOS 电影库系统。

## 读者对象

初学 iOS 编程的自学者；

从事 iOS 开发的程序员；

iOS 编程爱好者；

大中专院校的教师和学生；

毕业设计的学生；

相关培训机构的老师和学员。

## 售后服务

为了更好地为读者服务，为大家提供一个完善的学习和交流平台，本书提供了读者交流 QQ 群，群号 28316661，大家可以在里面学习交流。另外，还提供了问题答疑和本书源程序的下载地址：[www.toppr.net](http://www.toppr.net)。

本书在编写过程中，得到了人民邮电出版社工作人员的大力支持，正是基于各位编辑的求实、耐心和效率，才使得本书在这么短的时间内出版。另外，也十分感谢我的家人，在我写作的时候给予的大力支持。由于作者水平有限，书中纰漏和不尽如人意之处在所难免，诚请读者提出意见或建议，以便修订并使之更臻完善。编辑联系和投稿邮箱为 [zhangtao@ptpress.com.cn](mailto:zhangtao@ptpress.com.cn)。

编者

# 目 录

第 1 章 iOS 开发入门	1
1.1 iOS 系统介绍	1
1.1.1 iOS 发展史	1
1.1.2 全新的版本——iOS 11	1
1.2 开始 iOS 11 开发之旅	2
1.3 工欲善其事，必先利其器——搭建开发环境	3
1.3.1 Xcode 介绍	4
1.3.2 下载并安装 Xcode 9	4
1.3.3 创建 iOS 11 项目并启动模拟器	6
1.3.4 打开一个现有的 iOS 11 项目	8
1.4 iOS 11 中的常用开发框架	8
1.4.1 Foundation 框架简介	8
1.4.2 Cocoa 框架简介	10
1.4.3 iOS 程序框架	10
第 2 章 使用 Xcode 开发环境详解	12
2.1 基本面板介绍	12
2.1.1 调试工具栏	12
2.1.2 导航面板介绍	13
2.1.3 检查器面板	15
2.2 Xcode 9 的基本操作	16
2.2.1 改变公司名称	16
2.2.2 通过搜索框缩小文件范围	16
2.2.3 格式化代码	17
2.2.4 代码缩进和自动完成	17
2.2.5 文件内查找和替代	18
2.2.6 快速定位到代码行	19
2.2.7 快速打开文件	19
2.2.8 自定义导航条	20
2.2.9 使用 Xcode 帮助	21
2.2.10 调试代码	21
2.3 使用 Xcode 9 帮助系统	22
第 3 章 Objective-C 语言基础	24
3.1 最耀眼的新星	24
3.1.1 究竟何为 Objective-C	24
3.1.2 为什么选择 Objective-C	24
3.2 Objective-C 的优点及缺点	25
3.3 一个简单的例子	26
3.3.1 使用 Xcode 编辑代码	26
3.3.2 基本元素介绍	27
3.4 数据类型和常量	31
3.4.1 int 类型	32
3.4.2 float 类型	33
3.4.3 double 类型	33
3.4.4 char 类型	34
3.4.5 字符常量	35
3.4.6 id 类型	36
3.4.7 限定词	37
3.4.8 总结基本数据类型	39
3.5 字符串	39
3.6 算术表达式	40
3.6.1 运算符的优先级	40
3.6.2 整数运算和一元负号运算符	41
3.6.3 模运算符	42
3.6.4 整型值和浮点值的相互转换	43
3.6.5 类型转换运算符	44
3.7 表达式	44
3.7.1 常量表达式	44
3.7.2 条件运算符	45
3.7.3 sizeof 运算符	45
3.7.4 关系运算符	46
3.7.5 强制类型转换运算符	46
3.8 位运算符	47
3.8.1 按位与运算符	47
3.8.2 按位或运算符	48
3.8.3 按位异或运算符	48
3.8.4 一次求反运算符	49
3.8.5 向左移位运算符	50
3.8.6 向右移位运算符	50
3.8.7 总结 Objective-C 的运算符	51
第 4 章 Swift 语言基础	52
4.1 Swift 概述	52
4.1.1 Swift 的创造者	52
4.1.2 Swift 的优势	52
4.1.3 最新的 Swift 4.0	53
4.2 数据类型和常量	54
4.2.1 int 类型	54
4.2.2 float 类型	55
4.2.3 double 类型	55
4.2.4 char 类型	55
4.2.5 字符常量	55

4.3 变量和常量	56	6.4.1 推出的背景	89
4.3.1 常量详解	56	6.4.2 故事板的文档大纲	90
4.3.2 变量详解	56	6.4.3 文档大纲的区域对象	91
4.4 字符串和字符	57	6.5 创建一个界面	91
4.4.1 字符串字面量	57	6.5.1 对象库	91
4.4.2 初始化空字符串	58	6.5.2 将对象加入到视图中	92
4.4.3 字符串可变性	58	6.5.3 使用 IB 布局工具	93
4.4.4 值类型字符串	58	6.6 定制界面外观	95
4.4.5 计算字符数量	59	6.6.1 使用属性检查器	95
4.4.6 连接字符串和字符	59	6.6.2 设置辅助功能属性	95
4.4.7 字符串插值	60	6.6.3 测试界面	96
4.4.8 比较字符串	60	6.7 iOS 11 控件的属性	96
4.4.9 Unicode	61	6.8 实战演练——将设计界面连接到代码（双语实现：Objective-C 版）	97
4.5 流程控制	63	6.8.1 打开项目	97
4.5.1 for 循环（1）	63	6.8.2 输出口和操作	98
4.5.2 for 循环（2）	64	6.8.3 创建到输出口的连接	98
4.5.3 while 循环	65	6.8.4 创建到操作的连接	100
4.6 条件语句	66	6.9 实战演练——将设计界面连接到代码（双语实现：Swift 版）	101
4.6.1 if 语句	66	6.10 实战演练——纯代码实现 UI 设计	102
4.6.2 switch 语句	67		
4.7 函数	68	<b>第 7 章 使用 Xcode 编写 MVC 程序</b>	104
4.7.1 函数的声明与调用	68	7.1 MVC 模式基础	104
4.7.2 函数的参数和返回值	69	7.2 Xcode 中的 MVC	105
4.8 实战演练——使用 Xcode 创建 Swift 程序	70	7.2.1 原理	105
		7.2.2 模板就是给予 MVC 的	105
<b>第 5 章 Cocoa Touch 框架</b>	72	7.3 在 Xcode 中实现 MVC	106
5.1 Cocoa Touch 基础	72	7.3.1 视图	106
5.1.1 Cocoa Touch 概述	72	7.3.2 视图控制器	106
5.1.2 Cocoa Touch 中的框架	73	7.4 数据模型	108
5.1.3 Cocoa Touch 的优势	73	7.5 实战演练——使用模板 Single View Application 创建 MVC 程序（双语实现：Objective-C 版）	109
5.2 iPhone 的技术层	73	7.5.1 创建项目	109
5.2.1 Cocoa Touch 层	73	7.5.2 规划变量和连接	110
5.2.2 多媒体层	76	7.5.3 设计界面	112
5.2.3 核心服务层	77	7.5.4 创建并连接输出口和操作	113
5.2.4 核心 OS 层	78	7.5.5 实现应用程序逻辑	114
5.3 Cocoa Touch 中的框架	78	7.5.6 生成应用程序	115
5.3.1 Core Animation（图形处理）框架	78	7.6 实战演练——使用模板 Single View Application 创建 MVC 程序（双语实现：Swift 版）	115
5.3.2 Core Audio（音频处理）框架	79		
5.3.3 Core Data（数据处理）框架	79	<b>第 8 章 文本框和文本视图</b>	116
5.4 Cocoa 中的类	80	8.1 文本框（UITextField）	116
5.4.1 核心类	81	8.1.1 文本框基础	116
5.4.2 数据类型类	82	8.1.2 实战演练——控制是否显示 TextField 中信息	116
5.4.3 UI 界面类	83	8.1.3 实战演练——实现用户登录框界面	118
5.5 国际化	85	8.1.4 实战演练——限制输入文本的长度	119
<b>第 6 章 Xcode Interface Builder 界面开发</b>	86		
6.1 Interface Builder 基础	86		
6.2 和 Interface Builder 密切相关的库面板	88		
6.3 Interface Builder 采用的方法	88		
6.4 Interface Builder 中的故事板——Storyboarding	89		

8.1.5 实战演练——实现一个 UITextField 控件 (Swift 版) .....	120	10.1.2 实战演练——使用素材图片实现滑动条特效 .....	148
8.2 文本视图 (UITextView) .....	121	10.1.3 实战演练——实现自动显示刻度的滑动条 .....	149
8.2.1 文本视图基础 .....	121	10.1.4 实战演练——实现各种各样的滑块 .....	150
8.2.2 实战演练——拖动输入的文本 .....	122	10.1.5 实战演练——自定义实现 UISlider 控件功能 (Swift 版) .....	152
8.2.3 实战演练——自定义设置文字的间距 .....	122	10.2 步进控件 (UIStepper) .....	153
8.2.4 实战演练——自定义 UITextView 控件的样式 .....	123	10.2.1 步进控件介绍 .....	153
8.2.5 实战演练——在指定的区域中输入文本 (Swift 版) .....	125	10.2.2 实战演练——自定义步进控件的样式 .....	154
8.2.6 实战演练——通过文本提示被单击的按钮 (双语实现: Objective-C 版) ..	126	10.2.3 实战演练——设置指定样式的步进控件 .....	155
8.2.7 实战演练——在屏幕中显示被单击的按钮 (双语实现: Swift 版) .....	126	10.2.4 实战演练——使用步进控件自动增减数字 (Swift 版) .....	156
<b>第 9 章 按钮和标签</b> .....	127	10.3 图像视图控件 (UIImageView) .....	157
9.1 标签 (UILabel) .....	127	10.3.1 UIImageView 的常用操作 .....	157
9.1.1 标签 (UILabel) 的属性 .....	127	10.3.2 实战演练——实现图像的模糊效果 .....	157
9.1.2 实战演练——使用 UILabel 显示一段文本 .....	127	10.3.3 实战演练——滚动浏览图片 .....	159
9.1.3 实战演练——为文字分别添加上划线、下划线和中划线 .....	129	10.3.4 实战演练——实现一个图片浏览器 .....	160
9.1.4 实战演练——显示被触摸单词的字母 .....	130	10.3.5 实战演练——使用 UIImageView 控件 (Swift 版) .....	162
9.1.5 实战演练——显示一个指定样式的文本 (Swift 版) .....	130	<b>第 11 章 开关控件和分段控件</b> .....	163
9.2 按钮 (UIButton) .....	131	11.1 开关控件 (UISwitch) .....	163
9.2.1 按钮基础 .....	132	11.1.1 开关控件基础 .....	163
9.2.2 实战演练——自定义设置按钮的图案 .....	132	11.1.2 实战演练——改变 UISwitch 的文本和颜色 .....	163
9.2.3 实战演练——实现了一个变换形状动画按钮 .....	134	11.1.3 实战演练——显示具有开关状态的开关 .....	164
9.3 实战演练——联合使用文本框、文本视图和按钮 (双语实现: Objective-C 版) .....	135	11.1.4 实战演练——显示一个默认打开的 UISwitch 控件 .....	165
9.3.1 创建项目 .....	135	11.1.5 实战演练——控制是否显示密码明文 (Swift 版) .....	165
9.3.2 设计界面 .....	136	11.2 分段控件 (UISegmentedControl) .....	166
9.3.3 创建并连接输出口和操作 .....	140	11.2.1 分段控件的属性和方法 .....	167
9.3.4 实现按钮模板 .....	141	11.2.2 实战演练——使用 UISegmentedControl 控件 .....	168
9.3.5 隐藏键盘 .....	142	11.2.3 实战演练——添加图标和文本 .....	170
9.3.6 实现应用程序逻辑 .....	144	11.2.4 实战演练——使用分段控件控制背景颜色 .....	171
9.3.7 总结执行 .....	145	11.2.5 实战演练——使用 UISegmentedControl 控件 (Swift 版) .....	172
9.4 实战演练——联合使用文本框、文本视图和按钮 (双语实现: Swift 版) .....	145	11.3 实战演练——联合使用开关控件和分段控件 (双版实现: Objective-C 版) .....	173
9.5 实战演练——自定义一个按钮 (Swift 版) .....	145	11.4 实战演练——联合使用开关控件和分段控件 (双版实现: Swift 版) .....	175
<b>第 10 章 滑块、步进和图像</b> .....	147		
10.1 滑块控件 (UISlider) .....	147		
10.1.1 Slider 控件的基本属性 .....	147		



第 12 章 Web 视图控件、可滚动视图控件和 翻页控件	176
12.1 Web 视图 (UIWebView)	176
12.1.1 Web 视图基础	176
12.1.2 实战演练——在 UIWebView 控件 中调用 JavaScript 脚本	177
12.1.3 实战演练——使用滑动条动态改 变字体的大小	178
12.1.4 实战演练——实现一个迷你浏览 器工具	179
12.1.5 实战演练——使用 UIWebView 控 件加载网页 (Swift 版)	181
12.2 可滚动的视图 (UIScrollView)	182
12.2.1 UIScrollView 的基本用法	182
12.2.2 实战演练——使用可滚动视图 控件	183
12.2.3 实战演练——滑动隐藏状态栏	186
12.2.4 实战演练——使用 UIScrollView 控件 (Swift 版)	186
12.3 翻页控件 (UIPageControl)	187
12.3.1 PageControll 控件基础	187
12.3.2 实战演练——自定义 UIPageControl 控件的外观样式	188
12.3.3 实战演练——实现一个图片播 放器	189
12.3.4 实战演练——实现一个图片浏览 程序	191
12.3.5 实战演练——使用 UIPageControl 控件设置 4 个界面 (Swift 版)	191
12.4 实战演练——联合使用开关、分段控件和 Web 视图控件 (双语实现: Objective-C 版)	193
12.4.1 创建项目	194
12.4.2 设计界面	194
12.4.3 创建并连接输出和操作	196
12.4.4 实现应用程序逻辑	197
12.4.5 调试运行	200
12.5 实战演练——联合使用开关、分段控件和 Web 视图控件 (双语实现: Swift 版)	200
第 13 章 提醒和操作表	201
13.1 UIAlertControlller 基础	201
13.1.1 提醒视图	201
13.1.2 操作表基础	201
13.2 使用 UIAlertControlller	201
13.2.1 一个简单的对话框例子	202
13.2.2 “警告”样式	203
13.2.3 文本对话框	203
13.2.4 上拉菜单	205
13.2.5 释放对话框控制器	207
13.3 实战演练	207
13.3.1 实战演练——实现一个自定义操 作表视图	207
13.3.2 实战演练——分别自定义实现提醒 表视图和操作表视图	208
13.3.3 实战演练——自定义 UIAlertControlller 控件的外观	209
13.3.4 实战演练——实现一个提醒框效 果 (Swift 版)	211
第 14 章 工具栏、日期选择器	212
14.1 工具栏 (UIToolbar)	212
14.1.1 工具栏基础	212
14.1.2 实战演练——联合使用 UIToolbar 和 UIView	213
14.1.3 实战演练——自定义 UIToolbar 控 件的颜色和样式	214
14.1.4 实战演练——创建一个带有图标 按钮的工具栏	215
14.1.5 实战演练——使用 UIToolbar 制作 一个网页浏览器 (Swift 版)	216
14.2 选择器视图 (UIPickerView)	218
14.2.1 选择器视图基础	218
14.2.2 实战演练——实现两个 UIPickerView 控件间的数据依赖	219
14.2.3 实战演练——自定义一个选择器 (双语实现: Objective-C 实现)	222
14.2.4 实战演练——自定义一个选择器 (双语实现: Swift 版)	229
14.2.5 实战演练——实现一个单列 选择器	229
14.2.6 实战演练——实现一个“星期”选 择框	230
14.3 日期选择控件 (UIDatePicker)	231
14.3.1 UIDatePicker 基础	231
14.3.2 实战演练——使用 UIDatePicker 控 件 (Swift 版)	233
14.3.3 实战演练——实现一个日期 选择器	234
14.3.4 实战演练——使用日期选择器自 动选择一个时间	240
第 15 章 表视图 (UITableView)	242
15.1 表视图基础	242
15.1.1 表视图的外观	242
15.1.2 表单元格	242
15.1.3 添加表视图	242
15.1.4 UITableView 详解	244
15.2 实战演练	246
15.2.1 实战演练——自定义 UITableViewCell	246

15.2.2 实战演练——在表视图中动态操作单元格 (Swift 版) .....	249	17.2.1 实战演练——给任意 UIView 视图四条边框加上阴影 .....	286
15.2.3 实战演练——拆分表视图 (双语实现: Objective-C 版) .....	251	17.2.2 实战演练——给 UIView 加上各种圆角、边框效果 .....	287
15.2.4 实战演练——拆分表视图 (双语实现: Swift 版) .....	252	17.2.3 实战演练——使用 UIView 控件实现弹出式动画表单效果 .....	288
<b>第 16 章 活动指示器、进度条和检索条</b> .....	253	17.2.4 实战演练——创建一个滚动图片浏览器 (Swift 版) .....	289
16.1 活动指示器 (UIActivityIndicator) .....	253	17.2.5 实战演练——创建一个产品展示列表 (双语实现: Objective-C 版) .....	290
16.1.1 活动指示器基础 .....	253	17.2.6 实战演练——创建一个产品展示列表 (双语实现: Swift 版) .....	291
16.1.2 实战演练——自定义 UIActivityIndicator 控件的样式 .....	253	<b>第 18 章 视图控制器</b> .....	292
16.1.3 实战演练——自定义活动指示器的显示样式 .....	255	18.1 导航控制器 (UIViewController) 基础 .....	292
16.1.4 实战演练——实现不同外观的活动指示器效果 .....	258	18.1.1 UIViewController 的常用属性和方法 .....	292
16.1.5 实战演练——使用 UIActivityIndicator View 控件 (Swift 版) .....	259	18.1.2 实战演练——实现可以移动切换的视图效果 .....	293
16.2 进度条 (UIProgressView) .....	260	18.1.3 实战演练——实现手动旋转屏幕的效果 .....	293
16.2.1 进度条基础 .....	261	18.2 使用 UINavigationController .....	294
16.2.2 实战演练——自定义进度条的外观样式 .....	261	18.2.1 UINavigationController 详解 .....	295
16.2.3 实战演练——实现多个具有动态条纹背景的进度条 .....	261	18.2.2 实战演练——实现一个界面导航条功能 .....	296
16.2.4 实战演练——自定义一个指定外观样式的进度条 .....	264	18.2.3 实战演练——创建主从关系的“主-子”视图 (Swift 版) .....	299
16.2.5 实战演练——实现自定义进度条效果 (Swift 版) .....	268	18.2.4 实战演练——使用导航控制器展现 3 个场景 (双语实现: Objective-C 版) .....	300
16.3 检索条 (UISearchBar) .....	269	18.2.5 实战演练——使用导航控制器展现 3 个场景 (双语实现: Swift 版) .....	303
16.3.1 检索条基础 .....	269	18.3 选项卡栏控制器 .....	304
16.3.2 实战演练——在查找信息输入关键字时实现自动提示功能 .....	270	18.3.1 选项卡栏和选项卡栏项 .....	304
16.3.3 实战演练——实现文字输入的自动填充和自动提示功能 .....	273	18.3.2 实战演练——使用选项卡栏控制器构建 3 个场景 .....	306
16.3.4 实战演练——使用检索控件快速搜索信息 .....	274	18.3.3 实战演练——使用动态单元格定制表格行 .....	310
16.3.5 实战演练——使用 UISearchBar 控件 (Swift 版) .....	277	18.3.4 实战演练——开发一个界面选择控制器 (Swift 版) .....	311
16.3.6 实战演练——在表视图中实现信息检索 (双语实现: Objective-C 版) .....	278	<b>第 19 章 实现多场景和弹出框</b> .....	313
16.3.7 实战演练——在表视图中实现信息检索 (双语实现: Swift 版) .....	281	19.1 多场景故事板 .....	313
<b>第 17 章 UIView 详解</b> .....	282	19.1.1 多场景故事板基础 .....	313
17.1 UIView 基础 .....	282	19.1.2 创建多场景项目 .....	314
17.1.1 UIView 的结构 .....	282	19.1.3 实战演练——实现多个视图之间的切换 .....	317
17.1.2 视图架构 .....	284	19.1.4 实战演练——使用第二个视图来编辑第一个视图中的信息 (双语实现: Objective-C 版) .....	320
17.1.3 视图层次和子视图管理 .....	284		
17.1.4 视图绘制周期 .....	285		
17.1.5 UIView 的常见应用 .....	285		
17.2 实战演练 .....	286		

19.1.5 实战演练——使用第二个视图来编辑第一个视图中的信息（双语实现：Swift 版）	323
<b>第 20 章 UICollectionView 和 UIVisualEffectView 控件</b>	324
20.1 UICollectionView 控件详解	324
20.1.1 UICollectionView 的构成	324
20.1.2 实现一个简单的 UICollectionView	325
20.1.3 自定义的 UICollectionViewLayout	327
20.1.4 实战演练——使用 UICollectionView 控件实现网格效果	328
20.1.5 实战演练——实现大小不相同的网格效果	331
20.1.6 实战演练——实现不同颜色方块的布局效果（Swift 版）	333
20.2 UIVisualEffectView 控件详解	333
20.2.1 UIVisualEffectView 基础	334
20.2.2 使用 Visual Effect View 控件实现模糊特效	335
20.2.3 使用 Visual Effect View 实现 Vibrancy 效果	336
20.2.4 实战演练——在屏幕中实现模糊效果	337
20.2.5 实战演练——在屏幕中实现遮罩效果	338
20.2.6 实战演练——编码实现指定图像的模糊效果（Swift 版）	339
<b>第 21 章 iPad 弹出框和分割视图控制器</b>	341
21.1 iPad 弹出框控制器（UIPopoverPresentationController）	341
21.1.1 创建弹出框	341
21.1.2 创建弹出切换	341
21.1.3 实战演练——弹出模态视图	342
21.2 探索分割视图控制器	343
21.2.1 分割视图控制器基础	343
21.2.2 实战演练——使用表视图（双语实现：Objective-C 版）	345
21.2.3 实战演练——使用表视图（双语实现：Swift 版）	349
21.2.4 实战演练——创建基于主从关系的分割视图（Swift 版本）	350
<b>第 22 章 界面旋转、大小和全屏处理</b>	352
22.1 启用界面旋转	352
22.1.1 界面旋转基础	352
22.1.2 实战演练——实现界面自适应（Swift 版）	353
22.1.3 实战演练——设置界面实现自适应（双语实现：Objective-C 版）	354
22.1.4 实战演练——设置界面实现自适应（双语实现：Swift 版）	354
22.2 设计可旋转和可调整大小的界面	355
22.2.1 自动旋转和自动调整大小	355
22.2.2 调整框架	355
22.2.3 切换视图	355
22.2.4 实战演练——使用 Interface Builder 创建可旋转和调整大小的界面	355
22.2.5 实战演练——在旋转时调整控件	357
22.2.6 实战演练——旋转时切换视图	360
22.2.7 实战演练——实现屏幕视图的自动切换（Swift 版）	363
<b>第 23 章 图形、图像、图层和动画</b>	364
23.1 图形处理	364
23.1.1 iOS 的绘图机制	364
23.1.2 实战演练——在屏幕中绘制一个三角形	365
23.1.3 实战演练——使用 CoreGraphic 实现绘图操作	366
23.2 图像处理	368
23.2.1 实战演练——实现颜色选择器/调色板功能	368
23.2.2 实战演练——在屏幕中绘制一个图像	369
23.3 图层	369
23.3.1 视图和图层	369
23.3.2 实战演练——实现图片、文字以及翻转效果	370
23.3.3 实战演练——滑动展示不同的图片	371
23.3.4 实战演练——演示 CALayers 图层的用法（Swift 版）	371
23.4 实现动画	372
23.4.1 UIImageView 动画	372
23.4.2 视图动画 UIView	372
23.4.3 Core Animation 详解	376
23.4.4 实战演练——实现 UIView 分类动画效果	376
23.4.5 实战演练——动画样式显示电量使用情况	378
23.4.6 实战演练——图形图像的人脸检测处理（Swift 版）	381
23.4.7 实战演练——联合使用图像动画、滑块和步进控件（双语实现：Objective-C 版）	382
23.4.8 实战演练——联合使用图像动画、滑块和步进控件（双语实现：Swift 版）	390

第 24 章 多媒体开发	391	26.3 获取位置	428
24.1 使用 AudioToolbox 框架	391	26.3.1 位置管理器委托	429
24.1.1 声音服务基础	391	26.3.2 获取航向	430
24.1.2 实战演练——播放指定的声音文件	392	26.3.3 实战演练——定位当前的位置信息	431
24.1.3 实战演练——播放任意位置的音频	393	26.4 加入地图功能	432
24.2 提醒和振动	393	26.4.1 Map Kit 基础	432
24.2.1 播放提醒音	394	26.4.2 为地图添加标注	433
24.2.2 实战演练——实现两种类型的振动效果 (Swift 版)	394	26.4.3 实战演练——在地图中定位当前的位置信息 (Swift 版)	434
24.2.3 实战演练——实用 iOS 的提醒功能	395	26.4.4 实战演练——在地图中绘制导航线路	435
24.3 AV Foundation 框架	401	26.5 实战演练——创建一个支持定位的应用程序 (双语实现: Objective-C 版)	436
24.3.1 准备工作	402	26.5.1 创建项目	437
24.3.2 使用 AV 音频播放器	402	26.5.2 设计视图	438
24.3.3 实战演练——使用 AV Foundation 框架播放视频	402	26.5.3 创建并连接输出口	438
24.3.4 实战演练——使用 AVAudioPlayer 播放和暂停指定的 MP3 播放 (Swift 版)	403	26.5.4 实现应用程序逻辑	438
24.3.5 实战演练——使用 AVKit 框架播放列表中的视频	404	26.5.5 生成应用程序	440
24.3.6 实战演练——使用 AVKit 框架播放本地视频	405	26.6 实战演练——创建一个支持定位的应用程序 (双语实现: Swift 版)	440
24.3.7 实战演练——使用 AVKit 框架播放网络视频	406	26.7 实战演练——实现地图定位 (双语实现: Objective-C 版)	441
24.4 图像选择器 (UIImagePickerController)	407	26.8 实战演练——实现地图定位 (双语实现: Swift 版)	442
24.4.1 使用图像选择器	407	第 27 章 读写应用程序数据	443
24.4.2 实战演练——获取照片库的图片	407	27.1 iOS 应用程序和数据存储	443
第 25 章 分屏多任务	410	27.2 用户默认设置	444
25.1 分屏多任务基础	410	27.3 设置束	444
25.1.1 分屏多任务的开发环境	410	27.3.1 设置束基础	444
25.1.2 Slide Over 和 Split View 基础	411	27.3.2 实战演练——通过隐式首选项实现一个手电筒程序 (双语实现: Objective-C 版)	445
25.1.3 画中画	412	27.3.3 实战演练——通过隐式首选项实现一个手电筒程序 (双语实现: Swift 版)	448
25.2 实战演练	413	27.4 直接访问文件系统	448
25.2.1 实战演练——使用 SlideOver 多任务 (Swift 版)	413	27.4.1 应用程序数据的存储位置	449
25.2.2 实战演练——使用 SplitView 多任务 (Swift 版)	415	27.4.2 获取文件路径	449
25.2.3 实战演练——开发一个分割多视图浏览器 (Swift 版)	419	27.4.3 读写数据	450
第 26 章 定位处理	422	27.4.4 读取和写入文件	450
26.1 iOS 模拟器调试定位程序的方法	422	27.4.5 通过 plist 文件存取文件	452
26.2 Core Location 框架	423	27.4.6 保存和读取文件	453
26.2.1 Core Location 基础	423	27.4.7 文件共享和文件类型	453
26.2.2 使用流程	423	27.4.8 实战演练——实现一个用户信息收集器 (双语实现: Objective-C 版)	454
26.2.3 实战演练——定位显示当前的位置信息 (Swift 版)	425	27.4.9 实战演练——实现一个用户信息收集器 (双语实现: Swift 版)	457


27.5 核心数据 (Core Data) .....	458
27.5.1 Core Data 基础 .....	458
27.5.2 实战演练——使用 CoreData 动态 添加、删除数据 .....	459
27.6 互联网数据 .....	460
27.6.1 XML 和 JSON .....	460
27.6.2 实战演练——使用 JSON 获取网站 中的照片信息 .....	463
<b>第 28 章 触摸、手势识别和 Force Touch</b> .....	466
28.1 多点触摸和手势识别基础 .....	466
28.2 触摸处理 .....	466
28.2.1 触摸事件和视图 .....	467
28.2.2 iOS 中的手势操作 .....	469
28.2.3 实战演练——触摸的方式移动 视图 .....	470
28.2.4 实战演练——触摸挪动彩色方块 (Swift 版) .....	470
28.3 手势处理 .....	474
28.3.1 手势处理基础 .....	474
28.3.2 实战演练——识别手势并移动屏 幕中的方块 (Swift 版) .....	477
28.3.3 实战演练——实现一个手势识别器 (双语实现: Objective-C 版) .....	480
28.3.4 实战演练——实现一个手势识别器 (双语实现: Swift 版) .....	485
28.4 全新感应功能——Force Touch (3D Touch) 技术 .....	485
28.4.1 Force Touch 介绍 .....	486
28.4.2 Force Touch APIs 介绍 .....	486
28.4.3 实战演练——使用 Force Touch .....	487
28.4.4 实战演练——启动 Force Touch 触 控面板 .....	489
28.4.5 实战演练——为应用程序添加 3D Touch 手势 (Swift 版) .....	489
<b>第 29 章 和硬件之间的操作</b> .....	491
29.1 加速计和陀螺仪 .....	491
29.1.1 加速计基础 .....	491
29.1.2 陀螺仪 .....	493
29.1.3 实战演练——使用 Motion 传感器 (Swift 版) .....	494
29.1.4 实战演练——检测倾斜和旋转 (双 语实现: Objective-C 版) .....	495
29.1.5 实战演练——检测倾斜和旋转 (双 语实现: Swift 版) .....	499
29.2 访问朝向和运动数据 .....	500
29.2.1 两种方法 .....	500
29.2.2 实战演练——检测当前设备的朝向 (双语实现: Objective-C 版) .....	502
29.2.3 实战演练——检测当前设备的朝 向 (双语实现: Swift 版) .....	503
<b>第 30 章 地址簿、邮件、Twitter 和短消息</b> .....	504
30.1 Contacts Framework 框架 .....	504
30.1.1 Contacts 框架的主要构成类 .....	504
30.1.2 使用 Contact 框架 .....	505
30.1.3 实战演练——使用 Contacts 框架获 取通信录信息 .....	505
30.2 Message UI 电子邮件 .....	507
30.2.1 Message UI 基础 .....	507
30.2.2 实战演练——使用 Message UI 发 送邮件 (Swift 版) .....	508
30.3 使用 Twitter 发送推特信息 .....	509
30.3.1 Twitter 基础 .....	509
30.3.2 实战演练——开发一个 Twitter 客 户端 (Swift 版) .....	509
30.4 实战演练——联合使用地址簿、电子邮件、 Twitter 和地图 (双语实现: Objective-C 版) .....	511
30.4.1 创建项目 .....	511
30.4.2 设计界面 .....	512
30.4.3 创建并连接输出和操作 .....	512
30.4.4 实现通信录逻辑 .....	513
30.4.5 实现地图逻辑 .....	513
30.4.6 实现电子邮件逻辑 .....	514
30.4.7 实现 Twitter 逻辑 .....	514
30.4.8 调试运行 .....	514
30.5 实战演练——联合使用地址簿、电子邮件、 Twitter 和地图 (双语实现: Swift 版) .....	515
30.6 使用 Messages.framework 框架 .....	515
30.6.1 Messages.framework 框架介绍 .....	515
30.6.2 实战演练——调用并使用 Messages.framework 框架 (Swift 版) .....	515
<b>第 31 章 开发通用的项目程序</b> .....	517
31.1 开发通用应用程序 .....	517
31.1.1 在 iOS 6 中开发通用应用程序 .....	517
31.1.2 在 iOS 6+ 中开发通用应用程序 .....	518
31.1.3 图标文件 .....	524
31.1.4 启动图像 .....	524
31.2 实战演练——使用通用程序模板创建通用应 用程序 (双语实现: Objective-C 版) .....	524
31.2.1 创建项目 .....	525
31.2.2 设计界面 .....	525
31.2.3 创建并连接输出 .....	526
31.2.4 实现应用程序逻辑 .....	526
31.3 实战演练——使用通用程序模板创建通用 应用程序 (双语实现: Swift 版) .....	527
31.4 实战演练——使用视图控制器 .....	527
31.4.1 创建项目 .....	527

31.4.2 设计界面	528	34.2 安装 CocoaPods	567
31.4.3 创建并连接输出口	528	34.2.1 基本安装	567
31.4.4 实现应用程序逻辑	528	34.2.2 快速安装	568
31.4.5 生成应用程序	529	34.3 使用 CocoaPods	568
31.5 实战演练——使用多个目标	529	34.3.1 在自己的项目中使用 CocoaPods	568
31.5.1 将 iPhone 目标转换为 iPad 目标	529	34.3.2 为自己的项目创建 podspec 文件	570
31.5.2 将 iPad 目标转换为 iPhone 目标	530	34.3.3 生成第三方库的帮助文档	571
31.6 实战演练——创建基于“主—从”视图的应用程序	530	34.4 实战演练——打开一个用 CocoaPods 管理的开源项目	571
31.6.1 创建项目	530	<b>第 35 章 使用扩展 (Extension)</b>	574
31.6.2 调整 iPad 界面	531	35.1 扩展 (Extension) 基础	574
31.6.3 调整 iPhone 界面	532	35.1.1 扩展的生命周期	574
31.6.4 实现应用程序数据源	533	35.1.2 扩展和容器应用的交互	575
31.6.5 实现主视图控制器	535	35.2 实战演练——使用 Photo Editing Extension (照片扩展)	575
31.6.6 实现细节视图控制器	536	35.3 实战演练——使用 TodayExtension (今日提醒扩展)	581
31.6.7 调试运行	537	35.4 实战演练——使用 Action Extension 翻译英文	583
<b>第 32 章 推服务和多线程</b>	538	35.5 实战演练——使用 Share Extension 扩展实现分享功能	586
32.1 推服务	538	<b>第 36 章 游戏开发</b>	592
32.1.1 推服务介绍	538	36.1 Sprite Kit 框架基础	592
32.1.2 推服务的机制	539	36.1.1 Sprite Kit 的优点和缺点	592
32.1.3 iOS 中 PushNotificationIOS 远程推送的主要方法	539	36.1.2 Sprite Kit、Cocos2D、Cocos2D-X 和 Unity 的选择	592
32.1.4 在 iOS 中实现远程推送通知的步骤	540	36.2 实战演练——开发一个 Sprite Kit 游戏程序	593
32.1.5 实战演练——在 iOS 系统中发送 3 种形式的通知	543	36.3 实战演练——开发一个射击游戏	601
32.2 多线程	545	<b>第 37 章 watchOS 4 智能手表开发</b>	607
32.2.1 多线程基础	545	37.1 Apple Watch 介绍	607
32.2.2 iOS 中的多线程	547	37.2 WatchKit 开发详解	608
32.2.3 线程的同步与锁	551	37.2.1 搭建 WatchKit 开发环境	608
32.2.4 线程的交互	552	37.2.2 WatchKit 架构	609
32.3 ARC 机制	553	37.2.3 WatchKit 布局	610
32.3.1 ARC 概述	553	37.2.4 Glances 和 Notifications (快速预览信息)	610
32.3.2 ARC 中的新规则	554	37.2.5 Watch App 的生命周期	611
32.4 实战演练——实现后台多线程处理 (双语实现: Objective-C 版)	554	37.3 开发 Apple Watch 应用程序	612
32.5 实战演练——实现后台多线程处理 (双语实现: Swift 版)	556	37.3.1 创建 Watch 应用	612
<b>第 33 章 Touch ID 详解</b>	557	37.3.2 创建 Glance 界面	612
33.1 开发 Touch ID 应用程序	557	37.3.3 自定义通知界面	612
33.1.1 Touch ID 的官方资料	557	37.3.4 配置 Xcode 项目	613
33.1.2 开发 Touch ID 应用程序的步骤	558	37.4 实战演练——实现 AppleWatch 垂直列表界面布局	615
33.2 实战演练——使用 Touch ID 认证	559	37.5 实战演练——演示 AppleWatch 的日历事件	616
33.3 实战演练——使用 Touch ID 密码和指纹认证	560		
33.4 实战演练——Touch ID 认证的综合演练	564		
<b>第 34 章 使用 CocoaPods 依赖管理</b>	567		
34.1 使用 CocoaPods 基础	567		

37.6 实战演练——在手表中控制小球的移动	620
37.7 实战演练——实现一个倒计时器	621
<b>第 38 章 HealthKit 健康应用开发</b>	623
38.1 HealthKit 基础	623
38.1.1 HealthKit 介绍	623
38.1.2 市面中的 HealthKit 应用现状	623
38.1.3 接入 HealthKit 的好处	624
38.2 HealthKit 开发基础	624
38.2.1 HealthKit 开发要求	624
38.2.2 HealthKit 开发思路	625
38.3 实战演练——读写 HealthKit 数据信息	626
38.4 实战演练——心率检测 (Swift 版)	626
38.5 实战演练——获取行走的步数	629
38.6 实战演练——获取步数、跑步距离、体重和身高 (Swift 版)	630
<b>第 39 章 在程序中加入 Siri 功能</b>	632
39.1 Siri 基础	632
39.1.1 iOS 中的 Siri	632
39.1.2 HomeKit 中的 Siri 指令	632
39.2 在 iOS 应用程序中使用 Siri	633
39.2.1 iOS 对生态整合与 Extension 开发的努力	633
39.2.2 Siri 功能将以 Extension 扩展的形式存在	633
39.2.3 创建 Intents Extension	634
39.3 实战演练——在 iOS 程序中使用 Siri	638
39.4 实战演练——在支付程序中使用 Siri (Swift 版)	641
<b>第 40 章 开发 tvOS 程序</b>	645
40.1 tvOS 开发基础	645
40.1.1 tvOS 系统介绍	645
40.1.2 tvOS 开发方式介绍	645
40.1.3 打开遥控器的模拟器	646
40.2 使用 Custom App 方式	646
40.2.1 Custom App 方式介绍	646
40.2.2 实战演练——开发一个简单的按钮响应程序 (Swift 版)	646
40.2.3 实战演练——开发一个猜谜游戏 (Swift 版)	647
40.2.4 实战演练——在 tvOS 中使用视图 (Swift 版)	649
40.3 使用 TVML Apps 方式	650
40.3.1 使用 TVML Apps 方式开发	651
40.3.2 实战演练——开发一个可响应的 tvOS 程序 (Swift 版)	659
40.3.3 实战演练——电影播放列表 (Swift 版)	663
<b>第 41 章 使用 Apple Pay</b>	665
41.1 Apple Pay 介绍	665
41.2 Apple Pay 开发基础	665
41.2.1 Apple Pay 支付流程	665
41.2.2 配置开发环境	666
41.2.3 创建支付请求	667
41.2.4 授权支付	669
41.2.5 处理支付	671
41.3 实战演练——Apple Pay 接入应用程序	671
41.3.1 准备工作	671
41.3.2 具体实现	672
41.4 实战演练——使用图标接入 Apple Pay	676
41.5 实战演练——使用图标接入 Apple Pay (Swift 版)	678
<b>第 42 章 开发 AR 虚拟现实程序</b>	681
42.1 虚拟现实和增强现实	681
42.2 使用 ARKit	681
42.2.1 ARKit 框架基础	681
42.2.2 ARKit 与 SceneKit 的关系	682
42.2.3 ARKit 的工作原理	682
42.3 实战演练——自定义实现飞机飞行场景的 AR 效果	683
42.3.1 准备工作	683
42.3.2 具体实现	684
42.4 实战演练——实现 3 种 AR 特效捕捉功能	686
42.4.1 实现水平捕捉功能	686
42.4.2 实现飞机随镜头飞行效果	688
42.4.3 实现环绕飞行效果	688
42.5 实战演练——实现 5 种 AR 特效 (Swift 版)	689
<b>第 43 章 tvOS 电影库系统</b>	695
43.1 tvOS 电影库系统介绍	695
43.2 系统介绍	695
43.3 使用 Objective-C 实现	697
43.4 使用 Swift 实现	703
43.5 系统扩展——优酷和土豆视频	703
<b>第 44 章 分屏多视图播放器</b>	704
44.1 分屏多视图系统介绍	704
44.2 创建工程	704
44.3 分屏具体实现	705
44.3.1 实现主视图界面	705
44.3.2 显示某个视频的基本信息	708
44.3.3 播放视频	709
44.3.4 播放网页嵌入式视频	711

iOS是一个强大的系统，被广泛地应用于苹果公司的系列产品iPhone、iPad和iTouch设备中。iOS通过这些移动设备展示了多点触摸、在线视频以及众多内置传感器的界面。本章将带领大家认识iOS这款系统，为读者步入本书后面知识的学习打下基础。

## 1.1 iOS 系统介绍

 知识点讲解光盘:视频\知识点\第1章\iOS系统介绍.mp4

iOS是由苹果公司开发的手持设备操作系统。苹果公司最早于2007年1月9日的Mac World大会上公布这个系统，最初是设计给iPhone使用的，后来陆续用到iPod touch、iPad以及Apple TV等苹果产品上。iOS与苹果的Mac OS X操作系统一样，本来这个系统名为iPhone OS，直到2010年6月7日WWDC大会上才宣布改名为iOS。

### 1.1.1 iOS 发展史

iOS最早于2007年1月9日的苹果Mac World大会上公布，随后于同年的6月发布第一版iOS操作系统，当初的名称为“iPhone运行OS X”。

2007年10月17日，苹果公司发布了第一个本地化iPhone应用程序开发包（SDK）。

2008年3月6日，苹果发布了第一个测试版开发包，并且将“iPhone runs OS X”改名为“iPhone OS”。

2008年9月，苹果公司将iPod touch的系统也换成了“iPhone OS”。

2010年2月27日，苹果公司发布iPad，iPad同样搭载了“iPhone OS”。

2010年6月，苹果公司将“iPhone OS”改名为“iOS”，同时还获得了思科iOS的名称授权。

2010年第四季度，苹果公司的iOS占据了全球智能手机操作系统26%的市场份额。

2011年10月4日，苹果公司宣布iOS平台的应用程序已经突破50万个。

2012年6月，苹果公司在WWDC 2012上推出了全新的iOS 6，提供了超过200项新功能。

2013年6月10日，苹果公司在WWDC 2013上发布了iOS 7，去掉了所有的仿实物化，整体设计风格转为扁平化设计。

2014年6月3日，苹果公司在WWDC 2014开发者大会上正式发布了全新的iOS 8操作系统。

2015年6月9日，苹果公司在WWDC 2015开发者大会上发布了全新的iOS 9操作系统。

2016年6月13日，苹果开发者大会WWDC在旧金山召开，会议宣布iOS 10的测试版将在2016年夏天推出，正式版将在秋季发布。

2017年6月6日，苹果公司在圣何塞McEnery会议中心召开了WWDC2017全球开发者大会，会上发布了iOS 11系统的测试版本，正式版于2017年秋季发布。

### 1.1.2 全新的版本——iOS 11

2017年6月6日，苹果公司在圣何塞McEnery会议中心召开了WWDC2017全球开发者大会，5000名



开发者来到了发布会现场。本次发布会全程秉承“只发产品不废话”的原则，节奏极快。本次大会上苹果公司正式公布了最新版iOS系统版本iOS 11，并在随后开放了iOS 11 beta1开发者预览版下载。iOS 11系统最突出的新特性如下。

### (1) 黑暗模式 (Dark Mode)

如果仅仅是“夜间模式”的功能，相信大家应该都在第三方的阅读APP上用过。开启此功能之后，系统的主色调变黑，文章反白，不仅能缓解阅读疲劳，同时还能够提供不错的护眼效果。

### (2) 更强大的Siri

无论是目前最火的人工智能，还是与苹果其他设备的整合功能，苹果都让Siri变得更加强大和智能。比如给出餐厅建议并发送导航或推送用户感兴趣的内容、创建对话，而且还将有望引入人声验证功能、Siri数据同步到iCloud，从而与Google Assistant和亚马逊Alexa抗衡。

### (3) P2P支付 (类似微信红包)

也许是来自微信的灵感，苹果iOS 11中正在测试P2P支付业务，将其整合到iOS 11上，并重新设计钱包 (Wallet) 应用，使得用户可通过Apple Pay或iMessage应用等转账给其他人，像发红包一样给好友转账。

### (4) FaceTime多人聊天

FaceTime将新增群组通话功能，支持最多5人视频通话，向Skype、Facebook Messenger和Google Hangouts靠齐。FaceTime Audio也将成为默认预设的拨号或呼叫方式，就像优先发送iMessage一样，优先使用FaceTime Audio通话，从而获得高品质、高速网络的通话体验。

### (5) ARKit

iOS SDK 11中Apple给开发者，特别是AR相关的开发者带来了一个很棒的礼物，那就是ARKit。ARKit利用单镜头和陀螺仪，在对平面的识别和虚拟物体的稳定上做得相当出色。ARKit极大降低了普通开发者玩AR的门槛，也是Apple现阶段用来抗衡VR的选项。

## 1.2 开始 iOS 11 开发之旅

### 知识点讲解光盘:视频\知识点\第1章\开始iOS 11开发之旅.mp4

要想成为一名iOS开发人员，首先需要拥有一台计算机，并运行苹果的操作系统。对于iOS 11开发人员来说，需要安装最新的MacOS 10.13系统。硬盘至少有6GB的可用空间，开发系统的屏幕越大，就越容易营造高效的工作空间。对于广大读者来说，还是建议购买一台Mac机器，因为这样的开发效率更高，也避免一些因为不兼容所带来的调试错误。除此之外，还需要加入Apple开发人员计划 (Developer Program)，拥有一个Apple账号。

其实无须任何花费即可加入到Apple开发人员计划，然后下载iOS SDK (软件开发包)、编写iOS应用程序，并且在Apple iOS模拟器中运行它们。但是毕竟收费与免费之间还是存在一定的区别：免费会受到较多的限制。例如将编写的应用程序加载到iPhone中或通过App Store发布它们，需支付会员费。本书的大多数应用程序都可在免费工具提供的模拟器中正常运行，因此，接下来如何做由你决定。

---

**注意：**如果不确定成为付费成员是否合适，建议读者先不要急于成为付费会员，而是先成为免费成员，在编写一些示例应用程序并在模拟器中运行它们后再升级为付费会员。但是，模拟器不能精确地模拟移动传感器输入和GPS数据等。

---

如果读者准备选择付费模式，付费的开发人员计划提供了两种等级：标准计划 (99美元) 和企业计划 (299美元)，前者适用于要通过App Store发布其应用程序的开发人员，而后者适用于开发的应用程序要在内部 (而不是通过App Store) 发布的大型公司 (雇员超过500)。

---

**注意：**无论是公司用户还是个人用户，都可选择标准计划 (99美元)。在将应用程序发布到App Store时，如果需要指出公司名，则在注册期间会给出标准的“个人”或“公司”计划选项。

---