

TURING 图灵程序设计丛书

Apress®



中文版累计销量超过8万册

全球数百万开发者交口称赞的iOS开发圣经

精通iOS开发

(第8版)

【美】Molly Maskrey 【英】Kim Topley 【美】David Mark 著
【瑞典】Fredrik Olsson 【美】Jeff LaMarche
周庆成 译



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

精通iOS开发

(第8版)

【美】Molly Maskrey 【英】Kim Topley 【美】David Mark 著

【瑞典】Fredrik Olsson 【美】Jeff LaMarche

周庆成 译



Beginning iPhone Development with Swift 3
Exploring the iOS SDK, Third Edition

人民邮电出版社
北京

图书在版编目 (CIP) 数据

精通iOS开发 : 第8版 / (美) 莫莉·马斯克里
(Molly Maskrey) 等著 ; 周庆成译. — 北京 : 人民邮
电出版社, 2017. 7
(图灵程序设计丛书)
ISBN 978-7-115-45924-4

I. ①精… II. ①莫… ②周… 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130673号

内 容 提 要

本书是 iOS 应用开发基础教程, 内容翔实, 语言生动。作者结合大量实例, 使用 Swift 语言循序渐进地讲解了适用于 iPhone/iPad 开发的基本流程。新版介绍强大的 iOS 10 操作系统, 涵盖 Xcode 8 的新功能, 书中所有案例全部重新编写。

本书具有较强通用性, iOS 开发新手可通过本书快速入门进阶, 经验丰富的 iOS 开发人员也能从中找到令人耳目一新的内容。

-
- ◆ 著 [美] Molly Maskrey [英] Kim Topley
[美] David Mark [瑞典] Fredrik Olsson
[美] Jeff LaMarche
译 周庆成
责任编辑 朱 巍
执行编辑 温 雪
责任印制 彭志环
- ◆ 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>
三河市海波印务有限公司印刷
- ◆ 开本: 880×1230 1/16
印张: 36
字数: 1186千字 2017年7月第1版
印数: 1—4 000册 2017年7月河北第1次印刷
著作权合同登记号 图字: 01-2017-4141号
-

定价: 119.00元

读者服务热线: (010)51095186转600 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

致 谢

首先我想要感谢所有给予我支持的朋友，坚持写完一本书不是件容易事，由于他们的帮助我才能够坚持写到最后。感谢 Sam、Brittany、Amanda、Kristie、Pyper、Peter、Mikes、Kelly 以及我在丹佛的 Galvanize-Platte 整个团队。

感谢科罗拉多儿童医院和行动分析中心，他们让我有幸了解了他们所做的工作对于年青人脑中风和其他行走姿势失常研究的重要性。在这里了解的知识可以让我集中力量去帮助那些需要的人。

感谢 Global Tek Labs 的客户还有伙伴，允许我在本书中引用一些他们的项目作为教学展示。感谢过去一年陪我探讨并向我提供一些建议的许多朋友，比如 John Haley 就曾与我分享他个人对于 Xcode 中自动布局功能不甚理解的苦恼。这些实际经验能够帮助我确认重要问题。

最后我想要感谢本书之前版本的所有作者。正是由于你们为本书建立了扎实的基础，我所做的微小工作才能让更多人看到。谢谢大家！

译者序

在 2016 年美国旧金山召开的 WWDC (Worldwide Developers Conference, 苹果全球开发者大会) 上, 苹果公司发布了全新的桌面操作系统 macOS Sierra 和移动操作系统 iOS 10。iOS 10 改变了以往的使用习惯, 提升了用户体验, 沿用了扁平化界面设计风格并增强了系统的稳定性, 在功能上进行了诸多改进与优化。此外, iOS 10 系统还开放了更多的 API 接口, 以增强未来应用程序的功能并减少开发人员的工作量。

在此次开发者大会中, 苹果公司推出了 Swift 3 的预览版本。之后在秋季发布会 Swift 3 正式版与 Xcode 8 一同面世, 与之一同发布的还有 iPhone 7 和 iPhone 7 Plus。Swift 是苹果公司于 2014 年发布的全新开发语言, 以逐步替代 Objective-C。在此之前, 苹果平台推荐的开发语言一直是 Objective-C。它是基于 C 语言的扩展, 由苹果公司负责维护的面向对象语言, 出现时间比 C++ 还要早。随着 OS X 系统与 iOS 平台的不断发展, 之后越来越多的移动开发者开始学习这门语言, 因而其市场份额不断增大, 排名也一度超越 C++、C#、JavaScript 等主流语言。在 2011 年与 2012 年, Objective-C 凭借快速上升的使用率连续两次赢得了 TIOBE 的年度编程语言大奖。

如今苹果公司又推出了 Swift 语言, 逐步取代 Objective-C, 必然有其重大意义。尽管 Objective-C 是一种非常神奇的语言, 甚至比互联网的年龄还要长, 不过这个已为苹果贡献多时的编程语言, 想要上手却并不容易, 对新程序员来说更是如此。初学者难以掌握它的主要原因在于, Objective-C 的语法风格独特, 看起来和其他的 C 系语言大相径庭。

为了方便在苹果平台上的学习, 降低开发应用的负担, 苹果公司推出了全新的 Swift 语言。Swift 语言的主要目标是简单易学、便捷高效, 其执行速度比 Python 和 Objective-C 程序更快。Swift 还与 Xcode 8 的 Playground 功能配合, 可以像脚本语言一样, 具有实时预览功能。开发者在输入代码后, 右边屏幕会实时显示代码效果。此外, 开发者还可以看到应用程序运行过程中的代码效果, 从而使测试过程更加方便。

Swift 语言一经推出, 便立刻引起了全世界开发者的关注, 流行程度与使用数量一直稳步上升。Swift 汲取了许多语言的特点并加以改进。对于编程老手来说, 适应这种新的语言非常容易; 而对于新人来说, Swift 的风格更加友好。开发者问答网站 Stack Overflow 的调查显示, 苹果公司的 Swift 语言以极大的优势成为了最受开发者欢迎的编程语言。

苹果公司对于 Swift 语言的未来发展仍有很高的期待。仅仅一年之后, 在 2015 年的 WWDC 上, 苹果公司不仅发布了升级版的 Swift 2, 还宣布将 Swift 语言开源, 以接纳开源社区中更多开发者的建议, 共同对其进行语法上的完善和效率上的提升。苹果公司向广大开发者伸出了橄榄枝, 这一重要举措引起了业界的一致好评。它意味着开发者将更容易掌握这门强大的编程语言, 并且拓展了更广泛的实用方向。Swift 还有很多发展的空间, 在苹果公司与开源社区的共同努力下, Swift 将会为开发者提供一种全新的体验。在未来, 将会有更多的人去接受这一语言, 全球开发者也将会越来越重视苹果平台上的应用开发。

这一次开发者大会, 苹果公司还发布了 iPad 版的 Swift Playground 学习应用。Swift Playground 针对的不仅是开发从业人员, 也包括对编程感兴趣的青少年。晦涩枯燥的开发语言编码过程将以游戏娱乐的方式展现, 用户在解决任务通过关卡的同时, 也不自觉地学习了开发程序的基础知识, 以及如何制作一个完整的应用。这种方式能让青少年在游戏中体会编程的乐趣。在本书出版时, Swift Playground 已经支持中文, 国内无论是程序员还是编程爱好者——甚至小朋友——都能够毫无障碍地接触这种寓教于乐、老少咸宜的编程学习方式。

本书中的内容讲解和示例代码采用了 Swift 3.0 语言, 部分代码中会引用到 Objective-C 框架的功能。我们都

会对此详细讲解。如果读者刚刚开始了解 Swift 语言，可以同时阅读图灵社区出版的其他 Swift 开发图书以快速入门。而已经掌握了 Swift 的开发人员，在阅读过程中可以很快重新熟悉新版的 Swift 3。通过书中层层递进的讲解，以及富有代表性的示例应用，读者能够逐步掌握 iOS 的软件开发。Swift 将能够帮助你开发出优秀的 iOS 应用程序。

本书的原作者是经验极为丰富的开发人员，在国外这本已经是 iOS 与 Swift 系列书籍的第 3 版。这一系列的书籍在国内外的销量一直长盛不衰，有越来越多的程序员依靠其步入了 iOS 开发的殿堂。当前这一版的内容极为丰富，全面涵盖了 iOS 系统的许多功能，并进行了深入的探索。如此庞大厚重的一本书堪称 iOS 开发学习的圣经。这本书适用于所有对 iOS 应用设计感兴趣的人，无论你拥有多年的开发经验还是第一次学习编程，都可以藉由此书领略到 iOS 的魅力。希望你能够通过本书的学习，开启通往熟练驾驭 iOS 应用设计的旅程。

在翻译本书的过程中遇到了许多困难，在此要感谢所有给予过我帮助的人。感谢我的家人给予我精神上的支持，感谢那些替我分担了压力的朋友，还有工作上的伙伴，尤其感谢我的同事 Simon 用他丰富的英语经验帮助我解决了很多难题。也要感谢图灵公司参与了本书编辑与校对等工作的每个人，正是由于你们的努力才能保证它的质量。尽管如此，书中依然难免有疏漏之处，希望读者能够包涵并向我们提出宝贵的建议，也希望你们的应用能够早日登上 App Store。

目 录

第 1 章 欢迎来到 iOS 和 Swift 世界	1	3.5 小结	62
1.1 关于本书	1	第 4 章 更丰富的用户界面	63
1.2 必备条件	2	4.1 动态控件、静态控件和被动控件	66
1.2.1 选择开发者计划	3	4.2 创建 Control Fun 应用程序	66
1.2.2 必备知识	5	4.3 实现图像视图和文本框	67
1.2.3 iOS 应用程序的特点	5	4.3.1 添加图像视图	67
1.3 本书内容	8	4.3.2 调整图像视图的大小	69
1.3.1 新版增加内容	9	4.3.3 设置视图属性	71
1.3.2 Swift 和 Xcode 版本	9	4.3.4 添加文本框	72
1.3.3 准备好了吗	9	4.3.5 添加约束	78
第 2 章 创建第一个 App	10	4.3.6 创建并关联输出接口	79
2.1 创建“Hello, World!”项目	10	4.4 关闭键盘	80
2.1.1 Xcode 项目窗口	13	4.4.1 按下 Done 按钮关闭键盘	81
2.1.2 深入了解项目	20	4.4.2 触摸背景关闭键盘	82
2.2 Interface Builder 简介	21	4.4.3 添加滑动条和标签	83
2.2.1 文件格式	22	4.4.4 创建并关联操作方法和输出接口	85
2.2.2 storyboard	22	4.4.5 实现操作方法	85
2.2.3 实用工具	23	4.5 实现开关、按钮和分段控件	86
2.2.4 在视图中添加标签	24	4.5.1 添加两个带标签的开关	87
2.2.5 属性修改	27	4.5.2 为开关创建并关联输出接口和操作方法	87
2.3 画龙点睛——美化 iPhone 应用程序	28	4.5.3 实现开关的操作方法	88
2.4 启动界面	32	4.5.4 控件状态	90
2.5 在设备上运行应用	33	4.5.5 为按钮创建并关联输出接口和操作方法	91
2.6 小结	36	4.6 实现分段控件的操作方法	91
第 3 章 基本的用户交互	37	4.7 实现操作表单和警告视图	92
3.1 MVC 模式	37	4.7.1 显示操作表单	92
3.2 创建 ButtonFun 应用	38	4.7.2 显示警告视图	95
3.3 视图控制器	39	4.8 小结	96
3.3.1 输出接口和操作方法	40	第 5 章 自动旋转	97
3.3.2 精简视图控制器代码	41	5.1 自动旋转机制	98
3.3.3 设计用户界面	42	5.1.1 点、像素和 Retina 显示屏	98
3.3.4 测试 ButtonFun 应用	49	5.1.2 控制旋转	98
3.3.5 布局预览	57	5.2 创建 Orientations 项目	99
3.3.6 改变文本样式	59	5.2.1 应用级支持的方向	99
3.4 应用程序委托	60		

5.2.2 独立控制器的旋转支持	101	7.6.3 实现控制器	179
5.3 创建布局项目	102	7.6.4 最后的细节	182
5.3.1 覆盖默认的约束	105	7.7 小结	185
5.3.2 与屏幕等宽的标签	107	第8章 表视图简介	186
5.4 创建自适应布局	109	8.1 表视图基础	186
5.4.1 创建 Restructure 应用程序	109	8.1.1 表视图和表视图单元	186
5.4.2 设置 iPhone 的横向 (wC hC) 配置 布局	116	8.1.2 分组表和连续表	187
5.4.3 设置 iPad (以及 iPhone Plus 横向) 的 wR hR 配置	124	8.2 实现一个简单表	188
5.5 小结	131	8.2.1 设计视图	188
第6章 创建多视图应用	132	8.2.2 实现控制器	190
6.1 多视图应用的常见类型	132	8.2.3 添加一个图像	193
6.2 多视图应用的体系结构	135	8.2.4 表视图单元样式	194
6.2.1 根控制器	137	8.2.5 设置缩进级别	197
6.2.2 内容视图剖析	137	8.2.6 处理行的选择	197
6.3 构建 View Switcher 项目	137	8.2.7 更改字体大小和行高	199
6.3.1 重命名视图控制器	137	8.3 定制表视图单元	200
6.3.2 添加内容视图控制器	139	8.4 实现自定义表视图应用程序	201
6.3.3 修改 SwitchingViewController.swift	140	8.4.1 创建 UITableViewCell 子类	201
6.3.4 创建拥有工具栏的视图	140	8.4.2 从 XIB 文件加载 UITableViewCell	205
6.3.5 连接工具栏按钮和视图控制器	142	8.5 分区分区和索引分区	210
6.3.6 编写根视图控制器	143	8.5.1 构建视图	210
6.3.7 实现内容视图	147	8.5.2 导入数据	211
6.3.8 过渡动画效果	150	8.5.3 实现控制器	212
6.4 小结	152	8.5.4 添加索引	214
第7章 分页栏与选取器	153	8.5.5 添加搜索栏	215
7.1 Pickers 应用程序	153	8.5.6 视图调试器	220
7.2 委托和数据源	156	8.6 小结	222
7.3 创建 Pickers 应用程序	157	第9章 表视图中的导航控制器	223
7.3.1 创建视图控制器	157	9.1 导航控制器基础	223
7.3.2 创建分页栏控制器	158	9.1.1 栈的概念	223
7.3.3 首次模拟器测试	161	9.1.2 控制器栈	224
7.3.4 实现日期选取器	162	9.2 简单的字体浏览器: Fonts	225
7.4 实现单滚轮选取器	164	9.2.1 Fonts 应用的子控制器	225
7.4.1 构建视图	164	9.2.2 Fonts 应用的基础框架	227
7.4.2 将控制器实现为数据源和委托	168	9.2.3 创建根视图控制器	230
7.5 实现多滚轮选取器	170	9.2.4 初始化 storyboard	232
7.5.1 构建视图	170	9.2.5 第一个子控制器: 字体列表视图	233
7.5.2 实现控制器	170	9.2.6 设计字体列表的 storyboard	235
7.5.3 滚轮内容根据环境变化	172	9.3 创建字体尺寸视图控制器	237
7.6 使用自定义选取器创建一个简单游戏	178	9.3.1 设计字体尺寸视图控制器的 storyboard	238
7.6.1 编写控制器头文件	178	9.3.2 对字体列表视图控制器的转场 进行设置	238
7.6.2 构建视图	178		

9.3.3 创建字体信息视图控制器	239	13.2 文件保存方案	308
9.3.4 设计字体信息视图控制器的 storyboard	240	13.2.1 单文件持久化	308
9.3.5 调整字体列表视图控制器的转场	243	13.2.2 多文件持久化	309
9.3.6 我的收藏字体	244	13.3 属性列表	309
9.3.7 补充功能	244	13.3.1 属性列表序列化	309
9.3.8 实现轻扫删除	244	13.3.2 创建 Persistence 应用程序的第一个版本	310
9.3.9 实现拖动排序	246	13.4 对模型对象进行归档	314
9.4 小结	247	13.4.1 遵循 NSCoding 协议	314
第 10 章 集合视图	248	13.4.2 实现 NSCopying 协议	315
10.1 创建 DialogViewer 项目	248	13.4.3 对数据对象进行归档和取消归档	316
10.1.1 配置自定义单元	249	13.4.4 归档应用	316
10.1.2 配置视图控制器	252	13.4.5 使用 iOS 嵌入的 SQLite3	318
10.1.3 提供内容单元	252	13.4.6 创建或打开数据库	319
10.1.4 实现流动布局	254	13.4.7 使用绑定变量	320
10.1.5 实现标题视图	255	13.5 创建 SQLite3 应用程序	321
10.2 小结	257	13.6 使用 Core Data	325
第 11 章 iPad 应用中的分割视图和浮动窗口	258	13.6.1 实体和托管对象	326
11.1 创建基于 UISplitViewController 的分割视图应用程序	260	13.6.2 Core Data 应用	329
11.1.1 在 storyboard 中定义结构	262	13.6.3 修改 AppDelegate.swift 文件	332
11.1.2 使用代码定义功能	263	13.7 小结	338
11.1.3 Master-Detail 模板应用程序的工作原理	266	第 14 章 文档和 iCloud	339
11.1.4 添加总统信息	268	14.1 使用 UIDocument 管理文档存储	340
11.1.5 创建浮动窗口	272	14.1.1 构建 TinyPix	340
11.2 小结	277	14.1.2 创建 TinyPixDocument 类	340
第 12 章 应用设置和用户默认设置	278	14.1.3 主控制器代码	342
12.1 设置捆绑包入门	278	14.1.4 设置 storyboard	348
12.2 Bridge Control 应用程序	279	14.1.5 创建 TinyPix 视图的类	350
12.2.1 创建项目	282	14.1.6 设计 storyboard 的详情视图	353
12.2.2 使用设置捆绑包	283	14.2 添加 iCloud 支持	356
12.2.3 读取应用中的设置	295	14.2.1 创建配置描述文件	356
12.2.4 在应用中修改默认设置	298	14.2.2 如何查询	359
12.2.5 注册默认值	300	14.2.3 保存位置	360
12.2.6 保证设置有效	300	14.2.4 将偏好设置保存到 iCloud	361
12.2.7 切换到 Settings 应用程序	302	14.3 小结	363
12.3 小结	303	第 15 章 用 Grand Central Dispatch 进行多线程编程	364
第 13 章 数据持久化基础知识	304	15.1 创建 SlowWorker 应用程序	365
13.1 应用程序的沙盒	304	15.1.1 线程基础知识	367
13.1.1 获取 Documents 和 Library 目录	307	15.1.2 工作单元	368
13.1.2 获取临时目录	308	15.1.3 GCD: 底层队列	368
		15.1.4 改进 SlowWorker	369
		15.2 后台处理	372
		15.3 应用生命周期	373

15.4 状态更改通知	373	18.2.1 响应事件	439
15.4.1 创建 State Lab 项目	374	18.2.2 转发事件：保持响应者链的活动状态	440
15.4.2 探索执行状态	375	18.3 多点触控体系结构	440
15.4.3 利用执行状态更改	376	18.4 4个手势通知方法	440
15.4.4 处理不活跃状态	377	18.5 TouchExplorer 应用	441
15.4.5 处理后台状态	380	18.5.1 创建 Swipes 应用程序	445
15.4.6 进入后台时保存状态	382	18.5.2 使用触摸事件检测轻扫	445
15.5 小结	385	18.5.3 自动手势识别	447
第 16 章 图形与绘制	386	18.5.4 实现多指轻扫	448
16.1 Quartz 2D 基础概念	386	18.5.5 检测多次轻点	450
16.2 Quartz 2D 绘图方法	387	18.5.6 检测捏合和旋转	453
16.2.1 Quartz 2D 图形环境	387	18.6 小结	456
16.2.2 坐标系统	388	第 19 章 确定位置	457
16.2.3 指定颜色	389	19.1 位置管理器	457
16.2.4 在环境中绘制图像	390	19.1.1 设置期望精度	457
16.2.5 绘制形状：多边形、直线和曲线	390	19.1.2 设置距离筛选器	458
16.2.6 Quartz 2D 样例：图案、渐变色、虚线图	391	19.1.3 获取使用定位服务的权限	458
16.3 QuartzFun 应用程序	392	19.1.4 启动位置管理器	458
16.3.1 构建 QuartzFun 应用程序	392	19.1.5 合理使用位置管理器	458
16.3.2 添加 Quartz 2D 绘制代码	398	19.2 位置管理器委托	459
16.3.3 优化 QuartzFun 应用程序	402	19.2.1 获取位置更新	459
16.4 小结	404	19.2.2 使用 CLLocation 获取纬度和经度	459
第 17 章 SpriteKit 制作简单游戏	405	19.2.3 错误通知	461
17.1 创建 TextShooter 应用	405	19.3 创建 WhereAmI 应用程序	461
17.1.1 自定义初始场景	408	19.3.1 更新位置管理器	465
17.1.2 玩家移动	411	19.3.2 将移动路线展现在地图上	467
17.1.3 创建敌人	415	19.3.3 更改定位服务权限	470
17.1.4 在场景中加入敌人	416	19.4 小结	471
17.1.5 开始射击	417	第 20 章 设备方向与动作	472
17.1.6 物理攻击敌人	420	20.1 加速计物理特性	472
17.1.7 完成关卡	421	20.2 陀螺仪旋转特性	473
17.1.8 自定义碰撞	423	20.3 Core Motion 和动作管理器	473
17.1.9 粒子系统	426	20.3.1 创建 MotionMonitor 应用程序	473
17.1.10 向场景中加入粒子	428	20.3.2 主动动作访问	476
17.1.11 游戏结束	430	20.3.3 陀螺仪和方位结果	478
17.1.12 创建开始场景	432	20.3.4 加速计结果	479
17.1.13 添加音效	434	20.4 检测摇动	480
17.1.14 添加力场：为游戏提高一些难度	434	20.4.1 内嵌的摇动检测	480
17.2 小结	437	20.4.2 摇动与破碎	480
第 18 章 轻点、触摸和手势	438	20.5 将加速计用作方向控制器	482
18.1 多点触控术语	438	20.5.1 Ball 应用程序	482
18.2 响应者链	439	20.5.2 实现 BallView 类	484

20.5.3 计算弹珠运动.....	486	22.2 字符串文件	500
20.6 小结	488	22.2.1 字符串文件的格式.....	500
第 21 章 摄像头和照片图库	489	22.2.2 本地化的字符串函数.....	501
21.1 图像选取器和 UIImagePickerController.....	489	22.3 创建 LocalizeMe 应用.....	502
21.1.1 图像选取器控制器.....	489	22.3.1 本地化项目.....	505
21.1.2 实现图像选取器控制器委托.....	491	22.3.2 本地化 storyboard.....	508
21.2 设计 Camera 界面	492	22.3.3 创建并本地化字符串文件.....	512
21.2.1 隐私选项	494	22.3.4 应用显示名称的本地化.....	516
21.2.2 实现摄像头视图控制器.....	495	22.3.5 添加其他本地化.....	517
21.3 小结	498	22.4 小结	517
第 22 章 本地化翻译应用.....	499	22.5 全书总结	518
22.1 本地化体系结构.....	499	附录 A Swift 简介	519

第1章

欢迎来到iOS和Swift世界

开发者为苹果移动设备编写应用可以得到一笔不菲的收入。应用不仅可以改变人们的生活（如图1-1所示），还可以让你有机会结识志同道合的朋友。在学习语言、工具和流程的过程中，难免会遇到一些困难，克服这些困难不但可以让你了解崭新的iOS世界，还可以让你超越平凡，成就自我。



图1-1 作为iOS开发者最棒的感受之一，就是在现实世界中看到你的作品被他人使用

现在请把我当作与你一同踏上iOS探索之旅的伙伴。我非常荣幸可以通过本书作为起点帮助你走进iOS开发的世界，为iPhone、iPod Touch或iPad开发应用。iOS是一个令人赞叹的平台，自从2007年面世以来发展势头不减。移动设备的快速增长意味着身处各地的人们都在使用着软件，可能是通过手机，也可能是通过穿戴设备，比如Apple Watch。对于刚入门的开发者来说，随着iOS 10、Xcode 8、Swift 3以及最新版本的iOS SDK（Software Development Kit，软件开发工具集）的发布，一切都变得更加令人兴奋，成为开发者也将更加轻松。

1.1 关于本书

本书是一本入门指导，可以帮助你开发出自己的iOS应用，旨在帮助你入门，理解iOS应用程序的运行和构建方式。

在学习的过程中，你将会创建一些简单的应用程序。每个应用程序都会涵盖某些iOS特性，并展示如何使用这些特性。如果你扎实地掌握了书中的基础知识，加上自己的创造力和恒心，再借助条理清晰的苹果公司官方文

档来扩大知识面，就可以创建出专业的iPhone和iPad应用。

注意 在整本书中，我优先讲解iPhone和iPad应用开发，因为这两种设备是我们最常用到的。并非有意忽略iPod touch，这样做只是为了方便。

提示 作者为本书创办了一个论坛：<http://forum.learncocoa.org>。可以在这里遇到志趣相投的伙伴，相互答疑解惑。请务必来看看！

1.2 必备条件

开始编写iOS应用程序之前，你需要做一些准备工作。初学者需要一台基于Intel架构的Macintosh计算机，并安装有Yosemite（OS X 10.10）、El Capitan（OS X 10.11）和Sierra（macOS 10.12）或更高版本的操作系统。任何最近上市的基于Intel架构的Macintosh计算机（台式机或笔记本）均可。当然不只是硬件，软件方面也要做好准备。只要你拥有一个Apple ID，就可以学习如何开发iOS应用程序并获取所需的软件工具。如果你拥有一台iPhone、iPad或iPod，那么你十有八九已经拥有了一个Apple ID，假如你的确没有，可以访问网站<https://appleid.apple.com/account>来创建一个。用Apple ID再访问<https://developer.apple.com/develop>这个网址，就会看到与图1-2类似的页面了。



图1-2 苹果公司的开发者中心资源集合网站

点击上方的Downloads图标可以进入主要资源页面（见图1-3），其中有当前已发布的产品，有时还能看到当前的iOS测试版本。你可以在这里找到各类文档、视频、样例代码以及所有可以指导你进行iOS应用程序开发的相关资源。请滚动到页面底部并查看前往文档（Documentation）和视频（Videos）相关内容网页的链接。你还可以找到前往苹果开发者论坛的链接，在那里你可以参与所有iOS平台以及macOS、watchOS和tvOS等平台各类相关主题的讨论。注册成为苹果开发者就可以在论坛中畅所欲言。

注意 在2016年的WWDC（全球开发者大会）中，苹果公司将OS X的系统名称改成了之前曾采用的macOS，以符合四个主要操作系统中其他平台的命名风格。

iOS开发中最重要的工具是Xcode，它是苹果公司的IDE（Integrated Development Environment，集成开发环境）。Xcode提供了一些实用工具，用于创建和调试源代码，编译应用程序，以及对应用程序进行性能调优。

可以从图1-3所示的开发者Download页面的Xcode链接中下载当前的Xcode测试版本。如果想用最新的正式版，可以通过Mac的苹果图标菜单访问Mac App Store，从中下载Xcode。

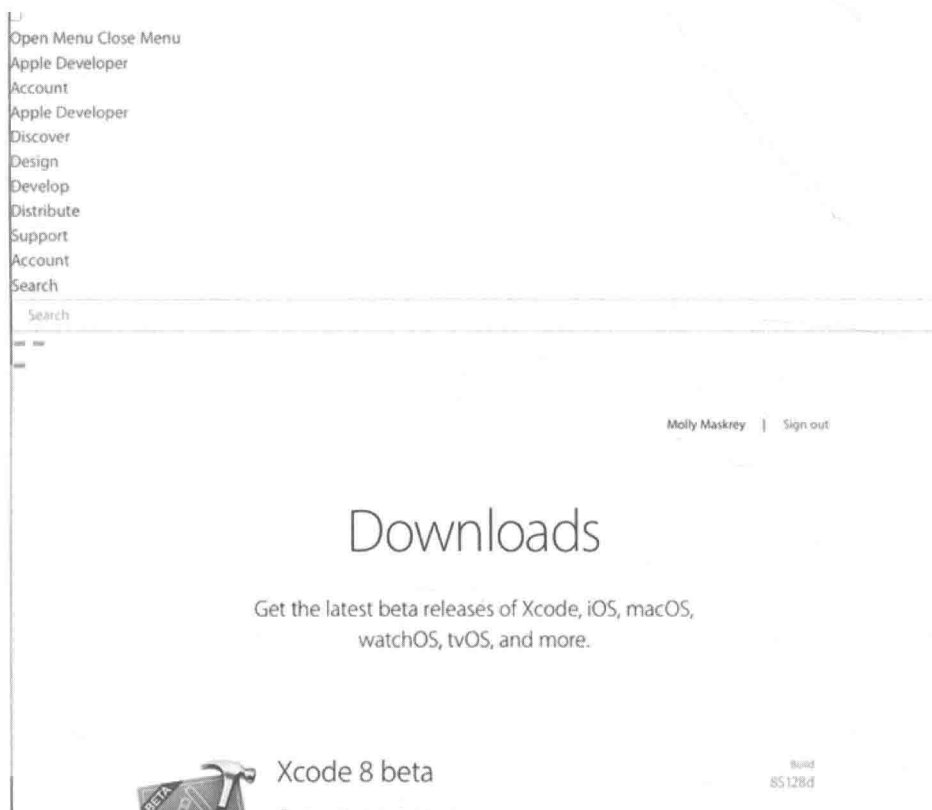


图1-3 只要用Apple ID登录，便可以从Downloads页面下载所有正式版和测试版开发工具

本书采用的SDK版本以及源代码示例

随着版本的不断更新，SDK和Xcode的下载方式也在发生变化。在最近几年内，苹果已经开始将当前较“稳定”版本的Xcode和iOS SDK放在Mac App Store中，同时在开发者网站上提供预览版供开发者下载。总之，如果想下载Xcode和iOS SDK的最新版本（即非测试版），应该使用Mac App Store。

本书面向当前最新版的Xcode和SDK。我们会在某些地方使用iOS 10中新引入的函数和方法，它们可能会与旧版SDK不兼容。

请务必从<http://www.apress.com>下载最新的源代码文档。每当有新版SDK发布时，我们会及时更新代码，你可以时常过来看看是否有变化。

1.2.1 选择开发者计划

免费下载的Xcode中包含一个模拟器，通过这个模拟器，就可以在Mac上创建并运行iPhone和iPad应用。这对于学习编写iOS程序极有帮助。不过，模拟器不支持那些需要依赖硬件的特性，比如加速计和摄像头。如果要测试使用了这些特性的应用程序，你需要iPhone、iPod touch或iPad。虽说iOS模拟器可以测试大部分的代码，但不代表所有的应用程序都可以。而且即便在模拟器上可以正常运行的应用，在决定发布出去之前仍然需要在真机上进行完整的测试。

以往版本的Xcode会要求你注册苹果开发者计划（需要付费）才可以在iPhone或其他真机上安装应用程序。但现在不这样了。从Xcode 7开始允许开发者在真机的硬件上测试应用程序，不需要购买苹果开发者计划的会员

资格（然而稍微有一些限制，之后我们会再提到）。这意味着在你的iPhone或者iPad上不需要付费就可以运行这本书中大部分的实例项目。但免费下载的SDK不支持把应用程序放到App Store上出售。如果想获取这些功能，需要从以下两个付费的开发者计划中选择一个。

- ❑ 标准版计划的价格为每年99美元。它提供了大量的开发工具和资源，以及技术支持，还可以通过苹果公司的App Store出售应用。你还可以为iOS、watchOS、tvOS和macOS开发并发布应用程序。
- ❑ 企业版计划的价格为每年299美元。如果一些公司需要开发只在企业内部使用的iOS应用程序，可以选择这个计划。

想要了解关于这些计划的详细信息，可以访问<http://developer.apple.com/programs/>（见图1-4）。如果你是一位独立开发者，通常不必购买标准版计划的会员资格。只有以下几种情况例外：你运行的应用程序使用了例如iCloud这种需要付费会员资格的功能，或者你想要在苹果开发者论坛中发帖提问，又或者你打算在App Store上部署自己的应用程序。

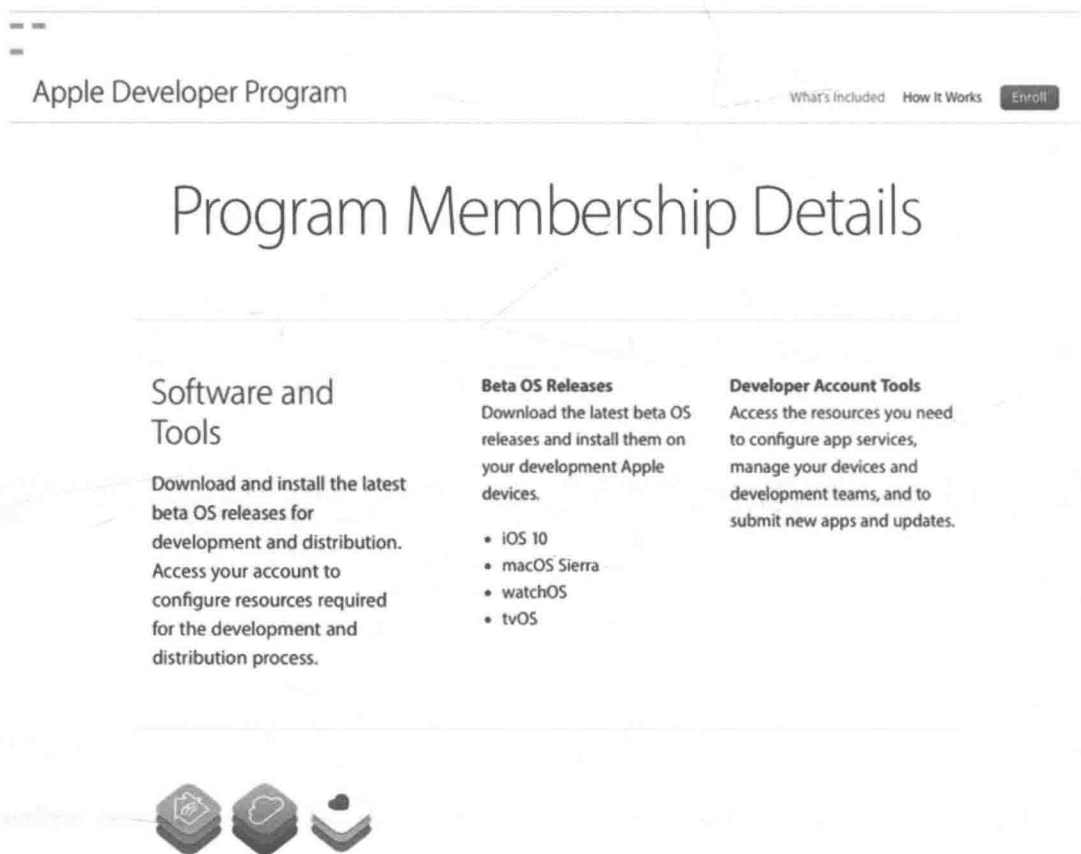


图1-4 注册成为付费会员就会获取访问测试版以及操作系统工具的权限

因为iOS支持的移动设备（比如iPhone）经常连接到其他公司的无线网络设施，所以苹果对iOS开发人员的限制比Mac开发人员严格得多（至少目前Mac开发人员完全不需要经过苹果的审查和批准就可以编写并发布程序）。虽然iPod touch和仅支持WiFi的iPad不使用其他公司的无线设施，但是它们也受到同样的限制。

苹果之所以添加这些限制，主要是为了尽量避免发布恶意程序和蹩脚程序，因为这类程序可能会在共享网络中降低性能。开发iOS应用似乎有很高的门槛，不过苹果已经为简化开发过程付出了巨大的努力。值得一提的是，99美元的价格比任何付费版本Visual Studio（微软公司的软件开发IDE）的价格低得多。

1.2.2 必备知识

我们假定本书的读者已经掌握了一些编程的常用知识，尤其是面向对象编程的内容（比如知道类、对象、循环、变量这些概念分别指什么）。不过你也许还不熟悉Swift编程语言。本书的附录会同时介绍Swift和Xcode中全新的Playground（游乐园）特性，便于让你更快地掌握这些知识。阅读完附录的内容后如果还想更多地了解Swift，最佳的方案就是直接浏览苹果公司的官方编程语言参考指南*The Swift Programming Language*，可以从iBooks商店或者iOS开发者站点下载到，网址是https://developer.apple.com/library/ios/documentation/Swift/Conceptual/Swift_Programming_Language/index.html。

你还需要以用户的身份来熟悉iOS。为了编写应用程序，你应该知道iPhone、iPad和iPod touch的细微差别以及各自的特点。利用一些时间来熟悉iOS的界面，以及苹果公司的iPhone和iPad上应用程序的视觉效果和使用体验。

一开始接触各种术语可能会让你产生困惑，我在表1-1列出了各个集成开发环境、API以及你所开发的操作系统平台所采用的编程语言之间的关系。

表1-1 平台、工具以及语言之间的关系

操作系统	IDE	API	语 言
macOS	Xcode	Cocoa	Objective-C, Swift
iOS	Xcode	Cocoa Touch	Objective-C, Swift

1.2.3 iOS 应用程序的特点

如果从未使用Cocoa写过程序，你可能会发现Cocoa Touch（用于编写iOS应用程序的应用程序框架）比较新奇。它与其他常用应用程序框架（比如用于构建.NET或Java应用程序的框架）之间存在一些根本差异。起初你可能有点不得要领，不过不必担心，只要多加练习，很快就能够对Cocoa Touch运用自如了。

注意 本书将多次提到“框架”。这个术语的意义有些难以限定，根据上下文会有不同的用法。框架是一堆“工具”的集合，其中包括了单个或多个库、脚本、用户界面元素和其他工具的集合。框架的工具通常与一些特定功能有关，比如位置服务会用到的CoreLocation框架。

如果曾经使用Cocoa写过程序，你会发现iOS SDK中有许多熟悉的身影。有很多类都是从Mac OS X版本的Cocoa中原样移植过来的。即便是那些不同的类，也遵循相同的基本原则和相似的设计模式。但是，Cocoa和Cocoa Touch之间还是有一些不同的。

无论你的知识背景如何，都需要时刻牢记iOS开发与桌面应用程序开发之间的重要差异。接下来的几个小节将讨论这些差异。

1. 通常情况下只能运行一个应用

在iOS中，一次只能有一个应用处于活动状态并显示在屏幕上。从iOS 4开始，用户按下Home键后，应用程序可以在后台继续运行，但这只限于少数情况，而且必须专门为此编写代码（第15章将告诉你如何做到）。从iOS 9开始，苹果公司加入了在同一屏幕上运行两个应用程序的功能，不过需要用户使用较新的iPad。苹果将这个特性称为多任务，第11章会介绍。

不处于活动状态或者在后台运行时，应用程序不会占用任何CPU资源，这会严重干扰其与开放式网络的连接。iOS允许后台处理，不过要使应用程序在此状况下良好运行则需要开发者自己的努力。

2. 只能使用一个窗口

在台式机和笔记本的操作系统中，多个程序可以同时运行，每个程序还可以创建并控制多个窗口。然而，除非应用程序经过特别编码，并且连接另一个屏幕或者使用AirPlay镜像，否则iOS只允许应用程序操作一个窗口。应用程序与用户的所有交互都在这个窗口中完成，而这个窗口的大小就是iOS设备屏幕的固定大小，除非你启用

了多任务功能,在这种情况下你的应用程序需要留出部分屏幕空间以运行其他应用程序。

3. 为安全起见设置访问权限

通常,用户能够访问的内容,台式机和笔记本上的程序也可以访问。然而,iOS严格限制了应用程序的访问权限。

iOS的文件系统会为每个应用分配一块独立的区域,称为沙盒(sand box)。每个应用只能读写自己沙盒内的文件。沙盒就是应用程序用于存储文档、偏好设置等任何有效数据的地方。

应用程序还会受到其他方面的限制。比如不能通过iOS访问端口号较小的网络,也不能进行在台式机中需要根用户权限或管理员权限的操作。

4. 有限的响应时间

由于使用方式特殊,iOS需要快速响应各种事件,你的应用程序也应如此。启动应用程序时,要立即打开它,载入偏好设置和数据,并把主视图显示到屏幕上。你的应用应该只有较短的延迟。

注意 我们提到的延迟指的并不是速度。速度和延迟通常可以互换,但意思并不完全等同。延迟指的是执行操作到发生结果之间的等待时间。用户按下Home按钮后,iOS就会返回主屏幕界面,你必须在应用程序退到后台之前尽快保存一切内容。如果没有在5秒之内保存必要的数据并放弃对系统资源的控制,无论是否已经保存好,应用程序进程都会被终止。然而如果你知道如何使用某个API,就可以在应用程序终止前请求多一些时间来完成必要的工作。因此通常情况下,想要快速结束也就意味着要舍弃不重要的信息。

5. 有限的屏幕尺寸

iPhone的屏幕显示效果非常出色。刚进入市场时,iPhone是当时分辨率最高的手持电子设备。不过,iPhone的显示空间并不大,比现代计算机的屏幕空间要小很多。最初几代iPhone的屏幕分辨率只有320像素×480像素,从iPhone 4的Retina屏幕开始,分辨率增加到了640像素×960像素。目前最大iPhone(即iPhone 6 Plus)的屏幕分辨率达到了1080像素×1920像素。这听起来像是个很平常的数字,不过要记住如此高密度的像素点(苹果公司用术语Retina来表示)被塞入了非常小的单位中,这对于iPhone和iPad上各种应用程序以及交互体验有重大的影响。表1-2列出了在编写本书时iOS 10支持的所有设备的屏幕尺寸。

表1-2 iOS设备屏幕尺寸

设 备	硬件尺寸	软件尺寸	精密程度
iPhone 5和5s	640×1136	320×568	2×
iPhone 6/6s	750×1334	375×667	2×
iPhone 6/6s Plus	1080×1920	414×736	3×
iPhone SE	640×1136	320×568	2×
iPad 2和iPad mini	768×1024	768×1024	1×
iPad Air、iPad Air 2、iPad Retina和iPad mini Retina	1536×2048	768×1024	2×
iPad Pro	2732×2048	1366×1024	2×

硬件尺寸指的是屏幕的实际物理尺寸,以像素为单位。不过在编写软件时只需要注意软件尺寸一栏的数字。如你所见,软件尺寸大都是实际硬件尺寸的一半。这种情况是苹果公司引入第一代Retina设备后开始出现的,其横向与纵向的像素数量都是上一代的两倍。如果苹果公司不做任何处理的话,所有现有的应用程序在新的Retina屏幕上都会只占据一半的尺寸,无法正常使用。因此苹果公司选择将应用程序绘制的所有内容都在内部乘以2,这样不需要作任何代码改动就可以将其铺满新屏幕。所有使用Retina屏幕的设备都采用了内部乘以2的机制,但iPhone 6 Plus有些例外,这是因为它拥有更高精密度的屏幕,需要乘以3。通常来说,你无需担心应用程序实际放大的倍数,只需要根据软件屏幕尺寸进行开发,剩下的事情交给iOS来做就行了。

唯一不适用于这个规则的是位图。由于位图自身的尺寸是固定的,你无法让同一张图片在Retina屏幕和非