

iOS

SDK 编程实战

何孟翰 著
武海军 改编

全面!

使用 iOS+ Xcode +
Objective-C2.0 进行开发

专业!

精辟的技巧分享，直达开发捷径

图解!

完整且清晰的系统架构、流程图及程序
运行图，拓展读者的开发思路



ios

SDK 编程实战

何孟翰 著
武海军 改编

人 民 邮 电 出 版 社
北 京

图书在版编目 (C I P) 数据

iOS SDK编程实战 / 何孟翰著. — 北京 : 人民邮电出版社, 2014. 5
ISBN 978-7-115-34831-9

I. ①i… II. ①何… 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(2014)第040731号

版 权 声 明

本书为台湾精诚资讯股份有限公司悦知文化授权人民邮电出版社出版。此中文简体字版本仅限在中华人民共和国境内(不包括香港、澳门特别行政区及台湾地区)印刷、发行及经销。

本书原版权属精诚资讯股份有限公司悦知文化。

版权所有, 侵权必究。

内 容 提 要

本书是初学者进入 iOS 开发领域的敲门砖; 是让身为程序员的读者可以早点完成工作, 早日让自己的 App 上架赚大钱; 是开发者必备的学习手册和学以致用权威的指南。

本书除了一般 iOS 程序设计书籍比较常见的基本技术, 如 Objective-C 与 MVC 设计模式的内容外, 在本书的后半段着重介绍页面的连贯与页面管理, 包括利用 Storyboard 与 View Controller 的交互, 做到多样化、层级与一致的页面呈现, Storyboard 的设计样式让设计 App 就像在讲故事, 连贯而顺畅, 这同时也是 iOS 开发最经典的地方之一。另外, 在多媒体实现、触控动画及图像处理等内容中, 读者可以从实例演练中学到 iOS 程序开发的概念与技巧, 达到事半功倍的效果, 并学会自行扩展 iOS 程序开发技能的深度和广度, 注入个人的创意, 创造出全新的价值。

本书适合 iOS 初学者阅读, 也适合程序员使用, 以及作为大专院校的师生用书和相关培训学校的教材。

-
- ◆ 著 何孟翰
改 编 武海军
责任编辑 张 涛
责任印制 程彦红 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>
北京中新伟业印刷有限公司印刷

- ◆ 开本: 787×1092 1/16

印张: 42.5

字数: 1180 千字

印数: 1—2 500 册

2014 年 5 月第 1 版

2014 年 5 月北京第 1 次印刷

著作权合同登记号 图字: 01-2012-9279 号

定价: 99.00 元

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

反盗版热线: (010)81055315



前言

两三年前，在偶然的场合认识了位好朋友，他具有强烈而显著的个人特质，除了事业经营非常成功外，对于网络营销与科技新知也保持了高敏锐度。那一阵子，常常在开始上班前和我分享新鲜事外，对于工作经验和人生也都成为彼此谈论的话题。在那段忙碌的程序开发与教学日子中，让我获得了许多不易取得的新信息与教学灵感。

约莫在一年前，我和这位好朋友在一次的会面时，得到了一个新的观念——curation，也就是在这个信息爆炸且泛滥的时代中，对于任何未知内容，只要在搜索引擎中键入关键词，几乎都会得到超过 10,000 条的数据时，作为一个内容提供者的程序设计讲师与作者，该如何定位自己的想法。curation 这个全新观念，也成为了我对于本书与之后移动程序设计系列课程规划的方针。

当然，我并非专门研究 curation，无法给它一个很明确的字面解释，但就我得到的信息和认识，curation 就是如何在庞大的信息“森林”中，按照自己的经验与见解提供导引来帮助课堂上的学生，或是正在阅读本书的读者，能更快速明确地了解程序设计的概念与彼此间的关联。也就是说，这种价值是基于信息的整理、归纳、分析与典藏（archive）后，能给人更大的影响。

按照 curation 的概念，我也一一整理了 iOS 程序的课程设计，从最基本的语言探讨到开发框架，进而介绍 iOS 应用程序的用户界面与基本 CocoaTouch 框架的应用。

在开设的 iOS 实战课程中，我对于课程的期许是，即便是不懂 C 语言或面向对象基础的工程师，在通过一个个简单具体的实例学习，都让学员在这片信息汪洋中建构起个人的知识框架，进而将与 iOS 开发相关的知识串连起来，拥有自己建构新知识的能力。

本书还是以入门为主，但除了定位成敲门砖外，更希望这本书还能像是开发 iOS 过程中，信息“森林”中的 breadcrumb，能够让你建立起相关知识的关联外，还能随着内容的进行，学会自行扩展 iOS 程序开发技能的深度和广度。

本书由 21 章组成，集合了我和学员们超过 150 个小时的授课内容，以及与许多资深工程师研习讨论得到的心得。本书维持实战系列的风格，希望通过简单而具体的示例，让读者能在最短时间内了解书中所要传达的意思，进而让大家能快速开发项目。

章节的顺序为由浅到深，并从应用程序通用的开发面开始探讨，再往单独的主题做较为深入的研究。章节本身虽具有前后的关联性，但每章的主题皆为独立。若你是已有基础的读者，想了解某些单独功能的话，不妨直接跳到该主题进行阅读。如果你想按部就班从头学起，则可以按以下的说明循序阅读。

如果你正站在书店内考虑是否要购买本书，建议你翻阅一下“第 0 章在项目开发之前”。这一章的内容是关于在开发 iOS 之前，所需要准备的工作（包含你的心情），也可能在买完本书后，你还需要采购相关的软/硬设施与参考书籍，待万事具备后，就可以开始准备“大啃”苹果了。

万丈高楼平地起，想要开发充满互动性的 iOS 应用程序必须对基础知识有相当的认识，因此，在“第 1 章 XCode 与开发环境”将会讨论主要的开发工具——Xcode；并在“第 2 章 Objective-C 与对象基础”中说明 iOS 开发所使用的语言——Objective-C；从“第 3 章 iOS 应用程序与生命周期”开始就通过项目向导实际操作，让你对窗口应用程序有更实际且深入的认识。

打好基础后，在“第 4 章 Objective-C 进阶”会讨论更多的 Objective-C 语法，例如，在真实开发情景中最常用到的 Collection 框架等。“第 5 章用户界面设计”开始讨论窗口用户界面 GUI 的设计与程序的交互。接着在“第 6 章实现多界面应用程序”中通过探索多个项目模板，开始实践页面与页面间的连贯，帮助你在“第 7 章页面管理”时能够设计出多页面的应用程序。

Mac/OSX 与 iOS 一直以优异的用户界面著称，这样的概念也延伸到 iOS/OSX 用户界面的设计上，所以在“第 8 章使用 Storyboard 进行界面设计”中会大量使用 Storyboard 组件来做界面设计，以完整直观的方式来开发 iOS 应用程序。

要成为 iOS App 的好公民，必须看懂重点部分。为了要将 App 畅销到全世界，在“第 9 章实现多

语言支持”我们会讨论多语言的实现，并在“第 10 章 iOS 应用程序的常用行为”中，告诉你如何使用 URL 方式调用应用程序的基本行为，例如，拨打电话、发送短信、地图搜寻，甚至是存取用户偏好等。至此，我们就大概具备了开发 App 的基本能力，可以进一步探讨更丰富的主题了。

如果你想开发的是公用工具（utility）类型的应用程序，就必须使用表格来存取动态数据并配合数据存储的机制。在“第 11 章表格的使用”中，我们会讨论表格组件的使用，“第 12 章使用 CoreData”会说明数据库存取框架 Core Data 的使用技术。

但随着应用程序开发的进展，应用程序的需求与逻辑也会更为复杂，我们会在“第 13 章文件存取与多线程”开始讨论文件的存取与多线程的操作细节，以便在项目开发时，能具有更多灵活性来调整应用程序的流程设计与规划。

在“第 14 章 Block 与 GCD”导入了 Block 语法。这是一个基于 C 语言的延伸，除了能让应用程序更易阅读与开发外，了解这种语法的特性也能帮助加速学习其他 iOS 应用程序开发框架的调用。在本章中，我们也会介绍 GCD（Grand Central Dispatch）这种 Apple 提供的函数调用，好让你自由地使用 Block 语法来封装代码，并传递 Block 对象到不同的线程与队列中执行。

从“第 15 章使用 CoreGraphic 绘制图形”开始，将进入充满色彩与变化的内容。在第 15 章会说明如何绘制与填充自定义组件，“第 16 章切换效果与动画”将会讨论各式各样的动画组件，通过简单的属性定义，让 iOS 的框架能处理复杂且繁琐的内插运算，呈现出各种生动且活泼的动画效果。

从“第 17 章触控”一直到“第 19 章：影音播放”，主要讨论的是与硬件相关的开发内容。例如：iDevice 最大的卖点——触控机制。除了卓越硬件的性能外，iOS 框架也提供了非常强大的功能设计，在第 17 章中我们会讨论触控、多点触控、触控的判定以及各式触控事件所形成的手势（gesture）识别。“第 18 章相机和滤镜”会说明如何应用 iDevice 的相机照相，并通过图形的滤镜转换来实现各种特效。在“第 19 章影音播放”会说明影音播放时，所需编写的代码与实际中所需注意的要点。

在最后一部分，我们会讨论 iDevice 的对外连接，在“第 20 章地图与 GPS”中会说明以 iDevice 位置为主的信息系统实现（也就是 Location BasedService，缩写 LBS），包含：地图、GPS 定位、地图上景点的定制化图层建立等。在“第 21 章常用的网络存取功能”讨论各种 iDevice 的网络存取方式，例如：存取网页、直接通过 http 协议进行数据传递等，此外，对于应用层的协议，例如：JSON 以及 Facebook 链接功能，也会一一进行介绍。

上述的内容都是由我和学员们的互动讨论延伸而来，对于容易造成误解或较难理解的部分，在书中都会详加分析。

希望你在看了目录与导读之后，能够尽快找到所需要的信息，快速地运用于自己的项目之中。

致谢

首先，要感谢王胜德教授，在进行繁忙研究之际，还愿意帮本书撰写推荐序。同时，也要谢谢王教授的研究生 Terry 与 Charles，作为本课程专业的助教，并帮忙一起解决学员 lab 的问题。以便让本书顺利付梓。

再之，要谢谢张智凯，在百忙之中拨冗撰写推荐序，并提供许多完善的建议。

本书示例文件下载地址如下：

www.toppr.net

何孟翰

目 录

第一篇 iOS 开发基础知识

第 0 章 在项目开发之前	2
0.1 知识的准备	2
0.2 软件的准备	3
0.3 硬件的准备	4
0.4 设计模式的准备	5
0.4.1 Model-View-Controller 设计模式	5
0.4.2 Delegation 设计模式	7
0.4.3 Target-Action 设计模式	8
总结	9
第 1 章 Xcode 与开发环境	10
1.1 OS X 与开发相关的特点	10
1.1.1 使用 Dock 启动应用程序 与查看状态	11
1.1.2 使用 Finder 来显示文件和 目录的结构与信息	11
1.1.3 使用 Spotlight 来快速 搜索文件	13
1.1.4 使用 terminal 来进行文本 模式的命令行操作	13
1.2 认识 Xcode 与开发环境	13
1.2.1 Xcode 的偏好设置	14
1.2.2 Xcode 的工作空间	15
1.3 开发设备管理	17
1.4 程序代码库管理	18
1.5 项目目录管理	19
1.6 存档目标管理	20
1.7 说明文件管理	21
总结	22
第 2 章 Objective-C 与对象基础	23
2.1 命令行模式项目与 Foundation 框架	23

2.2 Objective-C 的类	26
2.2.1 类的声明	27
2.2.2 类的成员变量与函数	28
2.2.3 实现类的成员函数	29
2.2.4 声明与实现的切换	30
2.2.5 函数的多参数	32
2.3 Objective-C 的对象实例	33
2.3.1 对对象发送消息	34
2.3.2 从类生成对象	35
2.4 使用静态程序分析器	37
2.4.1 静态程序代码分析	37
2.4.2 对象的管理与 retaincount (保留计数器)	38
2.4.3 释放分配的内存	39
2.5 Property (属性) 的定义	39
2.5.1 属性的声明	39
2.5.2 属性的实现	40
2.5.3 设值函数的属性	42
2.5.4 @property 能设定的 其余属性	47
2.6 Objective-C 中的字符串	47
2.6.1 字符串的声明	48
2.6.2 使用格式化字符串来动态 产生字符串的内容	49
总结	51
第 3 章 iOS 应用程序与生命周期	52
3.1 创建一个空白的 iOS 项目	52
3.1.1 使用项目向导生成 iOS 空白项目	52
3.1.2 iOS 项目的运行	54
3.1.3 委托的设计模式	56
3.1.4 委托类与应用程序生命 周期的建立	57
3.2 XIB 文件与 Model View Controller (MVC) 设计模式	65

3.2.1	使用项目向导生成单一视角应用程序 (Single View Application)	65	4.2.9	使用 NSCountedSet 来获取重复对象的数量	105
3.2.2	xib 版面配置文件与 MVC 的设计样式	67	4.2.10	使用键值存取 Dictionary 中的内容	107
3.3	Storyboard 的版面配置文件	71	4.2.11	使用 NSDictionary 定义不可变的 dictionary	107
3.3.1	应用程序的初始化与 plist 文件	72	4.2.12	使用 NSMutableDictionary 定义可变的 dictionary	108
3.3.2	Storyboard 的操作与 MVC 的设计模式	74	4.2.13	使用 NSIndexSet 来获取数组的任意索引值	109
3.4	实现 MVC 界面	77	4.2.14	在 NSMutableIndexSet 中新增区域	111
3.4.1	新增应用程序初始主界面文件	77	4.2.15	在 NSMutableIndexSet 中删除区域	111
3.4.2	在界面文件中加载 View Controller 组件	78		总结	112
3.4.3	实现定制化 View Controller 组件	80	第 5 章	用户界面设计	113
	总结	83	5.1	认识 Xcode 的界面编辑器与开发用户界面	113
第 4 章	Objective-C 进阶	84	5.2	使用 IBAction 与 IBOutlet 链接程序和用户界面	116
4.1	iOS 与协议 Protocol	84	5.2.1	连接用户界面与 IBOutlet 变量	116
4.1.1	声明协议与要实现的函数	85	5.2.2	从用户界面生成 IBOutlet 变量	121
4.1.2	由其他对象实现的函数	86	5.2.3	链接用户界面与 IBAction 函数	124
4.1.3	协议之间的引用	89	5.2.4	从用户界面产生 IBAction 函数	129
4.2	Collection 框架	91	5.3	处理文本框	132
4.2.1	CocoaTouch 中的 Collection	91	5.3.1	文本框和虚拟键盘	132
4.2.2	使用数组 (NSArray) 来存取顺序的数据	91	5.3.2	让文本框能与程序交互	134
4.2.3	使用 NSArray 来定义不可变的数组	93	5.4	设置组件的位置和大小	138
4.2.4	使用快速枚举 (FastEnumeration) 的方式获取对象	96	5.5	使用代码生成组件	139
4.2.5	使用 NSMutableArray 来定义可变大小的数组	98	5.5.1	利用代码生成文本框	140
4.2.6	使用 Set 来存取不可重复的数据	101	5.5.2	按照对象的标识符来判定不同的文本框并做出反应	142
4.2.7	使用 NSSet 定义不可变的 Set	101	5.5.3	标签的属性设定	143
4.2.8	使用 NSMutableSet 定义可变大小的集合	104	5.5.4	设定按钮组件的属性	145
			5.5.5	IBAction 和 IBOutlet 的交互	150
				总结	154

第二篇 页面的切换

第6章 实现多界面应用程序 156

- 6.1 公用程序模板应用程序 156
 - 6.1.1 新增公用程序模板应用程序 156
 - 6.1.2 从应用程序模板向导新增使用 storyboard 的公用程序 (Utility Application) 159
- 6.2 标签式模板应用程序 161
 - 6.2.1 在 xib 下新增一个标签页面 162
 - 6.2.2 在 storyboard 下新增标签页面 166
 - 6.2.3 连接 storyboard 和类文件, 用代码控制用户界面 169
 - 6.2.4 多标签的处理 170
 - 6.2.5 标签栏模板的图标处理 171
- 6.3 页面模板应用程序 174
- 6.4 具备主从关系的模板文件 177
 - 6.4.1 数据的新增/删除 180
 - 6.4.2 应用程序的生命周期 181
 - 6.4.3 主从关系的 storyboard 项目模板 182
- 总结 186

第7章 页面管理 188

- 7.1 使用 View Controller 进行页面的旋转和转置 188
 - 7.1.1 xib 中可视化组件旋转后的摆放 191
 - 7.1.2 适应不同屏幕而提供不同的 xib 组件 195
- 7.2 使用 UINavigationController (导航控制器) 进行多页面的顺序遍历 199
 - 7.2.1 导航控制器的基本操作 200
 - 7.2.2 在单一视角项目 (Single View Project) 中新增导航栏 201
 - 7.2.3 xib 中对导航栏的模拟 204
 - 7.2.4 在标签栏项目中新增导航栏 204

- 7.2.5 利用 UINavigationController 进行页面切换 206

7.3 导航栏的应用 211

- 7.3.1 使用导航栏进入 View Controller 的编辑模式 212
- 7.3.2 自定义导航栏的按钮 213
- 7.3.3 导航栏的工具栏 215
- 7.3.4 导航栏工具栏的自定义按钮 217

7.4 建立滚动条窗口组件 218

- 7.4.1 新增滚动条组件 219
- 7.4.2 在滚动条组件中新增内容 221

7.5 取得委托对象的实例 223

- 总结 225

第8章 使用 Storyboard 进行界面设计 226

- 8.1 从 xib 到 storyboard 226
- 8.2 在空白项目中使用 storyboard 227
 - 8.2.1 新增 storyboard 文件 228
 - 8.2.2 设定项目主要的 storyboard 文件 229
 - 8.2.3 storyboard 的操作 230
 - 8.2.4 连接 storyboard 和 View Controller 232
 - 8.2.5 修改默认项目并运行 233
 - 8.2.6 导航栏与 storyboard 234
 - 8.2.7 标签栏与 storyboard 236
- 8.3 storyboard 与 View Controller 的交互 237
 - 8.3.1 在场景中加入用户交互组件 237
 - 8.3.2 载入 storyboard 中单独的 View Controller 237
 - 8.3.3 载入外部 xib 的 View Controller 240
- 8.4 在 Scene 中使用 Segue 设定界面的连贯 241
 - 8.4.1 Modal 类型的 Segue 242
 - 8.4.2 Push 类型的 Segue 244
- 8.5 Segue 与数据传递 246
- 8.6 从程序中驱动 Segue 249
- 8.7 Storyboard 和静态表格 251

8.7.1	设定静态表格的类型	252
8.7.2	分栏与区域标题结尾	255
8.7.3	静态表格与 Segue	256
8.8	Segue 与动态切换	257
	总结	258

第三篇 常用的 Cocoa Touch 框架

第 9 章 实现多语言支持 260

9.1	iOS 下多语言的设定	260
9.2	多语言字符串的处理	261
9.2.1	新增字符串文件	262
9.2.2	编写字符串文件	263
9.2.3	多语言测试时的错误排除	267
9.3	从现有程序产生多语言字符串资源文件	268
9.4	使用多语言的注释	269
9.5	取得默认的语言	271
9.6	xib 文件的多语言	272
9.6.1	多语言 xib 文件的选择	272
9.6.2	多语言 xib 文件的实现	273
	总结	277

第 10 章 iOS 应用程序的常用行为 278

10.1	调用的系统 URL	278
10.1.1	调用应用程序自定义的 URLScheme	279
10.1.2	调用浏览器 Safari 浏览链接	280
10.1.3	发送 E-mail 的调用	281
10.1.4	拨打电话的调用	283
10.1.5	发送短信的调用	285
10.1.6	访问地图的调用	286
10.1.7	播放优酷视频	290
10.1.8	打开 App Store 的调用	292
10.2	操作用户偏好来存放应用程序的常用设定	293
10.2.1	认识「设定」应用程序	294
10.2.2	定义用户偏好值	294
10.2.3	用户偏好的图标	296
10.2.4	用户偏好的多语言界面	298

10.2.5	plist 文件的编辑方式	299
10.2.6	新增用户偏好的组	300
10.2.7	用户偏好的文字字段	302
10.2.8	用户偏好的选项界面	303
10.2.9	用户偏好的开关选项	305
10.2.10	用户偏好的标题选项	306
10.2.11	用户偏好的滑杆选项	308
10.2.12	用户偏好的子选项	309
10.3	读取用户偏好的设定值	312
10.3.1	使用 NSUserDefaults 类读取用户偏好	312
10.3.2	检测用户偏好的变更	314
10.3.3	使用 NSUserDefaults 类设定用户偏好	315
10.3.4	用户偏好值的调试	318
	总结	320

第 11 章 表格的使用 321

11.1	使用 UITableViewController	322
11.2	使用 datasource 定义表格的数据	326
11.2.1	定制化表格的内容	330
11.2.2	表格单元格的附属类型	333
11.2.3	表格的编辑模式	334
11.3	表格的 delegate 对象	335
11.3.1	新增表格的内容	338
11.3.2	删除表格的内容	344
	总结	346

第 12 章 使用 CoreData 347

12.1	Core Data 的架构	349
12.2	在程序中加入 Core Data 的功能	349
12.3	新增数据	355
12.4	查询数据	357
12.5	删除数据	361
12.6	修改数据	362
	总结	367

第 13 章 文件存取与多线程 368

13.1	存取应用程序目录中的文件	368
13.1.1	使用 NSBundle 存取应用程序中的文件	369

- 13.1.2 读取 Bundle 中的文字..... 371
- 13.1.3 Bundle 和 Retina 资源
的准备..... 373
- 13.2 应用程序的文件和外部的交互 376
- 13.3 使用定时组件在特定时间
执行程序..... 378
 - 13.3.1 使用 NSTimer 定时组件... 379
 - 13.3.2 只执行一次的定时器 379
 - 13.3.3 在 NSTimer 中传入参数... 382
 - 13.3.4 持续执行的定时器..... 382
 - 13.3.5 结合定时器和
Run Loop..... 384
- 13.4 使用多线程进行异步执行..... 386
 - 13.4.1 使用 NSThread 新增线程
并执行..... 386
 - 13.4.2 在线程中使用参数..... 387
- 13.5 操作对象和操作队列..... 389
 - 13.5.1 使用 NSInvocationOperation
建立操作对象..... 390
 - 13.5.2 继承 NSOperation 类建立
操作对象..... 392
 - 13.5.3 操作对象之间的依赖
关系..... 394
- 总结..... 397

第 14 章 Block 与 GCD..... 398

14.1	定义与实现 Block	398
14.1.1	从函数到 Block	399
14.1.2	typedef 和 block 变量	400
14.1.3	Block 所能存取 的变量	401
14.1.4	Block 中变量的 设置与 block 变量	404
14.1.5	使用 Block 存取静态变量	406
14.1.6	Block 的语法	407
14.2	Block 的调试	408
14.3	Block 的应用	412
14.3.1	Block 用于 viewController	412
14.3.2	在 Collection 中使用 Block	414
14.3.3	Block 用于字典的循环	416
14.3.4	Block 应用于集合 (Set) 的循环	416

14.3.5	用于数组中对象的排序	417
14.3.6	Block 用于数组中对象的过滤	420
14.4	使用 GCD 执行 block 代码	422
14.4.1	GCD 与 Block	422
14.4.2	取得主线程的队列并顺序执行	424
14.4.3	取得其他执行的队列	425
14.4.4	同步执行队列的注意事项	427
14.4.5	GCD 与并行执行	430
14.4.6	延迟执行 block	432
	总结	433

第四篇 绘图、触控与多媒体

第 15 章 使用 CoreGraphic 绘制图形 436

15.1	绘图的概念	436
15.1.1	绘图上下文	437
15.1.2	CoreGraphics 的基本类型 与运算	437
15.2	使用 UIView 进行绘图	441
15.2.1	UIView 的绘图周期	441
15.2.2	UIView 和绘图上下文	442
15.2.3	在 UIView 中绘制贝兹 路径 (BezierPath)	442
15.2.4	路径的填充和样式	446
15.2.5	绘制圆弧与曲线与路径 端点	449
15.2.6	路径和转换运算	452
15.2.7	绘制贝塞尔曲线	458
15.2.8	使用 CoreGraphics 函数 进行绘图	462
15.2.9	使用 CurrentTransformMatrix 进行绘图坐标转换	464
15.2.10	填充的顺序与色彩 的调和	466
	总结	468

第 16 章 切换效果与动画 469

16.1 使用切换动画切换画面.....469

16.1.1 单一窗口的切换动画.....469

16.1.2	从一个窗口组件切换到另一个窗口组件	472
16.2	使用 UIView 制作基础动画	476
16.2.1	取得 UIView 下的 CALayer 进行动画	476
16.2.2	同时执行两个属性的动画或先后执行两个属性的动画	479
16.2.3	动画的参数与重复循环	480
16.2.4	多重动画的使用	481
16.2.5	取得 UIView 下的 CALayer 进行动画	483
16.3	使用 CALayer 制作动画	484
16.3.1	建立 CALayer 进行不需明确声明的动画	485
16.3.2	使用 CAShapeLayer 建立几何动画图层	487
16.3.3	使用 CATextLayer 建立文字动画图层	490
16.3.4	使用 CAReplicatorLayer 建立样式重复的对象	493
16.4	使用 Core Animation 进行明确声明的动画	495
16.4.1	使用 CABasicAnimation 实现线性改变属性的动画	497
16.4.2	使用 CAMediaTiming 类进行动画属性的控制	501
16.4.3	设定动画的委托对象与委托函数控制	502
16.4.4	使用 CAKeyFrameAnimation 进行关键界面的设定	504
16.4.5	在明确的动画声明中同步使用多重动画属性	505
16.4.6	动画执行完成后的响应与顺序动画	507
16.5	在 CALayer 的 2.5 度空间制作动画	509
16.5.1	Z 轴的变换与视点的建立	511
16.5.2	三度空间的旋转	514
	总结	516

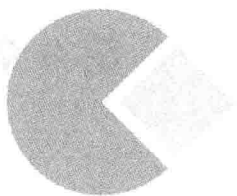
第 17 章	触控	517
17.1	触控与事件	517
17.1.1	UITouch 触控事件类	518
17.1.2	结合触控与用户界面	520
17.2	触控区域的判定	526
17.3	触控与手势	530
17.3.1	位移的手势识别	532
17.3.2	滑动的手势识别	535
17.3.3	旋转的手势识别	538
17.3.4	缩放的手势识别	539
17.3.5	点击的手势识别	542
17.3.6	长按的手势识别	544
17.3.7	识别器的委托函数	547
17.3.8	多重识别器的合并并使用	549
17.4	多点触控	550
17.5	结合触控与绘图	552
	总结	555
第 18 章	相机和滤镜	556
18.1	图片集与相片	556
18.1.1	UIImagePickerController 的初始化	556
18.1.2	使用相机功能	562
18.1.3	UIImagePickerController 的委托对象	565
18.1.4	储存相机提取的界面	569
18.2	CoreImage 与滤镜	571
18.2.1	滤镜与影像	571
18.2.2	内建的滤镜	572
18.2.3	滤镜的属性	575
18.3	滤镜的使用	582
18.3.1	使用 CIColorInvert 色彩反转滤镜	583
18.3.2	使用 CISepiaTone 深褐色调滤镜	584
18.3.3	使用 CIHueAdjust 色彩调整滤镜	585
18.3.4	使用 CIVignette 渐变滤镜	587
18.3.5	使用 CICheckerboardGenerator 产生棋盘效果	588

18.3.6 使用滤镜混合图形	591
18.4 多重滤镜的使用	593
总结	595
第 19 章 影音播放	596
19.1 声音播放	596
19.1.1 使用 AVAudioPlayer	
框架	596
19.1.2 取得声音文件的信息	599
19.1.3 播放多重音轨	603
19.1.4 AVAudioPlayer 的委托	
对象与中断处理	604
19.1.5 在后台执行音乐	606
19.2 影片播放	607
19.2.1 使用	
MPMoviePlayerController	
类播放影片	607
19.2.2 播放流媒体影音	610
19.2.3 使用 AVPlayer 类播放	
影片	614
19.2.4 影片与外接屏幕	617
总结	620

第五篇 地图与网络存取

第 20 章 地图与 GPS	622
20.1 使用 MKMapView 显示地图	622
20.1.1 地理信息相关的数据	
结构	623
20.1.2 在 MapView 中设定	
位置	625
20.1.3 在 MapView 移动	
中心点	626
20.1.4 使用 MKMapPoint 标记	
地图	628
20.1.5 地图的响应函数	630
20.2 地图的标记	631
20.2.1 建立遵循 MKAnnotation	
协议的对象	632

20.2.2 使用 MKAnnotationView	
自定义地标	634
20.2.3 设定加入产生的地标之后	
的响应函数	635
20.2.4 地标详细信息	637
20.2.5 地点标记的拖曳与移动	638
20.3 取得用户位置的标记	640
20.4 地图与定制化图层	641
20.4.1 绘制圆形的地图图层	642
20.4.2 绘制多边形线段的	
地图图层	643
20.4.3 绘制多边形的地图图层	644
20.5 使用地理信息编码	645
总结	646
第 21 章 常用的网络存取功能	647
21.1 显示网页	647
21.1.1 使用 UIWebView	647
21.1.2 使用 UIWebViewDelegate	
查看网页内容	649
21.2 同步与异步的数据存取	650
21.2.1 使用 NSURLConnection	
执行异步调用	651
21.2.2 使用 NSURLConnection	
执行同步调用	653
21.2.3 同步调用与 GCD 的异步	
执行	654
21.3 读取 JSON 格式	656
21.3.1 将 Foundation 对象转成	
JSON 对象	656
21.3.2 从网络上读取 JSON	
对象	658
21.4 整合 Facebook	660
21.4.1 下载 FacebookSDK	661
21.4.2 设定 Facebook 应用程序	
并取得 AppID	662
21.4.3 初始化 Facebook 对象	663
21.4.4 实现 Facebook 的登入	
与注销	665
总结	667



第一篇

iOS 开发基础知识

- 第 0 章 在项目开发之前
- 第 1 章 Xcode 与开发环境
- 第 2 章 Objective-C 与对象基础
- 第 3 章 iOS 应用程序与生命周期
- 第 4 章 Objective-C 进阶
- 第 5 章 用户界面设计

第0章 在项目开发之前

在开发 iOS 程序之前，首先，需了解相关准备工作，包含软件、硬件还有心情的准备。对大多数人来说，Apple 的 Mac 相关开发经验比起其他的平台来说，可能较为不足，因此，在找寻相关数据或技术文件时，常有不知如何下手的困扰。

但若能掌握基础知识，如了解相关术语或名词，大概就能将这些技术对应到已熟悉的开发框架上。若您已有其他程序的开发经验，即使目前没有开发过 iOS 或 OS X 应用程序的经验，也不会影响接下来的学习。在本章后面几小节中，我们会从较为轻松的角度（也就是还不用写代码）来认识 iOS 的开发环境与准备工作。

0.1 知识的准备

Mac OS X 是美国苹果公司麦金塔（Mac）操作系统 Mac OS 的其中一个版本。事实上，这也可能是紧接 Mac OS9 的下一代产品（因为 X 在罗马数字中代表的就是 10）。它在 2001 年问世，之后成为 Mac 最重要、也最成功的操作系统。

Mac OS X 是以 UNIX 中一个新分支 Mach 为核心，再加上 bsd 系统作为整个操作系统的主干，并集成 NextStep 后所合成的 Darwin 核心。这个 Darwin 系统加上 Aqua 的图形用户界面，便组成了 OS X 的操作系统。

OS X 的操作系统版本名称一直以猫科的动物来表示，对应的版本编号如下：

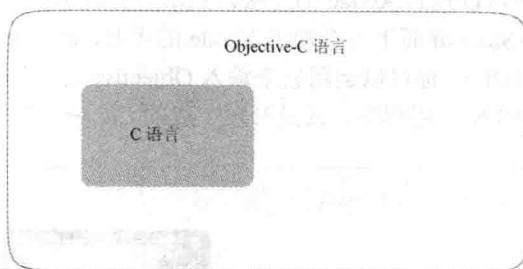
10.4 (2005)	Tiger
10.5 (2007)	Leopard
10.6 (2009)	Snow Leopard
10.7 (2011)	Lion
10.8 (2012)	Mountain Lion

在苹果官网与一般讨论区中，操作系统的版本和其动物代称常被混用，所以，在查询信息时，建议同时参照版本编号与它们对应的代码。

OS X 中的 Aqua 是一项极引人注目的设计。它优异的色彩和屏幕呈现再配合上苹果公司优秀的工业设计，让 Mac 个人计算机广受欢迎。而随着 OS X 的大放异彩，苹果公司也将这个操作系统移植到小型的嵌入系统，也就是最早的 iPhone 上，并将它命名为 iPhone OS。

后来有越来越多的设备，如 iPod Touch、iPad 与 iTV 也都应用了此操作系统，并在 2010 年，由苹果公司主办的 WWDC（全球开发者论坛）正式将这个操作系统命名为 iOS，也是本书要学习的主要目标。因为 iOS 和 OS X 系出同门，在这个操作系统的底层均属 NextStep 架构，并使用 Objective-C 作为原生语言。

Objective-C 是由 C 语言加上面向对象的特性扩充而来，所以，Objective-C 可视为 C 语言的 ProperSuperSet，也就是真母集合或严格母集合，如图 0-1 所示。



▲图 0-1 Objective-C 和 C 语言的示意图

Objective-C 语言并不等同 C 语言，但 C 语言的语法在 Objective-C 的环境下应不需修改便能兼容执行。

TIP

关于 Objective-C 较 C 语言新增的部分，我们会在第 2 章“Objective-C 与对象基础”中开始讨论。

本书定位为 iOS 程序设计的入门书，但并非一般程序设计的入门书。我们假设你对于图 0-1 中的 C 语言已有基本认知。

TIP

如果你需要一本 C 语言参考书籍，建议参考《The C Programming Language》(ISBN: 978-0-131-10362-7)，熟悉 C 语言的语法，不管是阅读本书或开发 iOS 程序，都会有很大的帮助。

0.2 软件的准备

iOS 开发者网站的网址如下：

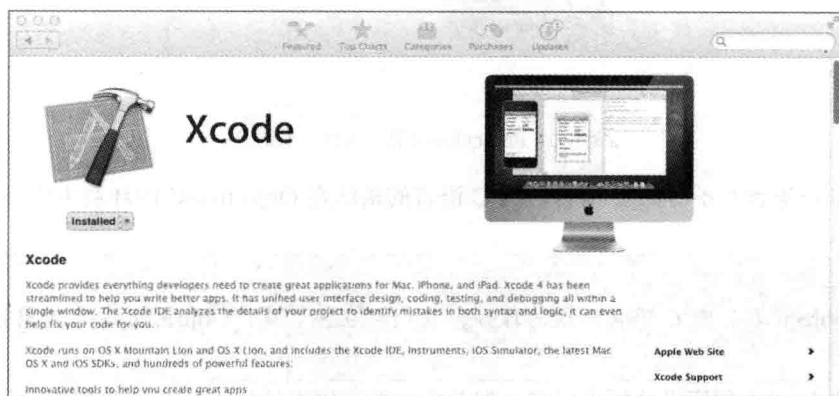
<https://developer.apple.com/>

这个网站中有三个主要开发资源，分别为 iOS 的开发、Mac 的开发和浏览器 Safari 相关的信息，如图 0-2 所示。你可以点击「iOS Dev Center」进入 iOS 开发者资源网页。这个网页需要申请账号，请先申请并登录。



▲图 0-2 美国苹果公司提供开发者的资源网页

登录后在开发者网页中可以找到 Xcode 的连接, 点击后会提示将使用 Mac 下的 App Store 来下载, 点击后在打开的 App Store 界面下方会列出 Xcode 的说明, 如图 0-3 所示。Xcode 为开发 iOS 与 OS X 的集成开发环境 (IDE)。你可以运用它来输入 Objective-C 的语言开发命令行或是直接使用窗口界面来开发 iOS 与 OS X 应用程序, 或是开发一般的 C/C++ 应用程序, 在 OS X 中运行。



▲图 0-3 打开 App Store 的 Xcode 界面

如果你尚未下载过 Xcode, 点击图 0-3 中的「Installed」按钮会看到蓝色「Free」按钮, 点击后会跳出绿色的「Install App」, 点击之后就会进行下载的动作, 下载完成即拥有开发 Ios/OS X 应用程序的 IDE 环境。

0.3 硬件的准备

OSX、iOS 与 Xcode 都是美国苹果公司的产品, 所以这几个环境非常紧密地结合在一起。因此, 若要开发 iOS5/iOS6 应用程序, 你必须要有能够运行 OS X Lion (10.7) 或 Mountain Lion (10.8) 操作系统的 Mac 计算机。虽说 Mac 计算机一般的硬件配备都算不错, 但在开发 iOS 程序时, 我们还会在 Mac 上运行 iOS 模拟器, 所以, 越大的内存和越快的硬盘通常也意味着越良好的执行表现。

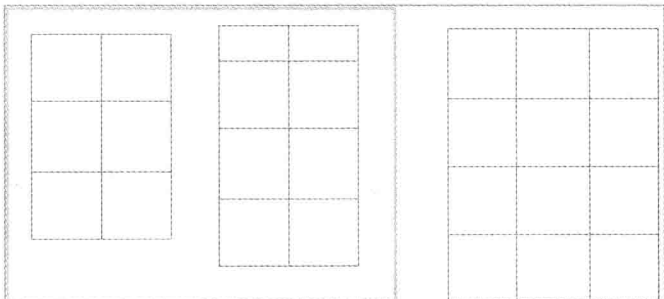
另外, 编写 Xcode 的同时也需执行模拟器的调试, 所以, 一个高分辨率的屏幕也不可少。新版 Mac book Pro 目前已配置了 Retina 显示器, 建议使用 2880×1800 的分辨率来进行开发。若能再外接一个屏幕来进行额外的显示, 也会提升开发的顺畅度。

虽然 Xcode 已提供 iOS 模拟器, 但在应用程序上线前, 使用实际的 iDevice 来测试应用程序在实体机上面的表现和性能, 也是非常重要的事。相对于其他平台, iDevice 的机种相对简单, 到目前为止, 只有三种比例, 具体为 3:2、4:3 与 16:9。

一般手持设备如 iPod Touch 与 iPhone 各系列均为 3:2 的屏幕, 分辨率为 480×320, 而配置 Retina 显示器的分辨率则是将每个尺寸都放大到两倍, 也就是 $(480 \times 2) \times (320 \times 2) = 960 \times 640$, 至于最新的 iPhone5 是以四英寸的屏幕配置 1136×640 的分辨率。另外, 像是平板计算机的 iPad、iPad2 以及 The new iPad, 则为 4:3 的屏幕, 分辨率是 1024×768。而配置 Retina 显示器的分辨率是 2048×1536, 其对比表参考图 0-4。

为了保护版权, 按照正常的渠道, iDevice 是不允许安装未经 Apple iTunes store 发布的软件。因此, 若准备了 iDevice 可运行的硬件, 想要在实体机上运行应用程序, 还必须成为 iOS 付费的开发会员, 并在以下网址进行登录:

<https://developer.apple.com/programs/ios/>



屏幕比例	3:2	16:9	4:3
一般分辨率	480*320	1136*640	1024*768
Retina 分辨率	960*640	1136*640	2048*1536
设备	iPod,iPhone	iPhone5	iPad, The new iPad

▲图 0-4 iDevice 的屏幕比例和分辨率列表

成功注册并缴交\$99 的年费后，在开发时，只要将 iDevice 通过 usb 连接到 Mac 开发计算机，Xcode 就会自动将这个设备的 UDID（Unique Device Identifier，UDID）连接到你的开发者账号，而一个账号最多可以连接一百个实体机设备，应用程序也不需经 Apple 的 iTunes store，便可安装到实际设备。

0.4 设计模式的准备

大致了解软硬件和 iOS/OS X 的背景后，来看看 iOS 开发时，需要了解的设计模式。设计模式（Design Pattern）是在软件设计和软件工程中，针对常常出现且一再被大家处理的问题所提出的解决方案。在此处，我们不会从最简单的面向对象的类/对象开始介绍起，而是假设你对对象、继承、接口等都有了基本认识，再来看 iOS 开发时，会用到的设计模式。

0.4.1 Model-View-Controller 设计模式

Model-View-Controller（模型—视图—控制器模式）是 iOS/OS X 窗口应用程序中三种组件的代表，也有文件会将这三个组件的前缀缩写成 MVC 设计模式。这个设计模式可说是 iOS 应用程序的基础。若想编写 iOS 应用程序，就必须对此设计模式有基本认识。当然，除了这个被动的理由之外，MVC 设计模式不只大量应用在 iOS/OS X 上，几乎所有窗口应用程序都采用这种架构来让组件能被重复使用，也更容易清楚定义组件之间的界面，当未来要扩充应用程序时，也会更具灵活性。

基本上，MVC 设计模式所要解决的问题便是视窗应用程序的设计模式。如图 0-5 所示，你可以想像所谓的窗口应用程序，就是一个介于用户和计算机之间的应用程序。



▲图 0-5 用户与计算机之间的窗口应用程序

而在图 0-5 所示中的这个窗口应用程序，会有一部分程序组件是与用户界面相关的，也就是