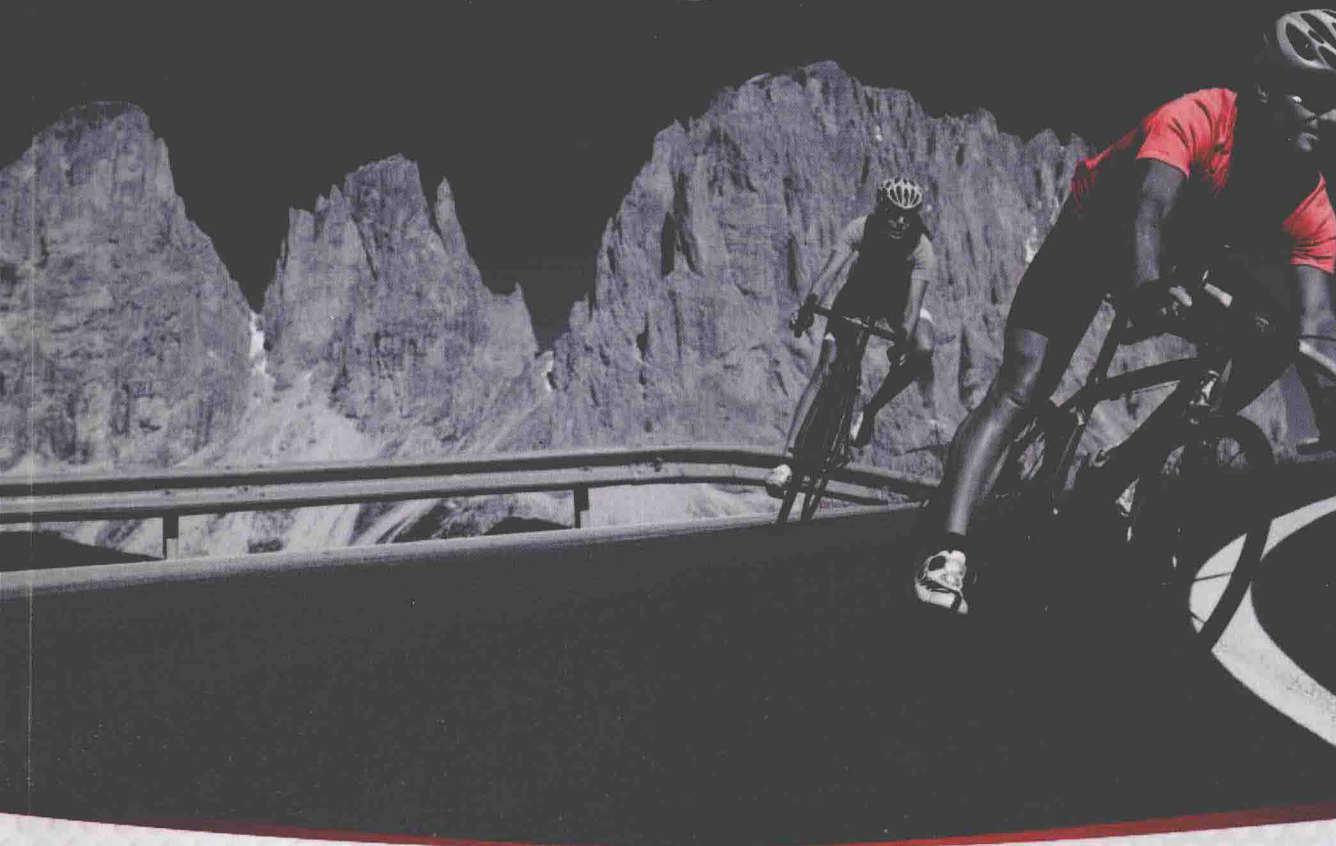


Join the discussion @ p2p.wrox.com



Wrox Programmer to Programmer™

移动开发经典丛书



Beginning Swift Programming

Swift编程入门经典

[美] Wei-Meng Lee 著
李泽鲁 译

清华大学出版社

移动开发经典丛书

Swift 编程入门经典

[美] Wei-Meng Lee 著

李泽鲁 译

清华大学出版社

北 京

Wei-Meng Lee

Beginning Swift Programming

EISBN: 978-1-119-00931-3

Copyright © 2015 by John Wiley & Sons, Inc., Indianapolis, Indiana

All Rights Reserved. This translation published under License.

Trademarks: Wiley, the Wiley logo, Wrox, the Wrox logo, Programmer to Programmer, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. Swift is a trademark of Apple, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc., is not associated with any product or vendor mentioned in this book. Beginning Swift Programming is an independent publication has not been authorized, sponsored, or otherwise approved by Apple, Inc.

本书中文简体字版由 Wiley Publishing, Inc. 授权清华大学出版社出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

北京市版权局著作权合同登记号 图字: 01-2015-2355

Copies of this book sold without a Wiley sticker on the cover are unauthorized and illegal.

本书封面贴有 Wiley 公司防伪标签, 无标签者不得销售。

版权所有, 侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目(CIP)数据

Swift 编程入门经典 / (美)李伟梦 著; 李泽鲁 译. —北京: 清华大学出版社, 2015

(移动开发经典丛书)

书名原文: Beginning Swift Programming

ISBN 978-7-302-40227-5

I. ①S… II. ①李… ②李… 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(2015)第 101269 号

责任编辑: 王 军 李维杰

装帧设计: 牛静敏

责任校对: 曹 阳

责任印制: 沈 露

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者: 北京鑫丰华彩印有限公司

装 订 者: 三河市溧源装订厂

经 销: 全国新华书店

开 本: 185mm×260mm 印 张: 15.5 字 数: 377 千字

版 次: 2015 年 6 月第 1 版 印 次: 2015 年 6 月第 1 次印刷

印 数: 1~3000

定 价: 49.80 元

产品编号: 062783-01

译者序

作为本书的译者，我倍感荣幸，同时作为一名“码农”也倍感骄傲，因为 Coding 可以给我带来快乐。在翻译本书的过程中，我不仅是一名译者，更是一名读者，在拜读原著时，也深深被 Swift 语言的魅力所吸引。在翻译本书之前，我自己也以“青玉伏案”的笔名发表过很多关于 iOS 开发和 Objective-C 的网络文章，并且结识了很多真正热爱技术的朋友。只要你对技术怀揣着一颗热诚之心，那么技术的大门也会为你敞开。

Swift 语言是苹果公司在 2014 年发布的一门新的编程语言。Swift 继承了 C 语言以及 Objective-C 的特性，且克服了 C 语言的兼容性问题，同时又增加了一些实用的特性，对于广大开发者来说，这无疑是一剂难以抗拒的兴奋剂。

本书为想进入 Swift 语言殿堂的开发人员而著，旨在穿过所有的技术术语直接进入 Swift 世界，带你感受 Swift 语言的魅力。当然，学习任何一门新的语言的最好方法就是理论结合实践，同时要学会与之前所学过的编程语言做类比，如果之前学过 C# 或 JavaScript 等语言，你会很快融入 Swift 世界当中，并且会亲身感受到 Swift 优雅所在。

学习一门新的编程语言，基础必不可少，尤其是细节问题，本书在编排上由浅入深，循序渐进。本书的所有代码在我翻译的时候都进行了编译，而且深知每个示例的重要性。想必作者在示例的构思和选择上面花了不少心思，而且示例与上下文联系紧密，紧贴生活，便于理解。本文虽是译者序，但是叫作读者序更为贴切，通过翻译本书我也学到了不少新的知识，而且对 Swift 语言有了更为系统和深入的了解，深深被 Swift 语言的强大和灵活所吸引。

在移动互联网蓬勃发展的今天，我相信 Swift 语言的未来是光明的，在不久的将来 Swift 语言将会在移动互联开发中扮演重要的角色。我相信 Swift 语言在移动互联网浪潮的推动下，将会把语言自身的优点发挥到极致，同时也会更好地促进移动互联网的蓬勃发展。在移动互联网时代，作为“码农”的我们必须走在技术的最前沿，鼓起勇气去尝试一些新鲜的东西。为了适应移动互联网大潮，我们必须不停地为自己充电，要做好时刻学习的准备。作为一名真正的 Coder，要对技术有种痴迷的状态，而不仅仅是一种谋生的手段，时时刻刻学习，要有被淘汰的危机感，本书就是学习优秀的 Swift 编程语言的引导者。本书为你展示了 Swift 的魅力所在，引导你揭开 Swift 语言的神秘面纱，并且让如此一门优秀的语言成为你的得力工具，帮助你畅游程序的海洋，体验编程的乐趣。

本书全部章节主要由李泽鲁翻译，参与本次翻译工作的还有李洁、明睿、于莲莲、郝凡宁、李如申、孙海金，在此一并表示感谢！

作者简介

Wei-Meng Lee 是一名技术专家，同时也是 Developer Learning Solutions (www.learn2develop.net)的创始人。Developer Learning Solutions 是一家培训最新的移动开发技术的科技公司。Wei-Meng 有多年的培训经验，并且他的培训课程特别推崇理论加实践的方法。以边学边做的方式来学习编程，这种方式使得编程相对一般的书本、教程或其他文档更容易理解。Wei-Meng 还是畅销书《Android 4 编程入门经典——开发智能手机与平板电脑应用》和《iOS 5 编程入门经典——开发 iPhone 与 iPad 应用》的作者，这两本书均已由清华大学出版社引进并出版，可以通过 weimenglee@learn2develop.net 联系到他。

技术编辑简介

Chaim Krause 是美国陆军司令部和 General Staff College 的仿真专家，他在众多的平台上开发各种软件产品，从 iOS 和 Android 设备到 Windows 桌面和 Linux 服务器等。Python 是他的首选语言，但他在 Java 和 JavaScript/HTML5/CSS 等其他语言上也有过实践。他非常幸运，因为他的职业生涯始于 Borland，他在这里是一名 Delphi 高级开发工程师。此外，他还是个计算机极客。Chaim 比较喜欢 techno 和 dubstep 音乐，以及他的摩托车和两个雪橇狗——Dasher 和 Minnie。

致 谢

我想借此机会感谢那些为了本书的出版而努力工作的幕后工作人员。首先非常感谢 Wrox 公司的执行编辑 Bob Elliott。当我向 Bob 提这本书的时候，他的第一反应就是我最快能什么时候完成。过去的都已成为历史。谢谢你，Bob，谢谢你给我按时交付这本书的信心！

当然，我也不能忘了 John Sleeva，他是本书的文字编辑，同时也是我的新朋友，他总是快乐地工作着。谢谢 John 对项目的指导和鼓励。

在这里也对本书的技术编辑 Chaim Krause 和 Matthew Eccles 表示深深的感谢。Chaim 总能发现书稿中的错误，我一直希望他能写一些建议来改善我的代码质量。谢谢 Chaim。Matthew 为本书代码做 eagle-eye 测试，确保本书代码基于最新的 Xcode 版本，谢谢 Matthew。

最后但并非最不重要，我想借此机会感谢我的家人。事实上，你手中拿的这本书是我在工作中受到很多人帮助的结果。我要感谢我的妻子 Sze Wa，在我为了赶时间而忙得焦头烂额的时候对我们的新宝贝 Chloe 的照顾。在她生产后，她一直无私地照顾 Chloe，我知道这对她的身体和心理是不小负担。谢谢你，亲爱的！

在我工作、出差和写作期间，父母也轮流来照顾我们的宝宝。他们对这些付出没有任何抱怨，为此我深表感激。我想对父母和家人说：我爱你们！最后我要感谢可爱的小狗 Ookii，感谢你忠实地呆在我的身边。

前言

世界在飞速发展，几乎每天都有变化，并且不时会有大事情发生并改变着整个行业。例如，iPhone 诞生于 2007 年，在一夜之间改变了手机行业，率先迈向智能手机的新时代。iPad 在发布三年后(2010 年)改变了我们使用计算机的方式，这使许多人预测，我们都登上了 PC 时代的顶峰。

在 20 世纪 80 年代 NeXT 公司成立后的很长一段时间内，Objective-C 被用于开发 NeXT 公司的 NeXTSTEP 操作系统。Mac OS X 和 iOS 都来源于 NeXTSTEP，因此 Mac OS X 和 iOS 开发语言自然就选择了 Objective-C。开发人员在刚接触 iOS 开发时，会经常抱怨 Objective-C 看上去并不像一门现代编程语言(如 Java 或 C#)并且很难写，通常要花大量的时间去学习。7 年来，苹果公司已经对开发语言和 iOS 框架进行了改善，通过引入一些有用的功能，使得开发开发人员用起来更为容易，比如用自动引用计数(Automatic Reference Counting, ARC)来做内存管理的苦差事，同时引入 Storyboard 来简化用户操作界面。然而，抱怨还是没有停止，此外，苹果公司需要一门新的语言来把 iOS 和 Mac 操作系统发展到更高水平。

2014 年，在苹果全球开发人员大会(WWDC)上，苹果公司发布了一门新的编程语言——Swift，这震惊了许多开发人员。7 年之后，苹果公司终于发布了一门新的语言来代替 Objective-C。正如你将在本书中看到的，Swift 是一门易于阅读的现代编程语言，并且对类型安全进行了严格控制。

本书是为忙碌的开发人员所准备的，旨在穿过所有的技术术语直接进入 Swift 语言的殿堂。当然，学习任何一门新的语言的最好方法就是查看代码示例，本书也为你呈现了这些示例。为了从本书中获取更多的知识，强烈建议你在阅读每章的示例时亲自实现一下。

本书读者对象

本书的读者对象是初级 iOS 开发人员和有经验的 Objective-C 开发人员。对于具备编程基础并且已经理解面向对象编程(Object-Oriented Programming, OOP)的开发人员来说，本书也能发挥巨大作用。

所有章节的代码示例都是在 Xcode 6 上编写的并且已经测试过。因为 Swift 语言仍处在发展当中，所以当本书面世时，Swift 语言应该会有一些小的调整。

本书组织结构

本书介绍使用 Xcode 6 进行 Swift 编程的一些关键主题，总共包括如下 12 章内容：

第 1 章：“初识 Swift”，涵盖 Swift 的基本语法以及如何配置开发环境，从而测试 Swift 代码。

第 2 章：“数据类型”，涵盖 Swift 支持的基本数据类型以及如何执行它们涉及的常见操作，并且介绍 Swift 新的元组和可选的数据类型。

第 3 章：“字符串和字符”，讨论在 Swift 语言中如何管理字符和字符串类型。特别强调 Swift 语言中的字符串类型是如何向后兼容 Objective-C 中的 NSString 类型的。同时介绍在处理字符集时必须知道的内容。

第 4 章：“基本运算符”，涵盖 Swift 语言中的常用运算符。此外，还讨论在 Swift 语言中一些运算符新的使用范围。

第 5 章：“函数”，介绍如何在 Swift 语言中进行函数定义以及在调用函数的时候如何使用内部参数和形参。

第 6 章：“集合”，涵盖 Swift 语言中支持的集合类型——数组和字典。

第 7 章：“控制流和循环”，介绍如何在 Swift 语言中作出判断以及如何使用循环语句执行重复的代码。

第 8 章：“结构体和类”，涵盖编程结构的基础知识，还演示如何在类和结构体中定义属性和方法。

第 9 章：“继承”，介绍如何在 Swift 语言中定义子类以及如何控制成员变量的访问权限，同时还展示如何扩展类。

第 10 章：“闭包”，涵盖你需要知道的有关功能块的所有知识以及如何使用 Swift 语言来实现它们。

第 11 章：“协议与代理”，讨论 Swift 语言中设计模式最重要的一部分，协议和代理模式是在 iOS 和 Mac OS 中大部分 API 上编程的基础。

第 12 章：“泛型”，本章描述在 Swift 语言中是如何包含“泛型”这个熟悉的编程概念的。泛型允许开发人员编写高度灵活的代码，促进代码共享和重用。

练习题的答案在书末的附录中。

使用本书的前提

为了实现本书中提供的代码示例，需要一台 Mac 电脑并装上 Xcode 6。在 Mac App Store 中，Xcode 6 是可以免费下载的。不需要用 iOS 设备测试本书中的代码。为了测试，可以创建 Playground 项目或 iOS 项目，然后在 iPhone 模拟器上进行测试。

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但是错误总是难免的，如果你在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

要在网站上找到本书的勘误表，可以登录 <http://www.wrox.com>，通过 Search 工具或书名列表查找本书，然后在本书的细目页面上，单击 **Book Errata** 链接。在这个页面上可以查看 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表，网址是 www.wrox.com/misc-pages/booklist.shtml。

如果在 **Book Errata** 页面上没有看到你找出的错误，请进入 www.worx.com/contact/techsupport.shtml，填写表单，发电子邮件，我们会检查你的信息，如果是正确的，就在本书的勘误表中粘贴一个消息，我们将在本书的后续版本中采用。

p2p.wrox.com

P2P 邮件列表是为作者和读者之间的讨论而建立的。读者可以在 p2p.wrox.com 上加入 P2P 论坛。该论坛是一个基于 Web 的系统，用于传送与 Wrox 图书相关的信息及相关技术，与其他读者和技术用户交流。该论坛提供了订阅功能，当论坛上有新帖子时，会给你发送你选择的主题。Wrox 作者、编辑和其他业界专家和读者都会在这个论坛上进行讨论。

在 <http://p2p.wrox.com> 上有许多不同的论坛，帮助读者阅读本书，在读者开发自己的应用程序时，也可以从这个论坛中获益。要加入这个论坛，必须执行下面的步骤：

- (1) 进入 p2p.wrox.com，单击 **Register** 链接。
- (2) 阅读其内容，单击 **Agree** 按钮。
- (3) 提供加入论坛所需的信息及愿意提供的可选信息，单击 **Submit** 按钮。
- (4) 然后就可以收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。



提示：不加入 P2P 也可以阅读论坛上的信息，但只有加入论坛后，才能发送自己的信息。

加入论坛后，就可以发送新信息，回应其他用户的帖子。可以随时在 Web 上阅读信息。如果希望某个论坛给自己发送新信息，可以在论坛列表中单击该论坛对应的 **Subscribe to this Forum** 图标。

对于如何使用 Wrox P2P 的更多信息,可阅读 P2P FAQ,了解论坛软件的工作原理,以及许多针对 P2P 和 Wrox 图书的常见问题解答。要阅读 FAQ,可以单击任意 P2P 页面上的 FAQ 链接。

源代码

如果想运行本书示例中的代码,需要把所有代码输入到 Xcode 中,然后运行并观察结果。记住,学习一门语言的最好方法就是亲自去实践并且犯错误。对于第 11 章,可以在网页 www.wrox.com/go/beginningswift 上找到 LBS 项目的源代码。在该页面上,只需找到这本书的书名(使用 Search 框或书名列表)并在本书的详细页面上点击 Download Code 链接,即可获得本书的所有源代码。

下载代码之后,用你最喜欢的压缩工具解压缩。也可以去 Wrox 主页下载本书的代码,下载地址为 www.wrox.com/dynamic/books/download.aspx,在上面还可以看到 Wrox 公司其他的图书。



注意: 因为许多书都有着类似的名字,用 ISBN 号查找更为容易,本书英文版的 ISBN 号为 978-1-119-00931-3。

目 录

第 1 章 初识 Swift	1	2.4.5 枚举函数	34
1.1 Swift 语言是什么	2	2.5 本章小结	35
1.2 Swift 语言为什么重要	3	第 3 章 字符串和字符	39
1.3 配置开发环境	3	3.1 字符串	39
1.3.1 创建 Playground 项目	5	3.1.1 可变的字符串	40
1.3.2 创建 iOS 项目	6	3.1.2 值类型的 String	40
1.4 Swift 语法	10	3.1.3 字符	41
1.4.1 常量	10	3.1.4 字符串的连接	42
1.4.2 变量	12	3.1.5 特殊字符	43
1.4.3 字符串插值：在字符串中 包含值	13	3.1.6 Unicode	44
1.4.4 声明	14	3.2 常见的字符串函数	45
1.4.5 输出	14	3.2.1 相等	45
1.4.6 注释	15	3.2.2 前缀和后缀	46
1.5 本章小结	16	3.2.3 长度	47
第 2 章 数据类型	19	3.2.4 子字符串	48
2.1 基本数据类型	20	3.2.5 将字符串转换为数组	50
2.1.1 整型	20	3.2.6 类型转换	50
2.1.2 浮点型数字	22	3.3 与 NSString 的互操作	51
2.1.3 类型别名	24	3.3.1 把 String 转换为 NSString	52
2.1.4 布尔型	24	3.3.2 直接使用 NSString	53
2.2 元组	25	3.3.3 使用 String 还是 NSString?	54
2.3 可选类型	26	3.4 本章小结	55
2.3.1 隐式打开可选类型	27	第 4 章 基本运算符	57
2.3.2 可选绑定	28	4.1 赋值运算符	57
2.3.3 用?符号打开可选值	29	4.2 算术运算符	58
2.4 枚举	29	4.2.1 加法运算符	59
2.4.1 在 Swift 语句中使用枚举	30	4.2.2 减法运算符	59
2.4.2 枚举原始值	31	4.2.3 乘法运算符	60
2.4.3 原始值自增	32	4.2.4 除法运算符	60
2.4.4 关联值	33	4.2.5 取模运算符	61
		4.2.6 递增和递减运算符	61

4.2.7	复合赋值运算符	62	6.1.6	将元素附加到数组中	88
4.2.8	nil 连接运算符	62	6.1.7	检查数组的大小	88
4.3	比较运算符	63	6.1.8	从数组中移除元素	89
4.3.1	等于和不等	63	6.1.9	数组的遍历	89
4.3.2	大于或大于等于	65	6.1.10	创建空数组	90
4.3.3	小于或小于等于	65	6.1.11	测试数组是否相等	91
4.4	区间运算符	65	6.2	字典	91
4.5	逻辑运算符	67	6.2.1	可变字典	92
4.5.1	逻辑非	67	6.2.2	获取字典中的元素	93
4.5.2	逻辑与	67	6.2.3	检查字典的大小	93
4.5.3	逻辑或	68	6.2.4	修改字典中的元素	93
4.5.4	组合逻辑运算符	69	6.2.5	移除字典中的元素	94
4.5.5	三元运算符	70	6.2.6	字典的遍历	95
4.6	本章小结	70	6.2.7	创建空字典	96
第 5 章	函数	73	6.2.8	测试字典是否相等	96
5.1	函数的定义与调用	73	6.3	数组和字典的复制行为	97
5.1.1	输入参数	74	6.4	本章小结	98
5.1.2	返回值	74	第 7 章	控制流和循环	101
5.1.3	返回多个值	75	7.1	控制流	101
5.1.4	函数参数名称	75	7.1.1	if 语句	102
5.1.5	形参名的简写方式	77	7.1.2	if-else 语句	103
5.1.6	默认参数值	77	7.1.3	switch 语句	104
5.1.7	可变(变量)参数	78	7.1.4	where 子句	111
5.1.8	常量和变量参数	79	7.2	循环	112
5.1.9	输入输出参数	80	7.2.1	for-in 循环	112
5.2	函数类型	80	7.2.2	传统的 for 循环	115
5.2.1	定义函数类型的变量	81	7.2.3	while 循环	116
5.2.2	调用函数类型的变量	81	7.2.4	do-while 循环	117
5.2.3	返回函数中的函数类型	82	7.2.5	控制转移语句	118
5.3	嵌套函数	83	7.3	本章小结	122
5.4	本章小结	83	第 8 章	结构体和类	125
第 6 章	集合	85	8.1	结构体	125
6.1	数组	86	8.1.1	成员级初始化器	127
6.1.1	可变数组	86	8.1.2	结构体的类型	127
6.1.2	数组数据类型	86	8.1.3	结构体比较	129
6.1.3	从数组中获取元素	87	8.2	类	129
6.1.4	向数组中插入元素	87	8.2.1	类的定义	130
6.1.5	修改数组中的元素	88	8.2.2	属性	130

8.2.3	初始化器	138	10.1.5	简写参数名称	181
8.2.4	初始化器与形参名	139	10.1.6	运算符函数	181
8.2.5	在初始化期间对变量和 常量进行初始化	141	10.1.7	尾部闭包	182
8.2.6	类是引用类型	142	10.2	使用数组的三个闭包功能	182
8.2.7	类中的方法	146	10.2.1	map()函数	182
8.2.8	类方法	149	10.2.2	filter()函数	184
8.2.9	结构体中的方法	150	10.2.3	reduce 函数	186
8.3	本章小结	152	10.3	在函数中使用闭包	187
第 9 章	继承	155	10.4	本章小结	190
9.1	认识继承	156	第 11 章	协议与代理	191
9.1.1	定义基类	156	11.1	理解协议	191
9.1.2	基类的实例化	156	11.1.1	协议的定义与使用	192
9.1.3	创建抽象类	156	11.1.2	可选方法	194
9.1.4	从基类中继承	158	11.1.3	遵循多个协议	195
9.2	初始化器的类型	165	11.1.4	必须实现的初始化器	196
9.2.1	默认初始化器	165	11.2	理解代理	197
9.2.2	指定初始化器	166	11.2.1	作为事件处理程序的 代理	197
9.2.3	便利初始化器和初始化 器链	168	11.2.2	协议和代理的示例	201
9.2.4	在子类中调用初始化器	170	11.3	本章小结	203
9.3	扩展	170	第 12 章	泛型	205
9.3.1	扩展方法	171	12.1	理解泛型	205
9.3.2	扩展属性	171	12.1.1	使用泛型函数	206
9.4	访问控制	172	12.1.2	多类型参数	207
9.4.1	Internal	173	12.1.3	指定类型约束	208
9.4.2	private	174	12.2	泛型类型	209
9.4.3	public	174	12.2.1	泛型类	209
9.5	本章小结	175	12.2.2	泛型结构体	210
第 10 章	闭包	177	12.2.3	泛型类型扩展	212
10.1	理解闭包	178	12.2.4	在协议中使用泛型	213
10.1.1	函数闭包	178	12.2.5	为相关类型指定需求	215
10.1.2	将闭包赋值给变量	179	12.3	本章小结	217
10.1.3	编写内置闭包	179	附录	习题答案	219
10.1.4	类型推断	180			

第 1 章

初识 Swift

本章主要内容:

- Swift 语言是什么
- Swift 为什么重要
- 搭建学习 Swift 语言的开发环境
- 如何创建 Playground 项目
- 如何创建 iOS 项目
- Swift 语法
- 如何声明常量
- 如何声明变量
- 使用字符串插值: 在字符串中包含值
- Swift 语句
- 如何在调试时打印变量的值
- 在 Swift 代码中如何插入注释

2014 年,苹果公司在全球开发人员大会(World Wide Developers Conference, WWDC)上发布了一门新的编程语言——Swift,这震惊了许多 Mac 和 iOS 开发人员。Swift 有着现代编程语言的语法,旨在代替 Objective-C 语言。同时也不用太过担心对 C 语言的兼容性。苹果公司本身就推崇用 Swift 语言取代 Objective-C。

因为开发人员已经习惯了 Objective-C 语言,所以可以预料到,在不久的将来 iOS 和 Mac OS X 开发仍然支持 Objective-C。然而,迹象表明,未来苹果公司旨在让 Swift 语言成为 iOS 和 Mac 开发语言的首选。

在这一章,你将学习 Swift 语言的基础知识以及如何配置开发环境。

1.1 Swift 语言是什么

Swift 是苹果公司为 Cocoa(Mac OS X)和 Cocoa Touch(iOS)编程而设计的一门新的语言。Swift 的语法类似于现代编程语言,例如 Java 和 C#,同时保留了 Objective-C 支持的一些核心功能,如命名参数、协议和代理。Swift 的清晰语法可以让代码的阅读和维护更为简单。

举个例子,考虑一下 Objective-C 中的如下方法:

```
-(int) addOneNumber:(int) num1 withAnotherNum:(int) num2
{
    return num1 + num2;
}
```

上面的方法返回两个数的和,可以发一条消息来调用上面的方法,如下所示:

```
int sum = [self addOneNumber:2 withAnotherNum:7];
```

注意,Objective-C 中的方法有着冗长的参数名,在 Swift 语言中相同的函数如下所示:

```
func addTwoNumbers(num1:Int, num2:Int) -> Int {
    return num1 + num2
}
```

前面的语句定义一个名为 addTwoNumbers 函数,该函数接受两个参数并返回一个整数值。可以使用下面的方式来调用该函数:

```
var sum = addTwoNumbers(2, 5)
```

正如你看到的,Swift 语法简洁并且便于阅读。

为了保持 Objective-C 的命名传统,可以在方法中使用命名参数:

```
func addTwoNumbers(num1:Int, secondNumber num2:Int) -> Int {
    return num1 + num2
}
```

现在可以用命名参数来调用该函数,如下所示:

```
var sum = addTwoNumbers(2, secondNumber:5)
```



注意: 在第 5 章会详细介绍函数和命名参数。

Swift 还被设计成一门类型安全的语言。变量在使用之前必须初始化。在大多数情况下,从一种类型转换为另一种类型时必须执行显式类型转换。变量在没有被赋值前是不能在语

句中使用的，否则会被标记为错误。

在 Swift 语言中，为了安全起见不允许隐式类型转换。将 Int 类型转换为 Float 类型(或 Double 类型)必须显式进行。例如，不能将 Int 类型的变量隐式转换为 Float 类型的变量：

```
var f:Float
var i:Int = 5
f = i //---error---
```

相反，需要显式地转换为 Float 类型的值：

```
f = Float(i)
```



注意：在第2章会对数据类型进行更详细的介绍。

1.2 Swift 语言为什么重要

毫无疑问，苹果公司创建 Swift 语言并不是为了创建一门新的编程语言。随着移动平台大战的逐渐升温，苹果公司迫切需要一门新的语言来确保其长期在移动平台市场的领先地位。在很多方面 Swift 语言就是苹果公司的战略所在，比如：

- 它解决了开发人员使用 Objective-C 时所反映的问题，特别是 Objective-C 的语法学起来很困难。取而代之的是可以快速学习的并且易于维护的一门新语言。
- 它是一门容易学习的语言，同时保留了 Objective-C 的精华且去除了其冗长的语法。
- 它是一门比 Objective-C 更为安全的语言，这有助于构建更为健壮的应用平台。
- 它能够与 Objective-C 共存，这使开发人员有足够的时间将他们的代码转向 Swift。

1.3 配置开发环境

为了测试本书中的 Swift 代码示例，需要 Swift 语言编译器。获取 Swift 语言编译器最简单的方法是从 Mac App Store 下载 Xcode 6(见图 1-1)。

下载 Xcode 6 后，安装到你的 Mac 电脑上并启动它(见图 1-2)。

测试本书中的代码时，有如下两种方式：

- **创建 Playground 项目：**Playground 是 Xcode 6 中的新特性，它使学习 Swift 语言变得简单而有趣。当输入每一行代码时，Playground 就会显示这行代码的运行结果。当跟踪每行代码时，也可以通过 Playground 观察每个变量的值。当给变量赋值并检查变量类型时，Playground 也非常有用。