Broadview
www.broadview.com.cn

Swifter

100个Swift开发必备Tip

王巍 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Swifter

100个Swift开发必备Tip

王巍 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内容简介

作者赴美参加了 Apple 的 WWDC 14，亲眼见证了 Swift 的发布，并从这门语言正式诞生的第一分钟就开始学习和钻研。在本书中作者将自己的经验加以总结和整理，以一个个的小技巧和知识点的形式揭示出来。全书共有 100 节，每一节都是一个相对独立的主题，涵盖了一个中高级开发人员需要知道的 Swift 语言的方方面面。

本书非常适合用作官方文档的参考和补充，相信也会是 iOS 中级开发人员很喜爱的 Swift 进阶读本。

图书在版编目（CIP）数据

Swifter：100 个 Swift 开发必备 Tip / 王巍著. — 北京：电子工业出版社，2015.5
ISBN 978-7-121-25796-4
I. ① S…II. ① 王…III. ① 程序语言－程序设计 IV. ① TP312

中国版本图书馆 CIP 数据核字（2015）第 065963 号

责任编辑：许　艳
印　　刷：北京丰源印刷厂
装　　订：三河市皇庄路通装订厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编 100036
开　　本：787×980　1/16　印张：17.5　字数：388 千字
版　　次：2015 年 5 月第 1 版
印　　次：2015 年 5 月第 1 次印刷
印　　数：3000 册　　定价：69.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

推荐序

让雨燕飞翔

在 2014 年 6 月之前，如果我们在 Google 中输入“Swift”进行查找，搜到的是美国创作型歌手、大美女泰勒 · 斯威夫特。今天我们再去做同样的检索，搜索结果是一门编程语言，这门编程语言的名字就叫作“Swift”，它的 Logo 是一只极速飞翔的雨燕。

Swift 是 Apple 公司在 2014 年 WWDC 大会上推出的一门新语言，用于在 iOS/OS X 平台上开发应用程序，之前独霸这个庞大平台的语言一直是 Objective-C。可以说 Swift 是我所见过关注度最高的新语言，刚推出即万众瞩目，媒体和开发者在数天之内对 Swift 进行了集中的报道和讨论，英文手册迅速被翻译成中文，即使是谷歌 2009 年推出 Go 语言时也没有如此浩大的声势。时至今日，已经有大量的独立应用是基于 Swift 开发构建的。

2007 年之前，Objective-C 一直是 Apple 自家后院的小众语言，iOS 移动设备的爆发让这门语言的普及率获得了火箭一般的蹿升速度，截止到今天，Objective-C 在编程语言排行榜上排名第三，江湖人称三哥。Apple 一直在不遗余力地优化 Objective-C，包括把 GCC 的编译链替换成 LLVM + GCC，又替换成 LLVM + Clang，做语法简化、自动引用计数、增加 Blocks 和 GCD 多线程异步处理技术……既然已经全盘掌握了 LLVM 和 Clang 技术，为什么不开发一门新语言呢？于是 Swift 语言诞生了。

Swift 的作者是天才的 70 后程序员 Chris Lattner，他同时是 LLVM 项目的主要发起人与作者之一、Clang 编译器的作者。Chris 毕业的时候正是 Apple 为了编译器焦头烂额的时候，因为 Apple 之前的软件产品都依赖于整条 GCC 编译链，而开源界的大爷们并不买 Apple 的账，他们不愿意专门为了 Apple 公司的需求优化和改进 GCC 代码，所以 Apple 经过慎重的考虑后将编译器后端替换为 LLVM，并且把 Chris 招入麾下。Chris 进入了 Apple 之后如鱼得水，不仅大幅度优化和改进 LLVM 以适应 Objective-C 的语法变革和性能要求，同时发起了 Clang 项目，旨在全面替换 GCC。这个目标已经实现了，从 OS X 10.9 和 XCode 5 开始，LLVM + GCC 已经被替换成了 LLVM + Clang。

Swift 是 Chris 在 LLVM 和 Clang 之后第三个伟大的项目！

Swift 是一门博采众长的现代语言，在设计的过程中，Chris 参考了 Objective-C、Rust、Haskell、Ruby、Python、C# 等优秀语言的特点，Swift 的语法特性最终形成。Swift 是面向 Cocoa 和 Cocoa Touch 的编程语言，编译型，类型安全，生产环境的代码都需要 LLVM 编译成本地代码才能执行，但是 Swift 又具备很多动态语言的语法特性和交互方式，支持各种高级语言特性，包括闭包、泛型、面向对象、多返回值、可选变量、类型接口、元组、集合等。

很显然，这是一门准备取代 Objective-C 的编程语言，它将吸引更多的开发者加入苹果的软件生态圈，为 iOS 和 OS X 开发出更为丰富的 App。如果你是 App Store 的开发者，推荐尽早学习和掌握这门苹果力推的新语言。对于大部分新事物来说，越早介入，收获越大。

Swift 入门并不困难，Apple 公司甚至为这门语言提供了所写即所得的 Playground 功能，不仅实现了很多脚本语言支持的交互式编程，而且提供控制台输出、实时图形图像、时间线（timeline）变量跟踪等功能，开发者除了可以看到代码的实时运行结果，还能根据时间线阅读某个变量在代码片段中值的变化。这真是太棒了！另外，阅读官方提供的《The Swift Programming Language》也是快速入门的途径，Cocoa 开发者社区甚至在第一时间提供了高质量的中译本。

问题的关键是入门了之后怎么办？当你读完教程学习了语法，自觉成竹在胸拔剑四顾的时候，突然发现 Swift 在实际的项目应用中会出现各种各样的问题，就像你手持一柄玄铁重剑，却无法洞悉剑诀的奥秘。如何让“雨燕”迅疾地飞翔？这就是《Swifter：100 个 Swift 开发必备 Tip》这本书要解决的问题。

本书作者王巍是我非常尊敬的一位 iOS 开发者，他的网络 ID 是“onevcat”，大家都叫他喵神。王巍毕业于清华大学，在校期间就对 iOS 开发一往情深，曾经开发出《小熊推金币》《Pomo Do》等一系列优秀的 iOS 游戏和应用。工作和开发之余，王巍也在参与 iOS 开发社区的建设，比如发起和组织翻译项目“objc 中国”，开源 Xcode 插件 VVDocumenter 项目等，这本《Swifter：100 个 Swift 开发必备 Tip》同样是他对社区的贡献之一。

王巍是一个在技术上对自己有要求的程序员，在涉及的每个领域，他都希望能够做到庖丁解牛，游刃有余。既能洞悉全局，又可直达细节。王巍 2014 年赴美参加了 Apple 的 WWDC 大会。可以说，从 Swift 诞生的那一分钟起，王巍就开始学习和研究这门语言。他在自己的博文《行走于 Swift 的世界中》阐述了大量 Swift 的语法细节和底层实现机制，并对这篇文章进行了持续的更新，这篇文章在 Swift 社区获得了巨大的反响。之后，王巍持续学习 Swift 语言，并进行了编程实践和项目实战，他把自己的学习心得和编程技巧进行了梳理和完善，最终形成了这本《Swifter：100 个 Swift 开发必备 Tip》。书中共有 100 个 Swift 编程技巧，几乎涵盖了 Swift 语言的所有细节，每篇独立成文，可拆可合，读者可以随时翻阅，也可以遇到实际问题后再来检索。

这本书最早的版本是电子书，我在它出版的第一时间就买了来读，之后随用随读，这本书让我对 Swift 语言有了更为深入的了解，也解决了我的团队在开发过程中的很多实际问题。所以，当获知王巍的这本书要出纸版的时候，我觉得我有责任让更多的人知道这本书。在目前这样一个知识版权认知匮乏的年代，优秀的原创作者总是值得尊敬，他们的图书作品也值得我们珍惜，我希望把这本书推荐给每一个 iOS 开发者，它值得我这么做。

目前王巍旅居日本，就职于即时通信软件公司 Line。他依然行走在修行的路上，孜孜以求创意之源。祝愿在未来的日子里，王巍能为这个世界呈现更好的软件产品和技术图书。落花无言，人淡如菊，书之岁华，其曰可读。这大概就是王巍目前的写照。

作为开发者，我们要做的就是找到这个领域的灯塔，阅读、学习，然后 Write the code, Change the world，并期待下一个收获的季节！

祝大家学得开心！

池建强

《MacTalk · 人生元编程》作者

微信平台 MacTalk 出品人

2015 年，春

序

虽然我们都希望能尽快开始在 Swift 的世界里遨游，但是我觉得仍然有必要花一些时间对本书的写作目的和适合哪些读者进行必要说明。我不喜欢自吹自擂，也无法承担“骗子”的骂名。在知识这件严肃的事情上，我并不希望对读者产生任何误导。作为读者，您一定想要找一本适合自己的书；而作为作者，我也希望找到自己的伯乐和子期。

为什么要写这本书

中文的科技书太少了，内容也太浅了。这是国内市场尴尬的现状，真正有技术的大牛不在少数，但他们很多并不太愿意通过出书的方式来分享他们的知识，一方面原因是回报率实在太低，另一方面是出版的流程过于烦琐。这就导致了市面上充斥着一些习惯于出版业务，但是却丝毫无视质量和素质的“流氓”作者，以及他们制造的“流水线”图书。

特别是对于 Swift 语言来说，这个问题尤其严重。iOS 开发不可谓不火热，每天都有大量的开发者涌入这个平台。而 Swift 的发布更使得原本高温的市场更上一层楼。但是市面上随处可见的都是各种《××× 开发指南》《××× 权威指南》或者《21 天学会 ×××》式的中文资料。这些图书大致都是对官方文档的翻译，并没有什么实质的见解，可以说内容单一，索然无味。作为读者，很难理解作者写作的重心和目的（其实说实话，大部分情况下这类书的作者自己都不知道写作的重心和目的是什么），这样的“为了出版而出版”的图书可以说除了增加世界的熵以外，几乎毫无价值。

如果想要入门 Swift 语言，阅读 Apple 官方教程和文档无论从条理性和权威性来说，都是更好的选择。而中国的 Cocoa 开发者社区也以令人惊叹的速度完成了对文档的高品质翻译，这在其他任何国家都是让人眼红的一件事情。因此，如果您初学程序设计或者 Swift 语言，相比起那些“泯灭良心”（抱歉我用了这个词，希望大家不要对号入座）的“入门书籍”，我

更推荐您看这份翻译后的官方文档[1]，这是非常珍贵的资源。

说到这里，可以谈谈这本《Swifter：100 个 Swift 开发必备 Tip》的写作目的了。很多 Swift 的学习者，包括新接触 Cocoa/Cocoa Touch 开发的朋友，以及之前就使用 Objective-C 的朋友，所面临的一个共同的问题是，入门以后应该如何进一步提高。也许你也有过这样的感受：在阅读完 Apple 的教程后，觉得自己已经学会了 Swift 的语法和使用方式，你满怀信心地打开 Xcode，新建了一个 Swift 项目，想写点什么，却发现实际上不是那么回事。你需要联想 Optional 应该在什么时候使用，你可能发现本已熟知的 API 突然不太确定要怎么表达，你可能遇到怎么也编译不了的问题但却不知如何改正。这些现象都非常正常，因为教程是为了展示某个语法点而写的，而几乎不涉及实际项目中应该如何使用的范例。本书的目的就是为广大已经入门了 Swift 的开发者提供一些参考，以期能迅速提升他们在实践中的能力。因为这部分的中级内容是我自己力所能及，有自信心能写好的，也是现在广大 Swift 学习者所缺乏和急需的。

这本书是什么

本书是 Swift 语言的知识点的集合。我自己是赴美参加了 Apple 的 WWDC 14 的，也正是在这届开发者大会上，Swift 横空出世。毫不夸张地说，从 Swift 正式诞生的第一分钟开始，我就在学习这门语言。虽然天资驽钝，不得其所，但是在这段集中学习和实践的时间里，也还算总结了一些心得，而我把这些总结加以整理和示例，以一个个的小技巧和知识点的形式，编写成了这本书。全书共有 100 节，每一节都是一个相对独立的主题，涵盖了一个中高级开发人员需要知道的 Swift 语言的方方面面。

这本书非常适合用作官方文档的参考和补充，也会是中级开发人员很喜爱的 Swift 进阶读本。具体每节的内容，可以参看本书的目录。

这本书不是什么

这本书**不是** Swift 的入门教程，也**不会**通过具体的完整实例引导你用 Swift 开发出一个像是计算器或者记事本这样的 app。这本书的目的十分纯粹，就是探索那些不太被人注意，但是又在每天的开发中可能经常用到的 Swift 特性。这本书并不会系统地介绍 Swift 的语法和特性，因为基于本书的写作目的和内容特点，采用松散的模式和非线性的组织方式会更加适合。

[1] *http://numbbbbb.gitbooks.io/-the-swift-programming-language-/*

换言之，如果你是想找一本 Swift 从零开始的书，那这本书不应该是你的选择。你可以在阅读 Apple 文档后再考虑回来看这本书。

组织形式和推荐阅读方式

100 个 Tip 其实不是一个小数目。本书每节的内容是相对独立的，也就是说你没有必要从头开始看，随手翻开到任何一节都是没问题的。当然，按顺序看是最理想的阅读方式，因为在写作时我特别注意了让靠前的节不涉及后面节的内容；另一方面，位置靠后的节如果涉及之前节的内容的话，我添加了相关节的交叉引用，这可以帮助迅速复习和回顾之前的内容。我始终坚信不断地重复和巩固，是真正掌握知识的唯一途径。

您可以通过目录快速地在不同节之间选择自己感兴趣或需要了解的内容。如果遇到您不感兴趣或者已经熟知的节，您也完全可以暂时先跳过去，这不会影响您对本书的阅读和理解。

建议您阅读本书时开启 Xcode 环境，并且对每一节中的代码进行验证，这有利于您真正理解代码示例想表达的意思，也有利于记忆的形成。每一段代码示例都不太长，却经过了精心的准备，能很好地说明本节内容，希望您在每一处都能通过代码和我进行心灵上的“对话”。

代码运行环境

书中每一节基本都配有代码示例的说明。这些代码一般来说包括 Objective-C 或者 Swift 的代码。理论上来说所有代码都可以在 Swift 1.1 （也就是 Xcode 6.1）版本环境下运行。当然因为 Swift 版本变化很快，可能部分代码需要微调或者结合一定的上下文环境才能运行，但我相信这种调整是显而易见的。如果您发现明显的代码错误和无法运行的情况，欢迎随时与我联系，我将尽快修正。

如果没有特别说明，这些代码在 Playground 和项目中都应该可以运行，并拥有同样表现。但是也存在一些代码只能在 Playground **或者**项目文件中才能正确工作的情况，这主要是因为平台限制的因素，如果出现这种情况，我都会在相关节中特别加以说明。

勘误和反馈

Swift 仍然在高速发展中，本书当前版本是基于 Swift 1.1 的。随着 Swift 的新特性引入及错误修正，本书难免会存在部分错误或者过时的情况。虽然我会随着 Swift 的发展继续不断完善

和修正这本书，但是这个过程亦需要时间。

另外由于作者水平有限，书中也难免会出现一些错误或者纰漏，如果您在阅读时发现了任何问题，可以直接向我反馈，我将尽快确认和修正。

致谢与提醒

首先想感谢您购买了这本书。我其实是怀着忐忑的心情写下这些文字的，小心翼翼地希望没有触动太多人。这本书所提供的知识我想应该是超过它的售价的，但在选择前还是请您再三考虑。

目录

Swift 新元素

Tip 1　柯里化 (Currying)

在 Swift 中可以将方法进行柯里化（Currying）[1]，也就是把接受多个参数的方法变换成接受第一个参数的方法，并且返回接受余下的参数、返回结果的新方法。举个例子，在 Swift 中我们可以这样写出多个括号的方法：

```
func addTwoNumbers(a: Int)(num: Int) -> Int {
    return a + num
}
```

然后通过只传入第一个括号内的参数进行调用，这样将返回另一个方法：

```
let addToFour = addTwoNumbers(4)    // addToFour 是一个 Int -> Int
let result = addToFour(num: 6)      // result = 10
```

或者：

```
func greaterThan(comparor: Int)(input : Int) -> Bool{
    return input > comparor;
}

let greaterThan10 = greaterThan(10);

greaterThan10(input : 13)    // 结果是 true
greaterThan10(input : 9)     // 结果是 false
```

柯里化是一种量产相似方法的好办法，可以通过柯里化一个方法模板来避免写出很多重复代码，也方便了今后维护。

举一个实际应用时的例子，在 Selector 一节中，我们提到了在 Swift 中 Selector 只能使用字符串生成。这面临一个很严重的问题，就是难以重构，并且无法在编译期间进行检查，其实这是十分危险的行为。但是 target-action 又是 Cocoa 中如此重要的一种设计模式，无论何

[1] *http://en.wikipedia.org/wiki/Currying*