

Swift 全解析

新式 iOS 实战开发

张云波 冼树华 丁美程 ◎编著



移动开发系列

Swift 全解析：新式 iOS 实战开发

张云波 冼树华 丁美程 编著



電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书从 Swift 4 开始,从 Swift 全面的语法,到 iOS 11 和 iOS 12 的入门级 App 开发,最后由一个非常实用的 iOS 小游戏开发结束。本书是一本适合新手从零开始,一步一步地坚实掌握 iOS 开发技术的一站式启蒙书。本书还配有语法和 iOS 基础开发的视频,可以有效地提升读者的学习效率和理解维度,增强学习兴趣。另外,读者可以与作者无缝交流,快乐地学习 iOS 开发。

本书适合广大编程爱好者及对 iOS 开发感兴趣的读者阅读。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

Swift 全解析:新式 iOS 实战开发/张云波,冼树华,丁美程编著. —北京:电子工业出版社,2018.7
(移动开发系列)

ISBN 978-7-121-34610-1

I. ①S… II. ①张… ②冼… ③丁… III. ①程序语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字(2018)第 142617 号

策划编辑:张 迪

责任编辑:底 波

印 刷:三河市良远印务有限公司

装 订:三河市良远印务有限公司

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:787×1092 1/16 印张:26.25 字数:672 千字

版 次:2018 年 7 月第 1 版

印 次:2018 年 7 月第 1 次印刷

定 价:99.00 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, 88258888。

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。

本书咨询联系方式:(010) 88254469; zhangdi@phei.com.cn。

前言

借 iOS 每年一次大幅更新的东风，经过几个版本的成长，Swift 4 已经非常成熟了，在 iOS 开发中的便利程度和第三方库的生态建设上，它实际已经大幅领先 Objective-C。苹果所有的官方文档和视频资料均以 Swift 语言为主。这个语言的原作者虽然离开了苹果公司，但 Swift 本身的发展丝毫没有减缓。笔者认为，除了维护垂垂老矣的 Objective-C 项目，新项目完全可以用 Swift 4 从头开始实施。拥抱强类型、类型推断、Optional、面向协议、泛型编程这些现代化编程语言必备的特性，是每个程序员必备的功课之一。

本书分为 3 篇：Swift 4 基础语法篇、iOS 11 基础开发篇和游戏实战篇。若读者在阅读本书时遇到问题，可以进入 QQ 群 532084830，与众多开发者沟通或下载相关资料，也可以在各大视频平台、小波说雨燕官网（xiaoboswift.com）观看本书配套的视频课程，这会使读者的学习效率大幅提升！

作者简介



小波

上海交通大学毕业,《小波说雨燕》系列 Swift 教学视频作者,表情包式的表演成功吸引了新手的注意。为人勇敢而懦弱,大气却苛刻,长不大的 IT 宅男。人生如戏,常靠临时演技,曾为看动漫而学习日语,后又莫名其妙地获得了东京的工作机会。目前致力于移动 App 领域的开发和培训工作。你有酒,我有故事,直播说给你听。



华仔

暨南大学珠海学院毕业, IT 大叔一枚,从.Net 到 Java, Android 再到 iOS,从 Winform 到 H5,桌面到 Web, Web 到移动 App,产品研发到数据分析等工作都有涉及,现致力于移动互联网金融相关工作,最近迷上小游戏。



丁 MC

毕业于安徽工业大学,自学 iOS 开发,参与或独立开发过多款 App 上架 APPStore,目前就职于一家小型软件科技公司任技术总监。一个不甘平凡却依旧平凡的平凡人,相信生活不止眼前的苟且,还有诗和远方。敢于冒险和折腾,正准备着走向创业的不归路。

参与本书编写的还有王倩影、张佳君、王韬懿、陈彬、贾建博、胡佳佳、王艳鹏、郭艾超。



第 1 篇 Swift 4 基础语法篇

第 1 章 关于 Swift	3
第 2 章 Swift 语法基础	7
2.1 变量和常量	7
2.2 注释	8
2.3 分号	9
2.4 整数	9
2.5 浮点数	10
2.6 类型安全和类型推断	10
2.7 数值型字面量	10
2.8 数值类型转换	11
2.9 布尔类型	13
2.10 元组	13
2.11 可选类型	14
2.12 错误处理	15
2.13 断言	16
第 3 章 Swift 基本运算符	17
3.1 术语	17
3.2 赋值运算符	17
3.3 算术运算符	18
3.4 组合赋值运算符	19
3.5 比较运算符	19
3.6 三元（目）运算符	20
3.7 空合运算符	21
3.8 区间运算符	22
3.9 逻辑运算符	23
第 4 章 Swift 字符串和字符	26

4.1	字符串字面量	26
4.2	初始化空字符串	26
4.3	字符串可变性	27
4.4	字符串都是值类型	27
4.5	使用字符	27
4.6	连接字符串与字符	28
4.7	字符串插入值	28
4.8	Unicode	29
4.9	计算字符数量	31
4.10	访问与修改字符串	31
4.11	比较字符串	33
第 5 章	Swift 集合类型	35
5.1	数组	35
5.2	集	39
5.3	字典	42
第 6 章	Swift 控制流	46
6.1	for 循环	46
6.1.1	for-in	47
6.1.2	for 条件递增	48
6.2	while 循环	48
6.2.1	while	49
6.2.2	repeat while	49
6.3	if 语句	50
6.4	guard 语句	51
6.5	switch 语句	52
6.6	控制转移语句	54
6.6.1	continue 语句	54
6.6.2	break 语句	55
6.6.3	fallthrough 语句	56
6.6.4	return 语句	57
第 7 章	Swift 函数	58
7.1	函数的定义和调用	58
7.2	函数的参数和返回值	59
7.2.1	多个输入参数的函数	59
7.2.2	没有参数的函数	59

7.2.3	没有返回值的函数	60
7.2.4	没有返回值的函数调用具有返回值的函数	60
7.2.5	多个返回值的函数	60
7.2.6	元组可选类型返回值的函数	61
7.3	函数的参数名称	62
7.3.1	外部参数名称	62
7.3.2	带默认值参数的函数	62
7.3.3	可变参数的函数	63
7.3.4	常量参数和变量参数	63
7.3.5	输入和输出参数	64
7.4	函数类型	65
7.4.1	使用函数类型	65
7.4.2	使用函数类型作为参数	65
7.4.3	使用函数类型作为返回值	66
7.4.4	使用函数类型作为常量	66
7.5	嵌套函数	67
第 8 章	Swift 闭包	68
8.1	闭包的定义	68
8.2	闭包表达式	69
8.2.1	sorted 函数	69
8.2.2	使用闭包表达式	70
8.2.3	根据上下文推断参数	70
8.2.4	单表达式闭包省掉返回关键字	70
8.2.5	参数名称缩写	71
8.2.6	运算符函数	71
8.3	挂尾闭包	71
8.4	闭包的值捕获	72
8.5	闭包是引用类型	73
第 9 章	Swift 枚举	74
9.1	枚举的定义、结构和使用	74
9.2	枚举在 switch 语句中的使用	75
9.3	枚举关联值	75
9.4	枚举原始值	76
第 10 章	Swift 类和结构体	77
10.1	类和结构体的对比	77

10.1.1	类和结构体的定义	78
10.1.2	类和结构体的实例	78
10.1.3	类和结构体的属性访问	78
10.2	结构体是值类型	79
10.3	类是引用类型	79
10.4	如何选择类和结构体	79
10.5	集合类型的赋值和复制	80
第 11 章	Swift 属性	81
11.1	存储属性	81
11.1.1	普通存储属性	81
11.1.2	延迟存储属性	82
11.2	计算属性	82
11.3	属性观察器	84
11.4	全局变量和局部变量	85
11.5	静态属性	85
第 12 章	Swift 方法	86
12.1	实例方法	86
12.1.1	使用类中定义的实例方法	86
12.1.2	使用结构体中定义的实例方法	87
12.1.3	使用枚举中定义的实例方法	88
12.2	静态方法	89
12.2.1	使用类中定义的静态方法	89
12.2.2	使用结构体中定义的静态方法	89
第 13 章	Swift 下标	91
13.1	下标的定义	91
13.2	下标的用法	92
13.3	下标脚本	93
第 14 章	Swift 继承	95
14.1	继承的定义	95
14.2	继承的一般用法	95
14.3	重写父类的方法和属性	96
14.3.1	重写父类的方法	96
14.3.2	重写父类的属性	97
14.4	防止父类的特性被重写	98

第 15 章	Swift 初始化	99
15.1	初始化的定义	99
15.2	为存储属性设置初始值	99
15.3	自定义构造函数	100
15.4	默认构造函数	101
15.5	值类型的初始化	102
15.6	引用类型的初始化	103
15.6.1	指定构造函数和便利构造函数	104
15.6.2	类的代理初始化调用	104
15.6.3	两段式构造过程	105
15.6.4	构造函数的继承和重写	107
15.6.5	自动构造函数的继承	108
15.6.6	指定和便利构造器的实战	109
15.7	可失败构造函数	111
15.8	必要构造函数	116
15.9	通过闭包或方法设置属性的默认值	116
第 16 章	Swift 析构	119
16.1	析构的原理	119
16.2	析构的实战	119
第 17 章	Swift 错误处理	122
17.1	创建错误类型	122
17.2	抛出错误	123
17.3	捕获和处理错误	124
17.4	不处理错误	126
17.5	将错误转为可选类型	126
第 18 章	Swift 泛型	128
18.1	泛型能解决的问题	128
18.2	泛型函数	129
18.3	类型参数	130
18.4	命名类型参数	131
18.5	泛型类型	131
18.6	泛型类型的扩展	133
18.7	类型约束	134
18.8	关联类型	136
18.9	条件语句	138

第 19 章 Swift 扩展 140

19.1 扩展的定义和语法 140

19.2 扩展计算属性 140

19.3 扩展构造函数 141

19.4 扩展方法 143

19.5 扩展下标 144

19.6 扩展嵌套函数 144

第 20 章 Swift 协议 146

20.1 协议的定义和语法 146

20.2 对属性的规定 147

20.3 对方法的规定 148

20.4 对变异方法的规定 149

20.5 对构造函数的规定 150

20.6 协议也是一种类型 151

20.7 代理 152

20.8 在扩展中添加协议 155

20.9 协议类型的集合 155

20.10 协议的继承 156

20.11 协议类型的组合 157

20.12 检查协议的一致性 158

第 2 篇 iOS 11 基础开发篇

第 21 章 入门级 163

21.1 开发环境配置 163

21.2 Playground 164

21.3 第一个 App 168

21.4 创建用户交互 178

第 22 章 初级 181

22.1 Auto Layout 181

22.2 Stack View 184

22.3 Size Class 192

22.4 Table View 197

22.4.1 原型单元格 199

22.4.2 代理模式 201

22.4.3 连接数据源 203

22.5	自定义单元格	204
22.6	单元格交互	216
22.7	单元格滑动菜单	219
22.8	导航和转场	226
第 23 章	中级	235
23.1	Model 重构	235
23.2	静态单元格	241
23.3	导航栏、状态条和动态字体的定制技巧	248
23.4	视差动画	256
23.5	特效、动画和手势	259
23.6	JSON 编/解码	271
23.7	录入新数据场景	276
23.8	真机部署	285
23.9	相册、拍照和录制视频	287

第 3 篇 游戏实战篇

第 24 章	卡牌斗兽棋	299
24.1	项目介绍与分析	299
24.2	项目框架搭建	301
24.3	首页和棋盘控制器的搭建	306
24.4	相关 model 和走棋逻辑的实现	330
24.5	单机双人对战的实现	337
24.6	人机对战的实现	360
24.7	联网对战的实现	371
24.8	联网对战的优化	400

第 1 章

关于 Swift



第 1 篇



Swift 4 基础语法篇

第 1 章

关于 Swift

Swift 是一门极其出色的编程语言，我们可以使用它编写苹果手机 App、基于 Mac 使用的桌面应用程序、苹果手表应用，甚至服务器端的应用程序等！它是一门安全、快速、交互式的编程语言，是现代先进的编程语言与苹果工程师的智慧和各大开源社区的贡献三者结合发展起来的。Swift 对于之前的版本，编译器和语言都进行了优化升级，其图标如图 1-1 所示。



图 1-1

Swift 是一门很好的编程语言，它兼具工业级别的编程语言质量，也具备脚本语言的编程易学易用的特点。我们使用 Swift 在 Playground 中编写代码，不需要任何编译和运行 App，就能直接看到编写代码后的结果，做到所写即所得。

Swift 采用现代编程模式，定义了大量的常见编程错误：

- 变量在使用前检查是否初始化；
- 数组使用索引前检查是否越界；
- 整数使用前检查是否溢出；
- 可选项检查空值是否已显式处理；
- 内存会自动管理；
- 错误处理允许从失败中恢复。

Swift 代码经过编译和优化，能更充分地使用现代的硬件。Swift 的语法和标准库是根据易于学习和使用这一设计思想指导而创建的。Swift 的安全性和运行速度，使它能够轻松编写出“Hello World”到整个操作系统应用级别的程序。

Swift 使用现代的轻量级的语法，集合强大的类型推断和匹配模式，使一个复杂的想法

通过一个简单、明确的方式表达出来。因此，代码不仅容易编写，也很容易阅读和维护。（代码是写给别人看的，不仅是自己能看懂。）

Swift 已经面世多年（已经到第 4 代了），它会继续发展新的特性和功能。我们对 Swift 充满信心，读者赶紧学习吧！

为了增加读者的自信心，下面开始我们的第一个 Swift 的应用程序。

（1）首先，打开 Xcode（写作的版本是 Xcode 9.2 正式版），选择“Get started with a playground”，如图 1-2 所示。



图 1-2

（2）然后选择“iOS”和“Blank”，如图 1-3 所示。

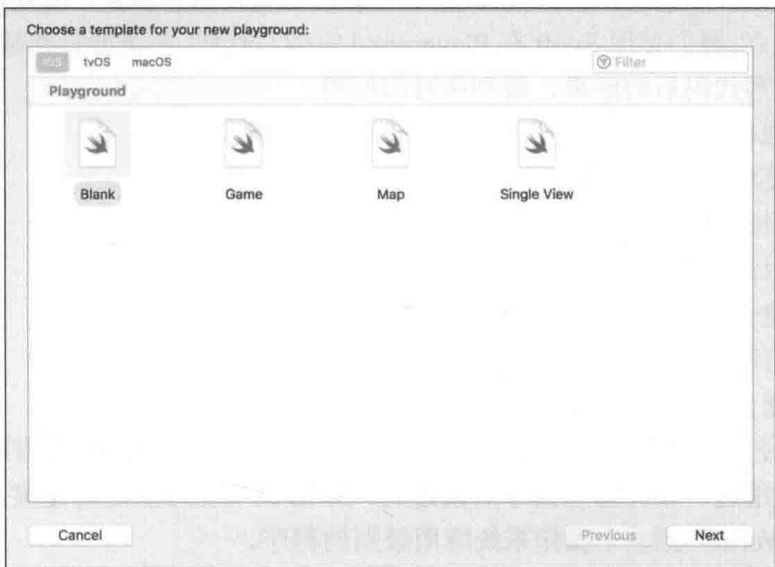


图 1-3

(3) 选择好保存的地方，单击“Create”按钮来创建程序，如图 1-4 所示。

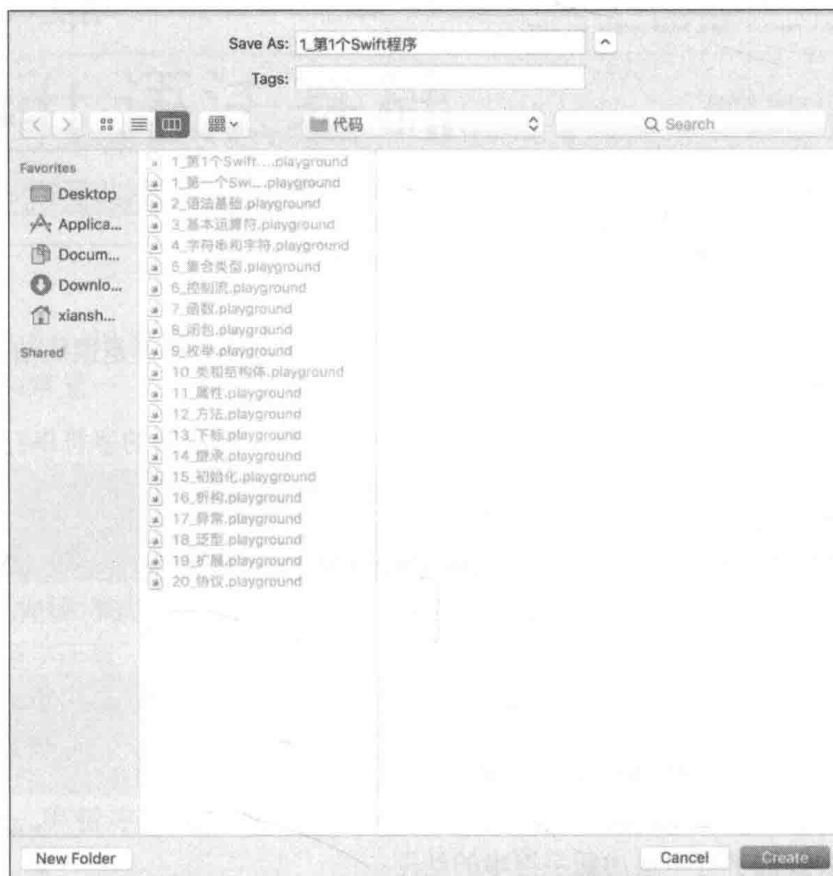


图 1-4

(4) 接下来会看到 Xcode 已经帮我们写了一段程序，如图 1-5 所示，图中标注 1 为代码区，标注 2 为显示代码执行结果的结果区

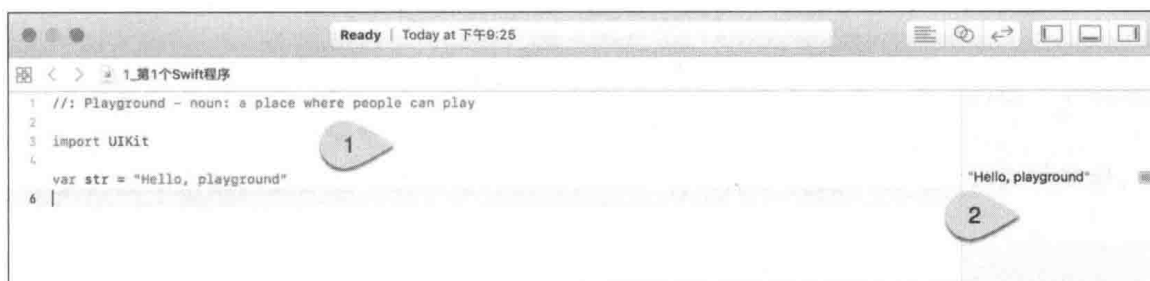


图 1-5

(5) 我们来更改一下程序，把代码在隐藏的调试区里打印出来，代码如下。

```
var str = "Hello, 大家一起学雨燕"
print(str)
```

(6) 打开隐藏的调试区，把打印的信息显示出来，如图 1-6 所示。