



Learn Swift 2 on the Mac

SECOND EDITION

Waqar Malik

Apress®

Learn Swift 2 on the Mac

Second Edition



Waqar Malik

Apress®

Learn Swift 2 on the Mac, Second Edition

Copyright © 2015 by Waqar Malik

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

ISBN-13 (pbk): 978-1-4842-1628-6

ISBN-13 (electronic): 978-1-4842-1627-9

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director: Welmoed Spahr

Lead Editor: Michelle Lowman

Technical Reviewer: Felipe Laso Marsetti

Editorial Board: Steve Anglin, Pramila Balan, Louise Corrigan, James T. DeWolf, Jonathan Gennick, Robert Hutchinson, Celestin Suresh John, Michelle Lowman, James Markham, Susan McDermott, Matthew Moodie, Jeffrey Pepper, Douglas Pundick, Ben Renow-Clarke, Gwenan Spearing

Coordinating Editor: Mark Powers

Copy Editor: Karen Jameson

Compositor: SPi Global

Indexer: SPi Global

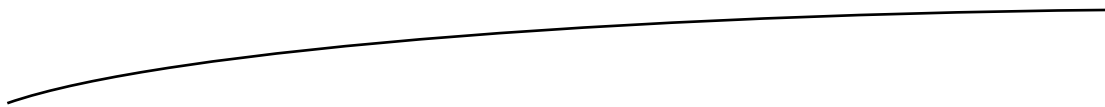
Artist: SPi Global

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a Delaware corporation.

For information on translations, please e-mail rights@apress.com, or visit www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales-eBook Licensing web page at www.apress.com/bulk-sales.

Any source code or other supplementary materials referenced by the author in this text is available to readers at www.apress.com/9781484216286. For detailed information about how to locate your book's source code, go to www.apress.com/source-code/. Readers can also access source code at SpringerLink in the Supplementary Material section for each chapter.



For my family, Irrum, Mishal, and Adam.

Contents at a Glance

About the Author	xvii
About the Technical Reviewer	xix
Acknowledgments	xxi
Introduction	xxiii
■ Chapter 1: Hello Swift.....	1
■ Chapter 2: The Swift Playground in Xcode	15
■ Chapter 3: Accessing Swift’s Compiler and Interpreter: REPL	29
■ Chapter 4: Constants, Variables, and Data Types	35
■ Chapter 5: Expressions.....	49
■ Chapter 6: Operators	59
■ Chapter 7: Flow Control	71
■ Chapter 8: Functions	87
■ Chapter 9: Closures	95
■ Chapter 10: Enumerations	101
■ Chapter 11: Classes and Structures	109
■ Chapter 12: Methods	121

■ Chapter 13: Inheritance.....	127
■ Chapter 14: Extensions.....	133
■ Chapter 15: Memory Management and ARC.....	139
■ Chapter 16: Error Handling.....	151
■ Chapter 17: Protocols.....	157
■ Chapter 18: Generics	167
■ Chapter 19: Access Control	177
■ Chapter 20: Interoperability with Objective-C.....	185
■ Chapter 21: Working with Core Data	201
■ Chapter 22: Consuming RESTful Services	219
Index.....	227

Contents

About the Authorxvii

About the Technical Reviewerxix

Acknowledgmentsxxi

Introductionxxiii

■ Chapter 1: Hello Swift..... 1

 Improvements over Objective-C 2

 Type Inference 2

 Type Safety 2

 Control Flow..... 2

 Optionals..... 3

 Strings 3

 Unicode..... 3

 Other Improvements 3

 Requirements 3

 Getting Xcode 4

 Quick Tour of Xcode..... 5

Quick Tour of Swift	9
Basic Types.....	9
Collection Types.....	10
Control Flow.....	10
Functions.....	11
Objects.....	11
Generics.....	12
Getting the Sample Code	13
Summary.....	13
■ Chapter 2: The Swift Playground in Xcode	15
Getting Started with a Playground	15
Rich Text Comments	17
Custom QuickLook Plug-Ins	20
To develop custom plug-ins.....	21
XCShowView.....	21
XCCaptureValue	21
XCPSetExecutionShouldContinueIndefinitely.....	21
Custom Modules for Playground	22
Importing Your Code	22
Summary.....	27
■ Chapter 3: Accessing Swift's Compiler and Interpreter: REPL	29
What Is REPL?	29
LLDB and the Swift REPL.....	31
Summary.....	33
■ Chapter 4: Constants, Variables, and Data Types	35
Type Annotation.....	35
Identifiers	36
Console Output.....	36
Integers	37
Floating-Point Numbers	38

Numeric Literals	38
Conversion	38
Booleans.....	39
Characters.....	39
Strings.....	40
Collection Types.....	41
Arrays	42
Sets.....	44
Dictionaries.....	44
Tuples	46
Optionals.....	46
Summary.....	48
Chapter 5: Expressions.....	49
Primary Expressions.....	50
Prefix Expressions	50
Try Operator.....	50
Postfix Expressions.....	51
Binary Expressions.....	51
Assignment Operator.....	52
Ternary Conditional	52
Casting Operators.....	53
Self and Super.....	53
Closures and Functions.....	54
Closures	54
Function Calls.....	56
Implicit Member Expression.....	56
Optionals	57
Optional Chaining	58
Summary.....	58

■ Chapter 6: Operators	59
Syntax	59
Notation	59
Precedence	60
Associativity	60
Swift Operators	60
Prefix	60
Infix	61
Postfix	68
Overloading Operators	68
Unary Operator	69
Binary Operators	70
Summary	70
■ Chapter 7: Flow Control	71
For Loops	71
For-in	71
For-conditional-Increment	73
While	74
Repeat-while	75
Branch Statements	75
Switch	77
Range Matching	79
Control Transfer Statements	81
Summary	85
■ Chapter 8: Functions	87
Defining Functions	87
Calling a Function	88
Parameter Names	89
Default Values	90
Variadic Parameters	91

Mutability of Parameters	91
In-Out Parameters	91
Function Types	92
Functions as Parameters.....	92
Functions as Return Values.....	93
Nested Functions	93
Summary	94
■ Chapter 9: Closures	95
Closure Syntax	95
Inferring Types from Context	97
Implicit Returns	97
Shorthand Argument Names	97
Trailing Closures.....	97
Capturing Values	98
Summary	99
■ Chapter 10: Enumerations	101
Syntax	101
Switch Statement and Enumerations.....	102
Associated Values	103
Raw Values.....	104
Recursive Enumerations	106
Summary	107
■ Chapter 11: Classes and Structures	109
Commonality	109
Definition	110
Initialization.....	111
Accessing Properties.....	112
Value Types vs. Reference Types.....	113
Choosing Between Classes or Structures	114

Properties.....	114
Stored Properties	114
Lazy Stored Properties	115
Computed Properties.....	116
Property Observers.....	117
Type Properties.....	118
Summary	119
■ Chapter 12: Methods	121
Instance Methods.....	121
Modifying Type State	123
Type Methods	124
Summary	125
■ Chapter 13: Inheritance	127
Terminology.....	127
Defining a Base Class.....	128
Subclassing	128
Properties.....	130
Preventing Overriding.....	131
Summary	131
■ Chapter 14: Extensions	133
Creating an Extension	134
Computed Properties.....	135
Initializers.....	135
Methods	136
Mutating Methods	136
Subscripts	137
Nested Types	137
Summary	138

■ Chapter 15: Memory Management and ARC	139
Object Life Cycle	140
Reference Counting	140
Object Ownership	140
ARC	141
Strong Reference Cycles	142
Resolving Strong Reference Cycles	143
Weak References	144
Unowned Reference	145
Strong Reference Cycles and Closures	147
Summary	149
■ Chapter 16: Error Handling	151
Representing Errors	151
Handling Errors	152
Propagating Errors	152
Handling Errors	154
Optional Handling	155
Asserting Errors	155
Cleanup Actions	155
Summary	156
■ Chapter 17: Protocols	157
Syntax	157
Properties	158
Methods	159
Initializers	160
Protocols as Types	161
Delegation	161
Conformance with Extensions	163
Protocols and Collection Types	164

Protocol Inheritance	164
Protocol Composition.....	165
Protocol Conformance.....	165
Optional Requirements.....	165
Summary	165
■ Chapter 18: Generics	167
Generic Functions	167
Generic Types	169
Extensions	171
Associated Types	171
Summary	175
■ Chapter 19: Access Control	177
Modules and Source Files	177
Access Levels.....	178
Syntax	178
Classes	179
Subclassing	180
Class Members.....	180
Functions.....	180
Enumerations	181
Nested Types	181
Getters and Setters	182
Initializers.....	182
Protocols	182
Extensions	183
Type Alias	183
Summary	183

■ Chapter 20: Interoperability with Objective-C	185
Import Process	188
Interoperability	188
Nullability and Optionals	190
Object Initialization.....	191
Failable Initializers	192
Properties.....	192
Methods	193
Blocks.....	193
Object Comparison	194
Type Compatibility	195
Objective-C Generics.....	196
Dynamic Dispatch	197
Selectors	198
Property Attributes	198
Namespaces and Class	198
Cocoa Data Types	199
Foundation Functions.....	199
Core Foundation	199
Interacting with C	200
Summary	200
■ Chapter 21: Working with Core Data	201
NSManagedObjectContext.....	202
NSManagedObject.....	202
NSManagedObjectModel.....	202
NSPersistentStoreCoordinator.....	202
NSFetchRequest.....	202
NSPredicate.....	203
Creating An Application	203

Defining Data Objects.....	205
Adding an Object Editor	208
Showing the Editor	214
Entity Classes	214
Summary	218
■ Chapter 22: Consuming RESTful Services	219
HTTP Methods	219
Being Stateless	220
Endpoints	220
Data Formats.....	220
Network Access.....	220
Security	225
Summary	226
Index.....	227

About the Author



Waqar Malik worked at Apple helping developers write Cocoa applications for the Mac during the early days of Mac OS X. Now he develops applications for various Apple OS platforms. He is the co-author of *Learn Objective-C on the Mac* (Apress, 2012).

About the Technical Reviewer



Felipe Laso Marsetti is an iOS programmer working at Lextech Global Services. He loves everything related to Apple, video games, cooking, and playing the violin, piano, or guitar. In his spare time, Felipe loves to read and learn new programming languages or technologies.

Felipe likes to write on his blog at <http://iFe.li>, create iOS tutorials and articles as a member of www.raywenderlich.com, and work as a technical reviewer for Objective-C and iOS-related books. You can find him on Twitter as @Airjordan12345, on Facebook under his name, or on App.net as @iFeli.



Acknowledgments

I'd like to give thanks to all the folks at Apress who helped complete this book. This book would not have been possible without their assistance.

Introduction

Whenever developers come to a new platform, they are faced with the task of getting to know unfamiliar development tools, design patterns, the standard frameworks available in the new environment, and perhaps even a new programming language.

Most of the time, this is all done while trying to deliver an application as soon as possible. In such situations, developers tend to fall back on the patterns and approaches they are familiar with from previous environments, which too often results in code that doesn't fit the new environment, or in duplicate code that might already be provided by the built-in frameworks. This can cause problems down the road or delays in delivery.

It would be great to have colleagues already familiar with the platform who could offer guidance to get you going in the right direction. Well, it's not always possible to have mentors to help you, and that's where this book steps in—to be your mentor.

The author of this book is a veteran of Apple's Developer Technical Services organization, and has answered countless questions from software engineers who are new to Apple technology. That experience results in a book that anticipates the most common misunderstandings and takes care to explain not only the how, but also the why of Apple's development platform.

Learn Swift 2 on the Mac provides a step-by-step guide that will help you acquire the skills you need to develop applications for OS X, iOS, watchOS, and tvOS.

Chapter 1

Hello Swift

Swift is a new language designed by Apple for developing iOS and OS X applications. It takes the best parts of C and Objective-C and adapts them with modern features and patterns. Swift-compiled programs will run on iOS7 or newer and OS X 10.9 (Mavericks) or newer.

The two main goals for the language are compatibility with the Cocoa and Cocoa Touch frameworks and safety, as you'll see in the upcoming chapters. If you've been using Objective-C, especially the modern syntax, Swift will feel familiar.

However, Swift's syntax is actually a major departure from Objective-C. It takes lots of cues from programming languages such as Haskell, C#, Ruby, and Python.

Some of the technologies we will cover in this book are the following:

- Automatic reference counting
- Closures (blocks)
- Collection literals
- Modules
- Frameworks
- Objective-C runtime
- Generics
- Operator overloading
- Tuples
- Namespaces
- Error Handling

Improvements over Objective-C

Let's take a quick look at some of the features that make Swift better than Objective-C. We will cover these topics in detail in later chapters.

Type Inference

One of the key features of Swift is type inference. But what is type inference? It means that you can figure out the type of the variable by the value it is assigned. There is usually no need to specify the type of variables (though you can always specify them); the types of the variables can be inferred by the value being set.

Type Safety

Swift variables must have a type and must be initialized before they can be used. Conversion between different types is done explicitly; you cannot simply assign a float to an integer.

```
let floatValue : Float = 4.0
let doubleValue : Double = floatValue // This is an error
```

There is not automatic conversion even though Double can hold the value that float holds. To correctly convert, we need to create a new value of type before assigning:

```
let doubleValue : Double = Double(floatValue)
```

Swift compiler knows more about types in method calls and uses table look-up for method dispatch instead of the dynamic dispatch that is used by Objective-C. Static method dispatch via table look-up enables more checks and validation at compile time, even in the playground. As soon as you enter an expression in the playground, the compiler evaluates it and lets you know of any possible issues with the statement; you can't run your program until you fix those issues. Here are some features that enhance safety:

- Variables and constants are always initialized before use.
- Array bounds are always checked.
- Raw C pointers are not readily available and are discouraged.
- Assignment statements do not return values.
- Overflows are trapped as runtime errors.

Control Flow

The switch statement has undergone a major overhaul. Now it can select based not only integers, but also on strings, floats, and ranges of items, expressions, enums, and so forth. Moreover, there's no implicit fall-through between case statements.

Also introduced a *guard* statement. A guard statement is like an if statement but with an additional requirement of always having an else statement.

Optionals

Variable values can be optional. What does that mean? It means that a variable will either be nil or it will have a valid value. The nil value is distinct from any valid value. Optionals can also be chained together to protect against errors and exceptions.

Strings

Strings in Swift are much easier to work with; they have a clear, simple syntax. You can concatenate strings using the `+=` operator. The mutability of the strings is defined by the language, not the `String` object. You declare a string as either mutable or nonmutable with the same `String` object, by using either the `let` or `var` keywords.

Unicode

Unicode is supported at the core: You can define variable names and function names using full Unicode. The `String` and `Character` types are also fully Unicode compliant and support various encodings, such as UTF-8, UTF-16, and 21-bit Unicode scalars.

Other Improvements

- Header files are no longer required.
- Functions are full-fledged objects; they can be passed as arguments and returned from other functions. Functions can be scoped similarly to instance variables.
- Comments can be nested.
- There are separate operators for assignment (`=`) and comparison (`==`), and there's even an identity operator (`===`) for checking whether two data elements refer to the same object.
- There is no defined entry point for programs such as `main`.

If all this sounds good to you (it does to me), let's go get the tools to start playing with Swift.

Requirements

Before you can begin playing with Swift, you need to download and install Xcode, the IDE that's used to build applications for iOS and OS X. You'll need Xcode 7.1 or later.

It's really easy to download and install Xcode. Here are the basic requirements:

- Intel-based Macintosh computer
- OS X 10.11 El Capitan (or later)
- 15GB of free disk space
- An Internet connection to download Xcode and documentation
- An iOS device running iOS 9 (or later)

Note As a rule, the later the version of the OS, the better. The examples in the book are developed using Xcode 7.1 running on OS X 10.11 El Capitan and for iOS 9.1 running on iPhone 6s.

Getting Xcode

Launch the App Store application and use the search bar on the top right to search for Xcode. You can then get more information by selecting Xcode, as shown in Figure 1-1, or install it by selecting the Install button.



Figure 1-1. Xcode on App Store

When you launch Xcode for the first time, it will download and install other required items in order to complete the installation. If you have multiple versions of Xcode installed, be sure to select Xcode version 7 or later for the command-line tools. You can do this by selecting **Xcode ► Preferences**, then choosing the **Locations** tab as shown in Figure 1-2.