Vorwort

Als Apple 2014 die neue Programmiersprache Swift vorstellte, fragten sich viele Entwickler: Brauchen wir wirklich eine neue Programmiersprache? Mittlerweile erübrigt sich die Frage. Swift ist der De-facto-Standard für neue Projekte im Apple-Universum. Objective-C ist damit nicht obsolet (vermutlich gibt es Milliarden Zeilen Code, der weiter gewartet werden muss), aber wer es sich aussuchen darf, wird neue Apps mit Swift entwickeln.

Warum Swift?

Swift ist für Apple ein Befreiungsschlag: Objective-C ist zwar noch immer das Fundament von macOS, iOS etc. – aber das ändert nichts daran, dass diese Sprache in den 1980er-Jahren entworfen wurde und in keinerlei Hinsicht mit modernen Programmiersprachen mithalten kann.

Swift ist dagegen ein sauberer Neuanfang. Bei der Vorstellung wurde Swift auch *Objective-C without the C* genannt. Natürlich ist Swift von Objective-C beeinflusst – schließlich muss Swift kompatibel mit den unzähligen Apple-Bibliotheken sein. Swift realisiert viele neue Ideen, greift aber auch Konzepte von C#, Haskell, Java, Python und anderen Programmiersprachen auf. Daraus ergeben sich mehrere Vorteile:

- Swift zählt zu den modernsten Programmiersprachen, die es momentan gibt.
- Code lässt sich in Swift syntaktisch eleganter formulieren als in Objective-C.
- Der resultierende Code ist besser lesbar und wartbar.
- Swift ist für Programmierer, die schon Erfahrung mit anderen modernen Sprachen gesammelt haben, wesentlich leichter zu erlernen als Objective-C. Vorhandenes Know-how lässt sich einfacher auf Swift als auf Objective-C übertragen.
- Swift ist ein Open-Source-Produkt und steht auch für Linux zur Verfügung. Der Entwicklungsprozess erfolgt offen und transparent.

Neu in Swift 5

Swift 5 wurde von Apple als die Version postuliert, mit der Swift gleichsam »erwachsen« wird. Apple garantiert mit Swift 5 die ABI-Stabilität. Das *Application Binary Interface* (ABI) beschreibt die Schnittstelle zwischen Programmen bzw. Bibliotheken auf binärer Ebene, also zur Laufzeit. Im Unterschied dazu betrifft das *Application Programming Interface* (API) »nur« den Quellcode.

Das Erreichen der ABI-Stabilität bedeutet, dass sich ein mit Swift kompiliertes Programm darauf verlassen kann, dass das Zusammenspiel mit einer zu einem anderen Zeitpunkt und von einem anderen Entwickler kompilierten Bibliothek funktioniert — auch dann, wenn die App mit Swift 5.0 kompiliert ist, die (z. B. unter iOS vorinstallierte) Bibliothek aber bereits mit Swift 5.1.

Das Erreichen der ABI-Stabilität hat insofern große praktische Auswirkungen, als Swift-Bibliotheken nun direkt als Teil von iOS, macOS usw. ausgeliefert werden. In Swift entwickelte Apps können sich ab jetzt darauf verlassen, dass die Standardbibliotheken auf der Zielplattform zur Verfügung stehen. Es ist nicht mehr nötig, mit jeder App diverse Bibliotheken mitzuliefern. Das macht Kompilate kleiner – und natürlich auch die daraus resultierenden Apps.

Davon losgelöst bietet Swift 5 natürlich eine Menge Neuerungen im Vergleich zu Swift 4. (Die folgende Aufzählung berücksichtigt auch Features, die bereits in Swift 4.1 oder 4.2 eingeführt wurden.)

- **Zeichenketten:** Swift unterstützt jetzt *Raw*-Zeichenketten, also Zeichenketten, die Sonderzeichen wie \ enthalten.
- **Umgang mit Fehlern:** Die Swift-Standardbibliothek enthält den neuen Datentyp `Result`, der dabei hilft, Ergebnisse einer Funktion oder Methode und eventuell aufgetretene Fehler gemeinsam zurückzugeben. Das ermöglicht einen einfacheren Umgang mit Fehlern, insbesondere in asynchronem Code.

  Eine praktische Verbesserung ist das *Optional Flattening* des Schlüsselworts `try?`: Es verhindert eine Verschachtelung von Optionals.
- **Aufzählungen:** Die Swift-Standardbibliothek wurde um diverse Methode ergänzt, die die Verarbeitung von Arrays, Dictionaries und anderen Aufzählungen vereinfacht. Dazu zählen `allSatisfy`, `compactMap`, `compactMapValues`, `count(where:)` und `removeAll`.
- **Zufallszahlen:** Swift 5 stellt endlich konsistente Methoden zum Erzeugen von Zufallszahlen zur Verfügung. `Int.random(in: 1...10)` liefert beispielsweise zufällige ganze Zahlen zwischen 1 und 10.

  Auch praktisch: `randomElement` liefert ein zufälliges Element eines Sets, eines Arrays bzw. ein zufälliges Zeichen aus einer Zeichenkette.
- **Equatable und Hashable:** Selbst definierte Strukturen und Enumerationen erfüllen jetzt in den meisten Fällen automatisch die Protokolle `Equatable` und `Hashable`. Selbst wenn dieser Automatismus nicht greift (z. B. bei eigenen Klassen), ist die Implementierung der `hash`-Methode aufgrund der neuen Methode `Hasher.combine` einfacher denn je.
- **Dynamische Eigenschaften:** Das neue Attribut `@dynamicMemberLookup` gibt Ihnen die Möglichkeit, Typen zu definieren, auf deren Daten Sie in der Form `obj.name [= ...]` zugreifen können, obwohl die Eigenschaft `name` gar nicht statisch definiert ist.

Was bietet dieses Buch?

Dieses Buch vermittelt einen kompakten Einstieg in die Programmiersprache Swift in der Version 5 (Xcode 10.2). Das Buch ist in vier Teile gegliedert:

- **Teil I** führt in die Grundlagen von Swift ein. Hier lernen Sie alle wichtigen Sprachdetails kennen. Die Themenpalette reicht vom Umgang mit Variablen und elementaren Datentypen bis hin zur Syntax der objekt- und protokollorientierten Programmierung.
- **Teil II** ist eine Einführung in die Entwicklung von Apps für iOS, macOS und tvOS. Hier erkläre ich Ihnen beispielsweise, wie der Storyboard-Editor funktioniert, wie Sie Ihre Oberfläche mit eigenem Swift-Code verbinden, eigene `ViewController`-Klassen entwickeln, Apps mit mehreren Dialogen/Views organisieren etc.
- **Teil III** fasst wichtige Programmiertechniken in Bausteinform zusammen. In Kurzanleitungen zeige ich Ihnen unter anderem, wie Sie auf Dateien zugreifen, XML-Dokumente auswerten, Webseiten anzeigen, Steuerelemente mit eigener Grafik gestalten, Listen und Tabellen in Apps darstellen, geografische Daten auswerten und Spiele mit SpriteKit programmieren. Sie lernen, wie Sie Ihre App, sobald sie zufriedenstellend funktioniert, tauglich für den App Store machen und dort einreichen.
- **Teil IV** zeigt anhand konkreter Beispielprojekte die Praxis der App-Programmierung. Die Apps decken eine ganze Palette von Themen ab: vom praktischen Währungsumrechner über den Icon-Resizer bis hin zu mehreren Spielen.

Neu in dieser Auflage sind nicht nur Swift-5-Features, sondern auch neue Beispiel-Apps sowie ein neues Kapitel zu Core Data und SQLite.

Materialien zum Buch

Selbstverständlich können Sie alle Beispieldateien und -projekte dieses Buchs herunterladen. Einen Download-Link finden Sie hier:

www.rheinwerk-verlag.de/4749

Um von diesem Buch maximal zu profitieren, benötigen Sie weder Vorkenntnisse in Xcode noch in der App-Entwicklung. Ich setze aber voraus, dass Sie bereits Erfahrungen mit einer beliebigen Programmiersprache gesammelt haben. Ich erkläre Ihnen in diesem Buch also, wie Sie in Swift mit Variablen umgehen, Schleifen programmieren und Klassen entwickeln, aber nicht, was Variablen sind, wozu Schleifen dienen und warum Klassen das Fundament der objektorientierten Programmierung sind. So kann ich Swift kompakt und ohne viel Overhead beschreiben und den Schwerpunkt auf die konkrete Anwendung legen.

Leseanleitung

1.300 Seiten – das kann schon abschrecken! Dazu besteht aber kein Grund. Ich habe mich beim Schreiben dieses Buchs bemüht, den Inhalt auf möglichst eigenständige Kapitel zu verteilen, aus denen Sie sich wie aus einem Baukasten bedienen können.

Wenn Swift für Sie vollständig neu ist, dann ist die Lektüre der ersten Kapitel aus Teil I natürlich unumgänglich. Besonders wichtig ist, dass Sie die Swift-spezifischen Eigenheiten beim Umgang mit elementaren Datentypen und Aufzählungen (Arrays, Dictionaries etc.) kennenlernen und das Konzept von Optionals verstehen. Interessanterweise hat sich herausgestellt, dass Sie für die Entwicklung erster Apps nicht unbedingt alle Feinheiten im Zusammenhang mit Vererbung, Protokollen etc. beherrschen müssen. Die Basics reichen zumeist.

Teil II richtet sich speziell an Programmierer, die erstmalig Apps für iOS, macOS oder tvOS entwickeln. Wenn Sie bisher Objective-C zur App-Programmierung verwendet haben, werden Sie in Teil II auf viel bekanntes Wissen stoßen.

Bei Teil III habe ich versucht, oft benötigte Programmiertechniken möglichst losgelöst von der Zielplattform zu beschreiben. Beispielsweise erfolgt der Umgang mit Dateien unter iOS ganz ähnlich wie unter macOS. Aus den Kapiteln in Teil III können Sie sich also bedienen, wie Sie es gerade brauchen.

Viele Detailprobleme treten erst dann auf, wenn man den Schritt von kleinen Beispielen hin zu »richtigen« Apps macht. Deswegen stellt Teil IV eine Reihe vollständiger Projekte vor. Auch wenn es unwahrscheinlich ist, dass Sie genau so eine App programmieren möchten wie eines der Beispiele aus Teil IV, so werden Sie in diesen Kapiteln vermutlich doch inhaltlich verwandte Anleitungen und Arbeitstechniken mit einem hohen Praxisbezug finden. Probieren Sie die Apps einfach einmal aus, und blättern Sie dann durch die entsprechenden Kapitel – Sie werden sicher über Details stolpern, die sich später als hilfreich herausstellen werden.

Viel Spaß bei der App-Entwicklung!

Eine neue Programmiersprache zu erlernen ist immer eine Herausforderung. Noch schwieriger ist es, einen Überblick über die schier unüberschaubare Fülle von Bibliotheken zu gewinnen, die Sie zur App-Entwicklung brauchen. Dieses Buch soll Ihnen bei beiden Aspekten helfen und Ihnen ein solides Fundament vermitteln.

Wenn Sie in die App-Entwicklung mit Swift einsteigen, haben Sie das Privileg, mit einer der modernsten aktuell verfügbaren Programmiersprachen zu arbeiten. Sobald Sie die ersten Schritte einmal erfolgreich absolviert haben, wird die Faszination für diese Sprache auch Sie erfassen. Bei Ihrer Reise durch die neue Welt von Swift wünsche ich Ihnen viel Spaß und Erfolg!

Michael Kofler (*https://kofler.info*)