Vorwort

Im März 2008 brachte Apple ein SDK für seinen Kassenschlager, das iPhone, heraus. Dieses SDK und der einige Monate später auf dem Markt erschienene App Store sorgten für eine Menge Aufmerksamkeit. Viele, die sich zum ersten Mal mit der Apple-Plattform beschäftigten, waren erstaunt, dass das gesamte Toolset auf einer obskuren, 25 Jahre alten Programmiersprache namens Objective-C basierte, anstatt auf einer häufiger verwendeten Sprache wie C++ oder Java.

Mac-Entwickler und alle, die Apple schon länger verfolgten, überraschte das nicht sonderlich. Objective-C war die primäre Sprache, in der Apps für den Mac geschrieben wurden, seitdem Apple 1996 NeXT aufgekauft hatte. Anders als die Allzwecksprachen C++ oder Java, die auf zahlreichen Plattformen laufen und für eine Vielzahl von Programmieraufgaben eingesetzt werden, entschied man sich dafür, das erste OS X und später iOS mit einer Sprache zu schaffen, die von Apple kontrolliert wurde. Die Sprache entwickelte sich langsam weiter und konzentrierte sich ausschließlich auf die Entwicklung von GUI-Apps für Apple-Betriebssysteme.

In vielerlei Hinsicht waren Objective-C und die NeXT-Frameworks zur Erstellung von Applikationen ihrer Zeit um Jahre voraus. Obwohl ihnen ein größerer kommerzieller Erfolg versagt blieb, waren NeXT-Tools für Applikationen bei allen unglaublich erfolgreich, die damit gearbeitet haben, denn damit konnte man Applikationen weitaus schneller erstellen als mit anderen Tools, die seinerzeit auf dem Markt waren.

Doch Objective-C ist mittlerweile über 30 Jahre alt, und in der Technologie ist das eine sehr lange Zeit. Zwar hat sich Objective-C verändert und weiterentwickelt, konnte aber nicht wirklich Schritt halten. Programmiersprachen und Compiler haben sich in den vergangenen Jahrzehnten stark weiterentwickelt, und viele Leute fordern seit geraumer Zeit eine »modernere« Programmiersprache zum Entwickeln von iOS- und OS X-Apps.

Bei der Apple Worldwide Developers Conference (WWDC) 2014 verblüfft Apple praktisch alle mit der Ankündigung, dass nun eine neue moderne Sprache namens Swift bereitsteht.

Swift wurde vier Jahre im Geheimen entwickelt und ist eine vollkommen andere Sprache. Sie hat viele interessante Features, die Objective-C fehlen, arbeitet aber mit der gleichen Laufzeitumgebung und kann alle vorhandenen Frameworks und Libraries nutzen, aus denen die iOS- und OS X-SDKs bestehen.

Doch Swift ist wirklich anders: Swift sieht anders aus und fühlt sich auch anders an. Man muss sogar ein wenig anders denken, um gut damit zu arbeiten.

Zum Glück halten Sie genau das richtige Buch in der Hand, um dieses Umdenken zu lernen. Matt und Dave haben hervorragende Arbeit dabei geleistet und nehmen Sie mit auf eine spannende Tour durch diese interessante, aber vielleicht auch ein wenig unheimliche

neue Sprache. Dabei erfahren Sie, wie und warum etwas so und nicht anders gemacht wird, umrunden geschickt viele Stolperfallen und kommen mit der Sprache immer besser zurecht.

Jeff LaMarche, Autor von *Beginning iPhone Development: Exploring the iPhone SDK* (Apress); Geschäftsführender Partner und Gründer von MartianCraft

Vorbemerkung

Dieses Buch stellt kurz und knapp die neue Programmiersprache Swift von Apple vor. Wir haben dieses Buch für Entwickler geschrieben, die aktuell Objective-C für Apps einsetzen, und für solche, die gerne bei der Programmierung von Apps einsteigen wollen und neugierig sind, was Swift bietet.

Dieses Buch richtet sich in seinem »ungeduldigen« Stil nach der Art, wie Cay Horstmann Themen leicht verständlich und klar strukturiert darstellt, damit Entwickler genug Infos bekommen, um sofort produktiv zu werden. Code wird in diesem Buch vor allem in kleinen Blöcken präsentiert, um bestimmte Konzepte zu veranschaulichen. Diese Codeschnipsel sind als schnelle Referenz gedacht, aber nicht als Rezeptur für fertige Beispiele, die man direkt nutzen kann.

Die Sprache Swift bietet viele spannende Features, die es in Objective-C nicht gibt, der Sprache, mit der hauptsächlich für Apple-Plattformen programmiert wird. Man hat deren Features so gestaltet, dass Entwickler produktiver sein können und der Code weniger fehleranfällig wird. Bei Swift können Entwickler auf ein sehr gutes Typsystem und eine moderne Syntax zurückgreifen, um auf einfache Weise leistungsfähige Applikationen zu erstellen. Die ersten beiden Kapitel »Einführung in Swift« und »Die Syntax von Swift« erläutern Syntax, Basistypen und Features von Swift, um diese Konzepte jenen Objective-C-Entwicklern schnell zugänglich zu machen, für die sie neu oder ungewohnt sind. In den darauf folgenden Kapiteln werden diese Features eingehender vertieft.

Wie Objective-C ist Swift primär eine objektorientierte Sprache. Das Kapitel 3, »Objekte und Klassen«, stellt die wichtigsten Objekttypen vor, die in praktisch jeder Swift-App verwendet werden. Die Arbeit mit Klassen bei Swift wird allen Entwicklern vertraut sein, die sich in anderen objektorientierten Sprachen auskennen. Außerdem verhindert das Swift-Typsystem bei anderen Sprachen häufig vorkommende Fehler.

Mit Kapitel 4, »Optionals«, steigen wir bei den Themen ein, die Entwicklern mit Objective-C-Hintergrund weniger vertraut sind. Kapitel 4 konzentriert sich auf optionale Typen. Dieses fundamental neue Konzept zwingt Objekte, auf Typebene zu deklarieren, ob ein Wert fehlt oder gleich null ist. In diesem Kapitel geht es auch darum, wie Swift den Schwerpunkt auf die Codesicherheit legt. Kapitel 5, »Generische Funktionen«, untersucht das Konzept der Generics, mit dem Entwickler Funktionalitäten abstrahieren, um Code wiederzuverwenden und gleichzeitig Typsicherheit zu gewährleisten. Bei Swift sind Funktionen Typen erster Klasse und deswegen können sie als Parameter übergeben und als Werte zurückgegeben werden. Kapitel 6, »Funktionen und Closures«, untersucht die Aspekte der funktionalen Programmierung bei Swift.

Ein wesentlicher Grund für das Erlernen von Swift ist, Apps für iOS und Mac OS X erstellen zu können. Kapitel 7, »Die Arbeit mit Objective-C«, konzentriert sich darauf, wie man Swift mit Objective-C und den vorhandenen Apple-Frameworks einsetzt. Swift ist so

aufgebaut, dass es sich einfach bei vorhandenen Objective-C-Projekten und den von Apple bereitgestellten Frameworks integrieren lässt. Aber ein paar wichtige Dinge müssen Sie lernen, bevor Sie sich daran machen, Swift mit C und Objective-C zu kombinieren. Darum geht es in Kapitel 7.

Wenn Sie sich bis Kapitel 8, »Häufig vorkommende Muster«, vorgearbeitet haben, besitzen Sie genug solide Kenntnisse, um mit Swift Apps zu programmieren und dessen neue Features zu nutzen. Also stellt Kapitel 8 praktische Use Cases in den Vordergrund. Nach Abschluss von Kapitel 8 wissen Sie über verschiedene Situationen Bescheid, auf die Sie bei der Entwicklung von Apps häufig treffen.

Am Ende eines jeden Kapitels stellen wir Übungen vor, mit denen Sie Ihre neu erworbenen Fähigkeiten gleich umsetzen können. Diese Übungen sind notwendig, um sich das Gelesene anzueignen. Wenn Sie bei einer Übung nicht weiterkommen, finden Sie Lösungen auf unserer englischen Website http://SwiftForTheReallyImpatient.com. Auch Errata des Buchs oder andere interessante Inhalte finden Sie dort. Die für Swift 1.2 optimierten Code-Beispiele dieser Ausgabe können Sie unter http://dpunkt.de/swiftfuerungeduldige herunterladen.

Hinweis

Wenn im Buch eine Codezeile für die Seite zu breit ist, finden Sie einen Pfeil (➥), um anzuzeigen, dass der Code in der nächsten Zeile fortgesetzt wird.