Apps für iOS 9 professionell entwickeln

Sauberen Code schreiben mit Objective-C und Swift. Stabile Apps programmieren. Techniken & Methoden von Grund auf verstehen

Bearbeitet von
Thomas Sillmann

2., aktualisierte und erweiterte Auflage 2015. Buch. 718 S. Hardcover
ISBN 978 3 446 44566 6
Format (B x L): 18,4 x 24,7 cm
Gewicht: 1407 g

HANSER

Leseprobe

Thomas Sillmann

Apps für iOS 9 professionell entwickeln

Sauberen Code schreiben mit Objective-C und Swift. Stabile Apps programmieren. Techniken & Methoden von Grund auf verstehen

ISBN (Buch): 978-3-446-44566-6

ISBN (E-Book): 978-3-446-44553-6

Weitere Informationen oder Bestellungen unter

http://www.hanser-fachbuch.de/978-3-446-44566-6

sowie im Buchhandel.

Inhalt

Vorwort

iOS ist und bleibt für Entwickler ein spannendes Feld, nicht zuletzt, da Apple mit dem App Store einen Weg geschaffen hat, mit dem auch einzelne Entwickler Software für einen internationalen Markt verbreiten können, ohne dass sie sich um Dinge wie Bezahlsysteme, Abrechnungen und Download Gedanken machen müssen. Ich selbst habe mich aufgrund dessen vor über fünf Jahren für die iOS-Entwicklung begeistern lassen und habe bis heute nichts von der Faszination für diese spannende und innovative Plattform verloren.

Beim Einstieg in die iOS-Entwicklung habe ich eines aber schmerzlich vermisst: einen einfachen, übersichtlichen und professionellen Einstieg. Ich habe mich mit viel Literatur auseinandergesetzt, war in vielen Foren unterwegs und habe schlicht und einfach viel ausprobiert. Da vieles von diesen Anfängen aber sehr umständlich oder im Nachhinein betrachtet sogar gänzlich falsch war, hat mich das viel Zeit und Lehrgeld gekostet. Und ich habe mir oft gewünscht, man hätte mich von Beginn an an die Hand genommen und mir nicht nur gezeigt, *wie* ich Apps für iOS entwickle, sondern wie ich *gute und professionelle* Apps entwickle, welche Besonderheiten, Best Practices und Design Patterns es gibt und wie ich effektiv und effizient mit der Entwicklungsumgebung arbeiten kann. Und dieser Wunsch hat den Grundstein für dieses Buch gelegt.

Dieses Buch vermittelt Ihnen alle essenziellen Grundlagen und Kenntnisse über die Entwicklung für iOS. Angefangen bei der Vorstellung des Betriebssystems selbst geht es weiter über die Programmiersprachen Objective-C und Swift, deren Struktur und jeweiligen Besonderheiten und all das, was sie ausmacht. Danach rückt die eigentliche Entwicklung für iOS in den Fokus. Sie erfahren alles über die wichtigsten Frameworks von Apple für die App-Entwicklung und lernen typische Best Practices kennen. Besonders wichtig ist hier auch die Vorstellung der Dokumentation, die für Sie als App-Entwickler Bestandteil Ihrer täglichen Arbeit sein wird. Denn letztlich beherbergt die Apple-Dokumentation alle Antworten auf die Fragen, wie Sie bestimmte Probleme in der iOS-Entwicklung angehen und welche Möglichkeiten Ihnen die einzelnen Frameworks von Apple liefern.

Nach der Vorstellung der Plattform und der Programmiersprache(n) geht es weiter mit der Entwicklungsumgebung Xcode, mit der wir unsere Apps für iOS entwickeln. Dabei war es mir besonders wichtig, den genauen Aufbau und die Struktur hinter Xcode vorzustellen sowie alle spannenden Möglichkeiten und Kniffe aufzuzeigen, die Xcode Ihnen bietet und Ihre tägliche Arbeit erleichtern. Auch der Umgang mit Grafiken oder die Lokalisierung Ihrer Anwendung sowie Debugging und Refactoring habe ich in dieses Kapitel mit aufgenommen.

Bis zu diesem Punkt habe ich mich rein mit den essenziellen Grundlagen beschäftigt und ich finde es wichtig, dass auch Sie diese Grundlagen verinnerlicht und verstanden haben, denn sie sind die Grundpfeiler für gute und erfolgreiche Apps. Dazu abschließend folgt im sechsten Kapitel die Vorstellung von MVC – Model-View-Controller. Dabei handelt es sich um eines der wichtigsten Design-Patterns in iOS und ist essenziell für die App-Entwicklung. Aufgrund dessen widme ich MVC ein eigenes Kapitel, stelle es im Detail vor und erkläre, wie und warum Sie es in Ihren eigenen Apps anwenden sollen.

Anschließend geht es im Speziellen um die Entwicklung für iOS und die Nutzung der Funktionen und Frameworks für Apple, unterteilt auf die drei Bereiche Controller, View und Model aus dem MVC-Pattern. Auch in diesen Kapiteln geht es darum, die grundlegenden Besonderheiten zu erläutern und aufzuzeigen, wie Sie mit den einzelnen Elementen arbeiten und diese in Ihren eigenen Apps verwenden. Sie lernen die wichtigsten Elemente aus den jeweiligen Bereichen kennen und erfahren, wie Sie selbstständig mit ihnen arbeiten können und worauf bei der jeweiligen Verwendung zu achten ist. Mit diesem Wissen gewappnet sind Sie imstande, sich selbst in neue Frameworks, Technologien und Themen anhand der Apple-Dokumentation einzuarbeiten und selbstständig Probleme zu lösen. Und genau das ist es, was einen guten und professionellen iOS-Entwickler ausmacht.

Kapitel 10 setzt sich im Detail mit den sogenannten Local und Push Notifications auseinander, die es Ihnen erlauben, Nachrichten aus Ihren Apps an Ihre Nutzer zu senden. Im darauffolgenden Kapitel erstelle ich Ihnen ergänzend dazu dann Extensions vor, die uns Entwicklern ganz neue und innovative Möglichkeiten an die Hand geben, unsere Apps zu erweitern und über das gesamte iOS-System heraus zugänglich zu machen.

Im Anschluss daran widme ich ein eigenes und umfangreiches Kapitel der Entwicklung für die Apple Watch. Die Apple Watch ist die neue spannende Plattform in Apples Ökosystem und ist – was die grundsätzliche App-Entwicklung betrifft – in vielen Dingen recht ähnlich zur iOS-Plattform. Da eine Apple Watch-App immer eine iPhone-App voraussetzt, war es nur passend, auch die Entwicklung für die Apple Watch in diesem Buch im Detail zu beleuchten und zu zeigen, wie Sie Ihre eigenen iPhone-Apps um ein Pendant für die Apple Watch erweitern können. Sie lernen alle Möglichkeiten und Einschränkungen kennen und werden so in die Lage versetzt, selbst mit der Entwicklung eigener Apple Watch-Apps zu beginnen.

Anschließend folgen die Themen Unit-Tests und Versionsverwaltung mit Git, die in jeder neuen Xcode-Version mehr und mehr in die Entwicklungsumgebung integriert und unterstützt werden. Ich zeige Ihnen dabei, was Unit-Tests sind, warum Sie sie in Ihren Apps verwenden sollten, wie Sie Unit-Tests schreiben und wie Sie Xcode in Sachen Unit-Tests unterstützt und Ihnen unter die Arme greift. Auch die zusammen mit iOS 9 neu eingeführten UI-Tests finden dabei Beachtung. Bei der Versionsverwaltung mit Git erfahren Sie alles über die integrierten Funktionen zur Arbeit mit Git in Xcode und wie Sie Änderungen im Code verfolgen und nachvollziehen können.

Zu guter Letzt geht es noch – wie könnte es anders sein? – um die Veröffentlichung Ihrer Apps im App Store und die integrierten Tools in Xcode, die Ihnen bei diesem Prozess unter die Arme greifen. Insbesondere erfahren Sie hier etwas über die Erstellung und Verwaltung Ihrer Apps in Apples iOS Developer Program.

Bei allen diesen Themen soll dieses Buch Sie unterstützen und Ihnen die Grundlagen und das essenzielle Praxiswissen vermitteln und mit auf den Weg geben. Es soll Ihnen nicht

Beispielprojekte aufzeigen und deren Verhalten und Eigenschaften erklären (davon gibt es nämlich von Apple selbst mehr als genug), sondern Ihnen das nötige Wissen mitgeben, um Sie in die Lage zu versetzen, Problemstellungen selbstständig zu lösen und zu verstehen, wie Sie gute und professionelle iOS-Apps entwickeln. Denn wenn Sie diesen Status erreicht haben, können Sie darauf aufbauen, experimentieren und eigene spannende und innovative iOS-Projekte umsetzen. Und ich bin gespannt, welche großartigen Apps wir von Ihnen erwarten dürfen.

Ich wünsche Ihnen viel Freude beim Lesen dieses Buches und viel Erfolg mit all Ihren iOS-Projekten.

Thomas Sillmann

thomassillmann.de/ios-buch

Unter dieser Adresse finden Sie einige Basis-Klassen, die aus den Code-Beispielen aus diesem Buch aufgebaut sind. Dieses Basis-Klassen dienen einerseits dazu, Ihnen einzelne Code-Schnipsel für bestimmte Aufgaben zur Verfügung zu stellen, und sind andererseits aber auch dazu geeignet, in Ihre eigenen Projekte übernommen und dort verwendet zu werden. Sie enthalten die Logik für verschiedene grundlegende Funktionen und Aktionen in der iOS-Entwicklung und können Ihnen daher möglicherweise des Öfteren eine nützliche Unterstützung sein. Schauen Sie einfach mal vorbei!

1 Über iOS

1.1 Was ist iOS?

Auch wenn diese Frage in der heutigen Zeit möglicherweise überflüssig erscheint (und auch in Anbetracht dessen, dass Sie sich dieses Buch gekauft haben), möchte ich zu Beginn doch zumindest kurz darauf eingehen, was eigentlich dieses iOS ist, für das ich mich – und Sie sich offensichtlich auch – als Entwickler so sehr interessiere. Dabei werde ich auch direkt den Spagat schlagen und Ihnen die Geräte vorstellen, auf denen iOS verfügbar ist, und beschreiben, wie sich das System im Laufe der Jahre entwickelt hat.

Zunächst einmal ist iOS ein Betriebssystem der Firma Apple. Seinen ersten großen Auftritt hatte es im Jahr 2007 zusammen mit der Vorstellung des allerersten iPhone, denn genau auf diesem System lief iOS (auch wenn es damals noch iPhone OS hieß). Mit dem iPhone krempelte sich der Markt der Smartphones maßgeblich um und heutzutage sieht man Touch-Smartphones mit dem Bildschirm als Hauptbedienelement allerorten.

Nach mehreren Hardware-Sprüngen des iPhone folgte im Jahr 2010 das nächste iOS-Device von Apple: Das iPad, welches – ebenso wie das iPhone zuvor den Smartphone-Markt – nun den Tablet-Markt ordentlich aufmischte und bis heute den Quasistandard im Bereich Tablets setzt. Auch auf dem iPad läuft Apples Mobil-Betriebssystem iOS (dessen Namensänderung ebenfalls im Jahr 2010 mit dem Erscheinen des iPad von iPhone OS zu iOS erfolgte).

Darüber hinaus läuft iOS auch auf dem iPod touch. Alle Apps, die Sie für das iPhone entwickeln, sind prinzipiell ebenfalls auf dem iPod touch lauffähig, lediglich die zugrunde liegende Hardware unterscheidet sich ein wenig; Telefonieren ist beispielsweise mit dem iPod touch nicht möglich. So kann sich aber ein iPod touch durchaus als günstiges Testgerät für iOS-Apps anbieten (das iPhone spielt da nun mal doch in einer etwas anderen Preisklasse).

Und direkt zu Beginn noch eine kleine Randnotiz: Auch auf dem Apple TV läuft eine angepasste Version von iOS, allerdings gibt es bisher für Entwickler keine Chance, eigene Anwendungen für dieses Gerät zu entwickeln und zu veröffentlichen. Aber vielleicht dürfen Sie und ich langfristig unsere Kenntnisse über die iOS-Entwicklung auch dafür nutzen, eigene Anwendungen für das Apple TV zu kreieren; warten wir ab, was die Zukunft noch so bringt (siehe Bild 1.1)!

Bild 1.1 iPhone und iPad sind die erfolgreichsten Geräte mit dem Betriebssystem iOS. Daneben verfügt auch Apples iPod touch über iOS als Betriebssystem. (Bild: developer.apple.com)

1.1.1 iOS und OS X

Zusammengefasst lässt sich also einfach sagen: iOS ist das Betriebssystem von Apples iPhone-, iPad- und iPod touch-Familie. Sicherlich wissen Sie aber auch, dass Apple nicht nur iOS-Geräte entwickelt und veröffentlicht (auch wenn das wohl aktuell das Geschäft ist, das Apple den größten Umsatz einbringt). Daneben gibt es noch die Mac-Familie, die Apples Produktplatte aus Notebooks und Desktop-PCs darstellt. Und besonders spannend ist hierbei, dass iOS auf OS X – dem Betriebssystem der Macs – basiert. So sind viele Frameworks, mit denen wir in der iOS-Entwicklung arbeiten werden, unter OS X in derselben oder in einer leicht abgewandelten Form verfügbar. Das bedeutet umgekehrt auch, dass der Einstieg in die OS X-Entwicklung leichter fällt, wenn Sie bereits für iOS entwickelt haben – und umgekehrt. Das aber nur als kleine Randnotiz und mögliche Motivation, sich nach der Lektüre dieses Buches eventuell auch mit der OS X-Entwicklung näher auseinanderzusetzen; Sie werden sehen, über das nötige Rüstzeug verfügen Sie dann. ☺

1.1.2 Besonderheiten der iOS-Plattform

Auch wenn iOS auf OS X basiert, so gibt es doch mannigfaltige Unterschiede zwischen den beiden Betriebssystemen (auch wenn sie sich unter der Haube relativ ähnlich sind).

Entscheidend anders sind die Bedienoberflächen und das Bedienkonzept gestaltet. Während OS X und jedes andere Desktop-Betriebssystem typischerweise mittels Maus und Tastatur

gesteuert werden, verfügen iOS-Geräte lediglich über einen Touchscreen, über den mittels Fingergesten und Berührungen alle Aktionen gesteuert werden. Hier gibt es also ganz neue Aspekte, auf die wir als Entwickler achten müssen, um gut funktionierende und intuitiv bedienbare Apps zu entwickeln. Denn ein Finger zum Bedienen eines Touchscreens ist nun mal etwas gänzlich anderes als eine Maus, die ich pixelgenau an jede Position bewegen kann. Ein Finger besitzt wesentlich mehr Fläche und allein das muss bereits beim Konzipieren und Entwickeln eigener Anwendungen für iOS maßgeblich beachtet werden.

Auch sind die Nutzer mit iPhone und iPad mobil unterwegs, was in heutigen Zeiten mit sehr gutem Ausbau des Mobilfunknetzes nichtsdestotrotz bedeutet: Nicht immer ist Internet verfügbar (mal ganz davon abgesehen, dass es das iPad auch in einer reinen WLAN-Version ohne Mobilfunkverbindung gibt) und den Nutzer dazu zu zwingen, eine Internet-Verbindung herzustellen, sollte nur wirklich dann erforderlich sein, wenn es gar nicht anders geht und ein Internet-Zugang zwingend für die Nutzung der eigenen App (oder der gerade benötigten Funktion) notwendig ist.

iPhone und iPad sind Mobilgeräte, und genau so werden sie auch genutzt, soll heißen: Viele Nutzer holen ihr Smartphone nur für den Bruchteil eines Augenblicks hervor, checken aktuelle Facebook- oder WhatsApp-Nachrichten und lassen das Handy dann wieder verschwinden. Auch für Sie als App-Entwickler gilt: Halten Sie den Nutzer bei dem, was er mit Ihrer App tun will, nicht auf. Weniger ist hier ganz klar mehr. Ihre App soll eine eindeutige Funktion erfüllen, bieten Sie diese darum dem Nutzer so komfortabel, übersichtlich und leicht zugänglich wie nur irgend möglich an.

Daneben gibt es noch einen weiteren wichtigen Aspekt, den wir als Entwickler immer berücksichtigen sollten: Schonender Umgang mit den Akku-Ressourcen. Wenn wir ununterbrochen Bluetooth in Beschlag nehmen und nach anderen Geräten suchen, saugen wir den Akku des Nutzers damit sehr schnell leer und dürfen uns wahrscheinlich im Umkehrschluss über schlechte Kritiken unserer App im App Store „freuen“. Hier gilt ganz klar: Weniger ist mehr, und Ihre App sollte sich immer auf genau die Aufgabe konzentrieren, für die sie gedacht ist.

Sie sehen also, Apps für iOS zu entwickeln besteht nicht nur darin, die Programmiersprache und SDKs zu beherrschen; es geht auch darum zu verstehen, wie die iOS-Gerätefamilie funktioniert, wie sie genutzt wird und wie Sie mit Ihren Apps den Nutzern das Leben erleichtern.

1.2 iOS für Entwickler

Apple hat mit dem App Store und iOS eine großartige Infrastruktur für uns Entwickler geschaffen. Wir können uns damit voll und ganz auf die Entwicklung unserer Apps konzentrieren, alle sonstigen Modalitäten wie Bezahlmethoden, Zahlungseingang oder Vertrieb übernimmt Apple für uns. Auch wenn Apple dafür einen Obolus in Form eines jährlichen Mitgliedsbeitrags im iOS Developer Program als auch 30 % der Erlöse pro verkaufter App fordert, so stellt der App Store doch eine großartige Möglichkeit dar, die eigene Anwendung binnen kürzester Zeit einem internationalen (und auch durchaus kauffreudigen) Nutzerkreis zum Download zur Verfügung zu stellen.

Was an dieser Stelle auch gleich gesagt sein muss: Es gibt von Apple keinen anderen vorgesehenen Weg zur Installation einer App auf einem iOS-Gerät außer dem offiziellen App Store. Höchstwahrscheinlich haben Sie bereits einmal von einem „Jailbreak“ gehört, der es ermöglicht, unter anderem den Weg über den App Store zu umgehen und dadurch Apps aus beliebigen Quellen (wie zum Beispiel direkt über die Website eines Anbieters) auf dem eigenen Gerät zu installieren; das sollte aber nicht Ihr bevorzugtes Vorgehen sein, wenn Sie Apps für iOS entwickeln möchten. Zum einen schiebt Apple den Lücken, über die sich ein solcher Jailbreak durchführen lässt, regelmäßig einen Riegel vor, zum anderen ist das schlicht und einfach nicht der Weg, diese Plattform zu nutzen. Apple hat die iOS-Geräte als relativ geschlossene und abgeschottete Systeme konzipiert, und genau so sollten sie auch betrachtet und genutzt werden. Denn auf der anderen Seite sollte nicht der Sicherheitsgewinn vergessen werden, den gerade iOS gegenüber anderen Mobil-Betriebssystemen innehat; Schadsoftware lässt sich nur schwer bis gar nicht auf den Geräten installieren. Möglicherweise halten Sie diese Einstellung für engstirnig und betiteln mich in Gedanken bereits als „Apple Fanboy“, ich versuche aber schlicht, die iOS-Plattform als das zu sehen, was sie ist, und sie so zu nutzen, wie es gedacht ist. Damit ist sowohl uns Entwicklern als auch all den Millionen iOS-Nutzern da draußen am meisten geholfen.

So weit, so gut, doch was benötige ich als Entwickler nun konkret, um mit der Entwicklung für iOS starten zu können?

1.2.1 Hardware für Entwickler

Um für iOS entwickeln zu können, benötigen Sie in jedem Fall einen gewissen Fuhrpark an Apple-Geräten. Zunächst wäre hier einmal der Mac genannt. Ja, ein Mac ist notwendig, um für iOS entwickeln zu können, denn nur unter OS X – dem Betriebssystem des Mac – stehen die SDKs und Frameworks zur Entwicklung für iOS zur Verfügung. Für welches Gerät Sie sich dabei im Detail entscheiden, ist gut und gerne Ihnen und Ihren persönlichen Vorlieben überlassen, leistungstechnisch eignen sich alle. Achten Sie im Idealfall am ehesten noch darauf, einen Mac mit mindestens 8 GB Arbeitsspeicher zu erstehen; für die Entwicklung und zum Kompilieren Ihrer Apps ist das ein sehr angenehmes Mindestmaß, glauben Sie mir (siehe Bild 1.2).

Bild 1.2 Die Macs von Apple eignen sich alle gleichermaßen zur Entwicklung für iOS, auch wenn man mindestens 8 GB Arbeitsspeicher beim Gerätekauf berücksichtigen sollte. (Bild: apple.com)

Tipp

Das MacBook Pro ist nicht nur ein extrem leistungsstarkes Notebook, es verfügt zudem auch über das sogenannte Retina-Display, welches Apple in dieser Form auch in seinen iPhones und iPads sowie dem neuen iMac verwendet. Damit werden für einen Bildpunkt die doppelte Anzahl Pixel verwendet, wodurch Bilder, Texte und Anwendungen gestochen scharf dargestellt werden. Nicht nur, dass dieses Display mit seiner Auflösung von 2560 × 1600 Pixeln eine technische Meisterleistung darstellt, nein, es erleichtert Ihnen auch den Umgang mit dem iOS-Simulator während der Entwicklung, denn dieser Simulator kann auch in der nativen Retina-Auflösung von iPhone und iPad dargestellt werden. Auf Bildschirmen ohne Retina-Display bedeutet das, dass beispielsweise für den iPad-Retina-Simulator eine Fläche von 2048 × 1536 Pixeln auf dem Bildschirm angezeigt werden muss; selbst auf einem iMac mit 21,5“-Bildschirm passt dieser Simulator im Hochformat nicht auf das komplette Display. Bei einem MacBook Air mit seinen noch kleineren Displays wird es dann wahrlich schwierig, mit diesen riesigen Simulatoren zu arbeiten. Wenn Sie also gänzlich unentschlossen bei der Wahl Ihres Entwickler-Mac sind, sollten Sie möglicherweise das MacBook Pro mit Retina-Display oder alternativ den neuen iMac mit Retina-Display in Betracht ziehen.

Neben Ihrem Mac sollten Sie auch mindestens ein iPhone und/oder iPad besitzen – eben je nachdem, ob Sie nur für eine oder für beide Plattformen Apps entwickeln. Zwar haben Sie die Chance, all Ihre entwickelten Apps auch in einem Simulator auszuführen und zu testen, Sie sollten aber in jedem Fall vor der Veröffentlichung Ihrer App im App Store diese auch auf einem richtigen iOS-Gerät ausführlich geprüft haben. Der Simulator nutzt nämlich die komplette Hardware-Power Ihres Mac, wodurch es passieren kann, dass zwar im Simulator alles schnell und fluffig läuft, aber auf einem richtigen (möglicherweise auch schon etwas betagteren) iOS-Gerät alles ruckelt oder sogar abstürzt.

1.2.2 Software für Entwickler

Das Programm, mit dem Sie als Entwickler die meiste Zeit verbringen werden, ist Xcode. Xcode ist kostenlos über den App Store Ihres Mac erhältlich und ist die komplette IDE von Apple zur Softwareentwicklung für OS X und iOS. Es liefert alle Werkzeuge, die Sie benötigen, inklusive aller SDKs und Simulatoren (sehen Sie dazu auch das fünfte Kapitel „Die Entwicklungsumgebung – Xcode“). Bild 1.3 zeigt das App-Icon von Xcode.

Bild 1.3
Xcode: Mit dieser IDE legen Sie in der iOS-Entwicklung richtig los!

Daneben gibt es noch weitere Software, die Ihnen in Ihrer täglichen Entwicklerarbeit das Leben erleichtern kann, zum Beispiel spezielle Clients zur Arbeit mit der Versionsverwaltung Git oder Apps zur Arbeit mit Datenbanken. Diese sind aber nicht notwendig und werden von mir an passender Stelle im Buch in der benötigten Tiefe vorgestellt.

1.2.3 Das Apple Developer Program

Wie eingangs erwähnt, ist es mit der Entwicklung einer App allein noch nicht mit der Veröffentlichung im App Store getan. Neben den 30% Verkaufserlös, die Apple automatisch von Ihren App-Verkäufen abzwackt, müssen Sie auch noch Mitglied im Apple Developer Program sein. Als Mitglied können Sie dann ein eigenes Entwicklerzertifikat beantragen und damit Ihre Apps signieren, was nötig ist, um diese in den App Store einzureichen als auch um sie auf einem eigenen iOS-Gerät testen zu können. Insgesamt gibt es drei verschiedene Formen des Apple Developer Program, die hier nun einmal in Kürze vorgestellt werden sollen:

Apple Developer Program Individual

Das ist das Entwicklerprogramm für alle, die alleine bzw. freiberuflich oder selbstständig Apps für iOS entwickeln und veröffentlichen möchten. Die Mitgliedschaft kostet 99 $ pro Jahr und erlaubt den vollen Zugriff auf alle Entwicklerressourcen von Apple, einschließlich Vorabversionen der Entwicklungsumgebung Xcode als auch iOS selbst.

Apple Developer Program Company

Das Apple Developer Program Company entspricht im Großen und Ganzen dem Individual Program, nur dass dieses hier explizit für Firmen ausgelegt ist. Dieses Programm bietet daher auch die Möglichkeit, mehrere Teammitglieder und Entwickler mit verschiedenen Rollen anzulegen und zu verwalten. Die Kosten belaufen sich – ebenfalls wie bei Individual – auf 99 $ im Jahr.

Apple Developer Enterprise Program

Das Enterprise Program ist für Firmen gedacht, die Apps für interne Geschäftszwecke entwickeln und nutzen möchten. So erlaubt dieses Programm das Veröffentlichen von Apps auf beliebig vielen im Programm registrierten Geräten, es ist allerdings keine Veröffentlichung von Apps in Apples App Store möglich. Die Kosten für dieses Programm liegen bei 299 $ pro Jahr.

Apple Developer University Program

Das Apple Developer University Program ist das einzige kostenlose Entwicklerprogramm von Apple. Wie der Name bereits andeutet, richtet es sich an Studierende und Lehrkräfte von Hochschulen und Universitäten, um dort beispielsweise iOS-Entwicklung zu unterrichten (siehe Bild 1.4).

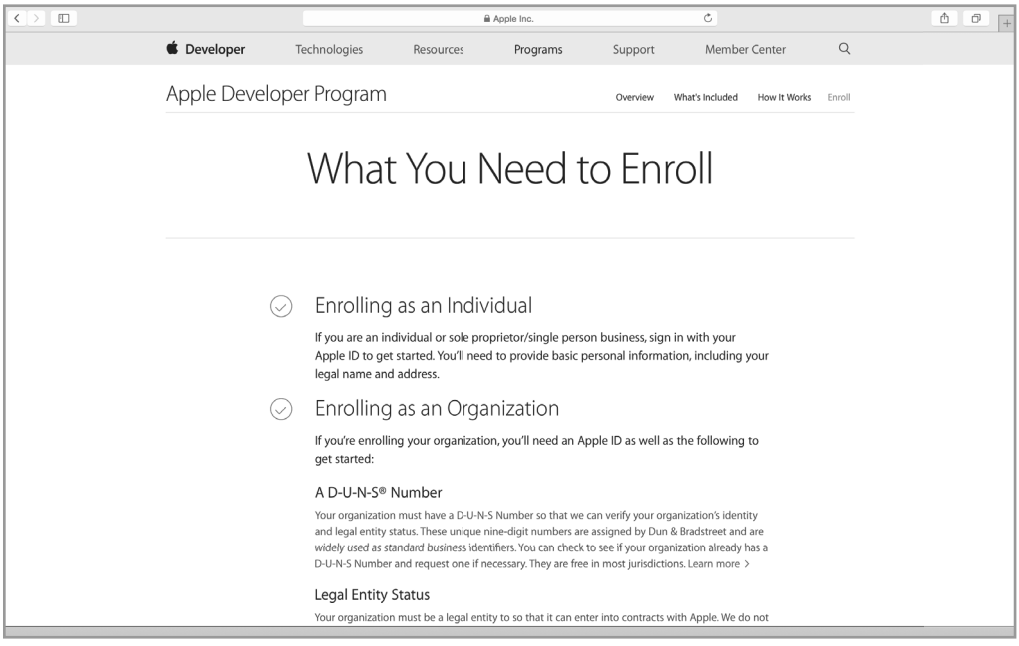

Bild 1.4 Auf https://developer.apple.com/programs/enroll/ finden Sie weitere Informationen zum Apple Developer Program und gelangen von dort auch zur Registrierung. (Bild: developer.apple.com)

Weitere Informationen zum Apple Developer Program sowie das Formular für die Registrierung finden Sie unter https://developer.apple.com/programs/enroll/.

1.3 Der Aufbau von iOS

Wir wissen nun also, was iOS ist, auf welchen Geräten es läuft und was diese Plattform ausmacht. Für uns als Entwickler ist aber besonders interessant, wie dieses System aufgebaut ist und wie die Architektur von iOS aussieht.

1.3.1 Die vier Schichten von iOS

iOS fußt auf insgesamt vier Schichten (den sogenannten Layern). Es gibt hier den Core OS Layer, den Core Services Layer, den Media Layer und den Cocoa Touch Layer (siehe Bild 1.5). All diese Schichten machen iOS zu dem, was es ist, und Sie als Entwickler nutzen die verschiedenen Funktionen der einzelnen Schichten, um Ihre Anwendungen zu entwickeln. Im Folgenden sollen diese Schichten einmal im Detail vorgestellt werden.

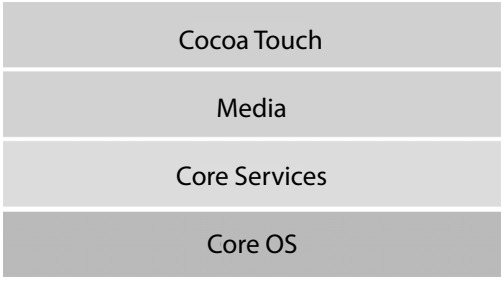

Bild 1.5
Die vier Schichten von iOS, chronologisch geordnet von unten nach oben.
(Bild: Xcode-Dokumentation)

Core OS

An allererster Stelle steht das Core OS (und es ist somit die unterste Schicht des Systems). Es beherbergt die grundlegenden Funktionen und Frameworks zur Arbeit mit iOS, wobei wir die meiste Zeit über mit den darüber liegenden Schichten zu tun haben. Es ist aber wichtig zu wissen, dass es diese Funktionen und Frameworks gibt und wofür sie gut sind. Auf diesem Layer finden sich so beispielsweise das Core Bluetooth, das Security und das External Accessory FrameworkNetzwerkkommunikation, Input/Output, Zugriffe auf das Dateisystem und mathematische Operationen.

So wichtig all diese Funktionen auch sind, so werden wir aber meist in unserer täglichen Arbeit – wie eben erwähnt – eher mit Frameworks arbeiten, die auf dem Core OS aufbauen und damit einen erleichterten und komfortableren Zugang auf diese Ressourcen erlauben. Sie sehen aber, dass Sie auch die Möglichkeit haben, selbst in diese Bereiche vorzudringen (sollte das nötig sein).

Core Services

Die nächste Schicht in der iOS-Systemarchitektur ist der Core Services Layer. Wie der Name bereits andeutet, stellt diese Schicht grundlegende Dienste zur Verfügung und setzt damit auf dem Core OS auf. Beispielsweise enthält dieser Layer die Peer-to-Peer-Services, mit

denen Verbindungen zwischen verschiedenen iOS-Geräten via Bluetooth hergestellt werden können. Wie Sie sehen, brauchen Sie für eine solche Aufgabe nicht zwingend das Core Bluetooth Framework aus dem Core OS zu nutzen; das entsprechende Multipeer Connectivity Framework aus dem Core Services Layer setzt darauf auf und bietet bereits Lösungen für den Verbindungsaufbau zwischen verschiedenen iOS-Geräten an.

Daneben enthält Core Services noch Frameworks zur Arbeit mit iCloud, die automatische Speicherverwaltung ARC (mehr dazu in Kapitel 4, „Programmierung für iOS"), Frameworks für In-App-Käufe und mehr.

Mit am wichtigsten für uns als Entwickler ist mit Sicherheit das Foundation Framework, welches grundlegende Datentypen wie Strings, Arrays etc., Funktionen zur Berechnung von Datum und Uhrzeit und weitere grundlegende Klassen liefert, die für die tägliche Arbeit unabdingbar sind, aber dazu später mehr.

Media

Hier wird es multimedial (im wahrsten Sinne des Wortes). Egal ob Frameworks zur Arbeit mit Bildern und Videos, zur Arbeit mit Animationen, zum Arbeiten mit Audio oder das Erstellen eigener Grafiken, im Media Layer findet sich all das Handwerkszeug, das Sie als Entwickler für alle Aufgaben rund um verschiedene Medien benötigen. Auch die Frameworks zur Arbeit mit AirPlay oder zum Verwalten eigener Game Controller finden sich auf dieser Schicht. Die meisten dieser Dinge werden aber ausschließlich für spezielle Einzelfälle benötigt, sodass wir in diesem Buch nur grundlegend auf das ein oder andere Framework näher eingehen werden (Sie werden aber sehen, dass Sie nach Lesen dieses Buches in der Lage sind, sich das nötige Know-how für all die anderen Media-Frameworks ohne Schwierigkeiten selbst anzueignen, versprochen).

Cocoa Touch

Mit Cocoa Touch erreichen wir die oberste Schicht in der iOS-Architektur und damit auch die Schicht, die den größten Unterschied zwischen iOS und OS X darstellt. Ganz grundlegend gesprochen stellt Cocoa Touch alle Frameworks und Funktionen zur Verfügung, um Apps für iOS zu entwickeln. Es enthält allen voran das UIKit Framework, welches verschiedene Views und Controller zum Bau von iOS-Apps enthält und womit dann das komplette Aussehen und Design einer App umgesetzt werden kann. Es enthält auch Frameworks und Funktionen für die grundlegende App-Infrastruktur, so beispielsweise ein System, um den Nutzer mittels Notifications zu informieren, oder die Arbeit mit verschiedenen Touch-Gesten. Viele enthaltene Frameworks bauen auf den Funktionen der darunter liegenden Schichten auf, womit Cocoa Touch einfach nutzbare und objektorientierte Schnittstellen bietet.

Zur Entwicklung von iOS-Apps ist dieses Framework essenziell und für grundlegende Apps auch ausreichend. Dennoch ist es wichtig, als Entwickler zu wissen, aus welchen Schichten sich iOS aufbaut und welche Funktionen Ihnen als Entwickler zur Verfügung stehen.

1.4 Die perfekte iOS-App

O je, solch ein reißerischer Titel direkt zu Beginn dieses Buches? Nun ja, auch wenn ich Ihnen hier kein Patentrezept zu einer erfolgreichen und ertragreichen App geben kann, so kann ich Ihnen aber sagen, was Sie beachten müssen, wollen Sie mit Ihrer App-Idee wirklich erfolgreich sein. Und ich teile diese Weisheiten bereits mit Ihnen hier zu Beginn dieses Buches, weil es grundlegende Aspekte sind, die Sie bei jedem Schritt der Entwicklung (und bereits davor) im Hinterkopf behalten und absolut verstehen sollten.

Zunächst einmal: Achten Sie bei jedem Schritt der Entwicklung auf die Besonderheiten der iOS-Plattform (wie im vorigen Abschnitt beschrieben). Sie haben es hier mit Mobilgeräten zu tun, die die Nutzer in unregelmäßigen Abständen nutzen und mit denen sie meist gezielt eine bestimmte Aufgabe erfüllen möchten. Halten Sie den Nutzer nicht mit ewig langen Hinweisen oder Meldungen auf und achten Sie auf die Akku-Ressourcen und gar zu stromhungrige Funktionen, die Sie eigentlich gar nicht benötigen.

Wenn Sie das beherzigen (und zwar am besten bereits in der Planungs- und Konzeptionsphase, bevor die erste Zeile Code geschrieben ist), haben Sie bereits einen essenziellen Grundstein für eine erfolgreiche App gelegt. Denn Sie machen sich keine Vorstellung, bei wie vielen Apps genau diese Aspekte bereits schieflaufen und zu entsprechend schlechten Kritiken im App Store führen. Darüber hinaus gibt es noch eine weitere Weisheit, die zwischen Erfolg und Misserfolg Ihrer App entscheiden kann und daher auch von Beginn an berücksichtigt werden sollte:

Konzentrieren Sie sich auf genau die eine Aufgabe, die Ihre App erfüllen soll! Denn dafür sind Apps für iOS gedacht: Für eine Aufgabe, die der Nutzer dank der App schnell, komfortabel und intuitiv lösen können soll. Was sich jetzt möglicherweise einfach und simpel anhört, ist in der Praxis bisweilen ein schmaler Grat und alles andere als einfach umzusetzen. Geben Sie sich daher im Vorhinein die Mühe und überlegen Sie, wie Sie sich selbst Ihre Traum-App vorstellen würden. Wie würde sie aussehen, welche Funktionen muss sie anbieten? Wie bedient sie sich, wie nutzt sie Animationen zum besseren Verständnis? Werden Sie sich über diese Dinge klar, bevor Sie beginnen, massiv Code zu schreiben, denn das geht meist nach hinten los. Und denken Sie daran, den Nutzer nicht mit unzähligen (und unnützen) Features zu überfrachten, die er gar nicht oder zumindest so gut wie gar nicht braucht; der Nutzer wird Ihnen diese Überfrachtung der App nicht danken. Deshalb betone ich es noch einmal, einfach weil es so wichtig ist und zwischen Erfolg und Misserfolg Ihrer Idee und Ihrer App entscheiden kann:

Wichtig

Konzentrieren Sie sich auf die eine Aufgabe, die Ihre App erfüllen soll …
und dann geben Sie Vollgas!

1.4.1 Apple Human Interface Guidelines

Bevor Sie dann mit der Entwicklung loslegen, ist eines auch ganz besonders wichtig: Werden Sie mit dem Betriebssystem und dem iPhone bzw. iPad als Zielplattform vertraut. Kennen Sie diese Geräte bisher lediglich aus der Presse und der Werbung und haben sich selbst noch nie mit ihnen über einen längeren Zeitraum beschäftigt, holen Sie das erst nach. Sonst kann es leicht passieren, dass Sie die Besonderheiten der Plattform und das typische Verhalten von iOS-Apps bei Ihrer Planung und Konzeption komplett missachten und schließlich eine App entwickeln, die sich vor schlechten Kritiken kaum retten kann. Eine große Hilfe an dieser Stelle sind die iOS Human Interface Guidelines, die wertvolle Tipps zum Aufbau und zur Gestaltung Ihrer App geben. In diesem Dokument hat Apple alle Informationen zu Design und Aufbau typischer für iOS optimierter Apps zusammengefasst, es ist daher nicht nur für Einsteiger in die iOS-Entwicklung einen Blick wert (siehe Bild 1.6).

Bild 1.6 Die iOS Human Interface Guidelines sind eine sehr gute Anlaufstelle für die perfekte Gestaltung von iOS-Apps; selbst als erfahrener Entwickler lohnt ein Blick in dieses Entwicklerdokument von Apple. (Bild: https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html)

Berücksichtigen Sie bitte diesen Ratschlag, denn ich kann aus eigener Erfahrung sagen: Erst wenn Sie wissen, wie Sie mit iPhone und iPad effektiv arbeiten und wie Sie die verschiedenen Gesten und Strukturen der Apps nutzen, werden Sie imstande sein, selbst eine großartige und innovative App für genau diese Plattformen zu entwerfen und zu entwickeln. Und schließlich soll das doch das große Ziel sein, nicht wahr?

Index

M

N

O

P

Q

R

S

T

U

V

W

X

Z