HSB
Hochschule Bremen
City University of Applied Sciences

Einstiegshilfe in das iPhone und Apple Watch
Projekt Sportakus

Erstellt von:	Emel Altmisoglu
	Florian Meinert
	Jannis Lindenberg
Dozent:	Prof. Dr. Thorsten Teschke
Modul:	Mobile Computing
Semester:	SoSe17 - 6. Semester
Institut:	Hochschule Bremen
	University of Applied Science

Inhaltsverzeichnis

Abbildungsverzeichnis

Einführung

Sportakus ist eine Fitness-App, entwickelt für das iPhone und die Apple Watch. Mit der iPhone App können Trainingspläne erstellt werden und jedem Trainingsplan können Übungen hinzugefügt werden. Der Nutzer hat hierbei die Möglichkeit aus einer Vielzahl an Übungen auszuwählen. Die Trainingspläne inklusive der hinzugefügten Übungen werden in eine Datenbank gespeichert. Des weiteren ist es Möglich diese Trainingspläne an die zugehörige Apple Watch zu senden. Mithilfe der Apple Watch kann der Nutzer nun sein Training durchführen. Das besondere an der Apple Watch ist, dass diese die Wiederholungen, die der Nutzer bei einer Übung durchführt automatisch, mithilfe der in der Watch verbauten Sensoren (Bewegungssensoren), zählt. Ein durchgeführtes Training kann dann wieder an die iPhone App geschickt werden. Dort wird es in einer Trainingsübersicht für den Nutzer visualisiert. Außerdem wird ein absolviertes Trainings zusätzlich noch in der auf dem iPhone integrierten Health App angezeigt, sofern der Nutzer dies zulässt.
Die Apps wurden mit xCode 8.3.3 und Swift 3 entwickelt.

Installation

Um das Projekt auszuführen, muss zunächst das Github Projekt geklont werden. (https://github.com/JLuka/Sportakus.git). Ist das Projekt geklont, kann es durch einen doppelklick auf die Projektdatei geöffnet werden.
Um die Apps im Simulator ausführen zu können oder um sie auf den Hardware Geräten zu installieren, muss zunächst das Signing Team zu Ihrem Account gewechselt werden. Im Besten Fall sollte xCode nun automatisch ihr Provisioning Profile einrichten. Das Team muss für alle drei Targets (iPhone, WatchKit und WatchKit Extension) geändert werden.

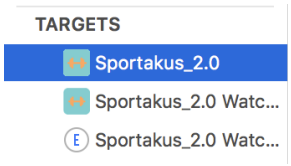

ABB. 1: TARGETS

Darüber hinaus muss noch der Bundle Identifier für alle drei Targets angepasst werden. Zum Beispiel von *de.jannis.Sportakus-2-0* zu *de.tteschke.Sportakus-2-0*. Außerdem kann es sein, dass der Bundle Identifier auch noch in den .plist Dateien geändert werden müssen. Und zwar in der WatchKit App .plist:

WKCompanionAppBundleIdentifier	String	de.jannis.Sportakus-2-0

ABB. 2: WATCHKIT APP .PLIST BUNDLE IDENTIFIER

Und in der WatchKit Extension .plist:

NSExtension	Dictionary	(2 items)
▼ NSExtensionAttributes	Dictionary	(1 item)
WKAppBundleIdentifier	String	de.jannis.Sportakus-2-0.watchkitapp

ABB. 3: WATCHKIT APP EXTENSION .PLIST BUNDLE IDENTIFIER

Nun sollte das Projekt soweit eingerichtet sein und kann im Simulator und auf einem Hardware Device installiert und gestartet werden.

Ausführen

Zum ausführen des Projekts, muss zunächst ein Device ausgewählt werden. Um die App sowohl auf dem Handy als auch auf der Uhr zu installieren/starten, muss der zu startende Device unter dem Reiter Sportakus_2.0 Watchkit App ausgewählt werden. Hier kann, wie in Abbildung 4 zu sehen ist, sowohl der Hardware Device, wie auch der Simulator ausgewählt werden. Ist die entsprechende Simulator Kombination nicht dabei, kann diese unter Add Additional Simulators hinzugefügt werden.

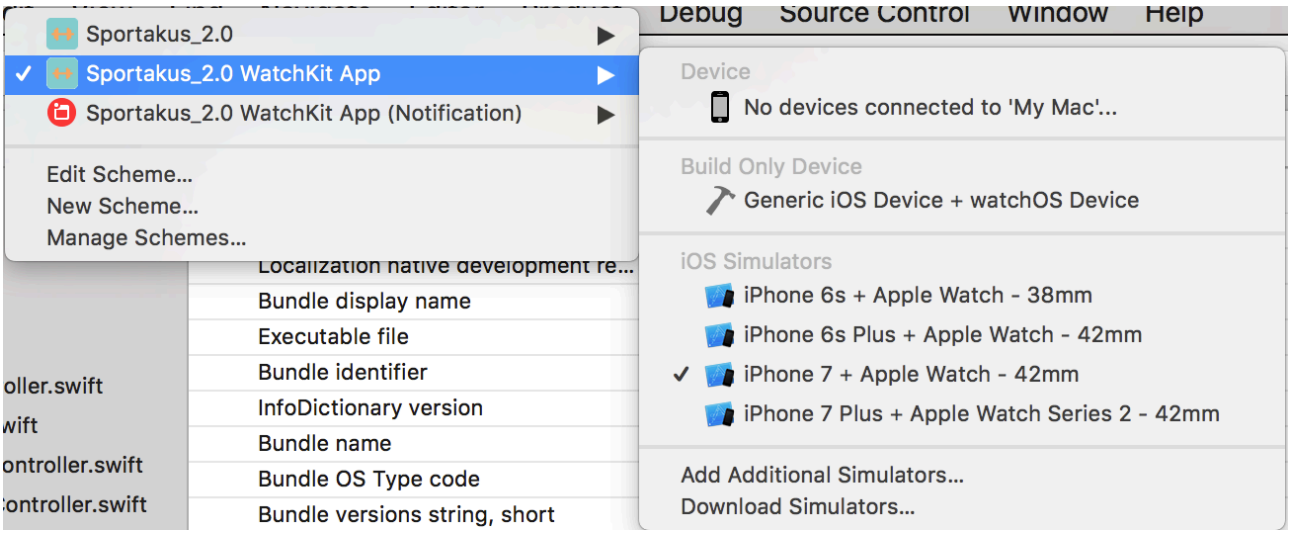

ABB. 4: SIMULATORWAHL ODER HARDWARE DEVICE AUSWAHL

iPhone

Die iPhone Applikation ist vom Grundaufbau selbsterklärend. Jedoch ist es nicht ersichtlich das ein Slider in den TableViews besteht.

Durch einen einfach Slide nach links, kann der Plan editiert werden. In der darauffolgenden View kann der Nutzer den Plan löschen oder den Namen ändern.
Darüber hinaus muss die Applikation der Apple Watch synchronisiert werden um einen Plan zu aktualisieren. Die Watch muss sich dazu in der Start View befinden.

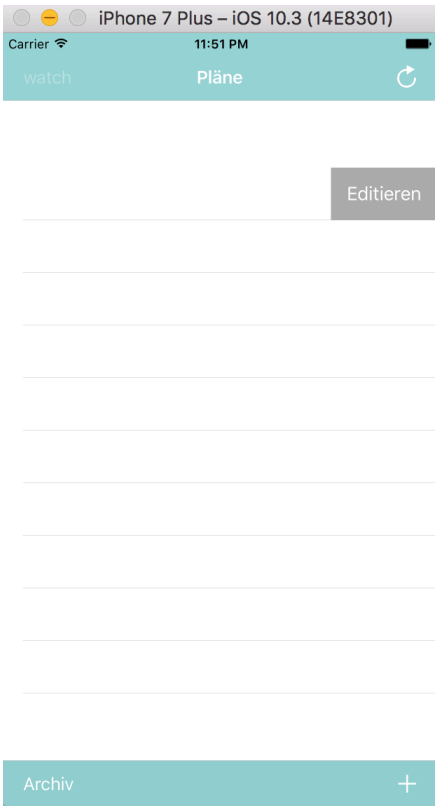

ABB. 5: ABBILDUNG APP SLIDER

ABB. 6: REFRESH BUTTON

ABB. 7: APPLE WATCH

Apple Watch

Die Apple Watch App ist aufgeteilt in die Sportakus WatchKit App und die Sportakus WatchKit Extension.
In der Sportakus Watchkit App befindet sich alles, was zur Darstellung der App wichtig ist. Also das Storyboard und der Ordner für die Assets des Projekts. Außerdem enthält die Watchkit App eine eigen .plist.
In der WatchKit Extension befinden sich dann alle Interface Controller. Also die Logik der App. Die Extension hat ebenfalls ihre eigene .plist und einen eigenen Assets Ordner, der hauptsächlich für Bilder, die bei einer Fehlermeldung angezeigt werden sollen.

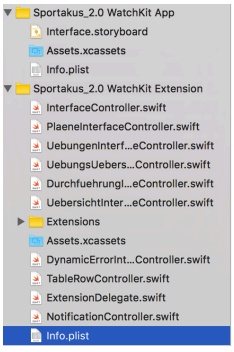

ABB. 8: ORDNERSTRUKTUR WATCH

Besonderheiten:

Ist die erste View der Apple Watch aktiv, können neue Daten des Handys empfangen werden. Wenn noch keine Pläne vorhanden sind, wird dem Nutzer eine Error View angezeigt. Im Simulator müssen die Pläne nach jedem Neustart der App (Play Button in xCode) neu übertragen werden. Grundsätzlich werden die aktuellen Pläne aber im UserDefault der Uhr gespeichert und müssen nicht bei jedem Start der App neu installiert werden.
In der Übungen View kann das aktuell durchgeführte Training an das Handy übertragen werden. Hierfür muss mindestens eine Übung durchgeführt worden sein. Das übertragene Training kann in der iPhone App jederzeit empfangen werden, sobald die App aktiv ist.

In der Durchführung View ist in der aktuellen Entwicklungsphase noch ein „Testbutton“, um gemachte Wiederholungen zu simulieren. Dieser ist notwendig, um alle Funktionen der Watch App auch im Simulator testen zu können. Der „Testbutton“ wird nach der Entwicklungsphase gelöscht.
Im Ordner WatchKit Extension befindet sich ein Unterordner „Extensions“ in dem sich unter anderem eine Extension des DurchführungInterfaceControllers befindet, die für das erkennen einer gemachten Wiederholung mithilfe der Sensoren in der Uhr ist.
Die Navigation innerhalb der Watch App funktioniert größtenteils durch Buttons. Der Navigation Back Button ist Standard in Watch Apps. Sobald ein Controller kein Initial View Controller ist, wird dieser mit dem Zurück Button ausgestattet. Dieser lässt sich in WatchOS 3 leider noch nicht ausblenden. In der Navigation durch die App sollte der Nutzer nach Möglichkeit auf die Nutzung dieses Buttons verzichten, da dieser leicht zu Verwirrungen in Bezug auf die Navigationsstruktur der App führen kann.
In der Datei HealthManager.swift in der Apple Watch Extension befinden sich alle wichtigen Methoden zur Autorisierung sowie zum starten und stoppen einer Workout-Session. Diese Klasse ist dafür zuständig, alle gesammelten Workout-Daten wie zum Beispiel die durchschnittliche Herzfrequenz während des Trainings in der HealthKit Daten Sammlung zu speichern und in der Health App anzuzeigen.

Sensoren

Für das Überprüfen der Arm Bewegungen wurde CMDeviceMotion aus dem Swift CoreMotion verwendet. Dieser bietet die attitude, gravity, user acceleration und die rotationRate. Diese können einzeln oder in Kombination für verschiedene Übungen genutzt werden. Im Ordner WatchKit Extension befindet sich in dem Unterordner „Extensions“. Dort ist eine Erweiterung namens „RepitionsExtension.swift“. In dieser wird jede Übung individuell aufgrund ihrer Funktionsweise mit der Bewegung des Anwenders verglichen und überprüft. Jede Wiederholung einer Übung besteht aus 2 Teilen. Beispiel: Hanteln - Heben, Senken. Es wird dann überprüft ob der Anwender die richtigen Bewegungen für eine Hantel Wiederholung ausführt. Nach jedem Heben-Senken wird der Zähler für die Wiederholungen erhöht, bis die zuvor definierte maximale Anzahl an Wiederholungen erreicht wird. Momentan sind folgende Übungen implementiert:
Butterfly, Butterfly Reverse, Hammer-Curls, Bizeps-Curls und Crunches. Für nicht implementierte Übungen, wie z.B. Beinpresse, wurde eine Zeitbasierte Wiederholungssteuerung implementiert. Diese lässt die Uhr vibrieren und spielt zwei verschiedene Sounds ab, wenn der Anwender beide Teile einer Wiederholung ausführen soll. Bei dem ersten Sound soll die Wiederholung gestartet werden und bei dem zweiten Sound soll die Wiederholung beendet werden. Anschliessen wird um 1 Wiederholung hochgezählt.
Für die zuvor aufgelisteten Übungen wurde die Bewegung des Anwenders mittels der CMDeviceMotion.Gravity (x, y und/oder z) geprüft. Für Butterfly, ButterflyReverse und Crunches wurde zusätzlich die CMDeviceMotion.Gravity.RotationRate benutzt.

Kommentare

Durch einen Klick mit der Maus, bei gedrückter CMD Taste, werden die Kommentare im QuickHelp Bereich nochmal übersichtlich dargestellt. (Abb. 6)

ABB. 9: QUICKHELP KOMMENTARANSICHT

Anhang

Property List Dateien - .plist

Ein auf XML basierendes Dictionary, in dem die wichtigsten Bundle Einstellungen gespeichert sind.

WatchKit

Ist ein Framework von Apple zum entwickeln von Interfaces für die Apple Watch.