

# **Weiterentwicklung der iOS Stundenplan App der Hochschule Hof**

## **Dokumentation, Spezifikation, Konstruktion**

Christian G. Pfeiffer, Johannes Franz,  
Normen Krug, Marcel Hagmann, Patrick Niepel,  
Carl Philipp Knoblauch, Angelina Scheler, Philipp Dümlein,  
Maximilian Sonntag, Bastian Kusserow

26.01.2018

Vorgelegt bei Prof. Dr. Sven Rill

# Inhaltsverzeichnis

<b>1</b>	<b>Projektablauf</b>	<b>1</b>
1.1	Ausgangssituation . . . . .	1
1.2	Überlegungen zu Projektbeginn . . . . .	2
1.2.1	GitHub / Branches . . . . .	2
1.2.1.1	Ausgangslage . . . . .	2
1.2.1.2	Verbesserungen . . . . .	2
1.3	Ziele für Version 4 . . . . .	3
1.4	Teams . . . . .	3
1.5	Kommunikation . . . . .	4
1.6	Projektfortschritt dokumentiert . . . . .	4
1.7	Fazit und Ausblick . . . . .	5
<b>2</b>	<b>Design</b>	<b>6</b>
2.1	Einleitung . . . . .	6
2.2	Stundenplan . . . . .	6
2.2.1	Vorschlag zur Verbesserung der Usability der Stundenplanansicht . . . . .	6
2.2.2	Entwicklung eines neuen Farbdesigns mit neuem Layout der Zellen . . . . .	7
2.2.3	Finales Design . . . . .	9
2.3	Onboarding . . . . .	11
2.4	Einstellungen . . . . .	12
2.4.1	Vereinfachung der Stundenplanauswahl . . . . .	13
2.4.2	Design Vorschlag . . . . .	14
2.4.3	Design Umsetzung . . . . .	14
2.5	Widget . . . . .	17
2.5.1	Design Vorschlag . . . . .	17
2.6	demo . . . . .	17
<b>3</b>	<b>Erweiterung durch eine Siri Integration</b>	<b>19</b>
3.1	Einleitung . . . . .	19
3.2	Istzustand . . . . .	20
3.3	Beispielhafte Anfragen . . . . .	20
3.4	Umsetzung . . . . .	20
3.5	Voraussetzung für SiriKit . . . . .	21
3.6	Einschränkungen . . . . .	21
3.7	Fazit . . . . .	23
<b>4</b>	<b>Push Notifications</b>	<b>24</b>
4.1	Einleitung . . . . .	24
4.2	Istzustand . . . . .	24
4.3	Projektablauf . . . . .	24

4.4	Erster Ansatz der Umsetzung . . . . .	26
4.4.1	MAMP als Virtuelle Umgebung . . . . .	26
4.4.1.1	MAMP . . . . .	26
4.4.2	Hilfsteools . . . . .	26
4.5	Vorbereitung der Umsetzung . . . . .	26
4.5.1	Server Installation . . . . .	26
4.5.2	MySQL Datenbank . . . . .	26
4.5.2.1	fcm_nutzer Tabelle . . . . .	27
4.5.3	Zertifikate . . . . .	28
4.5.4	CURL . . . . .	28
4.5.5	Sicherheit . . . . .	28
4.5.6	Builden der App . . . . .	28
4.6	Implementierung . . . . .	30
4.6.1	Server-Schnittstelle . . . . .	30
4.6.2	Registerung für Push Notification . . . . .	30
4.6.3	Sendne der Volesungsdaten zum Server . . . . .	30
4.7	Aufgetretene Probleme . . . . .	30
4.8	Fazit . . . . .	31
4.9	Weitere Arbeiten . . . . .	31
<b>5</b>	<b>Testen der Anwendung</b>	<b>32</b>
5.1	Test des Widgets . . . . .	32
5.2	Testfälle . . . . .	32
<b>6</b>	<b>Gruppe PMP</b>	<b>33</b>
6.1	Einleitung . . . . .	33
6.2	Überblick . . . . .	33
6.3	Erster Bug . . . . .	36
<b>7</b>	<b>Aufgaben Erweiterung</b>	<b>37</b>
7.1	Einleitung . . . . .	37
7.2	Planung und Mockup . . . . .	38
7.3	Funktionen . . . . .	39
<b>8</b>	<b>Kalenderschnittstelle</b>	<b>40</b>
8.1	Einleitung . . . . .	40
8.2	Überarbeitung . . . . .	40
<b>9</b>	<b>Onboarding</b>	<b>41</b>
9.1	Einleitung . . . . .	41
9.2	Umsetzung . . . . .	41
<b>10</b>	<b>Universal App Colors</b>	<b>42</b>
10.1	Beschreibung . . . . .	42
<b>11</b>	<b>Gruppe Widget</b>	<b>43</b>
11.1	Einleitung . . . . .	43
11.2	Umsetzung des Widgets . . . . .	43

11.3 Experimentell . . . . .	44
<b>12 Erweiterung um das Textfeld im JSON</b>	<b>45</b>
12.1 Einleitung . . . . .	45
12.2 Umsetzung . . . . .	45
<b>13 Studiengang Sortierung</b>	<b>46</b>
13.1 Einleitung . . . . .	46
13.2 Umsetzung . . . . .	46
<b>14 Framework</b>	<b>47</b>
14.1 Einleitung . . . . .	47
14.2 Umsetzung . . . . .	47
14.3 Wichtig . . . . .	47
<b>15 Website Termine</b>	<b>48</b>
15.1 Einleitung . . . . .	48
15.2 Umsetzung . . . . .	48
15.3 Wichtig . . . . .	48

# 1 Projektablauf

Johannes Franz & Christian Pfeiffer

## 1.1 Ausgangssituation

Im Wintersemester 2016/2017 begann die Entwicklung der iOS Stundenplanapp mit Studierenden des Studiengangs Mobile Computing im 5. Semester im Rahmen des Moduls "Fortgeschrittene Themen der Swift 3 Programmierung". Am Ende dieses Semesters wurde die Version V1 im AppStore veröffentlicht. Im Sommersemester 2017 wurde die App weiterentwickelt und schlussendlich die Version 2.0 veröffentlicht, welche eine Background Fetch Funktionalität einführte, lokale Notifications bei Vorlesungsverlegungen und Verbesserungen bei der Erkennung von einzelnen Vorlesungsterminen integrierte..

Im Wintersemester 2017/2018 übernahmen unser Studienjahrgang (Studierende des 5. Semesters) im Rahmen des Moduls "Fortgeschrittene Themen der Swift 3 Programmierung" die Pflege und Weiterentwicklung der iOS Stundenplanapp.

## 1.2 Überlegungen zu Projektbeginn

### 1.2.1 GitHub / Branches

GitHub ist eine Plattform zur effektiven Versionsverwaltung von Softwareprojekten. Branches stellen gewisse Softwareversionen dar. Das Verwenden mehrerer Branches ermöglicht es größeren Softwareteams gleichzeitig an verschiedenen Softwarefeatures zu arbeiten. Für die iOS Stundenplanapp wurde ein passendes Branch-Konzept vom Projektmanagement Team ausgearbeitet.

Der Großteil der Kommunikation lief über Issues und den Project Tab von GitHub ab.

#### 1.2.1.1 Ausgangslage

Zu Projektbeginn waren folgende Branches vorhanden:

- master
- development (obsolete)
- v2 (depricated)
- v3 (aktuell auf Deployment Target 10.0 entspricht iOS Version 10)

Von den vorherigen Projektteams wurde in GitHub ein Wiki angelegt. Link zum Wiki:

<https://github.com/HochschuleHofStundenplanapp/iOS-App/wiki>

Darin wurden in kurzer Form einige Auszüge aus den jeweiligen PDF Dokumenten zusammengetragen.

GitHub Branch Übersicht:

<https://github.com/HochschuleHofStundenplanapp/iOS-App/branches>

#### 1.2.1.2 Verbesserungen

Es wurde beschlossen, die vorherige Struktur beizubehalten. Dabei wurden Branches jeweils mit der Versionsnummer beschriftet.

Der "master" Branch wird immer mit der aktuellsten Version gefüllt.

Neue Branches sind eingeführt worden:

- v3.1: (Bugfixes, kleinere neue "Features". Der Branch bleibt auf Swift 3.1 / iOS 10.0)
- v3.2: Weitere Bugfixes. Der Branch bleibt auf Swift 3.1 / iOS 10.0
- v4: in Rahmen der Studienarbeit entwickelte Erweiterungen der App (Swift 4 / iOS 11)
- v4-widget: Subbranch der v4 zur Anpassung der App an ein Framework, welches Voraussetzung für die Implementierung eines Widget war.

Veränderungen an bestehenden Branches:

- master: Der master Branch beinhaltet immer die aktuell ausgelieferte Version aus dem Appstore.
- development: Der development Branch wurde als obsolet gekennzeichnet und deshalb entfernt.

Es wurde von der Projektgruppe entschieden beim iOS 10 Deployment Target zu bleiben, da zu Beginn des Semester iOS 11 erst veröffentlicht wurde und die Verteilung erst ein paar Monate dauerte. Zudem wurde für einige ältere Geräte, wie dem iPhone 5, iPhone 5C, and iPad 4, der Support eingestellt, weshalb einige Studierende keine App Updates mehr erhalten würden.

Letztendlich wurde entschieden das Projekt auf Swift 4 zu aktualisieren. Dies brachte u.a. Effizienzverbesserungen bei der String Manipulation. Diese wird beispielsweise in der Artificial Intelligence Klasse verwendet.

Zu Projektbeginn entstand die Idee, Branches nach Features zu benennen. Dabei war Design, Siri, Kalendersynchronisation, etc. angedacht. In der Projektphase hat sich zwischenzeitlich bewährt, Branches weiterhin nach Versionsnummern zu benennen und bei großen Änderungen der Versionsnummer ein Thema anzuhängen.

## 1.3 Ziele für Version 4

Folgende Aufgaben wurden als Ziel für die Version 4 festgelegt:

- Migration des Projektes auf Swift 4
- Onboarding
- Hausaufgaben Manager
- Push Notifications
- Widget
- iOS 11 Design
- Testkonzept für die App
- Auslagerung des Appmodels in ein Framework
- Parsen von vorlesungsfreien Tagen von der HS Webseite

## 1.4 Teams

Team 1: (Pfeiffer, Scheler)

- Design
- Buxfixes
- Swift 4 Konvertierung
- Universal App Color Persistenz

- Design des Onboarding

Team 2: (Franz, Krug):

- Siri
- Implementierung von Push Notifications
- Erweiterung der Schnittstelle

Team 3: (Hagmann, Knoblauch, Niepel):

- Verschiedene Fehlerbehebungen
- Hausaufgaben Manager
- Onboarding
- Universal App Color
- Kalenderschnittstelle überarbeitet

Team 4: (Kusserow, Sonntag, Dümmlin):

- Stundenplan Verbesserungen
- Erstellen eines Widgets
- Auslagerung des Appmodels in ein Framework
- Parseen von vorlesungsfreien Tagen von der HS Webseite

Team 5: (Pöhlmann):

- Testkonzept ausarbeiten
- Testen nach Protokoll

## 1.5 Kommunikation

Zur Verständigung untereinander und Festlegung der einzelnen Aufgaben wurden verschiedene Kommunikationsarten und Plattformen verwendet.

- Chat Gruppe mit allen Beteiligten
- Kommunikation während der Zeit in der Hochschule
- GitHub (Issues, Project Tab)
- Scrum ähnliche Vorstellung neu eingebauter Funktionen zu Beginn jeder Vorlesung

## 1.6 Projektfortschritt dokumentiert

Unter Zuhilfenahme des Projektfeature in GitHub konnte immer der Überblick über Subtasks der einzelnen Teams behalten werden und so der Fortschritt der Gruppen dokumentiert werden.

Projekt Tab in GitHub

<https://github.com/HochschuleHofStundenplanapp/iOS-App/projects>



## 1.7 Fazit und Ausblick

Zu Beginn des Semesters wurden die Nutzer mit Service Updates und Fehlerbehebungen versorgt. Hierfür entstanden die Versionen V3.1 und V3.2, welche die Stabilität der Stundenplanapp verbesserten und akute Probleme wie das Fehlen des roten Beschreibungstextes bei Stundenplanänderungen behob.

Bis zum Ende des Semesters waren wir in der Lage alle zielgesetzten Funktionalitäten in die V4 zu implementieren. Nur noch kleine Änderungen und die Übergabe der Server Komponente an den IT-Service zum Updaten der Produktiv Server Umgebung verzögerten die Veröffentlichung in den Appstore. Wie auch bei der vorherigen Version werden zu Beginn des Sommersemester 2018 weitere Qualitätsverbesserungen und Fehlerbehebungen vorgenommen werden müssen, damit die neue Version bei den Endnutzern als abgerundetes Produkt ankommt.

## 2 Design

Angelina Scheler & Christian Pfeiffer

### 2.1 Einleitung

Aufgabe des Design Teams war das Design der Stundenplanapp zu überarbeiten und zu verbessern, unter Berücksichtigung der Apple Human Interface Design Guideline. Dabei wurden für die V4 folgende Ziele gesetzt:

- Verbesserung der Stundenplanansicht
- Design eines Onboarding Assistenten
- Verbesserung und Vereinfachung der Einstellungen
- Design eines Widgets

### 2.2 Stundenplan

Die Ansicht des Stundenplans ist das primäre Feature unserer Stundenplanapp. Deshalb ist es besonders wichtig, dieses stetig zu verbessern und zu erweitern. Deshalb haben wir, um das Layout übersichtlicher zu gestalten, verschiedene Designkonzepte entworfen.

#### 2.2.1 Vorschlag zur Verbesserung der Usability der Stundenplanansicht

Zu Beginn des Semesters wurde ein Konzept entwickelt, innerhalb dessen die Stundenplanfunktion durch eine Wochenansicht erweitert werden sollte.

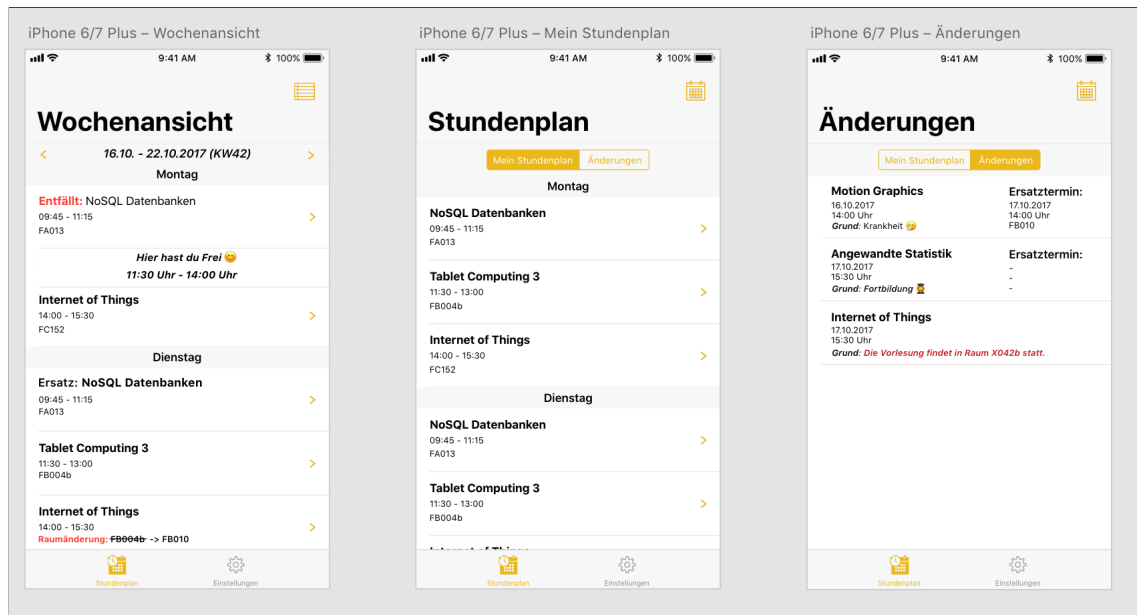


Abbildung 2.1: Neues Stundenplanbedienkonzept

Mit dem rechten oberen Navigationitem kann zwischen einer Wochenansicht und der klassischen Stundenplanansicht gewechselt werden. In der klassischen Stundenplanansicht kann der Benutzer durch ein ein Segmented Control Element unterhalb der Navigationbar zwischen 'Mein Stundenplan' und den Stundenplanänderungen wechseln. In der Wochenansicht werden 'Mein Stundenplan' und die Stundenplanänderungen in einer Ansicht für die aktuelle Woche kombiniert dargestellt.

### 2.2.2 Entwicklung eines neuen Farbdesigns mit neuem Layout der Zellen

In der vorherigen Version der Stundenplanapp hatte jede Funktion eine eigene, dem Hochschulfakultäten entsprechende, Farbe. Da das Wechseln der Farbe, bei Auswahl einer neuen Funktion inkonsistent ist, hat das Designteam ein Konzept entwickelt, durch welches der Appnutzer in den Einstellungen seine Fakultät wählen kann und die App damit mit der gewählten Fakultätsfarbe anpassen kann. Die folgenden Abbildungen zeigen die Stundenplanansicht in den jeweiligen Fakultätsfarben. Das Layout der Stundenplanzellen wurde darüber hinaus auch verbessert.

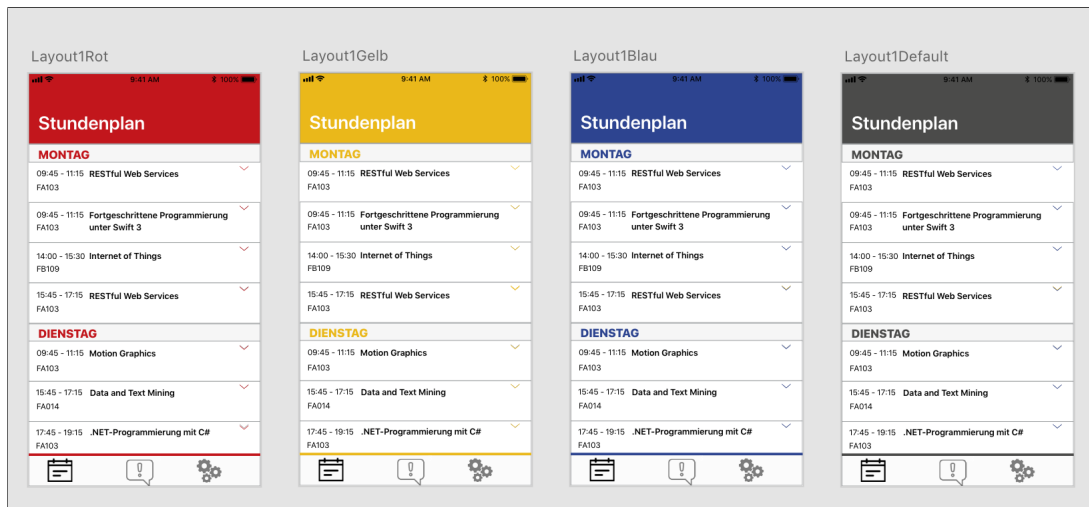


Abbildung 2.2: Design Konzept 1, Stundenplan Ansicht

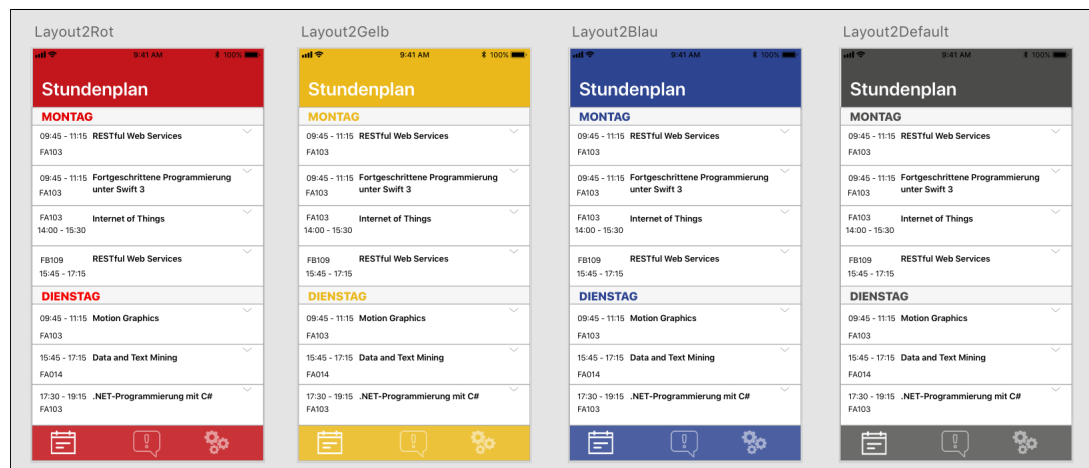


Abbildung 2.3: Design Konzept 2, Stundenplan Ansicht

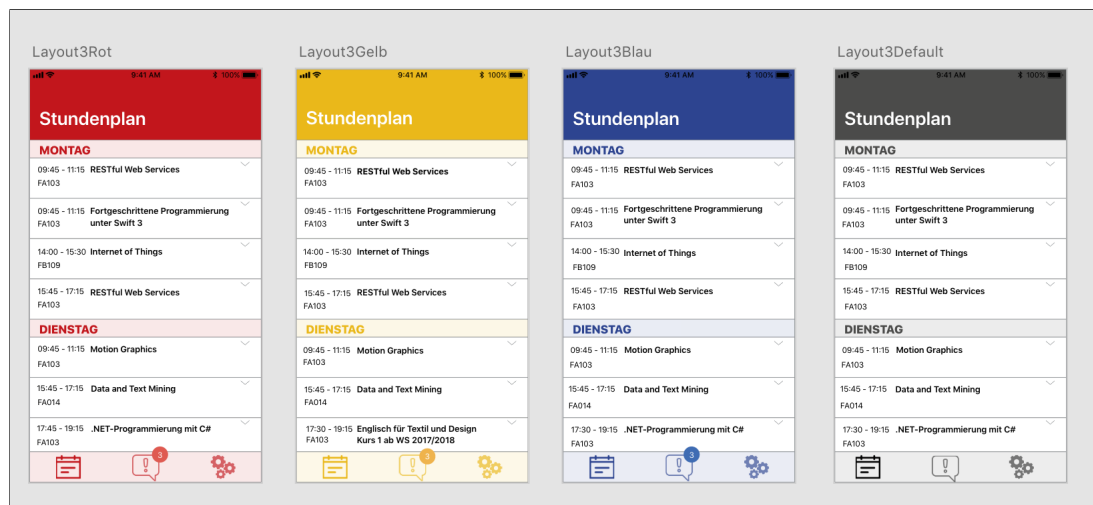


Abbildung 2.4: Design Konzept 3, Stundenplan Ansicht

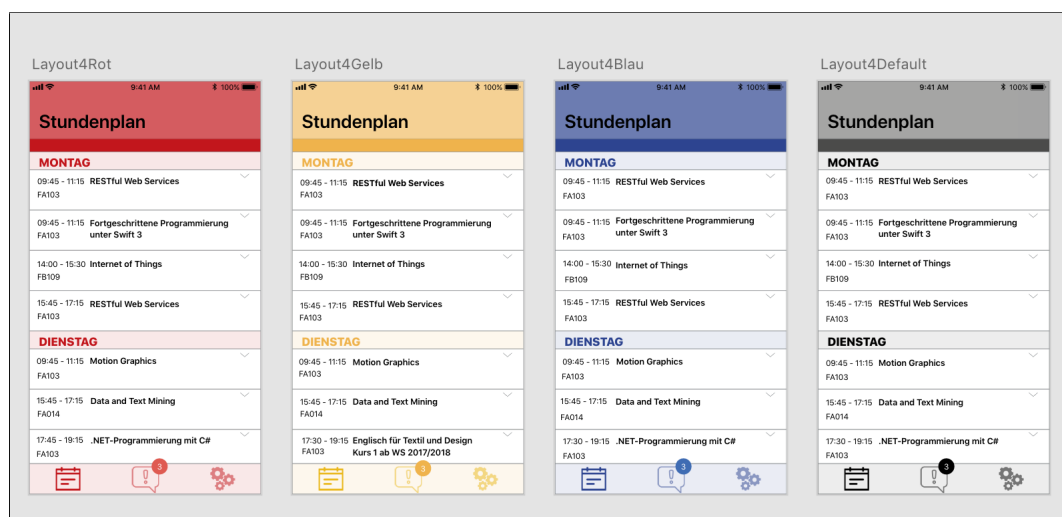


Abbildung 2.5: Design Konzept 4, Stundenplan Ansicht

## 2.2.3 Finales Design

Das Finale Design ist ein Kompromiss aus Altem und Neuem Layout, die Fakultätsfarben wurden eingeführt, Informationen übersichtlicher positioniert und farblich angepasst.

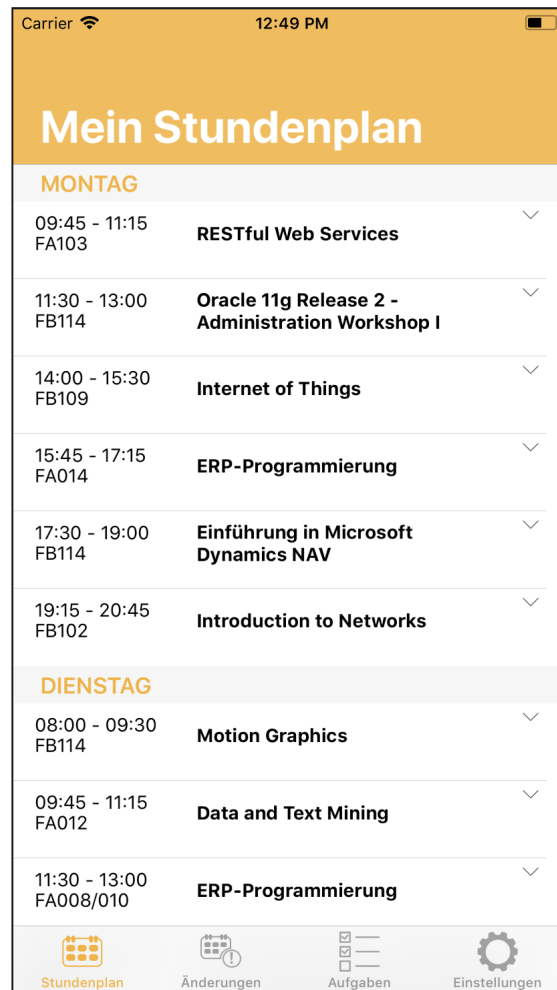


Abbildung 2.6: Finales Design, Stundenplan Ansicht

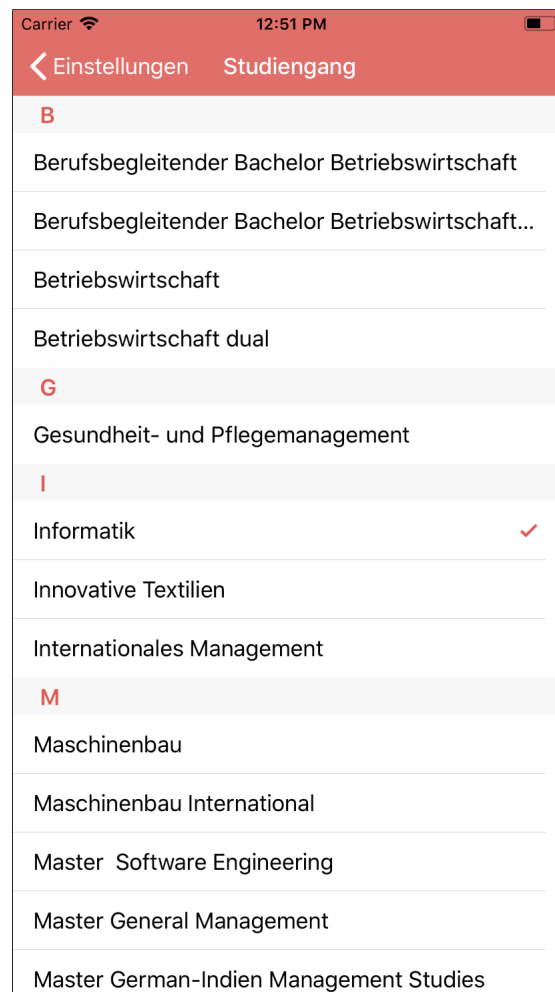


Abbildung 2.7: Finales Design, bsp. Auswahl Studiengänge

## 2.3 Onboarding

Um neuen Appnutzern eine geführte Einführung in die Funktionalität und Einstellungsmöglichkeiten zu bieten, ist die Implementation einer Onboardingfunktion sinnvoll. Deshalb wurden Mockups einer solchen Funktion entwickelt, die einen solchen Onboardingprozess darstellen.

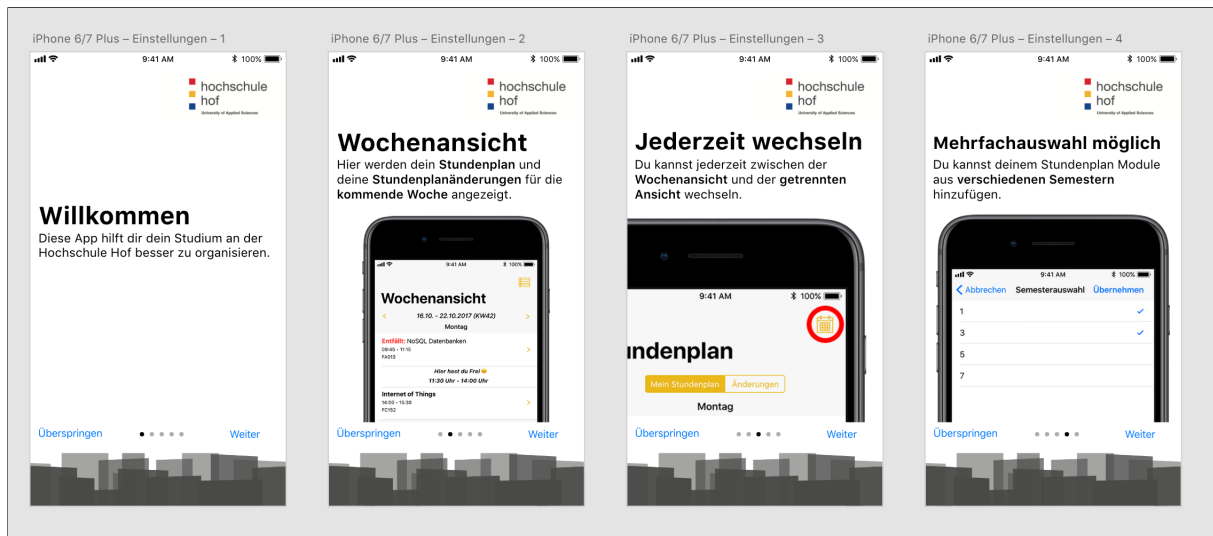


Abbildung 2.8: Neues Stundenplanbedienkonzept

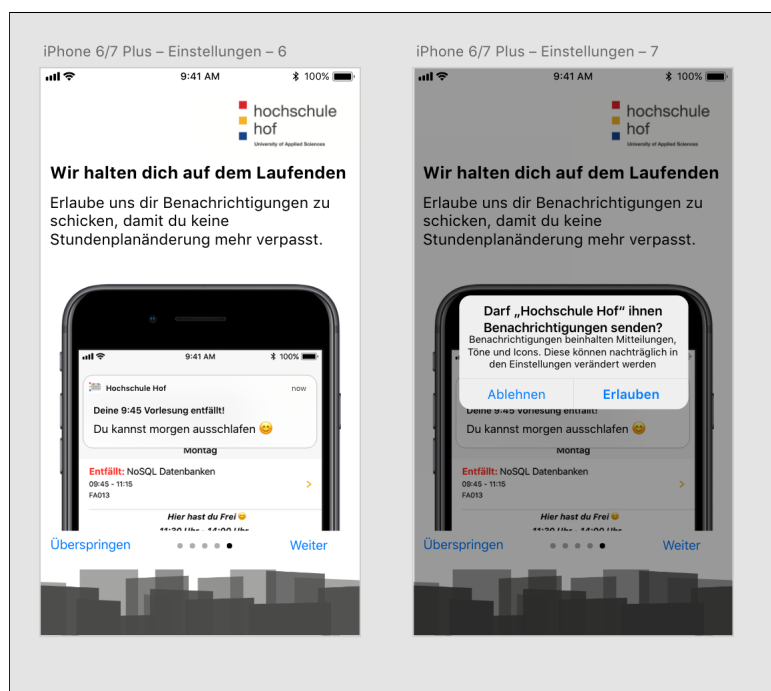


Abbildung 2.9: Neues Stundenplanbedienkonzept

## 2.4 Einstellungen

Das Einrichten der App unter Einstellungen sollte durch eine übersichtlichere Gestaltung vereinfacht werden.



## 2.4.1 Vereinfachung der Stundenplanauswahl

In der vorherigen Versionen der Stundenplanapp, war die Auswahl eines persönlichen Stundenplans

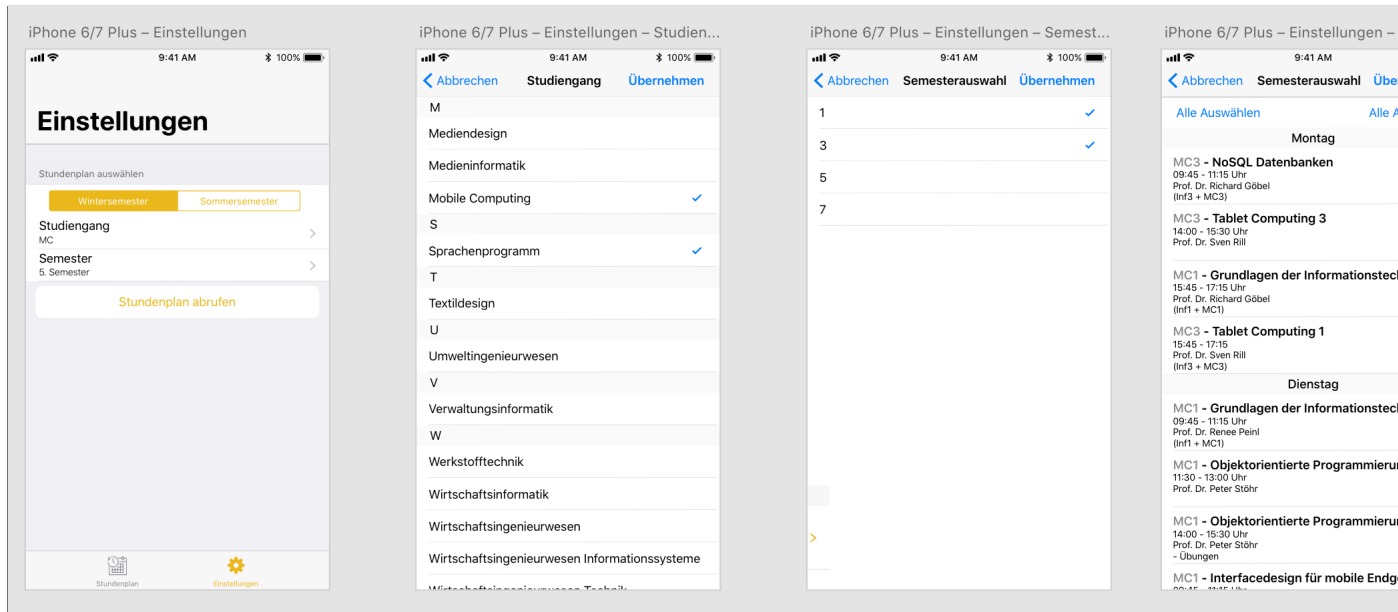


Abbildung 2.10: Optimierung der Schritte zur Stundenplanauswahl auf die nötigsten Schritte

## 2.4.2 Design Vorschlag



Abbildung 2.11: Design Konzept, Stundenplan Einstellungen

## 2.4.3 Design Umsetzung

Es wurde ein Menüpunkt für die Fakultätsauswahl und starten des Einführungsassistenten eingefügt. Desweiteren wurden in den jeweiligen Einstellungsscreens wie, Studiengang, Semester und Vorlesungen anpassungen vorgenommen, wie die Alphabetisierung siehe Abbildung 2.8.. Außerdem wurden alle Inhalte je nach Fakultätsauswahl farblich angepasst.

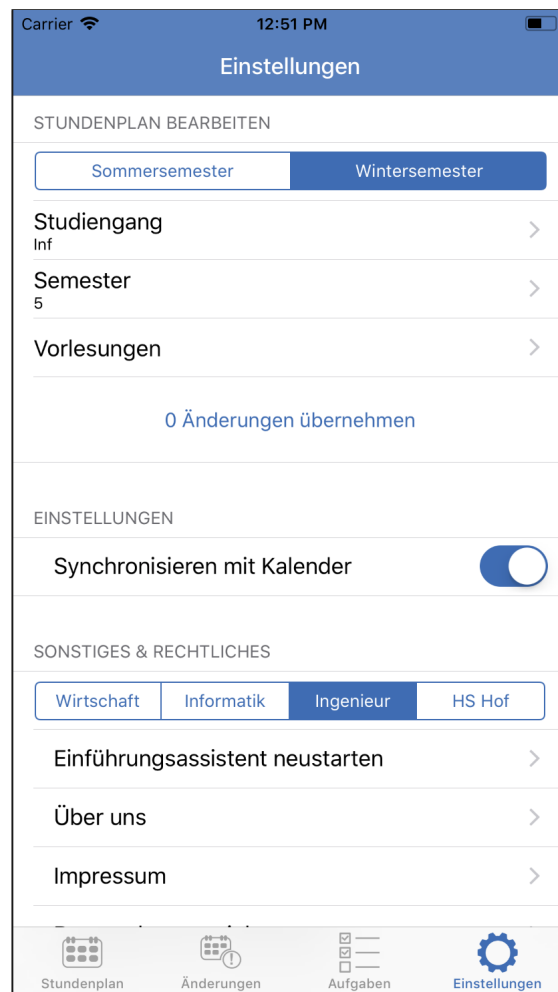


Abbildung 2.12: Finales Design, Stundenplan Einstellungen

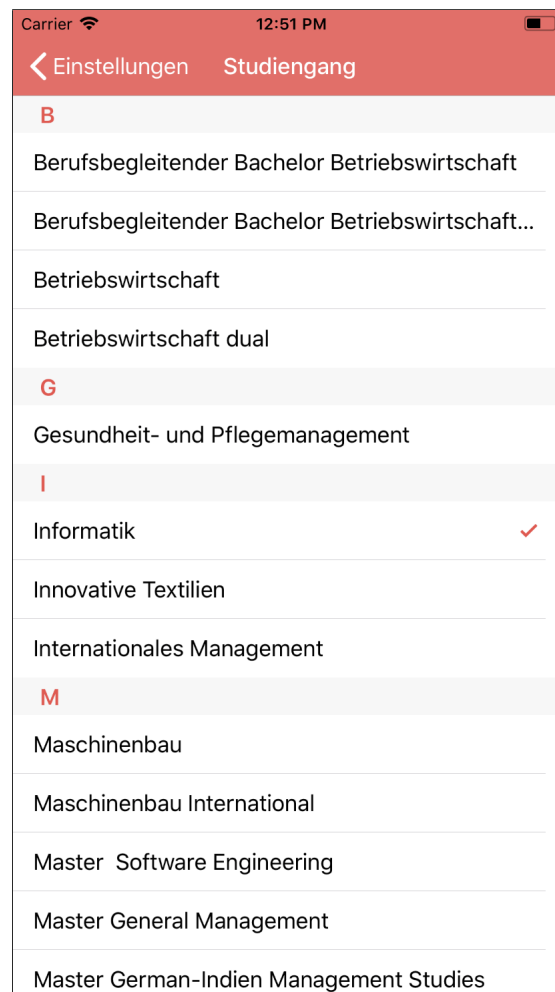


Abbildung 2.13: Finales Design, bsp. Auswahl Studiengänge

## 2.5 Widget

### 2.5.1 Design Vorschlag

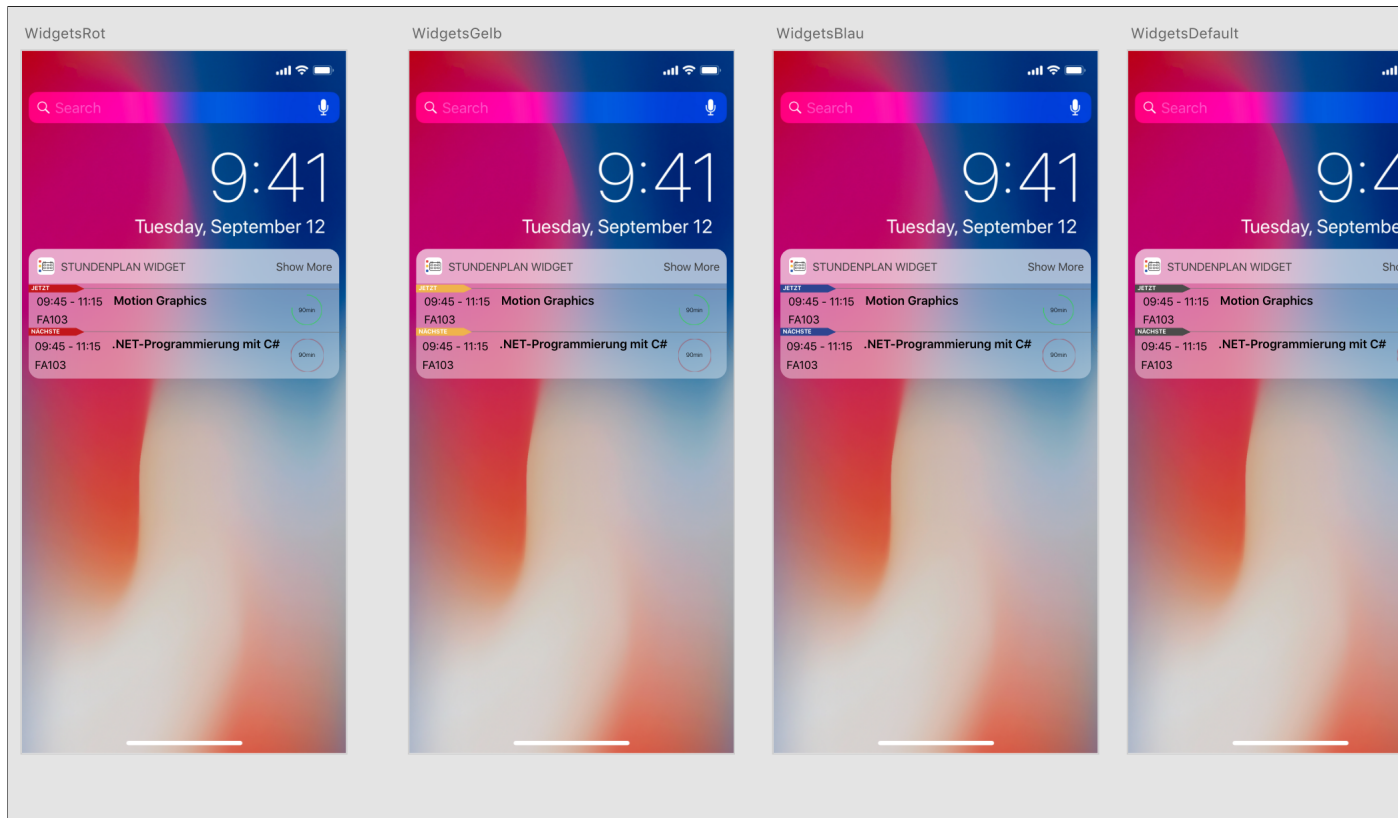


Abbildung 2.14: Design Vorschlag, Widget

## 2.6 demo

Eine weitere Aufgabe, die unsere Gruppe übernommen hat, war das Onboarding für die Stundenplan App. Im Onboarding kann der Nutzer gleich zum Start der App seine Einstellungen zu Fakultät, Studiengang, Semester, Vorlesung und Kalendersynchronisation vornehmen. Das Onboarding wird gestartet, wenn noch keine Angaben zu den genannten Einstellungen gemacht wurden.

- Neues Design
- besseres Design
- iOS Design

1. Fakultät (die die Farbe der App bestimmt)

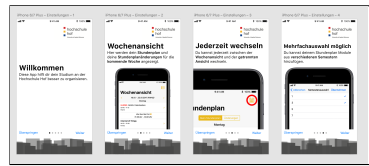


Abbildung 2.15: Mein schönstes Mockup

## 3 Erweiterung durch eine Siri Integration

Johannes Franz & Normen Krug

### 3.1 Einleitung

Da sprachbasierte Mensch-Maschinen-Interface immer beliebter und praktikabler werden, ist es naheliegend, dass die Stundenplan App um diese erweitert wird. Apple bietet mit Siri solch einen Sprachassistenten ein. Dieser ist tief im System eingebaut und wird daher von Apple gut unterstützt. Da es weiterhin das Ziel sein soll, die App zur Nutzung nicht öffnen zu müssen, ist eine Siri Integration der nächst logische Schritt zur Weiterentwicklung der Anwendung.

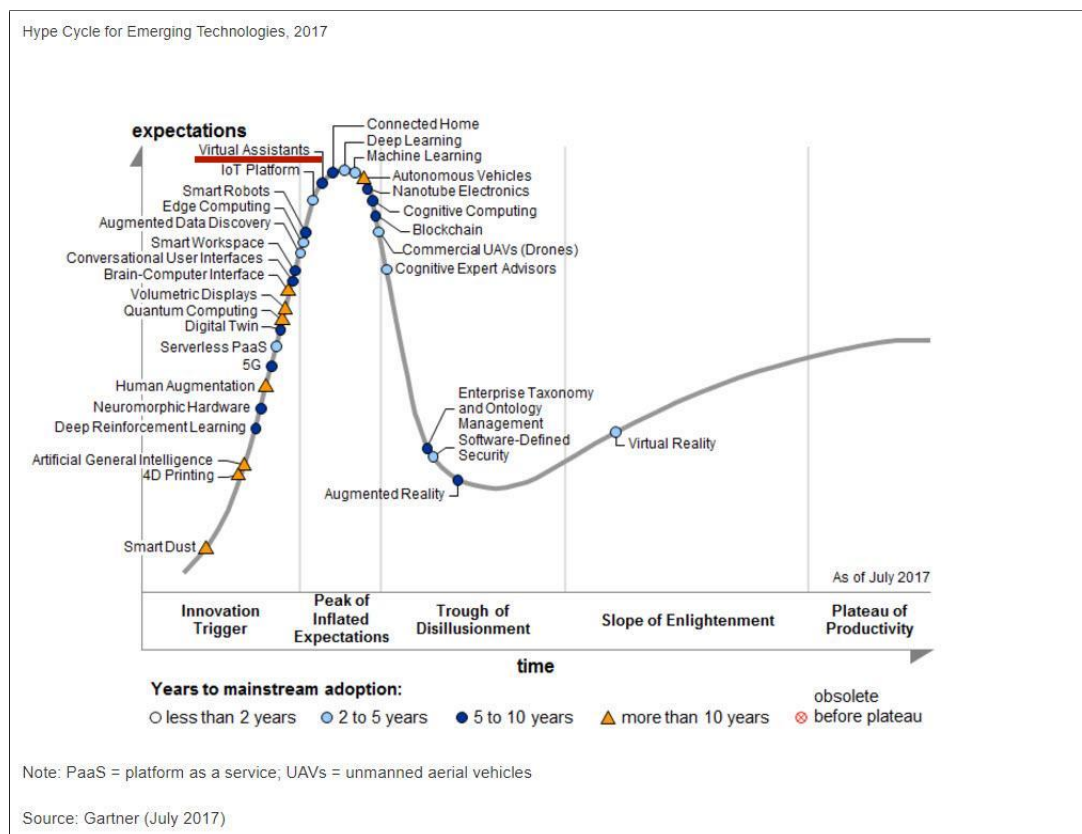


Abbildung 3.1: hype cycle for technologies 2017

## 3.2 Istzustand

Die App bietet aktuell keine Unterstützung für Sprachbefehle an. Eine Eingabe durch Sprachbefehle durch die Apple Watch ist so ebenfalls nicht möglich. Im Hinblick auf barrierefreie Bedienung hat die App daher noch Verbesserungspotential.

## 3.3 Beispielhafte Anfragen

Um die gängigen Anfragen abzudecken, müssen diese in der App vorher festgelegt werden. Um das Anliegen des Benutzers verstehen zu können müssen unterschiedliche Fragestellungen die zum selben Ergebnis führen abgedeckt werden.

Frage	Reaktion
Wann/Wo ist meine nächste Vorlesung?	Name der Vorlesung, Zeitspanne und Raumnummer vorlesen. UI zeigt diese Information noch einmal an.
Habe ich heute noch eine Vorlesung?	Ja/Nein Antwort, Name der Vorlesung mit Uhrzeit. Wenn ja wird im UI etwas angezeigt.
Fällt heute etwas aus?	Ja/Nein Antwort, Name der Vorlesung mit Uhrzeit mit Grund. Wenn ja wird im UI etwas angezeigt.
Welche Änderungen gibt es heute?	Aufzählung der Änderungen für heute. Auflisten dieser Änderungen im UI
In welchem Raum ist Vorlesung X? (optional/tricky)	Name der Vorlesung und Raum wird genannt. Informationen im UI werden angezeigt.

## 3.4 Umsetzung

Apple bietet für die Umsetzung des Sprachassistenten das SiriKit an. Mittels einer “Applications Extension” namens “Intents Extension” ist es möglich Hooks für eine Reaktion der Stundenplan App zu definieren. Für die Ausgabe im Lockscreen kann dabei ein angepasstes UI zur Verfügung gestellt werden, welches die Sprachausgabe ergänzt.



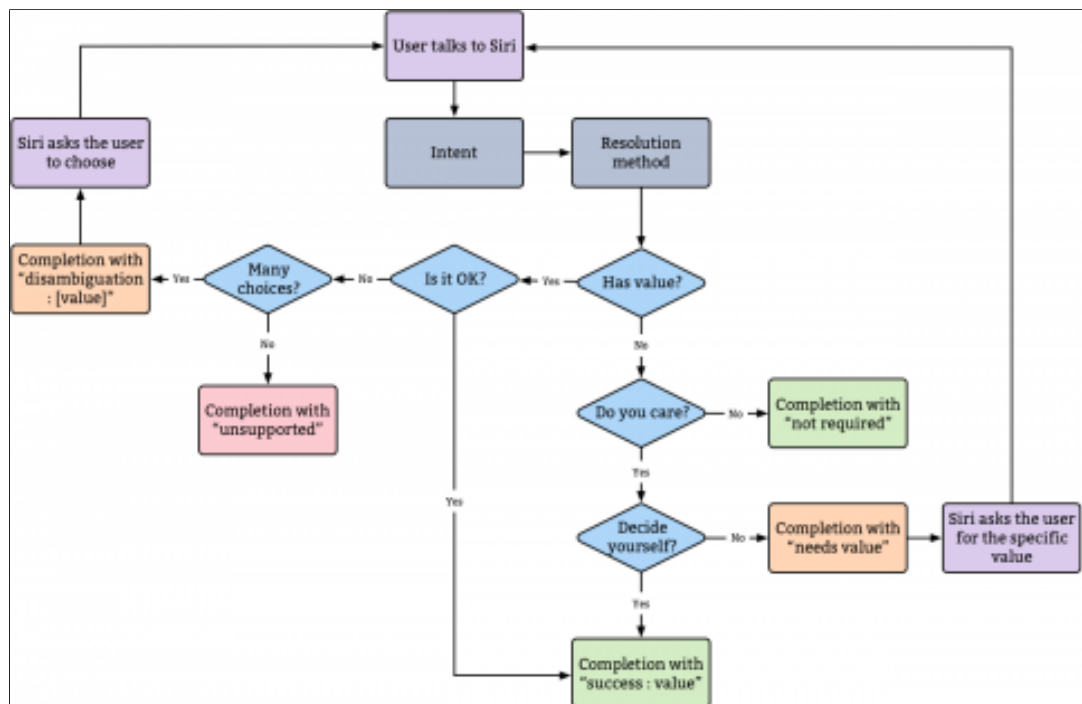


Abbildung 3.2: Siri Flowchart

### 3.5 Voraussetzung für SiriKit

Stand: 10.10.2017

Um SiriKit verwenden zu können, müssen die Kernbereiche der App in ein Framework ausgelagert werden. Apple empfiehlt das bei allen Erweiterungen. Dieser Schritt ist notwendig, weil der Benutzer eine Interaktion mit Siri starten kann auch wenn die App zurzeit nicht läuft.

### 3.6 Einschränkungen

Apple gewährt keinen vollständigen Zugriff auf Siri. Die zu entwickelte App muss in eine der folgenden Kategorien/Domains fallen:

- VoIP Calling
- Messaging
- Payments
- Lists and Notes
- Visual Codes
- Photos
- Workouts

- Ride Booking
- Car Commands
- CarPlay
- Restaurant Reservations

Die Schlüsselwörter welche Siri voraussetzt, um zu erkennen dass der Benutzer mit der App interagieren will, hängen von der Domain ab. Diese Schlüsselwörter müssen zwingend in der Anfrage des Benutzer enthalten sein.

Beispiel:

In der „Search Message“ Domain müssen die Wörter „Suche“ und „Nachrichten“ enthalten sein. Falls eines der beiden Wörter nicht in der Anfrage enthalten ist, erkennt Siri die Anfrage nicht.

Passender Blog-Post zu den Thema:

<https://swiftling.io/blog/2016/07/18/20-sirikit-can-you-outsmart-provided-intents/>

Mögliche Workarounds:

Es ist theoretisch möglich mit der „Lists and Notes“ Domain, die Funktion für die Stundenplan hinzubiegen. Da aber die bestimmten Schlüsselwörter enthalten sein müssen, wird aber kein natürlich sprachliche Interaktion möglich sein. <https://developer.apple.com/documentation/sirikit>

### 3.7 Fazit

Die vielversprechenden Idee die App um einen Sprachassistenten zu erweitern, ist zum aktuellen Zeitpunkt nicht realisierbar aufgrund der Limitierung von SiriKit. Das Projekt musste an dieser Stelle unterbrochen werden und das Team musste sich neu orientieren.

Da Apple den Sprachassistenten Siri stets erweitert, kann allerdings darauf gehofft werden, dass einer Umsetzung zu einem späteren Zeitpunkt keine Barrieren mehr im Wege stehen.

## 4 Push Notifications

Johannes Franz & Normen Krug

### 4.1 Einleitung

Push Notifications sind ein fester Bestandteil moderner Apps. Nutzer erwarten es häufig bei Änderungen oder Neuigkeiten informiert zu werden. Deswegen soll die Stundenplan App um solche erweitert werden. Ziel soll es sein den Benutzer über Stundenplanänderungen aktiv zu informieren, um speziell auf kurzfristige Änderungen reagieren zu können. Ein Server überprüft dabei eine Stundenplan Datenbank auf Änderungen und sendet eine Push Notification an alle iOS Geräte, die sich für die jeweilige Vorlesung registriert haben. Ein Apache Webserver stellt dabei mit einer MySQL Datenbank und entsprechenden Chronjobs das Backend bereit.

GitHub repository: <https://github.com/HochschuleHofStundenplanapp/iOS-App/>

### 4.2 Istzustand

Die iOS Version stellt nur lokale Push Notifications bereit. Dabei wird bisher keine Serverkomponente benötigt.

### 4.3 Projektablauf

Um den Projektfortschritt nachvollziehen zu können, wurde dieser tabellarisch in Verbindung mit dem aktuellen Datum aufgelistet.

Datum	Erreichter Meilenstein
Vorlesungsbeginn	Themenfindung
Siri	Einarbeitung in Siri. Erkennen erster Hürden.
Neue Themenfindung	Verwerfen von der Siri Projektidee und neue Themenfindung.
Festlegung	Thema Push Notifications festgelegt und Beginn der Einarbeitung.
28.10.2017	Push Notifications lassen sich per PHP Script an einen fest installierten Token schicken
30.10.2017	Eine Test iOS App kann per php Script mittels MAMP lokal eine Push Notification senden. Datenbank lokal in PhpMyAdmin angelegt. PHP Script schreibt bei Aufruf in die angelegte SQL Datenbank. Einarbeitung und Konvertierung der Dokumentation in Latex.
31.10.2017	Die Test iOS App wurde so erweitert, dass ein JSON File per POST Nachricht übermittelt werden kann. Das PHP Script parst nun das ankommende JSON File und fügt per insert die geparsten Daten in die Datenbank ein. Einarbeitung in bestehende Schnittstelle und Überlegungen wie das bestehende Backend erweitert werden muss.
Zwischenzeit	PHP Skripte der bestehenden (Android) Schnittstelle angepasst und getestet.
24.11.2017	Testserver wurde bereitgestellt.
2018	Push Notifications mit HTTP2 implementiert.

## 4.4 Erster Ansatz der Umsetzung

### 4.4.1 MAMP als Virtuelle Umgebung

#### 4.4.1.1 MAMP

Die Testumgebung MAMP (Akronym steht für: Mac, Apache, MySQL, PHP) virtualisiert die genannten Komponenten, um lokale Tests zu ermöglichen.

### 4.4.2 Hilfstools

Die MacOS App “Easy APNs Provider“ ermöglichte ersten Gehversuche, um Push Nachrichten an ausgewählte Token zu senden. Zu beachten ist das der “Easy APNs Provider“ das veraltete Push Interface benutzt.

Quelle: <https://itunes.apple.com/de/app/easy-apns-provider-push-notification-service-testing/id989622350?mt=12>

## 4.5 Vorbereitung der Umsetzung

Apple bietet für die Umsetzung von Push Nachrichten den Apple Push Notification service (APNs) an. Dabei handelt es sich um einen bei Apple gehosteten Dienst, der Push Notifications per API ermöglicht. Nur dieser Dienst ist es erlaubt, die Push Notification direkt an das iOS Gerät zu senden.

### 4.5.1 Server Installation

Zur Installation der Software sind Root Rechte notwendig. Der *wget* Befehl lädt dabei eine zum aktuellen Zeitpunkt sehr neue *curl* Version herunter. Diese hat die Besonderheit HTTP2 zu unterstützen, welches für die Push Notification Schnittstelle von Apple vorausgesetzt ist.

### 4.5.2 MySQL Datenbank

Die Tabelle *fcm\_nutzer* enthält eine Zuordnung von abonnierten Vorlesungsverlegungen mit dem jeweiligen Token des Gerätes. Dabei ist zusätzlich vermerkt welches Betriebssystem verwendet wird. Die *0* in der Spalte *os* steht dabei für Android während *1* iOS repräsentiert. Da die ausgewählte Sprache des Nutzers nicht bekannt ist, wird für die *language* Spalte an dieser Stelle auch null akzeptiert. Um bei einer späteren Version der App zwischen Nutzern die noch keine Sprache ausgewählt haben und allen anderen differenzieren zu können, wird hier trotz der aktuell nur in deutsch vorhandenen App der Wert nicht standardmäßig auf deutsch gesetzt.

```
CREATE TABLE 'fcm_nutzer' (  
  'id' int(11) NOT NULL AUTO_INCREMENT,  
  'token' varchar(255) CHARACTER
```

```

        SET utf8 COLLATE utf8_unicode_ci NOT NULL,
        'vorlesung_id' varchar(255) CHARACTER
        SET utf8 COLLATE utf8_unicode_ci NOT NULL,
        'os' int(11) DEFAULT '0',
        'language' varchar(2) DEFAULT NULL,
        PRIMARY KEY ('id'),
        KEY 'token' ('token'),
        KEY 'vorlesung_id' ('vorlesung_id')
    );

```

Da eine bestehende Infrastruktur die Grundlage dieses Projektes darstellt, musste diese Tabelle lediglich um *os* und *language* erweitert werden.

Auf Änderungen an anderen Tabellen konnte komplett verzichtet werden.

#### 4.5.2.1 fcm\_nutzer Tabelle

Diese Tabelle beinhaltet vor allem die Verknüpfung zwischen dem eingetragenen Token (*token*), der abonnierten Vorlesung (*vorlesung id*) und dem mobilen Betriebssystem (*os*). Diese Tabelle wurde beim Testsystem und beim Produktivsystem bereits um die neue Spalte *os* und *language* erweitert. Für letzere wurde die Schnittstelle, iOS und Android Version angepasst und getestet. Diese Funktion kann für zukünftige Implementierungen verwendet werden.





































			id	token	vorlesung_id	os	language
<input type="checkbox"/>				2160384	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	WiArBW\$anöfer_1%52030 \$ 2	1 de
<input type="checkbox"/>				2160385	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	WiArBW\$anöfer_1%52029 \$ 2	1 de
<input type="checkbox"/>				2160386	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	WiArBW\$anöfer_1%52028 \$ 2	1 de
<input type="checkbox"/>				2160387	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	WiArBW\$nöfer%54057 \$ 2	1 de
<input type="checkbox"/>				2160388	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	SouAP\$dorschner%53953 \$ 2	1 de
<input type="checkbox"/>				2160389	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	BetSem\$gsmola%52113 \$ 2	1 de
<input type="checkbox"/>				2160390	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	SouAP\$dorschner%53567 \$ 2	1 de
<input type="checkbox"/>				2160391	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	AktHEU\$froesch%53694 \$ 2	1 de
<input type="checkbox"/>				2160392	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	SouAP\$dorschner%53952 \$ 2	1 de
<input type="checkbox"/>				2160393	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	SouAP\$dorschner%53951 \$ 2	1 de
<input type="checkbox"/>				2160394	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	SouAP\$dorschner%53950 \$ 2	1 de
<input type="checkbox"/>				2160395	61a55c71301eb16a3fe373bbc3f065eca8dc065f80bf6d8e8f...	BetSem\$gsmola%52118 \$ 2	1 de

Abbildung 4.1: Tabelle fcm\_nutzer in phpmyadmin

```

wget https://curl.haxx.se/download/curl-7.54.0.tar.bz2
tar -xvjf curl-7.54.0.tar.bz2
cd curl-7.54.0/
./configure --with-nghttp2 --prefix=/usr/local --with-ssl=/usr/local/ssl
make
make install
ldconfig

```

```
ggf. php7.0-curl erneut installieren  
// Apache neu starten
```

### 4.5.3 Zertifikate

Um mit einer gesicherten Verbindung auf die Apple Push Notification Schnittstelle zuzugreifen, werden zwei Zertifikate benötigt. Dabei handelt es sich um ein öffentliches und privates Zertifikat. Diese Zertifikate müssen vor der Verwendung in das Zielformat *.pem* umgewandelt werden, bevor sie benutzbar sind.

Um die Zertifikate zu anzulegen und zu verwalten, stellt Apple eine Übersicht dem bei Apple angemeldeten Entwickler bereit. Bei dem verwendeten Zertifikat wird zwischen *Development* und *Production* unterschieden.

<https://developer.apple.com/account/ios/certificate/>

### 4.5.4 CURL

Der Server verwendet CURL um die Push Notification an den Service von Apple zu senden. CURL verwendet dabei das HTTP2 Protokoll.

```
curl --http2 -d '{"aps":{"alert":"[MESSAGE TEXT]","sound":"default"}}' --cert  
  ↪ "server_certificates_bundle_sandbox.pem":" " -H "apns-topic: iosapps.  
  ↪ hof-university.stundenplan" --http2 https://api.development.push.apple.  
  ↪ com/3/device/424  
  ↪ f676a488ee552061b085b7565fac714de9c864ffbe0855b1cce4c95ac24cb
```

### 4.5.5 Sicherheit

Das Thema Sicherheit spielt in der IT eine Zentrale, aber oft vernachlässigte Rolle. So ist beim der Einrichtung auf dem Server auf einige Punkte zu achten. Die hier verwendeten Software Versionen wie Apache2, MySQL oder CURL können auf dem Server auf die in der Zukunft aktuelle Version geupdatet werden. Die Dateirechte der PHP Dateien sowie Zertifikate müssen entsprechend angepasst werden und nur gewissen Usern angehören.

Da das Projekt öffentlich auf GitHub für jeden einsehbar ist, müssen die Passwörter so gewählt werden, dass sie keinem vorher verwendeten Passwort gleichen. Dies war bisher im Umgang mit dem Testsystem absichtlich nicht der Fall, was praktische Vorteile mit sich gebracht hat.

### 4.5.6 Builden der App

Bevor die App verwendet werden kann, muss speziell im Fall der Push Notifications auf gewisse Details geachtet werden.

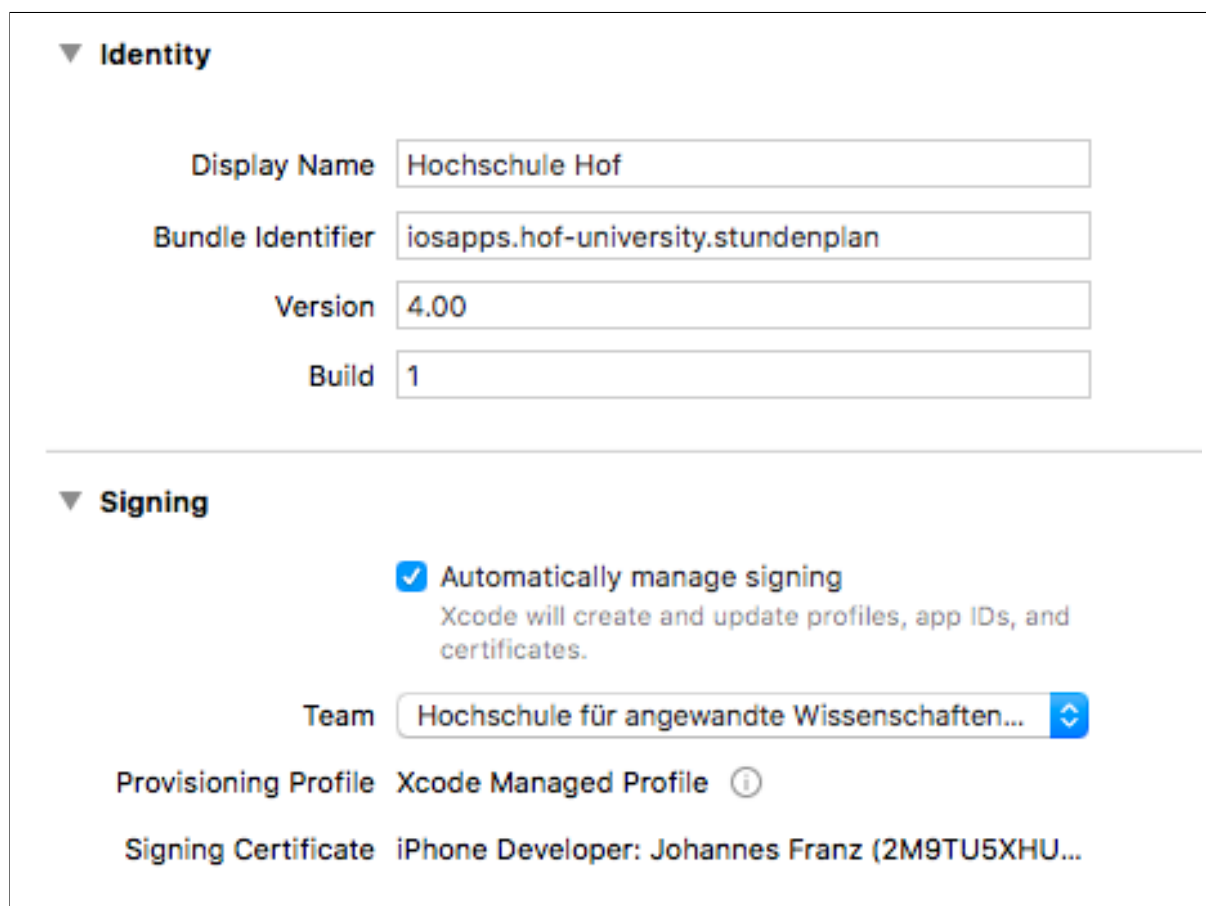


In der *Info.plist* stehen drei neue Parameter für Push Notifications bereit. Diese ermöglichen das Hin- und Herschalten zwischen dem Test- und Produktivserver mittels des *isPushTesting* Parameters. Die beiden anderen Parameter repräsentieren sprechend den jeweiligen Link. Bei dieser Einstellung ist darauf zu achten, dass sie sich nur auf die Anmeldung am jeweiligen Server bezieht. Informationen wie Stundenpläne und Änderungen kommen unabhängig von dieser Einstellung weiterhin vom Produktivserver ( <https://app.hof-university.de/soap/> ).

Key	Type	Value
▼ Information Property List	Dictionary	(21 items)
ProductiveURL	String	<a href="https://app.hof-university.de/soap/">https://app.hof-university.de/soap/</a>
TestURL	String	<a href="https://apptest.hof-university.de/soap/">https://apptest.hof-university.de/soap/</a>
isPushTesting	Boolean	YES

Abbildung 4.2: Anpassungen in der Info.plist

Das Developer Team muss wie folgt ausgewählt werden, damit Push Notifications zur Verfügung stehen:



The screenshot shows the Xcode settings for a project. Under the 'Identity' section, the 'Display Name' is 'Hochschule Hof', 'Bundle Identifier' is 'iosapps.hof-university.stundenplan', 'Version' is '4.00', and 'Build' is '1'. Under the 'Signing' section, 'Automatically manage signing' is checked, and the 'Team' is set to 'Hochschule für angewandte Wissenschaften...'. The 'Provisioning Profile' is 'Xcode Managed Profile' and the 'Signing Certificate' is 'iPhone Developer: Johannes Franz (2M9TU5XHU...'.

Abbildung 4.3: Wahl des Developer Teams

Capabilities müssen wie folgt angepasst werden:

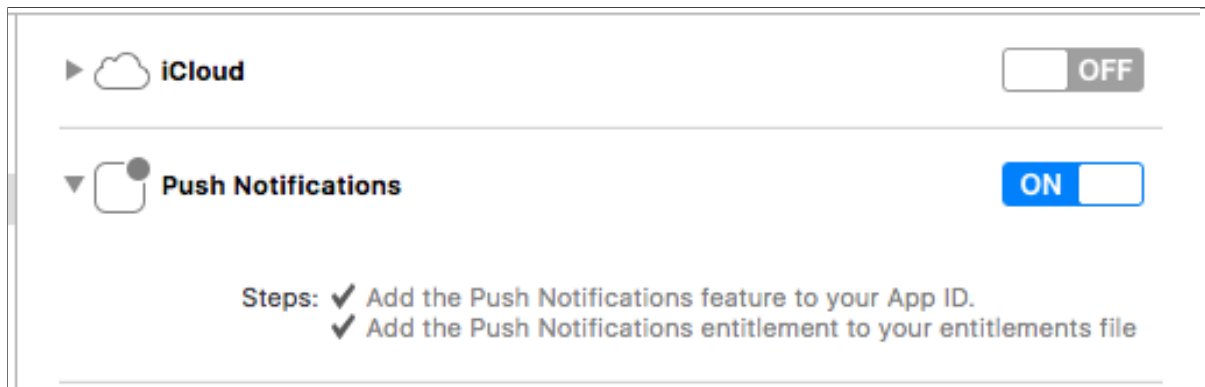


Abbildung 4.4: Angepasste Capabilities

## 4.6 Implementierung

### 4.6.1 Server-Schnittstelle

### 4.6.2 Registrierung für Push Notification

### 4.6.3 Sendne der Volesungsdaten zum Server

Code zum Registrieren am Server Um sich am Server für eine Vorlesung zu registrieren waren Veränderungen im Code notwendig.

```
func application(_ application: UIApplication,
    ↪ didRegisterForRemoteNotificationsWithDeviceToken deviceToken: Data) {
    print("DeviceToken1: \(deviceToken)")
    let token = deviceToken.map { String(format: "%02.2hhx", $0) }.joined()
    print("DeviceToken: \(token)")
}
```

## 4.7 Aufgetretene Probleme

Als Herausforderung zu sehende Punkte haben häufig sehr viel Zeit in Anspruch genommen oder den Projektfortschritt entschleunigt.

- Verzögerte Bereitstellung des Servers führte zu komplexen Reimplementierungen in schwer zu synchronisierenden getrennten VMs
- Beschränkte Rechte auf dem Testserver die nach und nach erweitert werden mussten
- Verschiedene Zertifikatarten sind sehr verwirrend
- Bundle Identifier und App Capabilities mussten nach jedem Git Pull wieder angepasst werden

- Komplexes Testen (Lokale Server, Erreichbarkeit im Labornetzwerk / Wi-Fi, unterschiedliche Datei- und Serverstände)
- Zu Beginn kein Zugriff auf die Git Schnittstellen Projekt
- Ablaufende Zertifikate
- Serverseitig unterschiedliche Übertragungsverfahren zwischen Android und iOS
- Erschwertes Debugging der PHP Scripte
- Die auf den Server vorhanden Versionen von apache2 und cURL, hatten kein HTTP2 unterstützt
- Struktur der Datenbank und der PHP Scripte war durch die Android App vorgegeben
- Veraltete APNs Variante war stark verbreitet, brachte allerdings viele Probleme mit sich und musste als Ansatz letztendlich verworfen werden

## 4.8 Fazit

Unser Team hat feststellen müssen, dass zu einem umfassenden iOS Projekt mehr als nur der Interface Builder und die Sprache Swift gehört. So sind Punkte zu nennen die positiv aus dem Projekt mitgenommen wurden.

- Linuxkenntnisse
- PHP und Debugging auf dem Server
- Beachten von Abhängigkeiten wie der Abwärtskompatibilität der Software zu früheren und bestehenden Android Versionen

## 4.9 Weitere Arbeiten

In diesem Projektabschnitt konnten alle gesteckten Ziele erreicht werden. Da solch ein relativ junges Projekt noch viele Möglichkeiten beinhaltet Funktionen zu verbessern und neue Funktionen einzuführen werden hier mögliche Punkte zur Anregung aufgelistet.

- Die Auswertung des Kommentarfeldes welche, aktuell bei der Android und iOS App auf dem Client Gerät stattfinden, könnten auf dem Server zentral bearbeitet werden. Das würde die Komplexität der beiden Apps verringern und erleichtert die Wartung des Codes an zentraler Stelle.
- Erweiterung der Push Notification Information um eine Priorität, Ablaufdatum,...
- Aufkommende Issues und Ideen aus dem öffentlichen Git Projekt der iOS App und Schnittstelle
- Erweiterung der App und Schnittstelle um andere Sprachen wie z.B. Englisch was in der Android App bereits angeboten wird

## 5 Testen der Anwendung

### 5.1 Test des Widgets

Philipp Dümlein & Maximilian Sonntag & Bastian Kusserow

### 5.2 Testfälle

Das Widget wurde über einen längeren Zeitraum getestet. Dabei wurden folgende Testfälle erarbeitet und überprüft:

- Aktuell keine Vorlesung - beide Vorlesungen in der nächsten Woche
- Aktuell keine Vorlesung - nächste Vorlesung am darauffolgenden Tag
- Aktuell keine Vorlesung - nächste Vorlesung in der nächsten Woche
- Aktuell keine Vorlesung - nächste Vorlesung übermorgen
- Aktuell eine Vorlesung - nächste Vorlesung nächste Woche
- Aktuell eine Vorlesung - nächste Vorlesung am darauffolgenden Tag
- Aktuell eine Vorlesung - nächste Vorlesung danach
- Aktuell eine Vorlesung - nächste Vorlesung übermorgen

## 6 Gruppe PMP

Patrick Niepel & Marcel Hagmann & Carl Philipp Knoblauch

### 6.1 Einleitung

Diese Gruppe besteht aus Patrick Niepel, Marcel Hagmann und Carl Philipp Knoblauch. In diesem Team wurde neben viel Debugging ...

- Aufgaben Erweiterung
- Überarbeitung der Kalenderschnittstelle
- Universal App Colors Funktion
- Onboarding

erarbeitet.

### 6.2 Überblick

Eine Übersicht über Arbeit der Gruppe über das Semester für Fortgeschrittene Programmierung unter Swift 3.

Datum	Aufgaben/Vorlesung	Was wir gemacht haben
KW 40 02.10.17 - 08.10.17	<ul style="list-style-type: none"> <li>• Einleitung der Vorlesung</li> <li>• GitHub Projekt vorgestellt</li> <li>• In Gruppen aufgeteilt</li> <li>• Erster Bug vorgestellt</li> <li>• <b>Neue Aufgabe:</b> Ersten Bug finden</li> <li>• <b>Neue Aufgabe:</b> Themensuche</li> </ul>	<ul style="list-style-type: none"> <li>• Diese Gruppe: Marcel Hagmann, Patrick Niepel, Carl Philipp Knoblauch</li> <li>• Marcel Hagmann findet ersten Bug</li> <li>• Themensuche Ideen: Aufgaben Erweiterung, Widget, Machine Learning, ...</li> <li>• Ausarbeitung der Idee Aufgaben Erweiterung (Vorstellungen, Aufbau, Mockup)</li> </ul>
KW 41 09.10.17 - 15.10.17	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> Aufgaben Erweiterung</li> </ul>	<ul style="list-style-type: none"> <li>• Marcel Hagmann stellt ersten Bug vor und behebt ihn</li> <li>• Vorstellung der Aufgaben Erweiterung</li> <li>• Programmierung der Aufgaben Erweiterung</li> </ul>
KW 42 16.10.17 - 22.10.17	<ul style="list-style-type: none"> <li>• Weiter an der Aufgaben Erweiterung arbeiten</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Push:</b> BugFix von Marcel (16.10.17)</li> <li>• <b>Push:</b> Aufgaben Erweiterung (20.10.17)</li> </ul>
KW 43 23.10.17 - 29.10.17	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> Kalenderschnittstelle</li> </ul>	<ul style="list-style-type: none"> <li>• Weiterer Bug entfernt, daySize (23.10.17)</li> <li>• Komplette Umstrukturierung der Kalenderschnittstelle</li> </ul>
KW 44 30.10.17 - 05.11.17	<ul style="list-style-type: none"> <li>• Weiter an der Kalenderschnittstelle arbeiten</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Push:</b> Kalenderschnittstelle (4.11.17)</li> </ul>
KW 45 06.11.17 - 12.11.17	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> Überarbeitung des Design und der Icons für die Aufgaben Erweiterung</li> </ul>	<ul style="list-style-type: none"> <li>• Bug gefunden: Datumsberechnungsfehler (nahezu Endlosschleife)</li> <li>• <b>Push (Bug):</b> Datumsberechnungsfehler behoben (10.11.17)</li> </ul>
KW 46 13.11.17 - 19.11.17	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> App Color Design</li> </ul>	<ul style="list-style-type: none"> <li>• Einarbeitung in Latex</li> <li>• Überarbeitung des Designs der Aufgaben Erweiterung</li> <li>• <b>Push:</b> des neuen Aufgaben Designs (13.11.17)</li> <li>• App Color Design (Farbe für App in Einstellungen auswählbar)</li> <li>• <b>Push:</b> App Color Design (17.11.17)</li> </ul>

Datum	Aufgaben/Vorlesung	Was wir gemacht haben
KW 47 20.11.17 - 26.11.17	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> Onboarding</li> </ul>	<ul style="list-style-type: none"> <li>• Einarbeitung und Planung des Onboardings</li> <li>• Programmierung des Onboardings</li> <li>• Vorstellung des Onboardings (24.11.17)</li> <li>• Kalenderschnittstellen Debugging (KalenderID wurde nicht persistent gespeichert)</li> </ul>
KW 48 27.11.17 - 03.12.17	<ul style="list-style-type: none"> <li>• Weiter am Onboarding arbeiten</li> </ul>	<ul style="list-style-type: none"> <li>• Programmierung des Onboardings</li> <li>• Debugging des Onboardings</li> </ul>
KW 49 04.12.17 - 10.12.17	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> Aufgaben auch im Kalender anzeigen (Termin)</li> <li>• <b>Bug:</b> Einstellungen Synchronisation: Man kann nichts drücken</li> <li>• <b>Bug:</b> Beim App schließen bricht Kalendersynchronisation ab (wird nur zum teil ausgeführt)</li> </ul>	<ul style="list-style-type: none"> <li>• Programmierung: Aufgaben auch im Kalender anzeigen (In den Notizen)</li> <li>• Debugging</li> <li>• Einarbeitung in Latex</li> </ul>
KW 50 11.12.17 - 17.12.17	<ul style="list-style-type: none"> <li>• <b>Bug:</b> Beim Löschen einer Aufgabe wird diese nicht aus den Notizen im Kalender entfernt</li> </ul>	<ul style="list-style-type: none"> <li>• Einarbeitung in Latex</li> <li>• Dokumentation schreiben</li> </ul>
KW 51 18.12.17 - 24.12.17	<ul style="list-style-type: none"> <li>• <b>Bug:</b> Farben im Onboarding nicht richtig übernommen</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Push:</b> Bei Änderungen an Tasks, werden diese nun auch in den Kalender übernommen (18.12.17)</li> <li>• Onboarding Farben gefixt</li> </ul>
KW 2 08.01.18 - 14.01.18	<ul style="list-style-type: none"> <li>• <b>Neue Aufgabe:</b> Push-Notifications in Onboarding integrieren</li> <li>• <b>Neue Aufgabe:</b> Roter Punkt bei jeder Vorlesung, für die eine Aufgabe offen ist.</li> </ul>	<ul style="list-style-type: none"> <li>• Aufgaben Erweiterung überarbeitet (Rote Punkte)</li> <li>• Onboarding erweitert (Push-Notifications)</li> </ul>
KW 3 15.01.18 - 21.01.18	<ul style="list-style-type: none"> <li>• Testen von verschiedenen Funktionen der App</li> </ul>	<ul style="list-style-type: none"> <li>• Dokumentation weiter schreiben</li> <li>• Debugging</li> </ul>
KW 4 22.01.18 - 28.01.18	<ul style="list-style-type: none"> <li>• Abgabe der Studienarbeit</li> </ul>	<ul style="list-style-type: none"> <li>• Dokumentation weiter schreiben</li> </ul>

## 6.3 Erster Bug

- Wintersemester Sommersemester segmented Control ist verbuggt.
- Änderungen werden gelöscht.
- Alle Vorlesungen mit Kommentar werden in den Kalender geschrieben, aber alle anderen sind im Kalender nicht vorhanden.



## **7 Aufgaben Erweiterung**

Patrick Niepel & Marcel Hagmann & Carl Philipp Knoblauch

### **7.1 Einleitung**

Die erste Aufgabe des Teams war die Erweiterung der Stundenplan App um ein Aufgaben Feature. Mit dem Aufgaben Feature kann der Nutzer seine Aufgaben aus Vorlesungen in die App eintragen, die dann mit dem Kalender synchronisiert werden.

## 7.2 Planung und Mockup

Zuallererst machte sich das Team Gedanken darüber, welche Informationen der Nutzer beim Hinzufügen seiner Aufgaben in der App angeben muss und möchte. Anhand dieser Erkenntnisse, fertigten das Team das Mockup an.

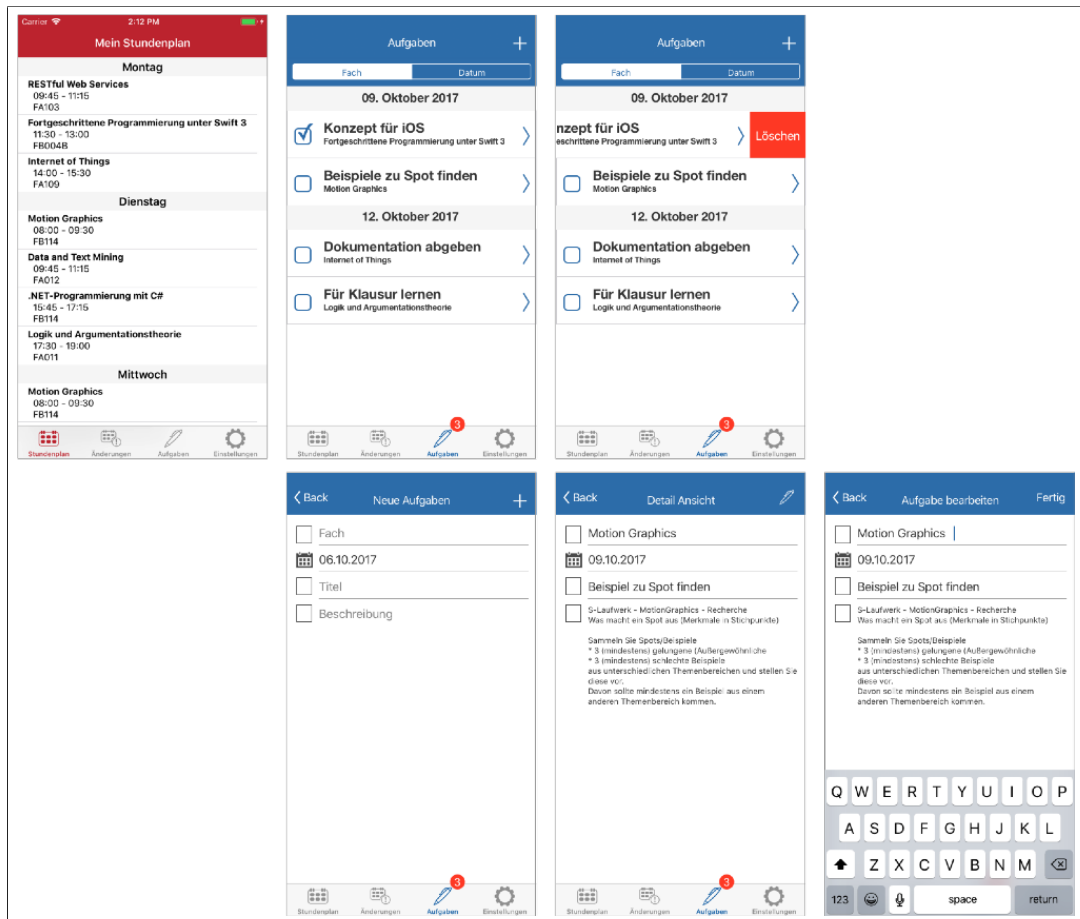


Abbildung 7.1: Mockup unserer Aufgaben Erweiterung

## 7.3 Funktionen

- Sortierung nach Datum und Fach
- Hinweis der noch zu erledigenden Aufgaben mit einem Badge in der Tab-Bar
- Hinzufügen, Löschen und Bearbeiten einer Aufgabe
- Vorlesungen in der Wochenübersicht, denen eine Aufgabe zugeteilt wurde hervorheben
- Aufgabe mit dem Kalender synchronisieren
- Speichern der Aufgaben in UserData

## 8 Kalenderschnittstelle

Patrick Niepel & Marcel Hagmann & Carl Philipp Knoblauch

### 8.1 Einleitung

Da Code öfters über die Zeit an Übersichtlichkeit und Korrektheit verliert, wie es in der Kalenderschnittstelle der Fall war, muss dieser in regelmäßigen Abständen kontrolliert und überarbeitet werden.

### 8.2 Überarbeitung

Eigentlich wollte das Team nur die folgenden Klassen überarbeiten:

- CalendarController
- CalendarIntervace

Allerdings waren die Änderungen des Teams so groß, dass folgende Klassen auch davon betroffen waren:

- DateExtension
- NotificationNameExtension
- SettingsController
- SettingsTableViewController

Als erstes überarbeitete das Team den einfachen Teil, die Übersichtlichkeit des Codes. Da Code von oben nach unten gelesen wird und beim Lesen des alten Codes viel hin und her gesprungen werden musste, ordnete das Team zu allererst die Reihenfolge der einzelnen Methoden an.

Danach überprüfte das Team jede Methode auf Logikfehler und bemerkten dabei, dass das Vorgänger-Team an der ein oder anderen Stelle gepfuscht haben, wodurch sich dann Folgefehler durch das gesamte Programm zogen. Einer dieser Fehler war Beispielsweise der, dass in gewissen Vorlesungsstunden der Vorlesungsbeginn in der Klasse **JsonLectures**, falsch berechnet wird und so anstelle von 2017 das Jahr 0017 ausgegeben wurde, was dazu führte, dass die Semestervorlesungsstunden-Berechnung nicht von 9 Millionen (dass ein paar Sekunden dauert) von 6 Milliarden berechnet wurde und somit dazu führte, dass die App für den Nutzer “einfriert”. Das Team fand diesen Fehler und entfernte diesen und damit auch die dadurch unnötig gewordenen Notifications, die die Vorgänger dazu verwendeten um vorzeitig aus dieser “fast Endlosschleife” auszubrechen.

Des Weiteren sorgte das Team für eine bessere Behandlung der Fehler die während der Ausführung der Operationen auftreten konnten, dass die Kalendersynchronisation im Hintergrund auch nach dem Schließen der App weiter läuft, der Kalender anhand seiner ID gesichert wird und der alte Kalender dementsprechend durch den neuen ausgetauscht wird.

## 9 Onboarding

Patrick Niepel & Marcel Hagmann & Carl Philipp Knoblauch

### 9.1 Einleitung

Eine weitere Aufgabe, die diese Gruppe übernommen hat, war das Onboarding für die Stundenplan App. Im Onboarding kann der Nutzer gleich zum Start der App seine Einstellungen zu Fakultät, Studiengang, Semester, Vorlesung und Kalendersynchronisation vornehmen. Das Onboarding wird gestartet, wenn noch keine Angaben zu den genannten Einstellungen gemacht wurden.

### 9.2 Umsetzung

Zuerst überlegte das Team uns, welche Informationen die App vom Nutzer benötigt, damit der Dienst im vollen Umfang verwendet werden kann. Dabei kam das Team auf folgendes Ergebnis:

1. Fakultät (die die Farbe der App bestimmt)
2. Abfrage der Fakultät
3. Abfrage des Studiengangs
4. Abfrage des Semesters
5. Abfrage der besuchten Vorlesungen
6. Ob eine Synchronisation mit dem Kalender gewünscht ist
7. Ob Push-Benachrichtigungen gewünscht sind

Das gesamte Onboarding befindet sich in einem separaten Storyboard (Onboarding.storyboard), damit die Übersichtlichkeit des aktuellen Projektes weiterhin gewährleistet wird. Jeder Schritt im Onboarding (Ausnahme: Kalendersynchronisation) kann nur fortgesetzt werden, wenn eine Auswahl getroffen worden ist, die für die weiteren Schritte zwingend notwendig ist.

## 10 Universal App Colors

Patrick Niepel & Marcel Hagmann & Carl Philipp Knoblauch

### 10.1 Beschreibung

Bisher hatte die Stundenplan-App für jeden Tab eine andere Farbe. Dieses Team hat sich darum gekümmert, dass der Nutzer in den Einstellungen zwischen den Hochschulfarben auswählen kann und diese dann für die ganze App übernommen werden. Die Hochschulfarben repräsentieren die Fakultät an der Hochschule. Vorher wurden die verschiedenen Farben im Storyboard gesetzt. Das hat zur Folge, dass bereits bei einer kleinen Farbänderung jedes Element einzeln im Storyboard geändert werden muss. Deshalb setzte das Team alle Design Element im Code, die über die von dem Team neu angelegte Klasse AppColor fungiert.

# 11 Gruppe Widget

Philipp Dümlein & Bastian Kusserow & Maximilian Sonntag

## 11.1 Einleitung

Diese Gruppe besteht aus Philipp Dümlein, Maximilian Sonntag und Bastian Kusserow. Die größte Aufgabe unserer Gruppe war das Widget für die Stundenplan App. Hiermit sollte es möglich sein seine derzeitige und kommende Vorlesung zu sehen. Somit soll der Nutzer nicht mehr die App öffnen müssen, um seine derzeit wichtigsten Termine anzuzeigen. Neben der Implementierung des Widgets wurden außerdem..

- Erweiterung der Stundenplan App um das zusätzliche Feld "text" im Stundenplan JSON File
- Einführung einer Sortierung bei der Studiengang Auswahl
- Auslagerung des Models in ein Cocoa Touch Framework
- Parsen und Anzeigen der Termine des aktuellen Semesters

in die Stundenplan App integriert.

## 11.2 Umsetzung des Widgets

Zuerst haben wir uns überlegt, welche die wichtigsten Informationen für den Nutzer sind, um diese dann im Widget anzuzeigen. Hierbei sind wir zu diesen Punkten kommen:

1. UITableView mit zwei Zellen, welche die derzeitige und kommende Vorlesung zeigen. TableView für Skalierbarkeit, falls noch etwas hinzugefügt werden soll
2. Erste Zelle mit Restzeitindikator, der den Fortschritt der derzeitigen Vorlesung anzeigt
3. Beide Zellen

Label für den Zeitpunkt der Vorlesung im Verhältnis zur derzeitigen Zeit

Namen der Vorlesung

Zeitspanne der Vorlesung

Raum der Vorlesung

Das Widget benötigt die Modeldaten aus der StundenplanApp, jedoch haben wir bemerkt, dass diese nicht direkt erreichbar sind. Deshalb wurde ein Framework erstellt, welches die nötigen Modelklassen beinhaltet, die für die Darstellung des Widgets nötig sind.

Beim implementieren des Widgets haben wir festgestellt, dass es viele verschiedene Möglichkeiten der Darstellung gibt. Daher sind diese hier zum Verständnis hier nochmals aufgeführt:

1. Erste Zelle zeigt die derzeitige Vorlesung an. Hierbei wird der Restzeitindikator angezeigt. Die Zweite Zelle zeigt die nächste Vorlesung am selben Tag. Das Label der ersten Zelle zeigt "Jetzt", das der zweiten zeigt "Nächste".
2. Erste Zelle zeigt die derzeitige Vorlesung an. Hierbei wird der Restzeitindikator angezeigt. Die Zweite Zelle zeigt die nächste Vorlesung am nächsten Tag. Das Label der ersten Zelle zeigt "Jetzt", das der zweiten zeigt "Morgen".
3. Erste Zelle zeigt die nächste Vorlesung an. Hierbei wird der Restzeitindikator nicht angezeigt. Die Zweite Zelle zeigt die darauf folgende Vorlesung am selben Tag. Das Label der ersten Zelle zeigt "Nächste", das der zweiten zeigt "Übernächste".
4. Erste Zelle zeigt die nächste Vorlesung am nächsten Tag an. Die Zweite Zelle zeigt die Vorlesung danach an. Das Label der ersten Zelle zeigt "Morgen", das der zweiten zeigt ebenso "Morgen".
5. Die nächsten Vorlesungen sind mehrere Tage entfernt (beispielsweise in den Ferien). Die Labels zeigen das Datum der nächsten Vorlesung an.

## 11.3 Experimentell

Da mittlerweile auch die für das aktuelle Semester relevanten Termine in die App integriert wurden, werden diese auch im Widget beachtet. Sollte an einer im Widget angezeigten Vorlesung ein vorlesungsfreier Tag sein, so wird dieser Termin in der Zelle angezeigt. Falls der Nutzer dies nicht möchte, kann das Anzeigen der Termine im Widget unter den Einstellungen der Stundenplan App deaktiviert werden.



## 12 Erweiterung um das Textfeld im JSON

Philipp Dümlein & Bastian Kusserow & Maximilian Sonntag

### 12.1 Einleitung

Die erste Aufgabe unseres Teams war das Hinzufügen der JSON Property "text". Diese musste danach noch in dem Tab "Änderungen" angezeigt werden.

### 12.2 Umsetzung

Für das Einfügen der Property wurde die Model Klasse "ChangedLecture" um das Feld text erweitert. Danach wurde in der Klasse "JsonChanges" die Property "text" extrahiert. Um diese dann in der Zelle anzuzeigen, musste die Zelle für die Änderungen angepasst werden. Außerdem sollte das Textfeld nur angezeigt werden, wenn in der text Property auch tatsächlich etwas steht. Dabei sind wir auf das Problem gestoßen, dass im Storyboard bei der Zelle einige Constraints falsch gesetzt wurden. Deshalb mussten die Constraints der kompletten Zelle noch einmal neu gesetzt werden. Nun kann das Height-Constraint für das Text Label durch Änderung des Constant Werts geändert werden.

## 13 Studiengang Sortierung

Philipp Dümlein & Bastian Kusserow & Maximilian Sonntag

### 13.1 Einleitung

Da wir anfangs Probleme mit dem Zugriff auf die Model Daten vom Widget aus hatten, war unsere nächste Aufgabe die Sortierung der Studiengänge nach dem jeweiligen Anfangsbuchstaben bei der Studiengangsauswahl.

### 13.2 Umsetzung

Dafür wurden der CourseTableViewDataSource und der CourseTableViewDelegate angepasst. Im DataSource wurde eine Methode implementiert, die die Studiengänge bei der Initialisierung sortiert. Außerdem wurde die didSelectRowAt Methode des Delegates auf die neu dazu gekommenen Sections angepasst.

## 14 Framework

Philipp Dümlein & Bastian Kusserow & Maximilian Sonntag

### 14.1 Einleitung

Um alle Daten die in der Stundenplan App angezeigt werden auch im Widget angezeigt werden können, musste, wie bereits erwähnt das Model auf dem die App basiert in ein Framework ausgelagert werden.

### 14.2 Umsetzung

Zu aller erst wurde das Framework erzeugt und die Dateien des Models in das Framework verschoben. Danach wurde die UserDataObjectPersistency Klasse angepasst, da der Document Path des Widgets und der der Stundenplan App verschieden waren und somit das Widget noch nicht auf die Daten der Stundenplan App zugreifen konnte. Deshalb wurde für das Projekt eine eigene AppGroup hinzugefügt. So konnte das Widget mit einigen Anpassungen der DataObjectPersistency auf diesen Container zugreifen und so die Daten der Stundenplan App abrufen. Da wir vermeiden möchten dass die App beim Update über den AppStore ihre Daten verliert, wurden außerdem noch einige Methoden implementiert, die die alten Daten der App an den neuen Speicherort verschiebt.

### 14.3 Wichtig

Wenn im Model neue Felder oder Methoden hinzugefügt werden sollen, muss immer der Zugriffsmodifizierer public hinzugefügt werden, da der standardmäßige Modifizierer internal ist und so weder App noch Widget auf die Methoden oder Felder zugreifen können.

## 15 Website Termine

Philipp Dümlein & Bastian Kusserow & Maximilian Sonntag

### 15.1 Einleitung

Eine kleinere Aufgabe unserer Gruppe war das Parsen der Termine aus der Hochschulwebsite. Hiermit sollten diese Termine in die Anzeige des Stundenplans und des Widgets integriert werden.

### 15.2 Umsetzung

Für das Parsen der Website haben wir den CocoaPod "SwiftSoup" verwendet. Hiermit war es möglich die div-Elemente der Tabellen auszulesen. Ebenso konnten die Tabellenzeilen einzeln ausgelesen werden. Auf diesen Zeilen basierend wurden die Daten ausgelesen. Hierbei wird sich darauf verlassen, dass die Darstellung der Daten dem Format "dd.MM.YYYY" entsprechen. Ebenso gibt es nicht nur Daten sondern auch Datenspannen, d.h. es gibt ein Start und ein Enddatum. Diese müssen dem Format "dd.MM.YYYY - dd.MM.YYYY" entsprechen. Die fertig geparsen Daten werden in das Model der Stundenplan-App gespeichert und können in einem eigenen Screen in den Settings angesehen werden. Das Herunterladen und speichern der Daten erfolgt beim ersten App Start. Sollte sich das Semester ändern werden die Termine erneut für das dann aktuelle Semester heruntergeladen.

### 15.3 Wichtig

Die im Projekt enthaltenen PodFiles müssen beim Checkout / Pull von GitHub zunächst neu installiert werden, damit das Projekt problemfrei funktioniert. Hierzu muss auf dem Gerät CocoaPods installiert sein. Vor dem Öffnen des Projekts muss dann im StundenplanFramework-Ordner "pod install" ausgeführt werden. Nun kann der Workspace geöffnet werden, welcher nun StundenplanNavigation, StundenplanFramework (als Unterprojekt) und Pods enthält. Sollte neben den Pods noch einmal StundenplanFramework erstellt worden sein, kann dieses aus dem Projekt gelöscht werden ("remove by reference!"). Nun sollte das Projekt fehlerfrei einsetzbar sein.

## Abbildungsverzeichnis

2.1	Neues Stundenplanbedienkonzept . . . . .	7
2.2	Design Konzept 1, Stundenplan Ansicht . . . . .	8
2.3	Design Konzept 2, Stundenplan Ansicht . . . . .	8
2.4	Design Konzept 3, Stundenplan Ansicht . . . . .	9
2.5	Design Konzept 4, Stundenplan Ansicht . . . . .	9
2.6	Finales Design, Stundenplan Ansicht . . . . .	10
2.7	Finales Design, bsp. Auswahl Studiengänge . . . . .	11
2.8	Neues Stundenplanbedienkonzept . . . . .	12
2.9	Neues Stundenplanbedienkonzept . . . . .	12
2.10	Optimierung der Schritte zur Stundenplanauswahl auf die nötigsten Schritte . .	13
2.11	Design Konzept, Stundenplan Einstellungen . . . . .	14
2.12	Finales Design, Stundenplan Einstellungen . . . . .	15
2.13	Finales Design, bsp. Auswahl Studiengänge . . . . .	16
2.14	Design Vorschlag, Widget . . . . .	17
2.15	Mein schönstes Mockup . . . . .	18
3.1	hype cycle for technologies 2017 . . . . .	19
3.2	Siri Flowchart . . . . .	21
4.1	Tabelle fcm_nutzer in phpmyadmin . . . . .	27
4.2	Anpassungen in der Info.plist . . . . .	29
4.3	Wahl des Developer Teams . . . . .	29
4.4	Angepasste Capabilities . . . . .	30
7.1	Mockup unserer Aufgaben Erweiterung . . . . .	38