

Praktikum iOS-Entwicklung

Wintersemester 2017 / 2018

Prof. Dr. Linnhoff-Popien

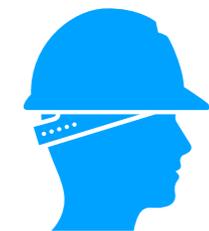
Kyrill Schmid, Markus Friedrich



KW	Tutorium	Projekt
42	18.10.17- Einführung & Organisation	Gruppeneinteilung
43	25.10.17- Einführung in Swift	1. Tutor-Treffen
44	01.11.17- Feiertag	Feiertag
45	8.11.17- Xcode: Erstes Projekt	2. Tutor-Treffen
46	15.11.17- Model-View-Controller	Definition Feature Listen
47	22.11.17- View-Controller	Definition Feature Listen
48	29.11.17 - Core-Data	3. Tutor-Treffen
49	06.12.17 - tba	4. Tutor-Treffen
50	13.12.17 - tba	5. Tutor-Treffen
51	20.12.17 - Präsentation Zwischenstände	6. Tutor-Treffen
52	27.12.17 - Ferien	-
1	03.01.18 - Ferien	-
2	-	7. Tutor-Treffen
3	-	8. Tutor-Treffen
4	Prüfungen	-
5	Prüfungen	-
6	07.02.18 - Abschlussveranstaltung	-



Ideenfindung



Implementierung

■ Pflichttermine (Tutor-Treffen sind immer verpflichtend)

Heute

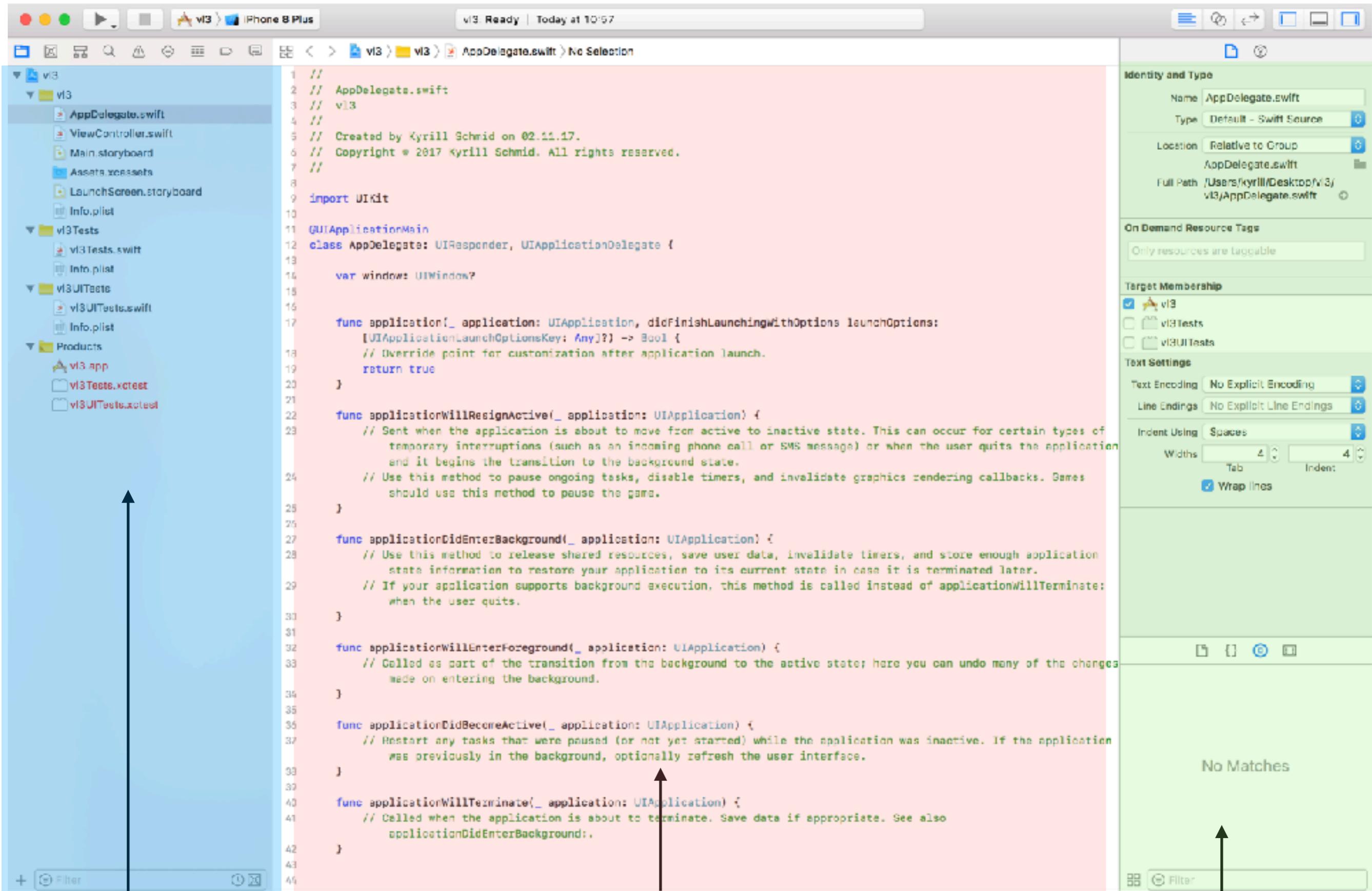
- ▶ Xcode
- ▶ Erstes Projekt
- ▶ Der AppDelegate
 - ▶ Das Delegation Pattern
 - ▶ Zustands Transitionen einer App
- ▶ ViewController
 - ▶ TableViewController

XCode

- Xcode beinhaltet im Grunde schon alles was man zum Entwickeln von iOS-Apps benötigt:
 - Software Development Kit (SDK)
 - Code Editor
 - UI Editor
 - Debugging Tools
 - Simulator
 - Version Control

Neues Projekt anlegen



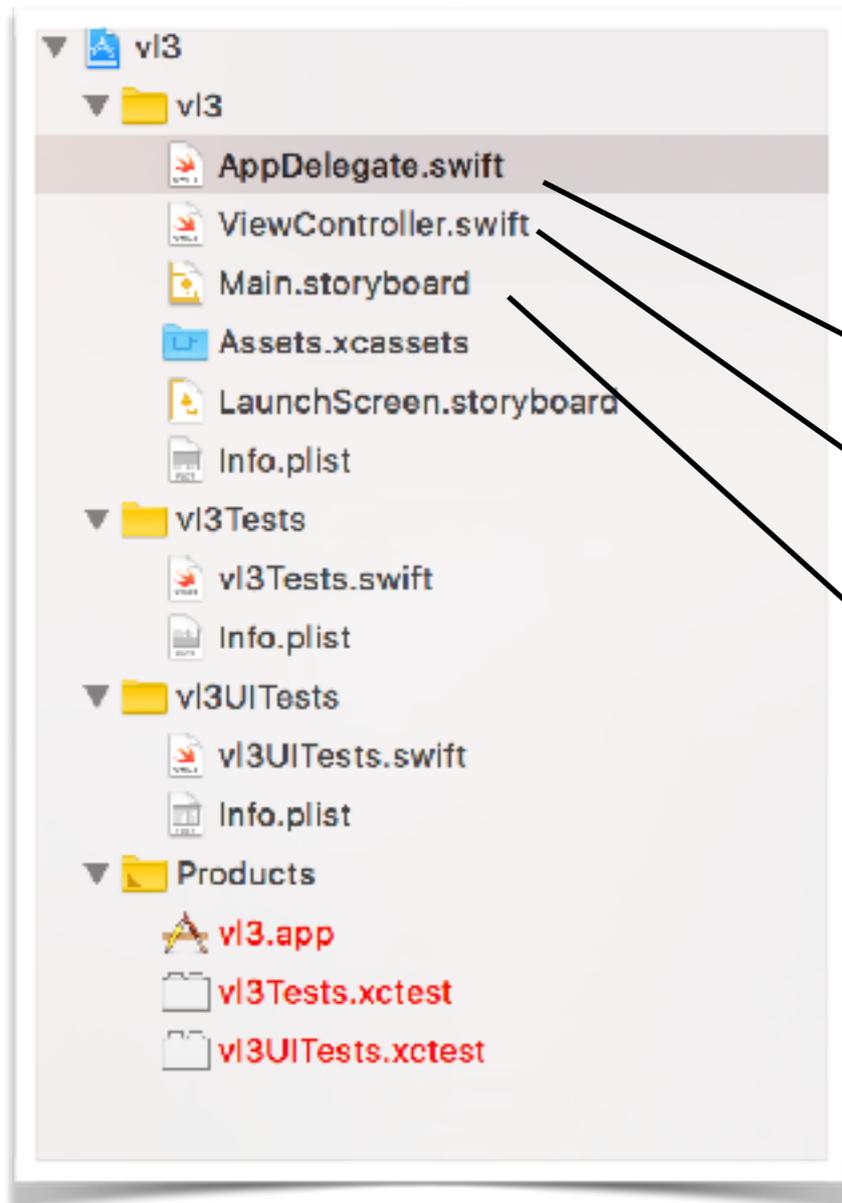


Project-Navigator

Project-Editor

Utilities

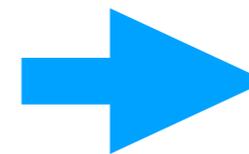
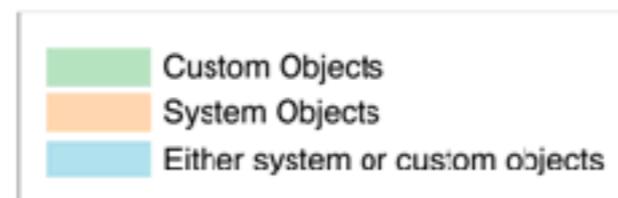
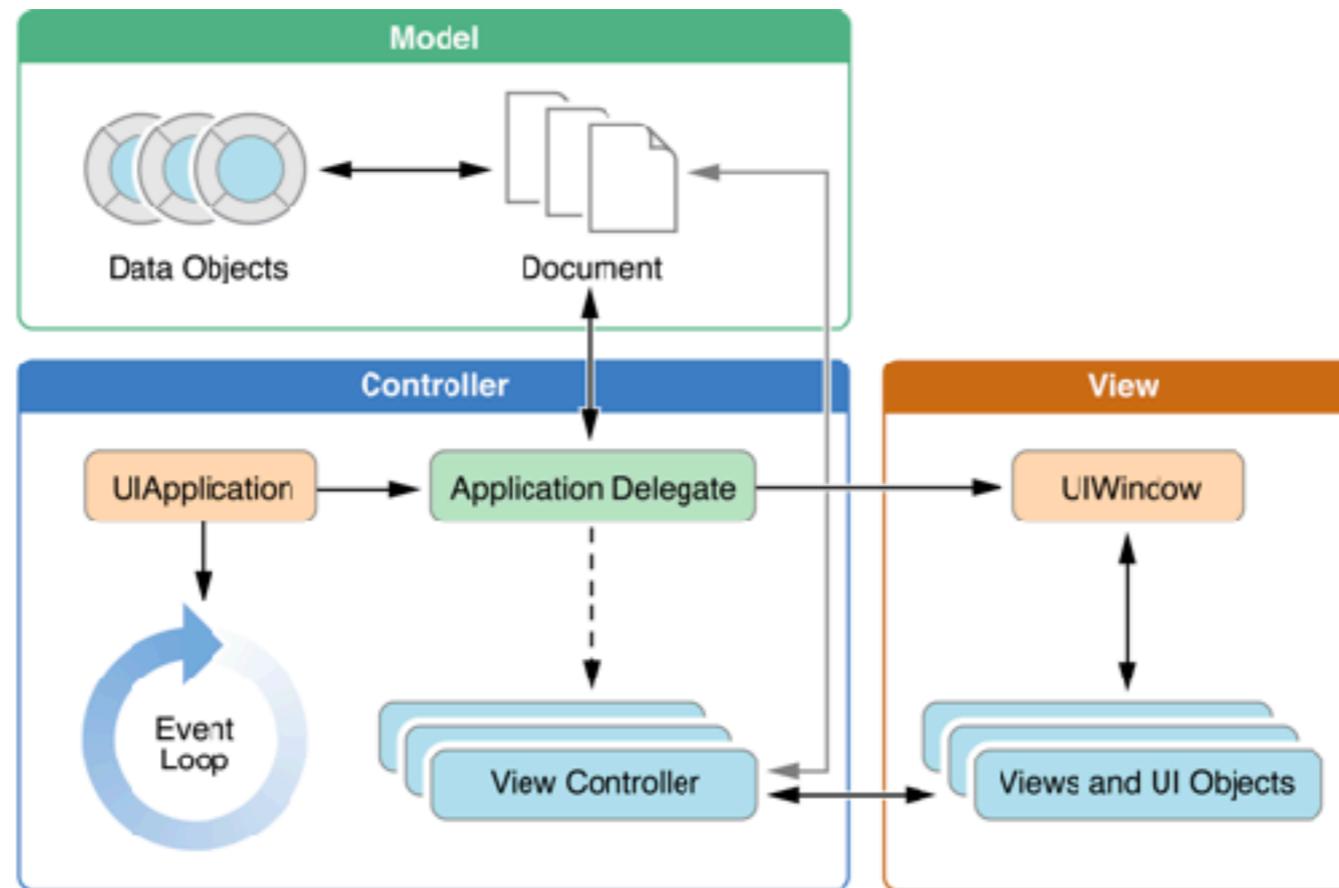
Neues Projekt: Single View Application



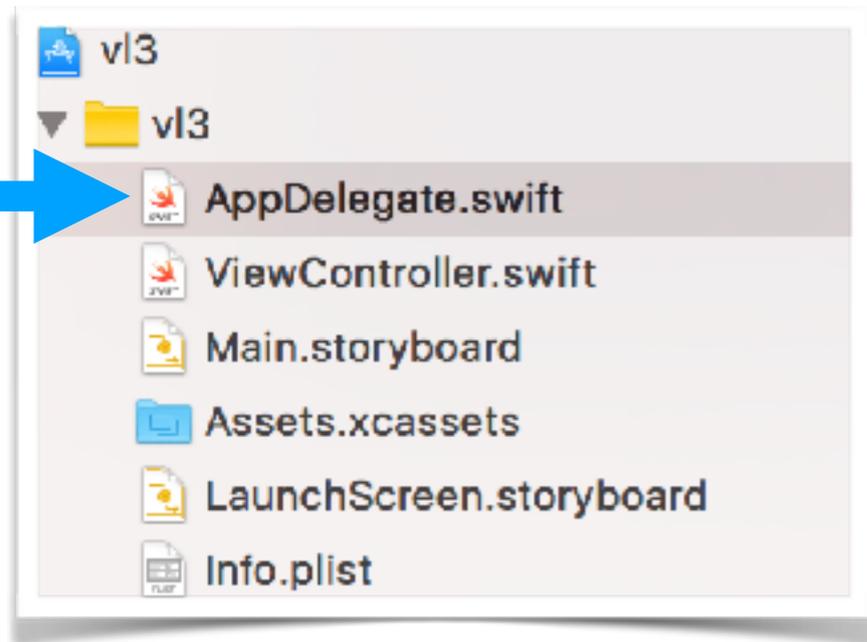
- XCode generiert beim anlegen eines neuen Projekts initiale Files:

- AppDelegate.swift
- ViewController.swift
- Main.storyboard
- ...

Bestandteile einer iOS-App



**Mehr zu
Model-View-Controller
nächste Woche**

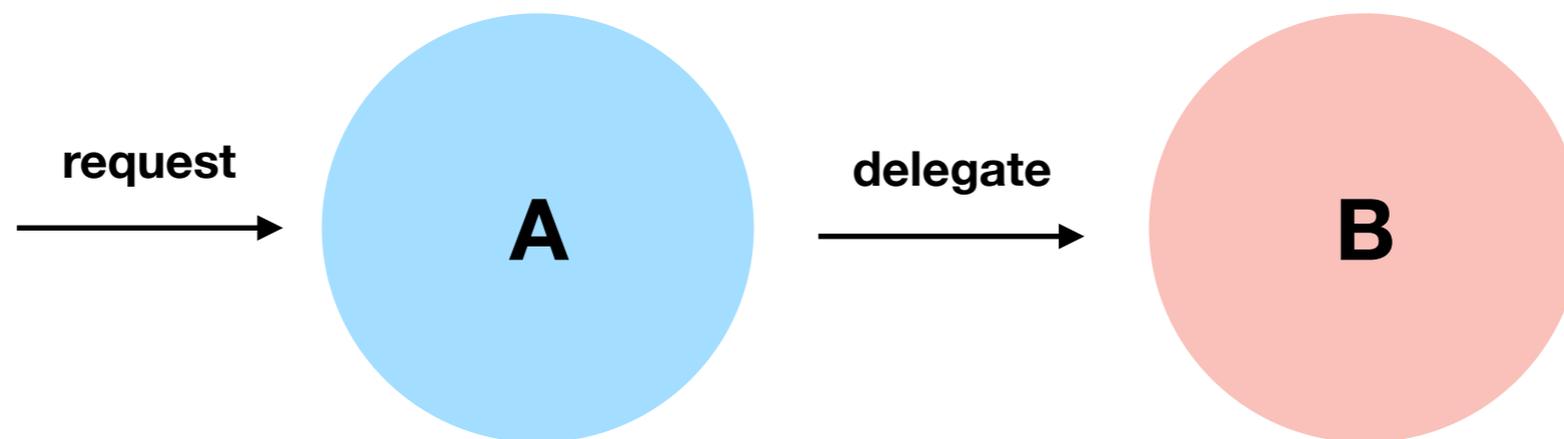


Der AppDelegate

- Der AppDelegate ist ein **Singleton** und das erste customisierbare Objekt einer iOS App
- AppDelegate ist „Ansprechpartner“ des Betriebssystems für einige wichtige Events
- AppDelegate ist der **Delegate** der Application
- Das **Delegation Pattern** ist zentral in iOS und findet an vielen Stellen Verwendung

Einschub: Das Delegation Pattern

- Delegation ist ein Pattern der Objekt-Orientierten Programmierung um Objekt-Komposition zu unterstützen
- Bei Delegation behandelt ein Objekt A eine Anfrage indem es an ein weiteres Objekt B delegiert



- In Swift wird Delegation durch die Verwendung von Protokollen unterstützt

Protokolle in Swift

- Protokolle sind ähnlich zu Interfaces in z.B. Java
- Klasse (oder Struktur) kann Protokoll konform sein d.h. die Anforderungen des Protokolls umsetzen

```
protocol SomeProtocol {  
    // protocol definition goes here  
    func requiredFunction()  
}
```

```
class SomeClass: SomeProtocol {  
    func requiredFunction() {  
        print("Have to do that")  
    }  
}
```

Protokolle & Delegation

```
protocol Assistant {  
    func writeEmail()  
}
```

Protokolle & Delegation

```
protocol Assistant {  
    func writeEmail()  
}  
  
class WorkerA : Assistant {  
    func writeEmail() {  
        print("WorkerA: I wrote an Email!")  
    }  
}
```

Protokolle & Delegation

```
protocol Assistant {
    func writeEmail()
}

class WorkerA : Assistant {
    func writeEmail() {
        print("WorkerA: I wrote an Email!")
    }
}

class Boss {
    var delegate : Assistant?

    func work(){
        print("Boss: I have no time to write Emails!")
        delegate?.writeEmail()
    }
}
```

Protokolle & Delegation

```
let boss = Boss()  
let worker = WorkerA()  
boss.delegate = worker  
boss.work()
```

```
// Boss: I have no time to write Emails!  
// WorkerA: I wrote an Email!
```

AppDelegate als Delegate der Application

- AppDelegate ist der **Delegate** der Application
- Das bedeutet konkret, dass er **UIApplicationDelegate** konform ist
- Eine wichtige Verantwortlichkeit, die der AppDelegate übernimmt ist das Reagieren auf Zustands-Transitionen der App

App Zustands-Transition

- Wenn die App in den Zustand ‚Background‘ übergeht müssen u.U. noch
 - Daten gesichert werden
 - Speicher frei gegeben werden (reduce memory footprint)
- Die letzte Chance dazu ergibt sich in der **`applicationDidEnterBackground(_:)`** Methode des AppDelegate

Home Button Press



App Zustands-Transition

- Weitere Methoden des AppDelegate:

[applicationWillTerminate\(_:\)](#)

[applicationDidBecomeActive\(_:\)](#)

[applicationWillEnterForeground\(_:\)](#)

[application\(_:willFinishLaunchingWithOptions:\)](#)

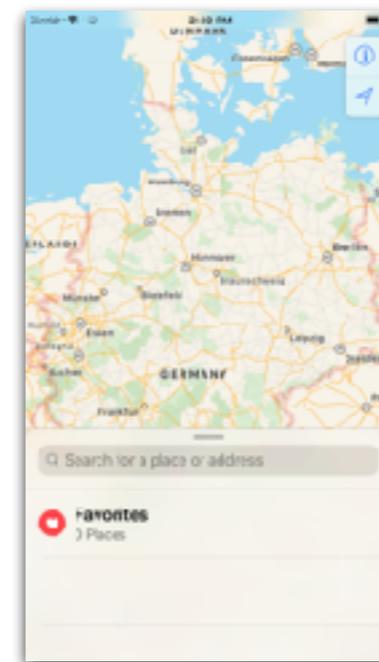
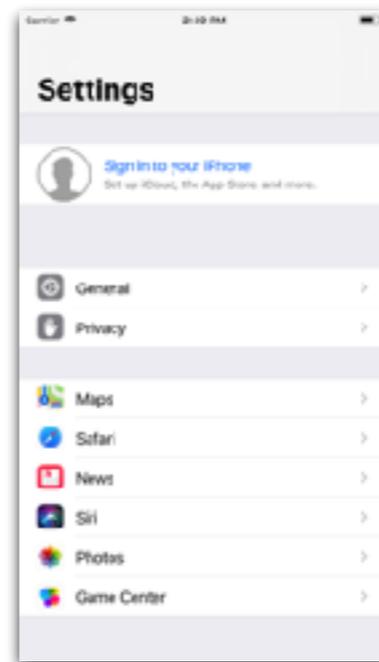
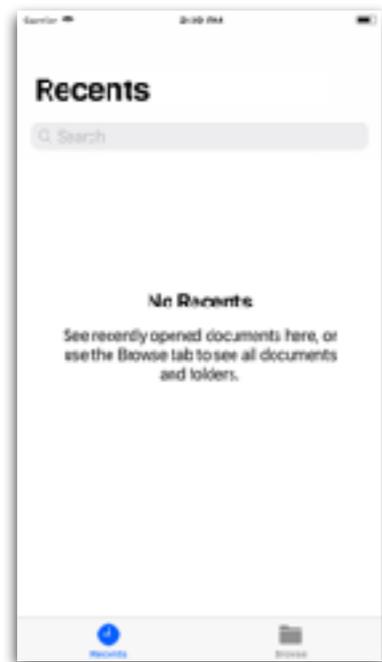
[applicationWillResignActive\(_:\)](#)

[application\(_:didFinishLaunchingWithOptions:\)](#)

User Interface

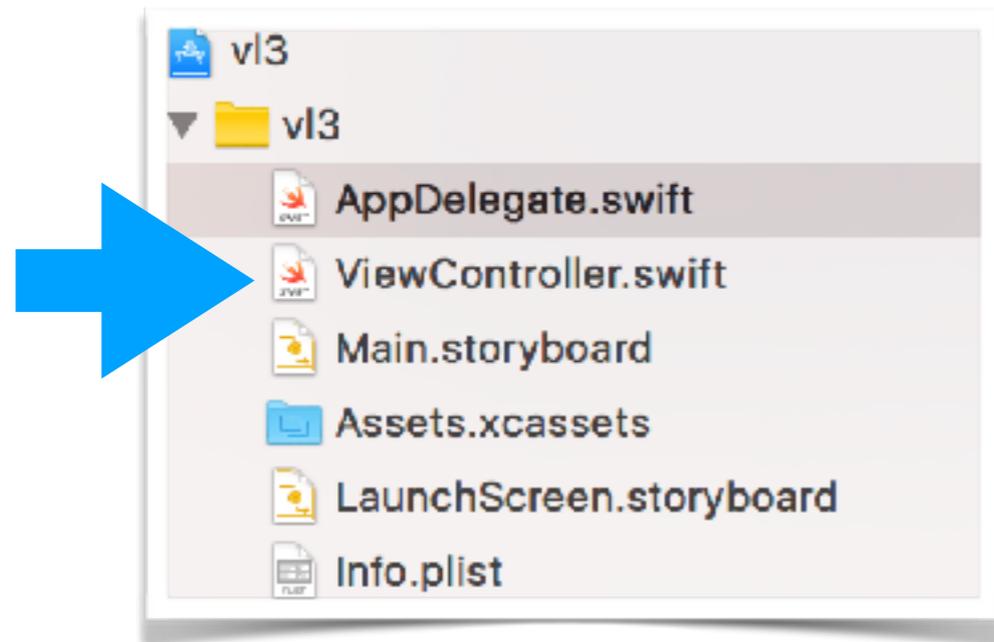
Views

- Eine App besteht aus einer Menge von Views durch die der Nutzer navigieren kann



- Views zeigen aber lediglich Daten an
 - Daten kommen aus Modell und werden von ViewController übermittelt
 - User Interaktion wird von ViewController interpretiert und an Model übermittelt

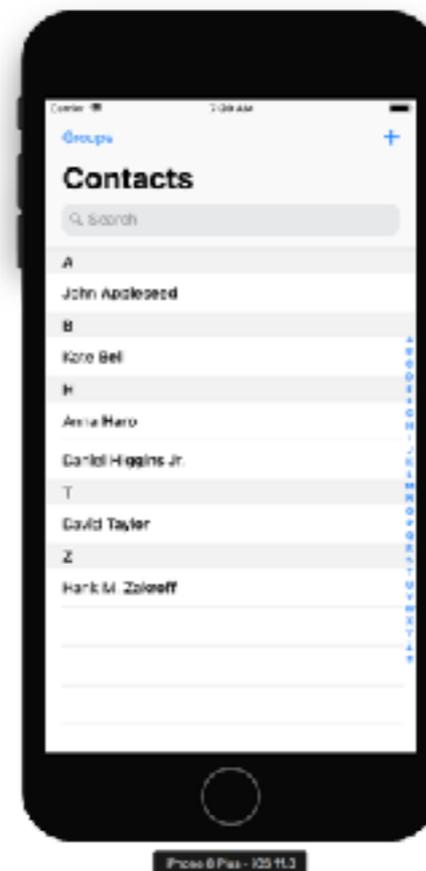
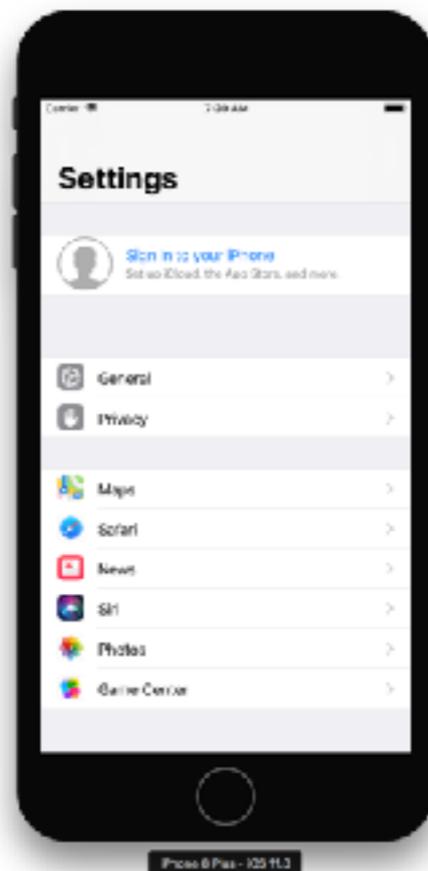
ViewController



- Jede **View** hat einen zugehörigen **Controller** -> ViewController
- ViewController sind zentrale Objekte für die iOS-Programmierung und übernehmen viele wichtige Aufgaben
- Geben eine bewährte Struktur vor für die Gestaltung eines Interface
- Viele verschiedene ViewController Typen, die sofort verwendet werden können (NavigationView, TableView, collectionView, etc.)

Beispiel: TableView

- Oft bietet sich eine Tabellen-Ärtige Darstellung von Daten an z.B. für Kontakte, Einstellungen, ...
- Dafür gibt es einen vordefinierten Controller: `UITableViewController`

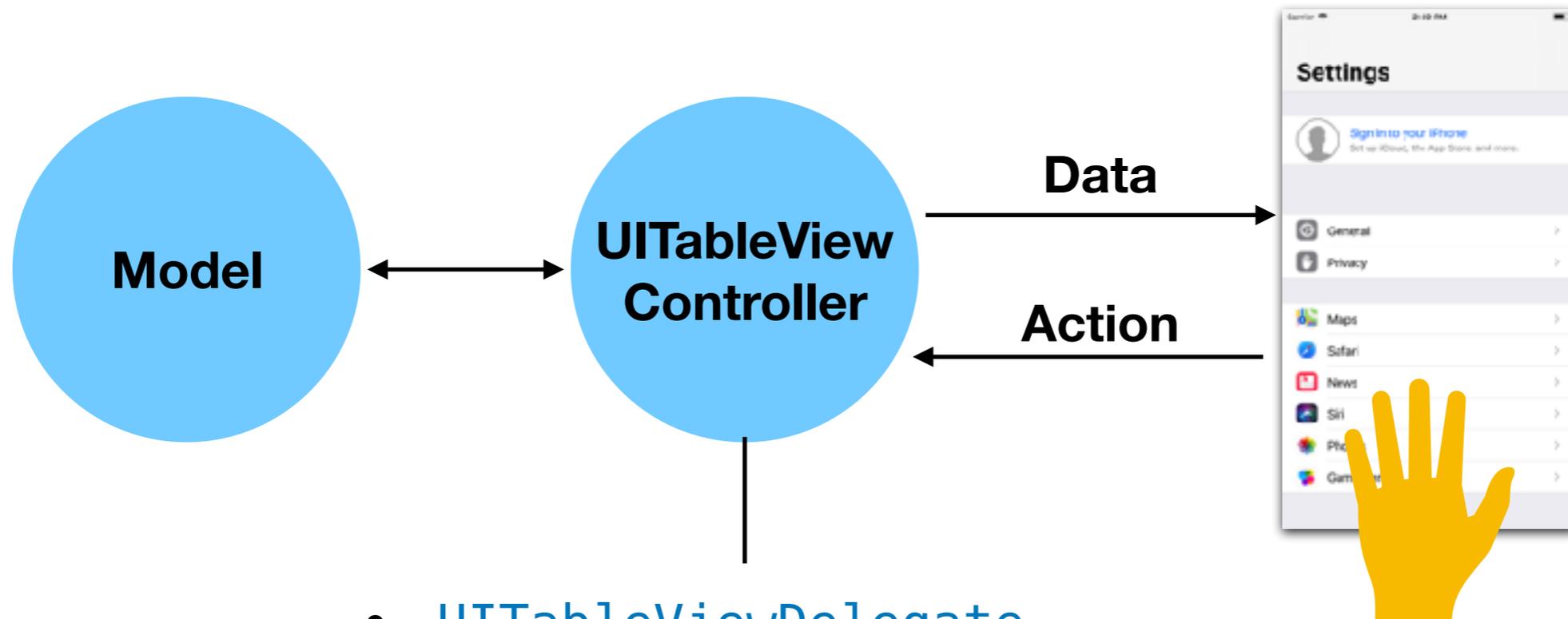


Beispiel: TableView

- Eine TableView:
 - Zeigt Daten aus einer Datenquelle (DataSource)
 - Muss auf User-Interaktion reagieren können (Scroll, Press, ...)
- Der `UITableViewController` bietet dazu eine Menge vordefinierter Funktionalität
- Dabei kommt wieder das Delegation Pattern zum Einsatz, denn der `UITableViewController` ist `UITableViewDelegate` und `UITableViewDataSource` konform

TableViewController

- Der UITableViewController muss also Daten liefern und auf User-Interaktion reagieren können



- UITableViewControllerDelegate,
UITableViewDataSource

TableViewController Beispiel

- Beispiel: Daten liegen in Form eines Arrays vor (Namen)
- Wir wollen Namen aus der Liste löschen
- Liste Umsortieren

