

Übungsblatt 3

Abgabe

Geben Sie Ihre Lösung zu Aufgabe 2 und 3 bis zum 16.5.2013 12:00 Uhr in Uniworx ab. Sie können das Übungsblatt entweder alleine oder zu zweit bearbeiten.

Aufgabe 1

Bilden Sie 4er-Teams für die Projektphase und tragen Sie sich in folgende Tabelle ein: <http://goo.gl/ls6Ha>

Aufgabe 2

Fassen Sie Ihre wichtigsten Erkenntnisse zu den Aufgaben 2a) und 2b) zusammen. Geben Sie hierfür eine PDF ab (1-2 Seiten).

- a) Setzen Sie sich mit dem XCode Debugger auseinander. Setzen Sie Breakpoints, versuchen Sie die Reihenfolge der involvierten Methoden zu verstehen und lesen Sie die Werte von Variablen zur Laufzeit aus. Es bietet sich an, diese Aufgabe parallel zur Aufgabe 3 zu erledigen.
- b) Setzen Sie sich mit einem (Bachelor) bzw. zwei (Master) der folgenden Themen auseinander. (Wenn sie das Übungsblatt zu zweit machen, müssen Sie *nicht* doppelt so viele Themen bearbeiten.)
 - App Life Cycle: [Grafik](#), [Dokumentation](#)
 - [Objective-C Coding Conventions](#)
 - [Delegates](#)
 - Submit your app to the App Store: [Dokumentation](#), Tutorial [Part 1](#), [Part 2](#)
 - [Build in-house apps](#)
 - [Build business-to-business apps](#)
 - [Basic data persistence](#) (Persistente Speicherung von wenigen/simplen Daten)
 - [CoreData](#) (Persistente Speicherung von Daten auf die professionelle Art)
 - [iOS User Experience Guidelines](#)
 - [Transitioning between mobile and desktop](#)
 - [iOS Technology Usage Guidelines](#)
 - [Icon and Image Creation Guidelines](#)
 - [App monetization, marketing, advertising, sales](#)
 - [Optimizing for high resolution \(Retina displays\)](#)
 - [Passbook](#)
 - [EventKit](#) (Calendar, Reminders integration)
 - [Accessibility](#)

Aufgabe 3

Bearbeiten Sie zwei der folgenden vier Teilaufgaben. (Wenn sie das Übungsblatt zu zweit machen, müssen Sie *nicht* doppelt so viele Teilaufgaben bearbeiten.)

a) Animationen und User Input

Erstellen Sie eine iOS Anwendung mit den folgenden Eigenschaften.

- Die App erkennt 3 unterschiedliche Arten von User Input Ihrer Wahl (z.B. Gestures, Accelerometer, Text Input, UI Controls). Verwenden Sie keine Elemente, die in den bisherigen Übungsblättern gefordert waren.
- Basierend auf den 3 Input-Arten reagiert die App mit 3 unterschiedlichen Animationen. Verwenden Sie dabei auch options, delay und completion. Für die Animationen können Sie beliebige Bildschirminhalte verwenden, z.B. PNGs aus dem Internet. (Denken Sie daran, dass die PNGs bei der Abgabe auch im Projekt sein müssen.)
- (Optional) Geben Sie außerdem Sound aus.
- Die Struktur der Anwendung (z.B. Single View, Tabbed) bleibt Ihnen überlassen.

b) Persistente Datenspeicherung (basierend auf Blatt 2)

Im 2. Übungsblatt haben Sie eine Navigation-based Anwendung erstellt. In der Detail View haben Sie dem Nutzer die Möglichkeit gegeben, das Objekt zu bewerten (z.B. mit einem Slider). Speichern Sie die Bewertung des Nutzers nun persistent, so dass die Bewertung bestehen bleibt, wenn man die Detail View verlässt und später erneut anzeigt. Recherchieren Sie hierfür z.B. folgende Komponenten (siehe auch [Tutorial](#) und [CoreData](#)):

- NSUserDefaults
- plist
- CoreData

c) Collection View

Bauen Sie eine ähnliche App wie im 2. Übungsblatt. Verwenden Sie allerdings dieses Mal keine Table Views, sondern Collection Views. Die Programmierung von Collection Views funktioniert fast gleich wie bei Table Views. Die Darstellung der Daten ist jedoch viel flexibler und lässt mehr Spielraum für Kreativität. Siehe [Introducing Collection Views \(WWDC 2012 Video\)](#), [Dokumentation](#), [Tutorial](#).

d) Tab Bar, Web View und Map View

Erstellen Sie eine iPhone App mit Tab Bar, Web View und Map View anhand der folgenden Schritte.

- Erstellen Sie ein neues XCode-Projekt (Tabbed Application, iPhone, Storyboards, ARC).
- Schauen Sie sich das Storyboard an und machen Sie sich mit dessen Elementen vertraut. Verschaffen Sie sich in der Dokumentation einen Überblick über die Klassen UITabBarController, UITabBar und UITabBarItem.

- Machen Sie die AppDelegate zu einem UITabBarControllerDelegate und fügen Sie in der didFinishLaunching...-Methode folgendes ein:

```
UITabBarController *tbc = (UITabBarController*) self.window.rootViewController;
tbc.delegate = self;
```

- Implementieren Sie in der AppDelegate außerdem die Methode didSelectViewController so, dass als Debug-Ausgabe (NSLog) der Index des aktuell ausgewählten View Controllers angezeigt wird.
- Ändern Sie im Storyboard die Beschriftung der Tab Bar Items von “First” zu “Web” und von “Second” zu “Maps”. Löschen Sie bis auf das Tab Bar Item alle Elemente der beiden View Controller. (Optional: Erstellen Sie eigene PNGs für die Tab Bar Items.)
- Ziehen Sie eine Web View auf den First View Controller. Legen Sie anschließend ein IBOutlet für diese Web View an. Fügen Sie der Web View durch folgende Zeilen statischen Inhalt hinzu und lassen Sie die Anwendung laufen:

```
NSString *html = @"Bacon ipsum dolor sit amet leberkas laboris chicken.";
[self.myWebView loadHTMLString:html baseURL:nil];
```

- Zeigen Sie nun eine wirkliche Webseite an. Kommentieren Sie dazu zunächst die beiden Zeilen wieder aus. Erzeugen Sie dann ein NSURL-Objekt für eine Webseite Ihrer Wahl, sowie ein NSURLRequest-Objekt. Finden Sie eine geeignete Methode der Klasse UIWebView, um die Seite anzuzeigen.
- (Optional) Verkleinern Sie die Web View und fügen Sie dem First View Controller zwei Buttons hinzu, um auf der Webseite vorwärts und zurück gehen zu können.
- Fügen Sie im Storyboard dem Second View Controller eine Map View hinzu und aktivieren Sie im Attributes Inspector der Map View “Shows User Location”. Legen Sie ein IBOutlet an und importieren Sie im Second View Controller <MapKit/MKMapView.h>. Fügen Sie Ihrem XCode-Projekt außerdem das MapKit hinzu (Projektübersicht > Build Phases > Link binary with libraries > + > MapKit.framework).
- Lassen Sie die Anwendung laufen. Im iOS Simulator kann die Location unter Debug > Location > Custom Location gesetzt werden.
- Zeigen Sie jetzt nicht die aktuelle User Location, sondern setzen Sie die Location im Code auf den Marienplatz oder einen anderen Ort Ihrer Wahl.
- (Optional) Erstellen Sie einen dritten View Controller und verknüpfen ihn mit dem TabBarController. Erzeugen Sie zwei Textfelder, in die man Koordinaten eintragen kann. Die Map View zeigt dann eine Location basierend auf diesen Koordinaten an.