Praktikum 1 – Grundlagen und erste Schritte mit iOS

Präambel

Analog zu Android, werden wir uns in diesem Praktikum erst einmal mit der generellen Verwendung der Werkzeuge zur Entwicklung von iOS-Anwendungen und der Sprache Swift vertraut machen.

Zur iOS-Entwicklung ist ein Mac-Computer erforderlich. Wir nutzen dazu die Rechner im Raum SI 0038. Hier ist die Entwicklungsumgebung **XCode** installiert. Sie sollten sich auf den Rechnern mit Ihrem Standard-Account anmelden können, wobei die initiale Anmeldung etwas länger dauern kann.

Nach Möglichkeit sollten Sie immer am gleichen Rechner arbeiten, da es sich um lokale Accounts handelt. Bitte sorgen Sie selbst für die Sicherung Ihrer Projekte (netcase, …).

MacOS

[Als erfahrener Mac-Nutzer können Sie diesen Punkt überspringen.]

Zwar kann an dieser Stelle keine komplette Einführung in macOS (Apple's Betriebssystem für Laptops und Desktops) gegeben werden, aber eine kleine Orientierung, die speziell auf Abweichungen vom Windows-Desktop eingeht:

- *Menüleiste*: die sehr kompakte **Menüleiste** verändert sich in Abhängigkeit der App im Vordergrund. Das ‚Apfel-Menü' ist immer konstant lokalisiert. Hier befindet sich der sog. ‚***Finder***', mit dem auf das Dateisystem zugegriffen werden kann.

- *Kontrollzentrum*: mit dem Icon neben der Systemuhr lassen sich wichtige Einstellungen wie Netzwerkzugang, die Bildschirmhelligkeit, die Lautstärke und weitere Systemfunktionen anpassen.

- *Dock*: über den sog. **Dock** ist ein Schnellzugriff auf Apps möglich. In gewisser Weise gibt es hier ein Analogie zum Startmenü unter Windows. Geöffnete Apps werden im Dock angeordnet und können komfortabel zugegriffen werden. Wenn es zu unübersichtlich wird, können die Apps auch entfernt werden. Kontext-Menüs gibt es dafür nicht: stattdessen wird die App mit der Maus aus dem Dock bewegt und ‚*geschüttelt*'. Dann erscheint ein Menü zum Entfernen der App aus dem Dock.

Die geöffneten **Fenster** können über die drei Symbole im linken oberen Bereich **gesteuert** werden. So kann mit dem roten Button beispielsweise die App beendet werden. Während bei Windows innerhalb der **Fenster** nach unten **gescrollt** wird, so ist das bei macOS genau umgekehrt. Auf einem Laptop können hier Gesten genutzt werden, die recht analog zu neueren Windows-Versionen sind (Scrollen durch Bewegen von *2 Fingern* auf dem Maus-Pad nach unten oder oben, mit *3 Fingern* werden die laufenden Apps sichtbar gemacht werden; alternativ kann hierfür auch der ‚*Mission-Control-Button*' in der Info-Leiste genutzt werden).

Der Standard-**Browser** unter macOS ist **Safari**. Zur Bedienung sind Keyboard-Shortcuts (in der Regel 2 Tasten … + …, bei vielen Befehlen wird die Taste *command* ⌘ verwendet) recht nützlich (*command + t*: neues Tab öffnen, *command + +*: Seite vergrößern etc.). Es können auch mehrere **Schreibtische** (*+*

Button in *Mission Control*) anlegt werden. Zum Umschalten zwischen diesen ist die Tastenkombination *control* + → (Pfeiltaste vor, rück analog) nützlich.

Kopieren und Einfügen funktioniert (von Dateien aus / in Ordner, aber auch für Inhalte aus Dateien) mit den Tasten *command + c* und *command + v*.

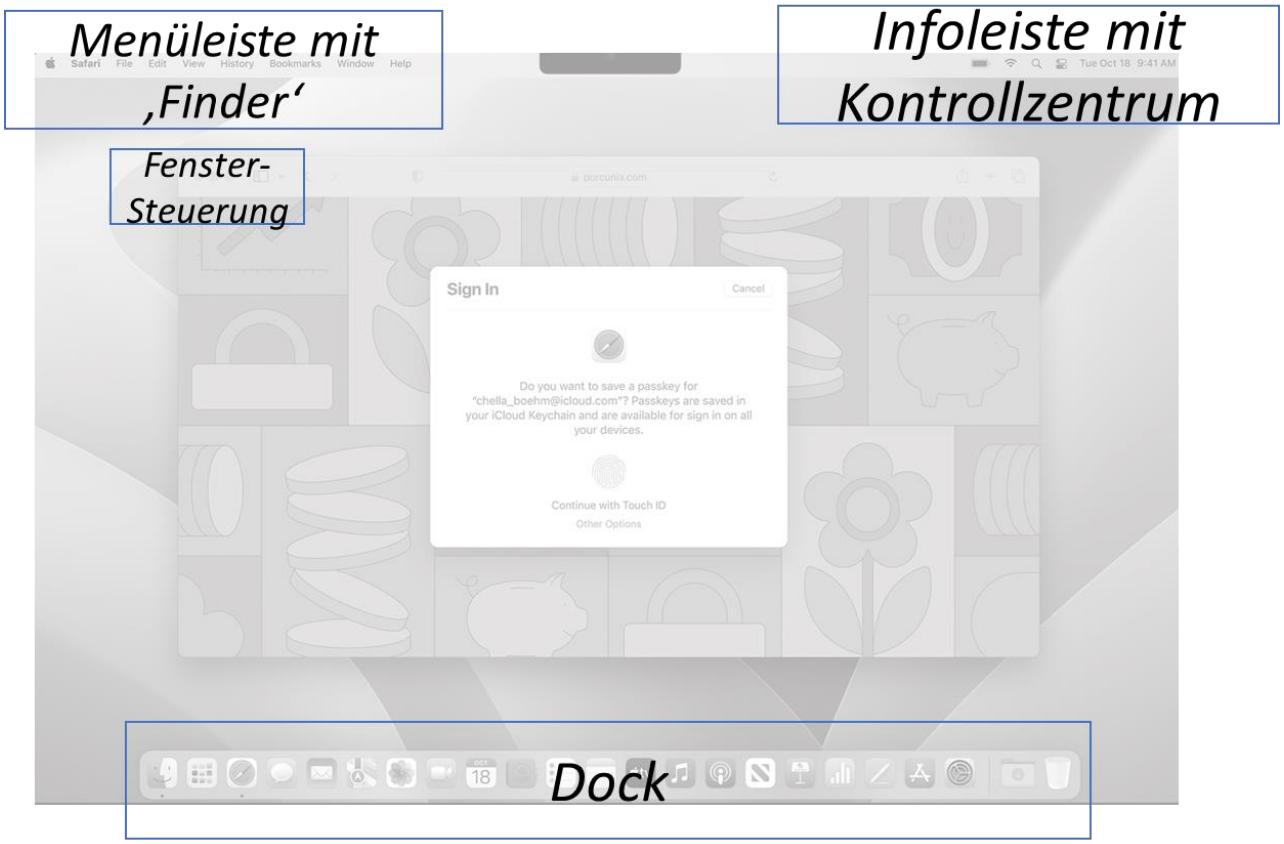
Speziell für die App-Entwicklung fehlen auf manchen Mac-Tastaturen (MacBooks) elementare Tasten, was später im Laufe des Praktikums noch auffallen dürfte. Die wichtigsten Tastenkombinationen in Verbindung mit der Taste *alt* (oder auch *option*) sind der Einfachheit halber hier dargestellt:

option + 7 erzeugt: | *option + Shift + 7* erzeugt: \ *option + L* erzeugt: @

option + 5 erzeugt: [*option + 6* erzeugt:]

option + 8 erzeugt: { *option + 9* erzeugt: }

Extrem nützlich ist die **Such-Funktion**: *command + space*. Hiermit können z.B. auch Apps gesucht werden. Eine nützliche Zusammenstellung von Shortcuts findet man unter: https://www.z-punkt.de/uploads/files/255/os_x_shortcuts.pdf.



Mit dieser sehr elementaren Übersicht sollte eine grundlegende Bedienung des Mac-Rechners möglich sein.

Einarbeitung Swift

In der Vorlesung wurden die elementaren Sprachkonstrukte von Swift vorgestellt, ohne die es bei der iOS-Entwicklung nicht geht.

Um das Verständnis von Swift zu vertiefen, sollen Sie zunächst ein Tutorial durcharbeiten.

Starten Sie dazu zunächst XCode (z.B. per Suche auf dem Rechner). Sollte es bei der Ausführung von Xcode zu Problemen kommen, dann bitte folgendes versuchen:

- öffnen Sie die App ‚*Self Service*'
- rufen Sie dort ‚*Xcode Fehler beheben*' auf

Das Tutorial `GuidedTour.playground.zip` befindet sich auch im Lernraum und wurde der Seite https://docs.swift.org/swift-book/GuidedTour/GuidedTour.html entnommen. Dieses Tutorial (dem Namen nach eine *Guided Tour*, wobei hier auch kleine Fragen und Experimente eingestreut sind) kann in XCode geöffnet werden. Dazu einfach die heruntergeladene Datei im ausgewählten Ordner (z.B. Download) auswählen.

Dabei wird quasi ein Dokument geöffnet, in dem auch Änderungen an den eingestreuten Code-Fragmenten vorgenommen werden können. Im unteren Bereich befindet sich ein Run-Button ▷. Daneben werden Fehler wie auch Ausgaben des Codes sichtbar gemacht. Durch längeres Drücken des Buttons werden die Optionen ‚*Automatically Run*' und ‚*Manually Run*' sichtbar.

Sie sollten die folgenden Kapitel durcharbeiten:

- *Simple Values*
- *Control Flow*
- *Functions and Closures*
- *Objects and Classes*
- *Error Handling*

Reflektieren Sie dabei die Unterschiede und Gemeinsamkeiten zu Kotlin. Die Kapitel ‚*Enumerations and Structures*' und ‚*Protocols and Enumerations*' sollten zu einem späteren Zeitpunkt durchgearbeitet werden.

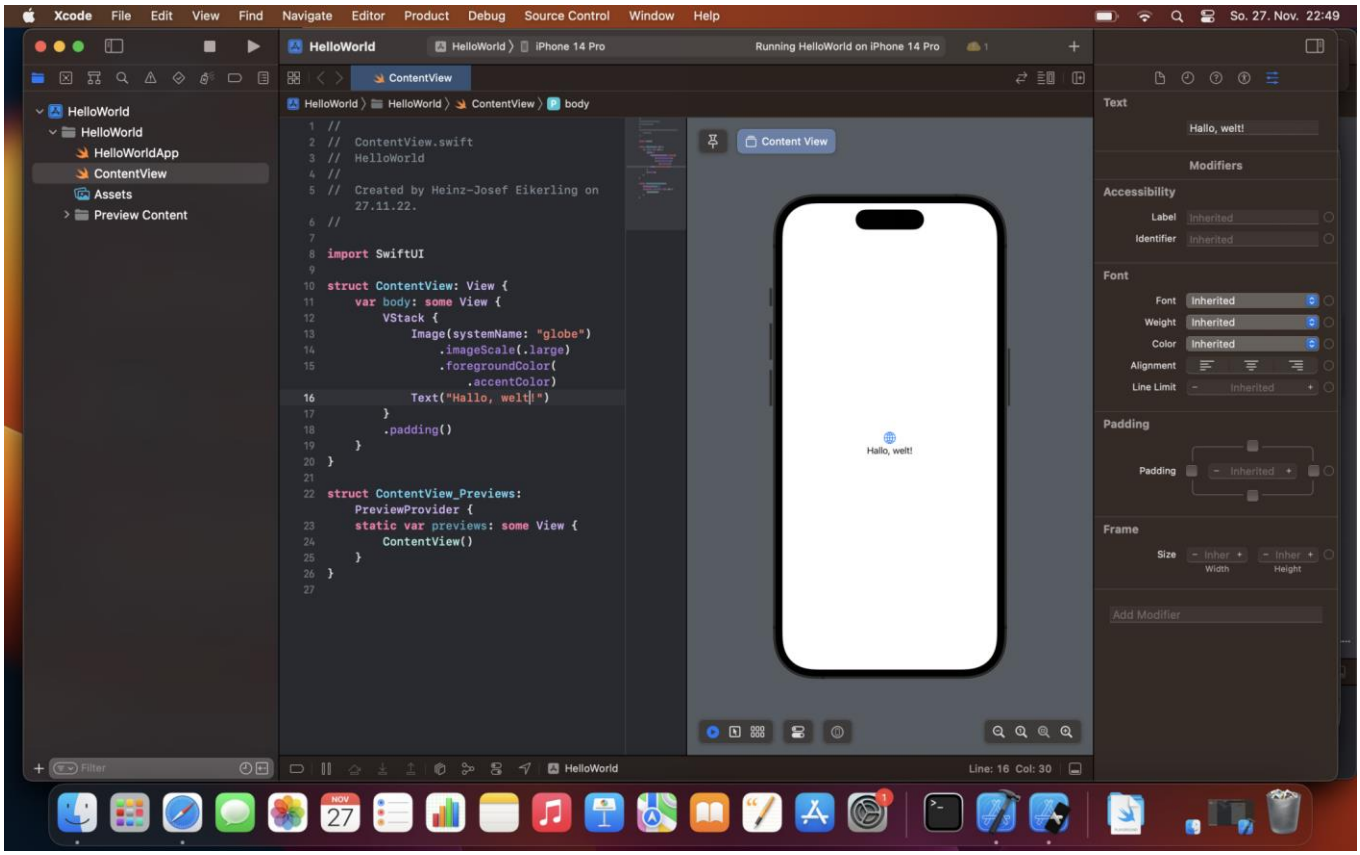
'*Hello World*' a la Swift

Eine erste Erfahrung mit XCode haben wir nun schon gesammelt. Die Realisierung einer Hello-World App ist schnell bewerkstelligt:

- Wählen Sie über die Menüleiste *File* → *New* → *Project* aus.
- In der nachfolgenden Auswahl geben wir als Ziel ‚*App*' an.
- Als Produkt-Name geben wir ‚*HelloWorld*' an. Weitere Einstellungen wählen wir nicht aus.
- Wir sehen hier das IDE mit (von links nach rechts): der *Projektstruktur*, dem *Source-Code* der ausgewählten Datei (z.B. Swift) und eine Vorschau (*Preview*) der App.
- Im oberen Bereich des IDE befinden sich die Steuerungselemente zum Starten (▷) der App im Simulator. Wir sehen dann in einem separatem Fenster eine Anmutung der Applikation

auf einem emulierten Zielgerät (z.B. *iPhone 14 Pro – iOS 16.1*). Alternativ kann auch *Product > Run* aufgerufen werden.

- Man kann nun etwas an der App ändern. Wählen Sie dazu die Swift-Datei `ContentView` aus. Ändern Sie die Zeichenkette `Text("Hello, world")` nach eigenem Gusto ab.
  Mit dem *command + b* (Build) wird das Projekt neu übersetzt, die Preview ändert sich.
  Rufen wir erneut ‚*Run*' auf, so ändert sich auch das Widget mit der App im Simulator, wenn wir zugestimmt haben, die laufende App im Simulator zu ersetzen.



Dice App mit SwiftUI

Arbeiten Sie nun das Tutorial unter: https://www.ralfebert.de/ios-app-entwicklung/swiftui-tutorial/ durch. Die ersten Schritte haben wir schon durchexerziert.

Das Tutorial adressiert die folgenden Punkte:

- Views definieren
- Handhabung der Preview
- Umgang mit Assets
- Kombination von SwiftUI-Views

SwiftUI Tutorial

Nun können Sie das Apple-Tutorial '*SwiftUI: Creating and combining views* ' durcharbeiten.

Sie finden es unter der URL: https://developer.apple.com/tutorials/swiftui/creating-and-combining-views

Der Aufwand des Tutorials wird mit ca. 40 Minuten angegeben.

Referenzen

- Es gibt auch ein **offizielles Handbuch** zu Swift, in dem alle wichtigen Mechanismen der Sprache vorgestellt werden. Das Handbuch ist in zwei Teile gegliedert, die Sie kostenlos aus dem *Apple Bookstore* (siehe entsprechende App) auf ein Apple-Gerät (z.B. Mac) laden und lessen können.

- Neben den Büchern von *Sillmann* (siehe Vorlesung) ist auch das Tutorial zu **SwiftUI** von Ralf Ebert (https://www.ralfebert.de/ios-app-entwicklung/swiftui-tutorial/) auf einem recht aktuellen Stand. Es ist speziell für Entwickler ohne iOS-Vorkenntnisse interessant.

- Offizielle Webseite zu Swift: https://developer.apple.com/swift/