User Interface Engineering PS A3

David Pape 01634454

1. Überblick

Die Idee ist eine Trinkwasser-Tracking-App, mit der Nutzer messen können, wieviel Trinkwasser sie getrunken haben und ob ihr Tagesziel schon erreicht wurde. Der Kniff bei dieser App sollte eine Funktionalität sein, die die bei neueren Smartphones zwecks Porträtmodus etc. vorhandenen Tiefensensoren und die bewährten Accelerometer einsetzt, um zu messen, wieviel Wasser in einem Glas ist.

Dies sollte dadurch geschehen, dass Nutzer ihr Gerät neben dem Wasserglas auf den Tisch legen und es dann bis zur Wassergrenze hochheben. Die erfassten Accelerometer- und Tiefendaten sollten zusammen mit milden Annahmen über die Form des Glases eine Schätzung der Wassermenge erlauben.

Natürlich ist dieses Feature nicht für jeden nützlich. Manche Nutzer werden genau wissen, wieviel Wasser in ein Bestimmtes ihrer Gläser passt, aber man kann sich auch vorstellen, dass Viele nicht genau wissen werden, welches Glas wieviele Milliliter genau fasst und bis wohin es gefüllt werden soll. Solchen Personen, die aber dennoch recht genau Buch über ihren Wasserhaushalt führen wollen, soll mit dieser App geholfen sein.

2. Implementation

Die Anwendung wurde mit Xcode und Swift, der Standard-Entwicklungsumgebung für iPhone implementiert. Diese Entscheidung wurde bewusst getroffen, da der MIT App Inventor nicht für iPhone bauen kann. Zur Implementation des Interfaces wurde Apples neue SwiftUI API verwendet.

```swift
struct ContentView: View {
    @State public var drank: CGFloat = 750
    @State private var showAccel: Bool = false
    var step: CGFloat = 360
    var mult:Int=1

    var body: some View {
        return ZStack {
            WavesBg(offs: 0, parent: self)
                .position(x: 250, y: UIScreen.main.bounds.size.height - 350)
                .animation(repeatingAnimation)

            if self.showAccel {

                // Show the accelerometer info box and start measuring...

            } else {

                // Show the "Settings"/"Add" button

            }
        }
        .frame(minWidth: .zero, maxWidth: .infinity, minHeight: .zero, maxHeight: .infinity)
        .edgesIgnoringSafeArea(.all)
    }
}
```

Screenshot A

SwiftUI erlaubt es, Interfaces in einem deklarativen Stil zu programmieren. Screenshot A illustriert, dass Elemente vordefinierte Konstruktoren wie `WavesBg()` haben, welchen dann mithilfe von Decorators wie `.position(x, y)` weitere Eigenschaften zugewiesen können. Erzeugte Elemente werden schlussendlich durch “Sammlungen” wie `VStack`, ein vertikaler Stack von Elementen, angeordnet und organisiert.

```
private var repeatingAnimation: Animation {
    Animation
        .easeInOut
        .speed(0.1)
        .repeatForever()
}
```

Screenshot B

Mit SwiftUI war es auch möglich, einen animierten Wellen-Hintergrund zu erstellen (in Anhang A: Screenshots und dem Video `video.mp4` sichtbar). Die Animationen werden als Klassen definiert (siehe den kleineren Screenshot) und schlussendlich mit dem Decorator `.animation(repeatingAnimation)` auf Elemente angewendet.

3. Limitierungen

Mit der Entscheidung für Xcode einhergehend war die Entwicklung der Anwendung natürlich aufwändiger und es konnten nicht mehr alle Features vollständig implementiert werden. Vielleicht auch im Vorhinein schon absehbar war es nicht möglich, eine zufriedenstellende Implementation des Mess-Features abzuliefern. Dies hängt mit der Komplexität der Sensor-APIs zusammen und auch damit, dass viel manuelles Tuning vonnöten gewesen wäre, um die Genauigkeit zu verbessern.

Der Hintergrund sollte außerdem den Fortschritt zum Tagesziel kommunizieren, indem der Wasserstand im Hintergrund gemeinsam mit dem Fortschritt zum Tagesziel ansteigt. Diese Funktion war aufgrund von Komplikationen mit SwiftUI und den Animationen leider auch nicht mehr möglich zu implementieren.

A. Screenshots

Folgende Screenshots sollten alles über das Konzept des Interfaces der App aussagen. Es ist weiters ein Video `video.mp4` inkludiert, in welchem die Animation der Wellen zu sehen ist.