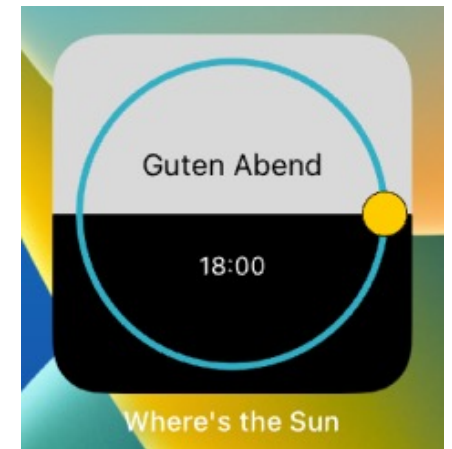


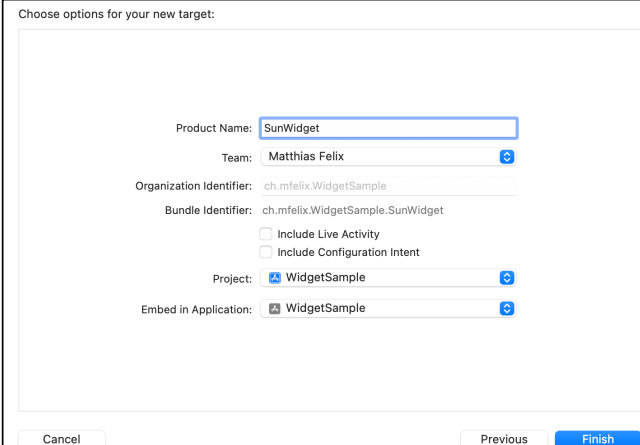
Übung 6

- Diese Übung befasst sich mit den Themen «Widget» und «Lokalisierung».
- Sie werden ein Widget erstellen, das jeweils den aktuellen Sonnenstand anzeigt sowie eine der Tageszeit entsprechende Begrüssung.



Widget Projekt

- Erstellen Sie ein neues Projekt **WidgetSample**: Wählen Sie **iOS App** als Template und **SwiftUI** als Interface.
- Fügen Sie als Erstes ein neues Target hinzu und wählen Sie als Template **Widget Extension**.
- Machen Sie sich v.a. mit dem Code im File `SunWidget.swift` vertraut; hier werden Sie ihr Widget implementieren.



Choose options for your new target:

Product Name:

Team:

Organization Identifier:

Bundle Identifier:

☐ Include Live Activity

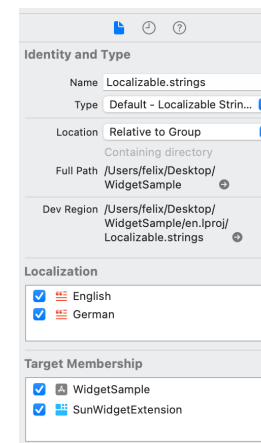
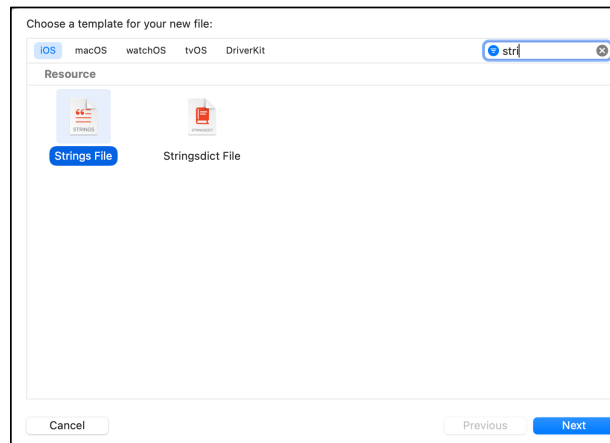
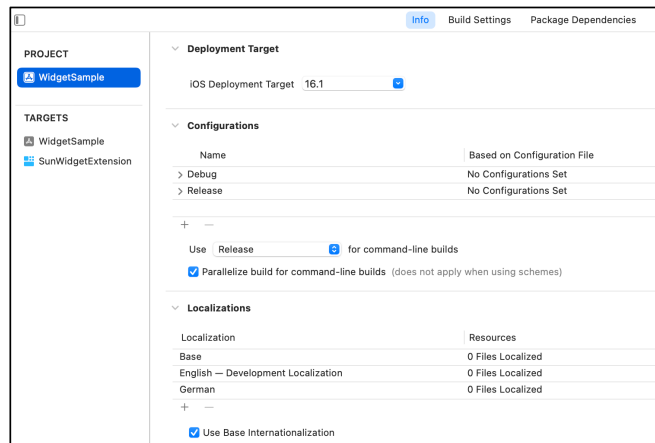
☐ Include Configuration Intent

Project:

Embed in Application:

Lokalisierung

- Fügen Sie in den Projekt-Settings unter **Localizations** eine zweite Sprache hinzu, sodass Deutsch und Englisch vorhanden sind.
- Fügen Sie danach ein neues Strings File `Localizable.strings` hinzu.
- Klicken Sie danach mit ausgewähltem File `Localizable.strings` rechts auf **Localize** und wählen Sie Deutsch und Englisch aus. Sie sollten nun zwei Versionen des Files haben.
- Achtung: Wählen Sie bei **Target Membership** beide Targets aus, damit auch die **SunWidgetExtension** Zugriff auf dieses File hat.



Timeline & TimelineEntry

- Konfigurieren Sie nun im File `SunWidget.swift` die Timeline ihres Widgets.
- Ihre `TimelineEntries` sollen folgende Konfigurationswerte enthalten:
 - `date: Date` → Der jeweilige Zeitpunkt des Updates
 - `greeting: LocalizedStringKey` → eine zur Tageszeit passende Begrüssung (resp. der Key dafür)
 - `sunAngle: CGFloat` → Der aus der Tageszeit errechnete Sonnenstands-Winkel.
- Ihr Widget soll alle 30 Minuten aktualisiert werden, und zwar immer zur vollen und halben Stunde. Erstellen Sie jeweils Entries für einen ganzen Tag, d.h. ihre Timeline sollte jeweils 48 Entries enthalten.
- Ihre `TimelineProvider`-Klasse ist demnach dafür verantwortlich, für die entsprechende Zeit die richtige Begrüssung sowie den Sonnenwinkel zu berechnen.

Timeline & TimelineEntry

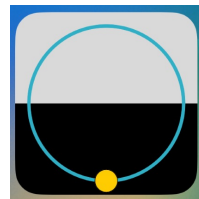
- **Bemerkungen:**

- `greeting`:

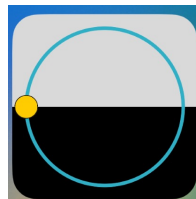
- Es soll je nach Tageszeit «Guten Morgen», »Guten Nachmittag», «Guten Abend» oder «Gute Nacht» (bzw. entsprechend auf Englisch) angezeigt werden. Sie müssen diese Texte also als Key-Value Pairs in beiden `Localizable.strings` Files erfassen, und dann im `TimelineEntry` den Key referenzieren.
 - Welche Tageszeit wann beginnt, dürfen Sie selber gemäss ihrem Tagesrhythmus entscheiden ;-)

- `sunAngle`:

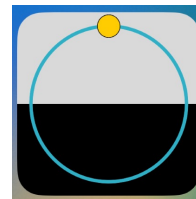
- Sie werden die Sonne auf einem Kreis darstellen. Die Position dafür kann mittels Geometrie relativ einfacher berechnet werden, dafür muss man jedoch den Winkel und den Kreistradius kennen. Unten finden Sie beispielhafte Zeit-/Winkel-Paare:



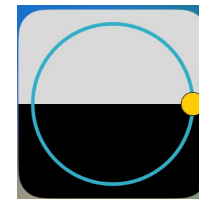
0:00 / 90°



06:00 / 180°



12:00 / 270°



18:00 / 0°

Widget-UI

- Nun werden Sie den Widget-View implementieren. Ihr `SunWidgetEntryView` hat bereits die eine benötigte Property `entry`. Darin sind alle benötigten Infos, um die Widget-UI zu implementieren.
- Es reicht, wenn sie die kleine Variante des Widgets implementieren. Wenn sie möchten, dürfen sie natürlich auch andere Grössen probieren. Um zu spezifizieren, welche Grössen ihr Widget unterstützt, benutzen sie diesen Modifier:

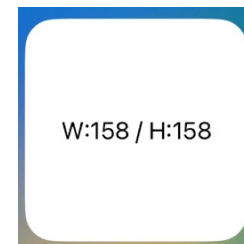
```
.supportedFamilies([.systemSmall])
```

Widget-UI

- Einige Tipps (1/2):
 - Ein `ZStack` eignet sich für diese View gut, da verschiedene Elemente übereinander gezeichnet werden (2-farbiger Hintergrund, Kreis, Texte, Sonne).
 - Um die Position der Sonne zu ermitteln, brauchen sie neben dem Winkel (im `entry`-Objekt) auch den Radius. Idealerweise benutzen sie dafür den SwiftUI-View `GeometryReader`, mit dem sie Zugriff auf die Size der View bekommen.

```
struct SunWidgetEntryView : View {
    var entry: Provider.Entry

    var body: some View {
        GeometryReader { proxy in
            ZStack {
                Color.white
                Text("W:\(Int(proxy.size.width)) / H:\(Int(proxy.size.height))")
            }
        }
    }
}
```



Widget-UI

- Einige Tipps (2/2):
 - Um die Sonne zu platzieren, verwenden sie am besten den Modifier `.offset()`. Wenn sie wie empfohlen einen `ZStack` verwenden, welcher standardmässig all seine Subviews in der Mitte platziert, können sie den x- und y-Offset mit einfacher Kreisgeometrie berechnen.
 - Farben / Styling: Fühlen Sie sich frei, mit dem Design zu experimentieren, andere Grössen/Farben etc. zu verwenden. SwiftUI eignet sich durch die Previews besonders gut, um zu experimentieren und verschiedene Designs rasch auszuprobieren.