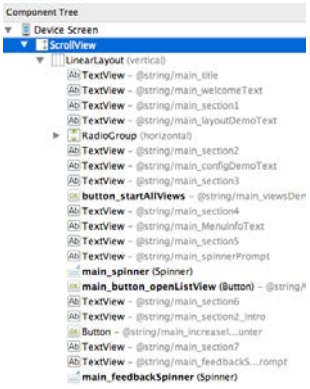
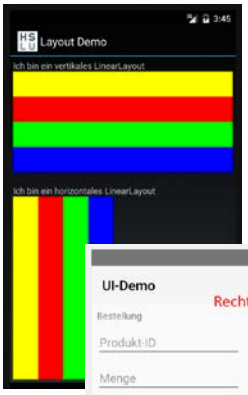
Übung 2: Benutzerschnittstellen & -interaktion

In dieser Übung verwenden wir verschiedene wichtige Klassen und Android-Mechanismen im Zusammenhang mit graphischen Benutzerschnittstellen und Benutzerinterkation. Konkret lernen Sie verschiede Views, Dialoge und Layout-Manager kennen, sowie die layout.xml-Dateien. Daneben kommen weiter Ressourcen wie Texte und Arrays vor. Sie verwenden View-Adapter und Sie lernen verschiede Möglichkeiten kennen, wie Benutzerereignisse (Events) mit Hilfe von Listeners behandelt werden können. Das Ziel dieser Übung ist es, die wichtigsten Android-Konzepte im Umfeld von Benutzerschnittstellen und –interaktion kennen zu lernen.

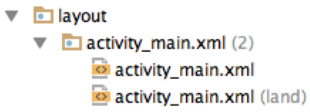
> Hinweis: Diese Übung besteht aus 7 Teilaufgaben, Teilaufgabe 0 (Setup neue App) ist Pflicht. Fürs Testat müssen Sie von den Teilaufgaben 1 bis 7 mindestens 4 lösen und präsentieren. Die Teilaufgaben sind analog zu den in der Vorlesung gezeigten Inhalten und Demos, konsultieren Sie also entsprechend die Folien.



0. **Neue App: UI-Demo & LayoutDemo**
   Erstellen Sie ein neues Android-Applikationsprojekt „UI-Demo“ („Phone and Tablet“, „Empty Activity“). Diese Übung besteht aus verschiedenen Teilaufgaben, welche in der `MainActivity` von diesem Projekt soweit als möglich resp. sinnvoll dargestellt werden. Diese Main-Activity (Layout-Datei: `activity_main.xml`) wird am Schluss ungefähr so aussehen wie der Screenshot rechts oben, verwenden Sie dazu eine `ScrollView` mit einem `LinearLayout`, siehe Screenshot rechts. Hinweis: Erstellen Sie das GUI nach und nach pro Teilaufgabe. Bei dieser ersten Aufgabe geht es nur darum, ein neues Android-Projekt anzulegen.



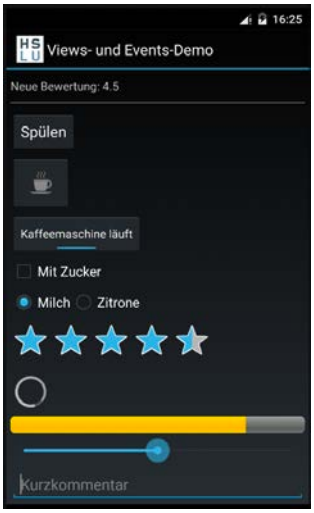
1. **LinearLayout & ConstraintLayout**
   Erstellen Sie mit Hilfe von zwei `RadioButtons` in einer `RadioGroup` eine Auswahl (siehe Screenshot rechts oben), in der das Layout für eine zu erstellende Activity `LayoutDemoActivity` ausgewählt werden kann. Das gewählte Layout soll als Code mittels Intent (Methode `putExtra`) an die `LayoutDemoActivity` übergeben werden. In dieser Activity soll in der `onCreate`-Methode aufgrund dieser Information entweder ein Beispiel-Layout für ein LinearLayout oder ein ConstraintLayout angezeigt werden (Methode `setContentView`). Erstellen Sie dazu zwei Layout-Dateien `layoutdemo_linearlayout.xml` und `layoutdemo_constraintlayout.xml`, mit welchen Sie entsprechende Bildschirmseiten wie in den beiden Screenshots rechts gestalten.



2. **Bildschirm-Orientierung**
   Erstellen Sie für den *Landscape*-Modus (= Bildschirm quer) neben dem `layout`- Ordner einen neuen Ordner `layout-land`. Und erstellen Sie darin eine Layout-Datei `activity_main.xml`, welche das Layout der `MainActivity` im Quer -Modus beschreibt. Im Quermodus soll die `MainActivity` ungefähr so aussehen wie im Screenshot rechts angegeben, also zweispaltig. Verwenden Sie dazu verschachtelte LinearLayouts. Testen Sie ihre App im Quermodus!

3. **Übersetzung nach Schweizerdeutsch**
   Erstellen Sie parallel zum vorhanden `values`-Ordner einen Ordner `values-de-rCH` für Schweizerdeutsche Übersetzungen. Kopieren Sie dort die bestehende Datei `strings.xml` hinein und modifizieren Sie deren Texte, um eine schweizerdeutsche Version ihrer App zu erhalten; siehe Screenshot rechts oben. Stellen Sie in ihrem Gerät (oder dem Emulator) die Sprache auf „Deutsch (Schweiz)“ um und testen Sie, ob die Sprachumstellung klappt.

4. **Views- und Events-Demo: RatingBar**
   Erstellen Sie eine Activity `ViewsDemoActivity` inkl. `activity_views_demo.xml` für das entsprechende Layout dieser Activity. Probieren Sie dort verschiede Views aus, z.B. verschiede Button-Typen oder `EditTexts` mit unterschiedlichen Input-Typen. Erstellen Sie auf jeden Fall eine `RatingBar`, auf welcher eine Bewertung von 0 bis 5 erfasst werden kann (siehe Screenshot rechts). Bei Bewertung soll in einer `TextView` oben auf dem Bildschirm ein entsprechender Text die neue Bewertung angeben. Registrieren Sie dazu, wie in der Vorlesung gezeigt, einen entsprechenden `Listener` auf der `RatingBar` und reagieren Sie wie beschrieben auf Bewertungs-Events.



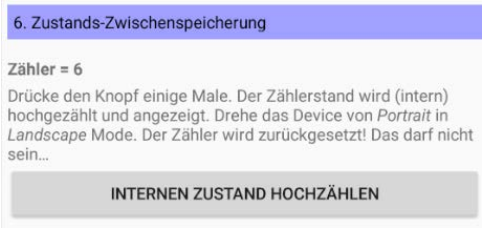
   Optional, aber empfohlen: Verwendung von DataBinding, resp. EventBinding, Observables + ViewModel.
   Siehe Folien + Google Doc zu „Android Data Binding“.

5. **Auswahl ermöglichen: Spinner**
   Fügen Sie ihrer `MainActivity` wie in der Vorlesung gezeigt einen Spinner hinzu, siehe Screenshot rechts. Erstellen Sie dazu eine Datei `res/values/arrays.xml` in welcher Sie im XML-Format mögliche String-Werte für den Spinner angeben. Bei Auswahl von einem Spinner-Item soll ein Toast mit dem entsprechenden Namen angezeigt werden. Implementieren Sie dazu die entsprechenden Event-Listener usw.

   Optional: Implementieren Sie eine analoge Liste mit Hilfe einer `ListActivity`.

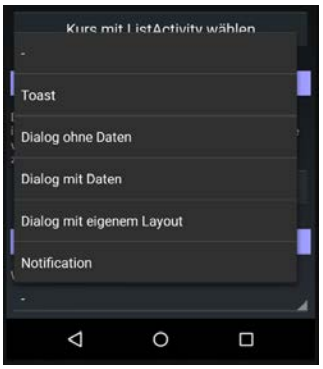
6. **Zustands-Zwischenspeicherung**

Legen Sie in der `MainActivity` ein Feld `int counter` an, welches durch Drücken eines Button um eins hochgezählt wird und dessen aktueller Wert auf einem Label angezeigt wird (siehe Screenshot rechts).
Nun soll dieser Zähler bei Darstellungsänderungen zwischen Portrait- und Landscape-Modus den Zustand nicht verlieren. Setzen Sie diese Funktionalität um, indem Sie ein `ViewModel` verwenden, um den Zustand zu halten.



7. **Dialoge**

Fügen Sie der `MainActivity` einen weiteren Spinner hinzu, auf welchem verschiedene Dialoge ausgewählt werden können, siehe Screenshot rechts. Implementieren Sie dann einen Dialog, welcher keine weiteren Daten benötigt und drei mögliche Antworten erlaubt, siehe Screenshot unten links für ein Beispiel. Bei Auswahl einer Antwort soll ein Toast mit der gewählten Antwort angezeigt werden. Implementieren Sie weiter einen Dialog, welcher in einer Liste mindestens drei verschiedene Optionen anbietet, siehe Screenshot unten rechts. Die gewählte Option soll wiederum in einem Toast angezeigt werden.



Optional: Experimentieren Sie mit weiteren Dialogen (z.B. mit Auswahl-Daten oder mit eigenem Layout) und mit Notifications.