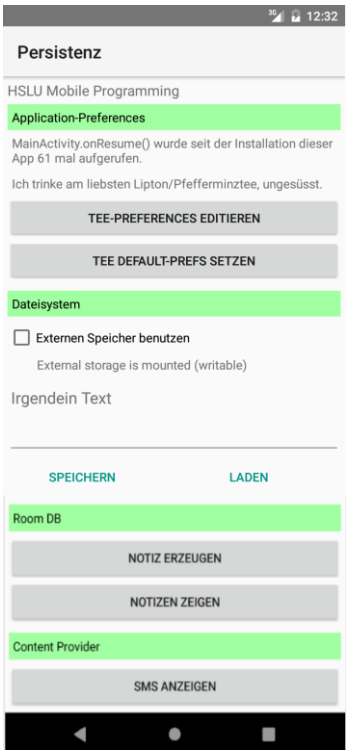
Übung 3: Persistenz – Preferences, Content Providers & Dateisystem, Room

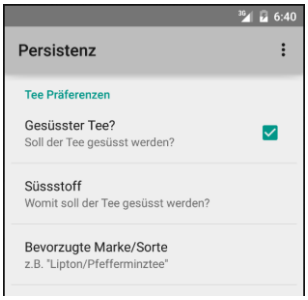


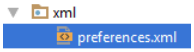
In dieser Übung verwenden Sie die drei Möglichkeiten, um auf Android Daten persistent zu speichern: *Preferences*, das *Dateisystem* und die *Room-Datenbank*. Für die Darstellung der Preferences verwenden wir dazu ein `PreferenceFragmentCompat`, welches für die Darstellung von Einstellungs-Bildschirmen praktische Unterstützung bietet. Danach speichern wir Daten in einer Datei und lesen diese wieder von dort ein. Anschliessend verwenden Sie einen Content Resolver, um via Content-Provider auf Daten einer anderen Applikation zuzugreifen. Zum Abschluss, speichern wir Notizen in einer Room-DB und lesen diese wieder aus.

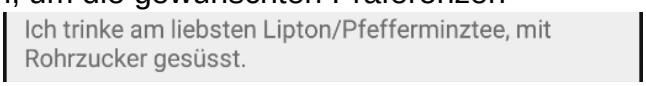
1. **Neue App: Persistenz**
   Erstellen Sie ein neues Android-Applikationsprojekt „Persistenz“. Diese Übung besteht aus verschiedenen Teilaufgaben, welche in dem `OverviewFragment` dieses Projekts soweit als möglich resp. sinnvoll dargestellt werden. Das `OverviewFragment` (Layout-Datei: `fragment_overview.xml`) sollte am Schluss ungefähr so aussehen wie der Screenshot rechts. Verwenden Sie dazu eine `ScrollView` mit einem `LinearLayout` (siehe Übung 2).

2. **Private Preferences: Resume Counter**
   Zählen Sie auf Ihrem `OverviewFragment` die Anzahl der Aufrufe der `onResume`-Methode. Verwenden Sie dazu einen Zähler, welchen Sie in den privaten Preferences der `MainActivity` persistent speichern. Geben Sie den aktuellen Zählerstand in einer `TextView` auf dem Bildschirm der Activity an, siehe Screenshot rechts oben. Wenn Sie die App beenden und wieder starten, muss der Counter erhalten bleiben.

3. **Tee-Präferenzen**
   Beim Klick auf den Knopf „Tee-Preferences editieren“ soll ein entsprechender Einstellungs-Bildschirm gemäss Screenshot rechts erscheinen, auf welchem die Tee-Präferenzen erfasst werden können. Verwenden Sie dazu, wie in der Vorlesung gezeigt, eine Ressourcen-Datei `preferences.xml`, in welcher diese Einstellungen spezifiziert sind. Erstellen Sie dazu weiter eine neues Fragment `TeaPreferenceFragment` und ein `SharedPreferencesViewModel`. Das Fragment soll von der Klasse `PreferenceFragmentCompat` erben, um die gewünschten Präferenzen anzuzeigen.







Hinweis: Der gewünschte Süssstoff soll nur abgefragt werden, wenn der Tee überhaupt gesüsst werden soll. Legen Sie für diese Aufgabe ebenfalls die dazu notwendigen String-Arrays in `arrays.xml` an. Die gesetzten Präferenzen sollen beim Verlassen des `TeaPreferenceFragment` auf dem `OverviewFragment` aktualisiert werden.

4. **Tee-Default-Präferenzen**
   Implementieren Sie einen Knopf, mit welchem ein vordefinierter Satz Tee-Präferenzen im App-Code gesetzt und danach entsprechend angezeigt werden.

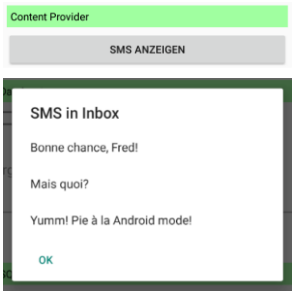
5. **Dateisystem: Text speichern / laden**
   Fügen Sie ihrer App ein Textfeld zum Erfassen und zwei Knöpfe zum Speichern und Laden eines Texts hinzu. Der erfasste Text soll dabei wahlweise intern oder extern Speicher gespeichert werden. Statusmeldungen (Speichern ok, Fehler beim Einlesen, usw.) sollen dabei direkt im Textfeld angezeigt werden. Die einzeln Code Stücke finden Sie in den Folien.

   Damit auf dem Emulator der externe Speicher funktioniert muss dieser mittels «adb» Kommando aktiviert werden, dafür können Sie folgend Befehl im Terminal im Android Studio eingeben:

   adb shell sm set-virtual-disk true

   Für die Abfragen im Dateisystem sollte ein ViewModel verwendet werden

6. **Externen Content Provider nutzen: SMS anzeigen**
   Durch Klick auf „SMS anzeigen“ sollen alle zurzeit in der Inbox vorhandenen SMS via `Content Provider` in einem `AlertDialog` angezeigt werden. Greifen Sie dazu mittels der URI-Konstanten `Telephony.SMS.Inbox.CONTENT_URI` auf den entsprechenden Provider zu.



   Hinweis: Vergessen Sie nicht, im Manifest die entsprechenden Permissions einzutragen und beim Benutzer zu erfragen!

   Tipp: Sie können dem Emulator SMS schicken, indem sie auf der Emulator-Toolbar *…* > *Phone* > *SMS Message* > *Send Message* benutzen

7. **[OPTIONAL] SQL Persistenz mit Room: Notiz erfassen & anzeigen**
   Durch Klick auf „Notiz erzeugen“ soll eine Notiz mit Titel und Text erfasst werden können. Erstellen Sie dazu ein neues Fragment `EditNoteFragment`, welches verwendet wird, um neue Notizen zu erfassen. Um bestehende Notizen anzuzeigen, erstellen sie ein Fragment `ShowNoteFragment`. Welches die Funktion `newInstance(noteId: Long): ShowNoteFragment` implementiert. Diese `noteId` sagt dem Fragment welche Notiz angezeigt werden sollte. Bei Klick auf „Notizen zeigen“ soll eine Liste mit allen verfügbaren Notiz-Titeln angezeigt werden (`NoteListFragment`, inkl. Verwendung eines selber definierten `NotesAdapter` mit einer `RecyclerView` und einer Datei `item_layout.xml` zur Spezifikation des Layouts eines Recyclerview-Items). Wird eine Notiz angeklickt, wird diese in dem `ShowNoteFragment` angezeigt. Hinweise: Erstellen Sie dazu wie in der Vorlesung gezeigt je eine Klasse `Note` (Entity), ein Interface `NoteDao` (Data Access Object) und eine abstrakte Klasse `NotesDatabase`. Für diese Übung ist es zulässig, DB-Abfragen im Main Thread auszuführen.