## Exercise 8.1: Store and restore (Currency Converter)



Use *Preferences* to store and restore the following settings:

* Source and target currency
* Entered value

If you leave your app by pushing the *Home* button and reopen the activity the values for the *Spinners* and *EditTexts* will be restored automatically anyway.

To test your functionality actually close the app: Push the software key in the lower right corner and then close the app using the close button.

## Exercise 8.2: Current Currencies on Program Start (Currency Converter)

When starting the currency converter app it uses the predefined conversion rates until they get updated with more current rates from the ECB website.

Extend your app so it persists all rates retrieved from the ECB and restores the most current local data when the app is started next time.

You can either use *preferences* or file access. If you feel brave you can also try to store the data in a SQLite database.

## Exercise 8.3: List all Songs

Use the MediaStore content provider to log a list of **all** songs on the device to the console.

To do so:

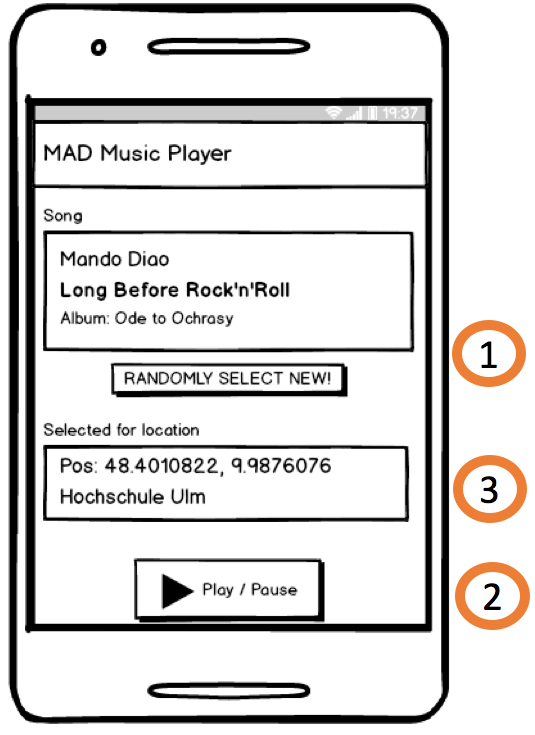
* Get a ContentResolver
* Query the URI MediaStore.Audio.Media.EXTERNAL\_CONTENT\_URI
* Iterate over the resulting Cursor and output artist, album, title and duration (see lecture slides and documentation at <https://developer.android.com/reference/android/provider/MediaStore.html>)

**Note:** You will need audio data on the device the app runs on. The emulator image does not contain audio by default. You can work around this in various ways:

* Test using a real phone
* Upload a directory of mp3 files to the emulator image using the following command from the command line: adb push <PathToDirectoryWithMp3> /sdcard/
* Alternatively query the URI MediaStore.Audio.Media.INTERNAL\_CONTENT\_URI to get a list of audio in the devices *internal* memory (usually ringtones).

## Exercise 8.4: Starting the MAD Music Player: Randomly Select a Song

**With this exercise, we will start a new project.**



**Project Vision:** The overall goal will be to create a special type of music player with the following requirements:

1. On button press the app randomly **selects a song from the list of all songs** on the device’s external memory.

The selection is **persisted** when the application is destroyed and recreated.

1. The app allows to **play** **the song** in an endless loop. Playback can be paused and resumed.
2. Optional: The app will be made **location-aware**: Song selections are only valid for the location where they were made. At a different location, a different song can be selected.

We will implement the player over the course of this and the following two exercises. Of course, you are free to change the concept as long as the elements song selection and playback are part of the app.

**Today:** Create a basic app that only implements UI and functionality for requirement 1.

**Note:**

* You can ask a Cursor about the number of found elements using the method getCount(). You can move to the element with the index position using moveToPosition(int position).