Playlist App

CSE 438

Spring 2019

Introduction

In this assignment, you will use a third party API to build a playlist application to search for music.

Details

- Due Date: February 28, 2019
- Grading: This entire lab is 100 points. The distribution of points is listed below in the requirements section.
- Submission: Please put your project into your forked repository. The repository can be found here. After you fork your repository, please provide admin access to the class account: cse438wustl@gmail.com. You can do this in the User Access section of the repository settings.

Last.fm API

You will be using the Last.fm API to query for music data. The API itself has a lot to offer, so consider using different functions for the creative portion (and documenting the appropriate API calls). You can find instructions for setting up an account and getting an API key here. We recommend getting your account setup and making a few sample calls in the browser to see what the JSON output looks like.

Description

When the app launches, there should be a GridView displaying the current top tracks (**Hint**: Take a look at the What's New section of the studio code and the API documentation for how to do this). The GridView should display at least 20 tracks if possible (with scrolling), and should display the track cover (if available) and title.

The app should have two tabs - the first should give the user the ability to search for different tracks by artist name. After the user searches for the artist, the app should display the top tracks by that artist in a GridView with the same format as with the top tracks view (track cover and title, at least 20 results if available). If no result were returned, the app should indicate that to the user.

If a user clicks on one of the entries into the GridView, a new activity should open up with that tracks cover, title, and three other pieces of information you can pick (**Hint**: You may need to make another API call for this information). Additionally, there should be an option to add the track to the playlist.

The second tab should have a users playlist. These should be songs that the user saved from searching for artists. The playlist should have the songs in a ListView with the name of each saved song and it's artist. This data should be stored and fetched from a SQLite database.

Helpful Advice

- Take a look at the studio code for how to make external API calls and process the JSON result.
- The API documentation is your friend - make sure to read it fully. Specifically, make sure you take a look at how to get your result formatted as a JSON.
- There are two ways to approach a GridView. One way is to use a GridView layout with a custom adapter, just like you did in the previous project. The other way is to use a RecyclerView with a GridLayoutManager. Either approach is acceptable for this assignment.
- To make the items in your grid view clickable, you can simply add a listener to each grid item. There is an example of this in studio 3, in the ResultListFragment.
- For testing purposes, consider using some dummy JSON data you save locally to test your GridView features instead of making an API call every single time.

We have covered nearly every component of this project in class. Below is a list of where you can find examples of concepts that you will need to complete this assignment:

- Searching: Studio 2
- API calls: Studio 2 and Trivia App
- SQLite setup: JokeDB and Studio 3
- TabLayout: JokeDB
- Gridlayout setup: Studio 2/3 (uses a RecyclerView with a LinearLayoutManager, which is very similar)
- Asyncronous calls: Studio 2, Trivia App

You may notice that the way this app is setup is very similar to the way that Studio 3 is set up. It is in your best interest to complete that studio before beginning the assignment.

Creative Portion

For every homework assignment, you will be asked to think of an additional feature to be added to the application that will improve the user experience and provide you an opportunity to learn about concepts that you are personally interested in. Put yourself in the shoes of your users: what features would they like to see in an app like this? Try to make it something new and substantially different from what the app already does - do not just rehash existing requirements.

When you submit your assignment, please include a ReadMe.txt file that explains your creative portion. You should explain what the feature is, why you chose to implement that particular feature, and how you went about implementing it.

To recieve full credit, your feature needs to be substantial as compared to the rest of the assignment. Examine the rubric below to get a feel for how much weight we are putting on the creative portion of the assignment.

Requirements

- (10 points) The app displays the current top tracks in a GridView on startup
- (10 points) The app uses a tab bar with two tabs, one for searching for tracks and one for looking at the playlist
- (10 points) Data is pulled from the API and processed into a GridView on the main page. Makes use of a Fragment to display the results seamlessly.

- (15 points) Selecting a track from the GridView opens a new activity with the track cover, title, and 3 other pieces of information as well as the ability to save it to the playlist.
- (5 points) User can change search query by editing text field.
- (10 points) User can save a track to their playlist, and the track is saved into a SQLite database.
- (5 points) User can delete a track from the playlist (deleting it from the SQLite database itself).
- (4 points) App is visually appealing
- (1 point) Properly attribute Last.fm API as source of data.
- (5 points) Code is well formatted and commented.
- (10 points) All API calls are done asynchronously and do not stall the application.
- (15 points) Creative portion: Be creative! (**Hint**: You have an entire API available to you - use it!)