CS478: Software Development for Mobile Platforms

Project #5

Due time: 11:59 pm on 4/30/2021
Submit using Blackboard web site
Total points: 100
Instructor: Ugo Buy

You are to code two Android apps. The first app, named *MusicCentral*, stores information about songs. This information includes for each song (1) the title of the song, (2) the name of artist or band who plays the song, (3) a picture (bitmap) associated with the song, and (4) a string denoting the URL of a web site containing an audio file for the song. *MusicCentral* stores information about *n* songs numbered 1 through *n*, with $n \geq 5$. The app exposes a service that supports functionality for downloading information about each song. Because the *MusicCentral* service has an effect on the user experience of the device, *this service should run in the foreground.*

The API of *MusicCentral*'s service should expose the following 3 pieces of functionality in an appropriate AIDL file:

1. Retrieve all information for all songs stored in the service.

2. Retrieve all information for one specific song by the song's number.

3. Retrieve the URL string of the site with the song's audio file.

The second app, *MusicClient*, contains a main activity that supports functionality for using the *MusicCentral* app by starting and binding to its service. Once bound to the service, *MusicClient* can use the service's API to retrieve information about songs. Once the service is bound, *MusicClient* allows an interactive user to download information about each song, display that information, download the song from the URL provided by the service and play the song.

In particular, the interactive user of *MusicClient*'s main activity uses key presses on appropriate buttons to bind and unbind from the service. The status of the service should be clearly displayed in the main activity as well. An additional UI element in the main activity allows the user to select a song from the service if the service is bound. The song will be played by *MusicClient*. In addition, when bound to the service, the client can request a list of the songs stored in the service or information about a specific song. The list of songs is displayed in a *RecyclerView* of a second activity of *MusicClient*. For each song (i.e., list item) the following information is displayed: (1) title of song, (2) name of the artist or band who plays the song, and (3) a picture relating to the song. A click on a list item starts the playback using the predefined *MediaPlayerService* to manage the playback.

You should enable and disable appropriately commands in the main activity of the *MusicClient* app depending on whether the activity is bound to the service or not. For instance, commads requiring binding should be disabled when the activity is not bound. You may "gray-out" disabled views to help users select the appropriate actions depending on the service state.

Hints. You are at liberty to choose the audio clips from segments pictures publicly-available (and not copyrighted or otherwise protected) on the Internet. When testing your application, make sure to upload *MusicCentral* app first, or else the client app may fail to initialize properly. Use methods *startForegroundService()* and *bindService()* to start the service and to bind to the service. Finally, in *MusicClient* use Android's built-in *MediaPlayerService* to play the music.

Implementation notes. As with the previous projects, use a Pixel 3 XL virtual device running the usual Android platform (API 30—Q). Define the layout of your client app in such a way that it will display best in portrait mode. You are not required to provide backward compatibility with previous Android versions or to support phone reconfigurations. Use the AIDL to expose the service's functionality. Make sure that the implementation of the service is *thread-safe*. This means that access to data structures shared by multiple clients should be protected by suitable mutual exclusion locks.

Be advised that we will stress test the robustness of your client app if the service suddenly dies. In this case, the playback should stop but the user of the client could restart the service and start a new playback without restarting the client app.

You must work alone on this project. Submit a zip archive containing two root directories; each directory contains the full Android Studio repository of the corresponding app. No late submissions will be accepted.