Services

Due Date: Monday, December 9th, 2019 @11:59pm

Project Details:

You are to code two Android apps. The first app, named ClipServer, stores a number of audio clips, such as songs or other recordings. The clips are numbered 1 through n, where n is the total number of clips, with $n \geq 5$. This app contains a service intended to be both started and bound, which exposes an API for clients to use. The API supports such functionality as playing one of the audio clips, pausing the clip, resuming the clip and stopping the playback altogether. The ClipServer app includes at least 5 audio clips of variable duration. The duration of Clip #1 should be at least 30 seconds but no more than 3 minutes. Because ClipServer's service has an effect on the user experience of the device, this service should run in the foreground.

The second app, AudioClient consists of an activity that exposes functionality for using the ClipServer and binds to the service for playing desired audio clips. AudioClient has the ability to start and stop the service. Once the service is started, AudioClient allows an interactive user to play one of the n audio segments. The AudioClient activity binds to the service for a duration of an audio clip. The activity unbinds from the service when either a clip has ended or the user stops the playback; however, the service is not stopped until the user decides to do so. (See UI spec below.) In other words, by default the service remains in the started state even if it is not playing an audio clip.

Implementation Details:

The interface of the AudioClient app should mininally include appropriate View elements for supportin the following functionality: (1) Starting the service, (2) Playing a given clip (by number), (3) Pausing the playback, (4) Resuming the playback, (5) Stopping the playback (which will also unbind the activity from the service), and (6) stopping the service. When the client activity is stopped, the service should continue playing; however, the service should be unbound if the activity is destroyed. Also, command (2) should be enabled only if the service has been started with command (1). Likewise, command (3) should be enabled only after (2), and so on. However, command (6) must stop a playback in progress and unbind from the service, after showing a dialog that the playback will be stopped. Do "gray-out" disabled views to help users select the appropriate actions depending on the service state.

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 ClipServer 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 ClipServer use Android's built-in MediaPlayerService to play the music.

As with the previous projects, use a Pixel 2 virtual device running the usual Android platform (API 28—Pie). Your client app layout 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. 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.

Submission Details:

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

Academic Integrity:

Unless stated otherwise, all work submitted for grading *must* be done individually. While we encourage you to talk to your peers and learn from them, this interaction must be superficial with regards to all work submitted for grading. This means you *cannot* work in teams, you cannot work side-by-side, you cannot submit someone else's work (partial or complete) as your own. The University's policy is available here:

https://dos.uic.edu/conductforstudents.shtml.

In particular, note that you are guilty of academic dishonesty if you extend or receive any kind of unauthorized assistance. Absolutely no transfer of program code between students is permitted (paper or electronic), and you may not solicit code from family, friends, or online forums. Other examples of academic dishonesty include emailing

your program to another student, copying-pasting code from the internet, working in a group on a homework assignment, and allowing a tutor, TA, or another individual to write an answer for you. It is also considered academic dishonesty if you click someone else's iClicker with the intent of answering for that student, whether for a quiz, exam, or

class participation. Academic dishonesty is unacceptable, and penalties range from a letter grade drop to expulsion from the university; cases are handled via the official student conduct process described at https://dos.uic.edu/conductforstudents.shtml.