Working with Audio Files

Mr. Prashanth BS

November 27, 2021

1 MediaPlayer-Playing Audio File

The Android multimedia framework includes support for playing variety of common media types, so that you can easily integrate audio, video and images into your applications. You can play audio or video from media files stored in your application's resources (raw resources), from standalone files in the filesystem, or from a data stream arriving over a network connection, all using MediaPlayer APIs. One of the most important components of the media framework is the MediaPlayer class. An object of this class can fetch, decode, and play both audio and video with minimal setup. It supports several different media sources such as,

- Local resources
- Internal URIs, such as one you might obtain from a Content Resolver
- External URLs (streaming)

Some of the APIs which comes under the MediaPlayer class are,

- To use the existing mediaplayer in the android, Android system provides MediaPlayer Class. Instantiate the object of this class as follows, it takes two arguments, they are the context and source of the file in the raw folder.

```
MediaPlayer mediaPlayer = MediaPlayer.create(context,
    source_file_mp3) ;
```

- *MediaPlayer* class provides a bunch of APIs Which can be used with mediaplayer object, they are

- *mediaPlayer.play*(): To play the audio file
- *mediaPlayer.pause*(): To pause the audio file
- *mediaPlayer.reset*(): To move to the start of the audio file
- *mediaPlayer.stop*(): To stop the audio file from playing
- *mediaPlayer.seekTo*(*position_in_millisecond*): To move to the location specified in the audio file in millisecond
- *mediaPlayer.getCurrentPostion*(): To retrieve the current position of the audio file while playing
- *mediaPlayer.release*(): Releases the audioplayer resources that is the audiofile
- *mediaPlayer.getDuration*(): To get the full duration of the audio file

2 Exercise

Create a media player application that will play media file saved on Android Phone. Demonstrate application with play, pause, fast forward, and rewind functionality

The following are the steps followed,

1. Create a raw folder under res directory and add the MP3 file with user specific name as shown in the figure 1,

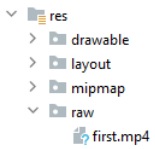

Figure 1: Folder Hierarchy

2. Create the layout as shown below in Figure 2 using LinearLayout with vertical orientation with buttons for play, pause, forward, rewind,restart and stop operation. The sample code for one button in activity_main.xml is as shown below,

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
```

Figure 2: Folder Hierarchy

```
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity"
android:orientation="vertical"
android:background="#7712">

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="@string/songname"
    android:id="@+id/songname"
    android:layout_margin="30dp"
    android:fontFamily="cursive"
    android:textSize="40dp"
    android:gravity="center"
    android:textStyle="bold"
    />
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginLeft="30dp"
android:layout_marginRight="30dp"
android:text="Play"
android:id="@+id/play"/>
</LinearLayout>
```

3. In the Mainactivityjava file, define the equivalent java objects as follows,

```
Button play, forward, rewind, pause, stop, reset;
MediaPlayer mediaPlayer ;
int starttime = 0 ; // startime is 0s
int stopttime = 0; // stoptime is 0s by default
int forwardtime = 5000 ; // 5s for forward
int backwardtime = 5000 ; // 5s for backwardtime
```

4. Instantiate *MediaPlayer* class and refer to the media file in the raw folder(Suppose the name is first.mp3) refer it as follows,

```
mediaPlayer = MediaPlayer.create(this, R.raw.first) ;
songtitle.setText("first.mp3");
```

5. Add an event click listener for all of the buttons as follows, Implement the same for pause, stop using play code as a template

```
play.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(MainActivity.this, "Playing
                    Media now", Toast.LENGTH_SHORT).show();
                mediaPlayer.start();
            }
});

reset.setOnClickListener(new View.OnClickListener() {
       @Override
       public void onClick(View v) {
         mediaPlayer.reset();
         mediaPlayer.start();
}
});
```

6. To implement the forward operation, first we need to determine the current position in which the audioplayer is playing which can be accessed using $mediaPlayer.getCurrentPostion()$ call.The full duration of the audiofile can be obtained by calling $mediaPlayer.getDuration()$. The audio can be forwarded only when the $(currentTime + forwardTime) < mediaPlayer.getDuration()$ which is as shown below,

```
forward.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        int currentpos = mediaPlayer.getCurrentPosition() ;
        if((currentpos+forwardtime) <= (stopttime =
            mediaPlayer.getDuration())){
              mediaPlayer.seekTo(currentpos+forwardtime);
         }
        }
});
```

7. Implement the backward operation on the similar lines.