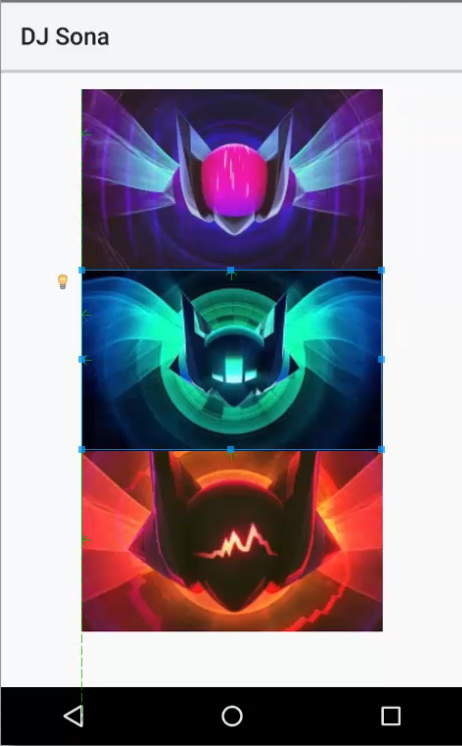
Tutorial 10 – DJ Sona

1. Head over to my github account and pick up the media files I’ve used. First go get the images at the following link: <https://github.com/amyork/Android_Studio_Tut_10_DJSona/tree/master/app/src/main/res/drawable>
2. Now, collect the .mp3 files: <https://github.com/amyork/Android_Studio_Tut_10_DJSona/tree/master/app/src/main/res/raw>
3. Create a new project named “DJSona” using API 21 (this is what I used). Keep the other options as their default.
4. Copy all 4 of the images provided, and paste them into the drawables folder, which is within the res folder.
5. For the .mp3 files, we need to create a new resource folder of raw type. Since raw is a subfolder or the res folder, Android automatically generates ID’s for the items within, which are stored within the R class. This gives Android’s classes and methods, even XML, access to these files. Right click on the res folder and select new > Android resource directory. Go to resource type and select “raw” and click OK. If the raw folder doesn’t appear right away, minimize the res folder and then expand it. Copy the .mp3 files, and paste them into the raw folder.
6. Now, let’s work on the design. Drag and drop 3 Image Button widgets onto the design. Stack them vertically and centered.
7. Select imageButton and find “src” in the properties tab select the ethereal image. Go to the very top of the properties window (scroll up) and change the layout width to 250dp, and change layout height to 150dp.
8. For imageButton1, repeat step 7, selecting the kinetic image.
9. For imageButton2, repeat step 8, selecting the concussive image.
10. To help arrange everything nicely, leave a tiny bit of room above the topmost image (ethereal) and then drag the other 2 buttons up against the bottom so there is no space in between them.



1. Select the second image button (kinetic), and find layout:margin in the properties tab. Expand the options and enter in 15dp into “top.” Do the same for the third image button (concussive). This will leave small gaps between each image making them evenly separated.
2. For each image button, select the button, then go to the near-bottom of the properties tab to find scaleType. Set each of the buttons up with the “centerCrop” scaletype.
3. Set the background image by clicking the activity anywhere that does not have a widget, then find the background property in the properties window and selecting the background\_music image. When doing this, you will notice that each image button has a white border.
4. If you like the white border, keep it. If you don’t like the white border, select each image and find the background property in the properties window and change the background color. This will cause the picture to expand over the white border. Finished Design result below:



1. Time to get coding. We have a few packages to import:

*// My Imports***import** android.view.View;  
**import** android.media.MediaPlayer;

1. Next, we need to declare 3 MediaPlayer objects. This is because we have three different songs we wish to play. Within the MainActivity class, add the following code:

**private** MediaPlayer **player1**, **player2**, **player3**;

1. In the onCreate method, we need to set our media players. The first parameter is the context, and the second parameter is the media from within our newly created raw file folder.

**player1** = MediaPlayer.*create*(**this**, R.raw.***ethereal***);  
**player2** = MediaPlayer.*create*(**this**, R.raw.***kinetic***);  
**player3** = MediaPlayer.*create*(**this**, R.raw.***concussive***);

1. What we do next is the real “meat” of the program. We’re going to create 3 different functions, one for each song. This way, we can go to our XML file and call the respective function, for each button.

**public void** playEthereal(View view)  
{  
 **if** (**player1**.isPlaying())  
 **player1**.pause();  
 **else  
 player1**.start();  
}  
  
**public void** playKinetic(View view)  
{  
 **if** (**player2**.isPlaying())  
 **player2**.pause();  
 **else  
 player2**.start();  
}  
  
**public void** playConcussive(View view)  
{  
 **if** (**player3**.isPlaying())  
 **player3**.pause();  
 **else  
 player3**.start();  
}

The above code is pretty basic, meaning that we don’t check to see if another song is playing. For the scope of this program, we won’t be checking. It is nice to know that we can play multiple media files at once though! Please see the video for a more “depthy” explanation of why this is programmed the way it is. For example, if we tried to stop a song that wasn’t playing, an error would occur, and the program would break.

1. We’re almost done here. Next, we need to manage what happens to our media player objects when the user leaves the application. Currently, the music would continue to play. To fix this, we need to release the objects… in other words, destroy them. Go below the playConcussive function and hold down the ALT key, then press Insert. In the menu that appears, select Override Methods. Search the list of methods until you find “onPause.” The application goes into a pause state when changing apps or going back to the phone’s home screen. Insert the override method and add the following lines of code to destroy the objects:

**player1**.release();  
**player2**.release();  
**player3**.release();

1. The last step! We need to go to our XML file and add a line to each ImageButton. Go to app > res > layout, then double-click on activity\_main.xml. Once here, add the following line to the corresponding ImageButton:

**android:onClick="playEthereal"**

**android:onClick="playKinetic"**

**android:onClick="playConcussive"**

1. Save and test your program. You’ve done it!

Video Link: <https://www.youtube.com/watch?v=c4ZJ28-8GHc&list=PLFVlCGwfyegYi8G0yxIVlGfjT3xGzCZOz&index=1>

Github Link: <https://github.com/amyork/Android_Studio_Tut_10_DJSona>