**University Of Delhi**

**PROJECT REPORT ON**

**RHYTHM**



**An Android Application to play songs**

|  |  |
| --- | --- |
|  |  |

ABSTRACT

The aim of the project is to develop an Android application that lets its users to play songs without internet . The idea for this application is easy and effective.

The basic concept of this mobile application is,

Whenever user want to listen music irrespective of place and situation with no internet and no knowledge of which song to hear, This Application would help with a variety of songs.

#### TABLE OF CONTENTS

**INTRODUCTION 8**

MOTIVATION 9

**.java Files Included 10**

Splash.java [Landing Screen] 10

Choice.java 11

Choice2.java 11

MainActivity.java 12

MusicList.java 13

PlayerActivity.java 15

**LIBRARIES INCLUDED 16**

**Features and files 23**

Word.java 23

MoodBoosters.java 23

Workout.java 24

Study.java 24

Indian.java 25

Western.java 26

**REQUIREMENT 28**

**IMPLEMENTATION 29**

Tools and Languages Used 29

1. Platform used 29
2. Languages used 29
3. Software Used 29

Implementation Issues and Challenges 31

**MERITS 32**

**DEMERITS 32**

**FUTURE IMPROVEMENTS 33**

**CONCLUSION 34**

**SUPPLEMENTARY MATERIAL 35**

CODE SNAPSHOTS 35

1.Workout.java 35

1. AndroidManifest 36

2.MoodBooster.java 39

3. Indian.java 41

4.javaMusicList. 43

5.Word.java 47

6.WordAdapter.java 48

7.Splash.java 50

**REFERENCES AND BIBLIOGRAPHY 53**

### INTRODUCTION

RHYTHM, an app to match your rhythm. Rhythm mobile application is an android based application to listen to the music of your choice. Music is not only a simple sensation of sound, but also it's a rhythm of musical notes which affects our mood, feelings, attitude, personality, and our actions a lot. Songs don't only is the thing which we like to do when we are free, but it is a thing which we like to do to make ourselves feel light in our hectic schedules and affects us psychologically too. And this is the thing which we took into consideration and made our project RHYTHM. It is an app to make people feel relaxed and an application to make the match with your rhythm.

Moreover, the application which we developed is an easy-going app that most people can use efficiently and easily. We focused more on the easily accessible content of their choice for our users and that's why we made fewer and straightforward options to make it easy while using. This app facilitates a user in different ways in listening to music. We have 3 categories in which the first one we have 5 sub categories user can choose according to their choice , the second is to listen to music randomly i.e songs will play randomly and the third one is to listen to songs from your device according to your choice.On clicking the application, we kept an animation to make our application look more attractive. It will appear just after we click the app icon. After this, a screen with three options will open up from where we provided different ways to play songs.

We had kept an option for the user to randomize the songs. The reason for that is to make users listen to songs even in a situation where they find themselves confused regarding their choices in songs. As many times, people face this situation and it leads to not listening to any song. We have also added some variety of songs through our code so that if users don't have any songs on their device, then they can also listen to at least some songs whenever they feel like without. Our application can be used without an Internet connection which will add to the community of people who don't have an internet connection to their phones at that very instant. We have added the songs which are mostly liked on the internet so that most of our users would definitely like them. This application will collect all your device music files in one single place which sometimes becomes hectic work to manage.This is an application which not at all directly helps you, but yes, deep down, it will help you in every work which you do in an indirect way.

#### MOTIVATION

As we all know, in recent times, smartphones have become a very important part of our lives. People are using many applications on their smartphones according to their needs. Students are taking their school and college classes through it whereas a working person is now working from home with the help of their smartphone and other devices. So, this motivated us to work on this and to develop an application.

Everyone knows nowadays how busy our lifestyle has become. Presently, people are getting more and more stress due to this and also becoming mentally unstable. It doesn't matter who you are and how old you are because everyone's lives are getting affected a lot in this stressful environment. And the study says that listening to songs relaxes the human brain and improves our brain's functionality. So, when it came to choosing what application we will develop, we got answers from the above statements. Therefore, here we got out the motivation for our project application "RHYTHM". We wanted to develop something which can help a person in a stressful situation and that's how our motive got a clear way to our application "RHYTHM". And also, we wanted to focus on a few things, especially those two are written below because we didn't find these in any other applications. So, this motivated us too to focus more on:

1. making an offline application with a small size.
2. a situation where a person is confused in choosing a song to listen to.

After all this, we finally came up with an idea for an application like "RHYTHM" which we made in our project.

**.java Files Included**

##### Splash.java [Landing Screen]

**android.animation.Animator :**

This is the superclass for classes which provide basic support for animations which can be started, ended, and have AnimatorListeners added to them.

**android.content.Intent :**

An intent is an abstract description of an operation to be performed. It can be used with [startActivity](https://developer.android.com/reference/android/content/Context#startActivity(android.content.Intent)) to launch an [Activity](https://developer.android.com/reference/android/app/Activity), [broadcastIntent](https://developer.android.com/reference/android/content/Context#sendBroadcast(android.content.Intent)) to send it to any interested

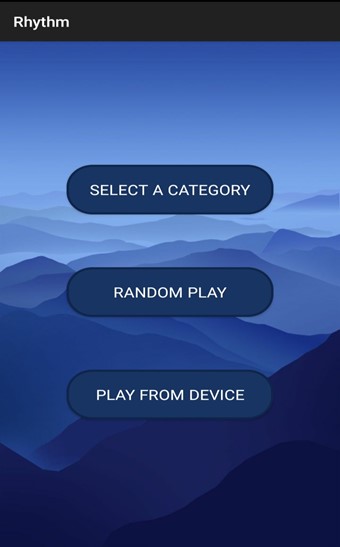
[BroadcastReceiver](https://developer.android.com/reference/android/content/BroadcastReceiver) components, and Context.startService (Intent) or

Context.bindService (Intent, ServiceConnection, int) to communicate with a background [Service](https://developer.android.com/reference/android/app/Service).



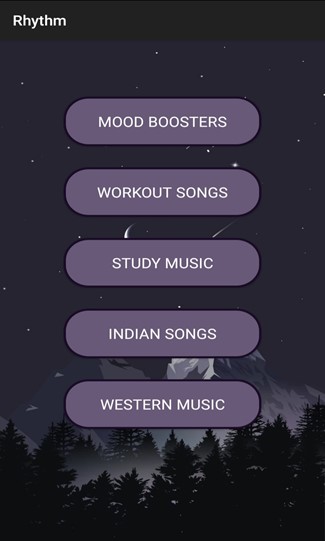
##### Choice.java

Next Activity to splash screen , with 3 options to be chosen from user whether the user wants to play songs category wise , random or directly from his/her device



##### Choice2.java

Activity with 5 different categories of songs to be selected from the user . This activity would come on clicking “SELECT A CATEGORY” from the previous activity



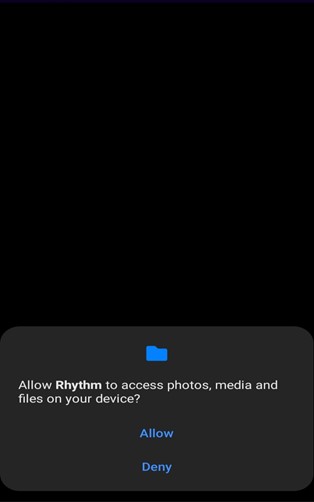
##### MainActivity.java

Opens when user clicks on the 2nd option provided on the 3 choice’s screen. In this activity we can play all the songs randomly with alternative clicks which is good in the situation when the user can’t decide any particular category to play.

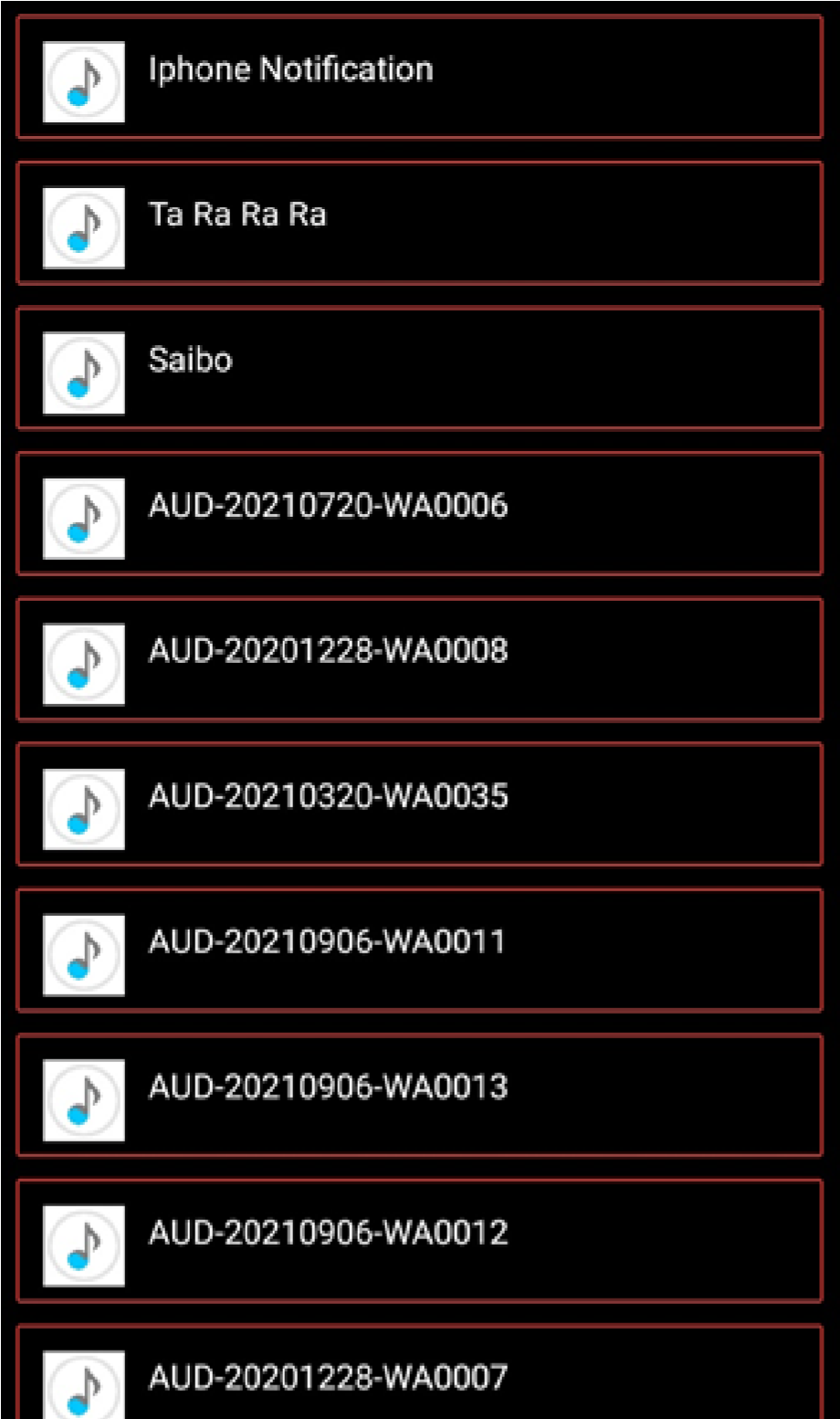


##### MusicList.java

Opens when the user clicks on the 3rd option provided on the 3 choice’s screen. On clicking , it would ask for the user’s internal storage permission to get mp3 and wav files.



In this activity we can play songs which are already there in user’s device , it makes it easy to find all mp3 ,audio files and wav files to play at single place which makes it convenient for the user.



##### PlayerActivity.java

This Activity is mainly designed to take the user to a beautiful visualized screen on selecting a song from the list of user’s device songs from the MusicList.java activity.

###### LIBRARIES INCLUDED

**androidx.annotation.NonNull :**

Denotes that a parameter, field or method return value can never be null.This is a marker annotation and it has no specific attributes.

**androidx.annotation.RequiresApi :**

Denotes that the annotated element should only be called on the given API level or higher.This is similar in purpose to the older {@code @TargetApi} annotation, but more clearly expresses that this is a requirement on the caller, rather than being used to "suppress" warnings within the method that exceed the { @code minSdkVersion} .

**android.content.Context :**

Interface to global information about an application environment. This is an abstract class whose implementation is provided by the Android system. It allows access to application-specific resources and classes, as well as up-calls for application-level operations such as launching activities, broadcasting and receiving intents, etc.

**android.content.Intent :**

An intent is an abstract description of an operation to be performed. It can be used with startActivity to launch an Activity, broadcastIntent to send it to any interested

BroadcastReceiver components, and Context.startService(Intent) or

Context.bindService(Intent, ServiceConnection, int) to communicate with a background Service.

An Intent provides a facility for performing late runtime binding between the code in different applications. Its most significant use is in the launching of activities, where it can be thought of as the glue between activities. It is basically a passive data structure holding an abstract description of an action to be performed.

Some examples of action/data pairs are:

* [ACTION\_VIEW](https://developer.android.com/reference/android/content/Intent#ACTION_VIEW) *content://contacts/people/1* -- Display information about the person whose identifier is "1".
* [ACTION\_DIAL](https://developer.android.com/reference/android/content/Intent#ACTION_DIAL) *content://contacts/people/1* -- Display the phone dialer with the person filled in.
* [ACTION\_VIEW](https://developer.android.com/reference/android/content/Intent#ACTION_VIEW) *tel:123* -- Display the phone dialer with the given number filled in. Note how the VIEW action does what is considered the most reasonable thing for a particular URI.
* [ACTION\_DIAL](https://developer.android.com/reference/android/content/Intent#ACTION_DIAL) *tel:123* -- Display the phone dialer with the given number filled in.
* [ACTION\_EDIT](https://developer.android.com/reference/android/content/Intent#ACTION_EDIT) *content://contacts/people/1* -- Edit information about the person whose identifier is "1".
* [ACTION\_VIEW](https://developer.android.com/reference/android/content/Intent#ACTION_VIEW) *content://contacts/people/* -- Display a list of people, which the user can browse through. This example is a typical top-level entry into the Contacts application, showing you the list of people. Selecting a particular person to view would result in a new intent { ACTION\_VIEW *content://contacts/people/N* } being used to start an activity to display that person.

**android.media.AudioAttributes :**

A class to encapsulate a collection of attributes describing information about an audio stream.

AudioAttributes supersede the notion of stream types (see for instance

AudioManager#STREAM\_MUSIC or AudioManager#STREAM\_ALARM) for defining the behavior of audio playback. Attributes allow an application to specify more information than is conveyed in a stream type by allowing the application to define:

* usage: "why" you are playing a sound, what is this sound used for. This is achieved with the "usage" information. Examples of usage are USAGE\_MEDIA and USAGE\_ALARM. These two examples are the closest to stream types, but more detailed use cases are available. Usage information is more expressive than a stream type, and allows certain platforms or routing policies to use this information for more refined volume or routing decisions. Usage is the most important information to supply in AudioAttributes and it is recommended to build any instance with this information supplied, see AudioAttributes.Builder for exceptions.
* content type: "what" you are playing. The content type expresses the general category of the content. This information is optional. But in case it is known (for instance CONTENT\_TYPE\_MOVIE for a movie streaming service or

CONTENT\_TYPE\_MUSIC for a music playback application) this information might be used by the audio framework to selectively configure some audio post-processing blocks.

* flags: "how" is playback to be affected, see the flag definitions for the specific playback behaviors they control.

**android.media.AudioFocusRequest :**

A class to encapsulate information about an audio focus request. An

AudioFocusRequest instance is built by Builder, and is used to request and abandon audio focus, respectively with AudioManager#requestAudioFocus(AudioFocusRequest) and AudioManager#abandonAudioFocusRequest(AudioFocusRequest).It is passed when building an AudioFocusRequest instance with its builder in the Builder constructor

AudioFocusRequest.Builder#Builder(int), or with

AudioFocusRequest.Builder#setFocusGain(int) after copying an existing instance with AudioFocusRequest.Builder#Builder(AudioFocusRequest)**.**

**android.media.MediaPlayer :**

MediaPlayer class is used to control playback of audio/video files and streams.

**android.media.RemoteControlClient :**

RemoteControlClient enables exposing information meant to be consumed by remote controls capable of displaying metadata, artwork and media transport control buttons.A remote control client object is associated with a media button event receiver. This event receiver must have been previously registered with

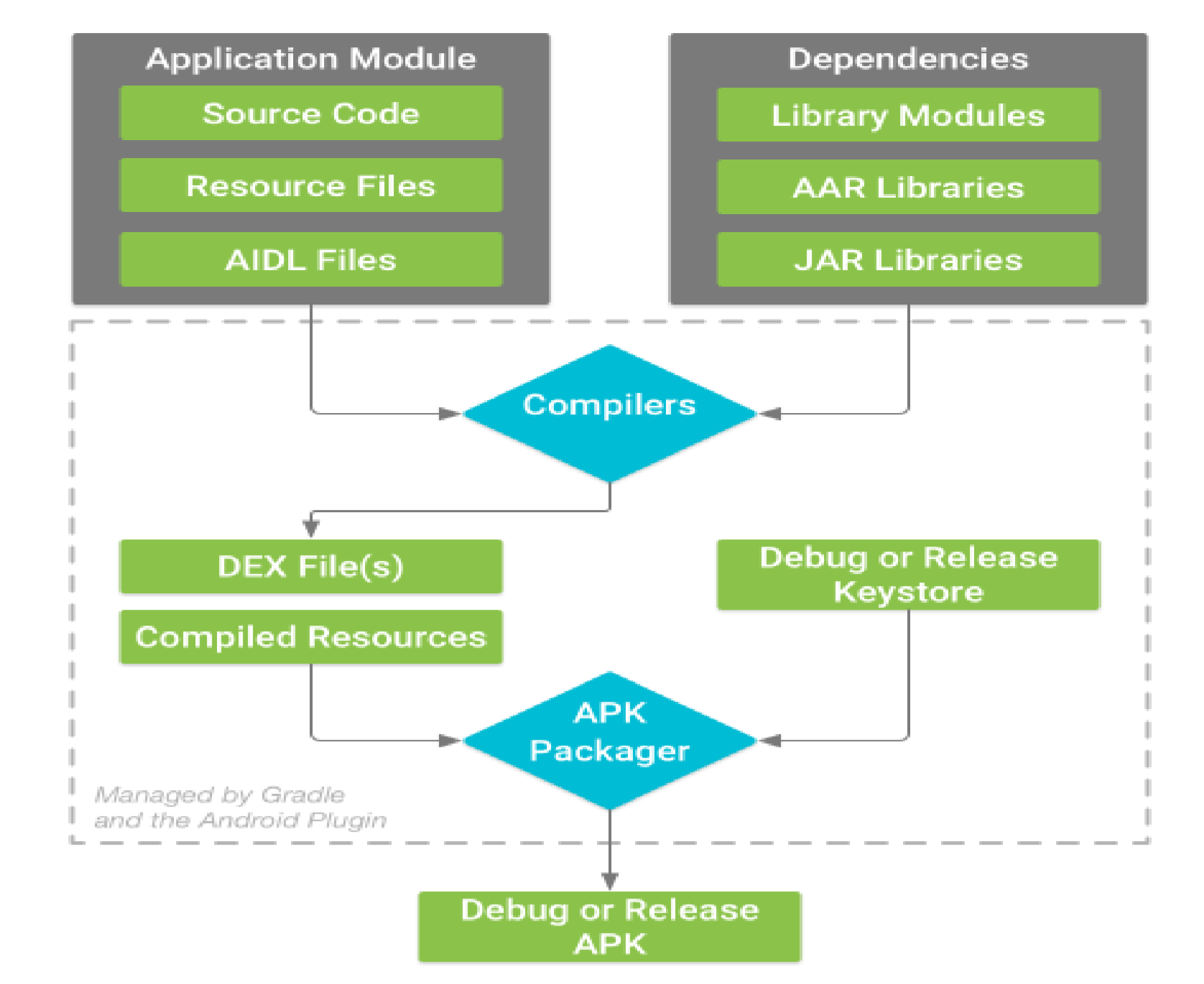
AudioManager#registerMediaButtonEventReceiver(ComponentName) before the

RemoteControlClient can be registered through

AudioManager#registerRemoteControlClient(RemoteControlClient).

**android.os.Build :**

The build process involves many tools and processes that convert your project into an Android Application Package (APK) or Android App Bundle (AAB). The build process is very flexible, so it's useful to understand some of what is happening under the hood.



**android.os.Bundle :**

Bundle is a utility class that lets you store a set of name-value pairs. You will always find this import along with the import for Activity class because both onCreate() and onFreeze() methods take Bundle as a parameter. Into a Bundle object, you can put integers, longs, strings, arrays, etc along with the keys to identify them. When needed, these values can be obtained by using those keys.So, as you can see, bundle object is ideal for storing a set of state variables in the onFreeze method; and the same state variables can be read back in the onCreate method.

**android.os.Handler :**

A Handler allows to send and process [Message](https://developer.android.com/reference/android/os/Message) and Runnable objects associated with a thread's [MessageQueue](https://developer.android.com/reference/android/os/MessageQueue). Each Handler instance is associated with a single thread and that thread's message queue. When we create a new Handler it is bound to a [Looper](https://developer.android.com/reference/android/os/Looper). It will deliver messages and runnables to that Looper's message queue and execute them on that Looper's thread.There are two main uses for a Handler: (1) to schedule messages and runnables to be executed at some point in the future; and (2) to enqueue an action to be performed on a different thread than your own.

**android.view.MenuItem :**

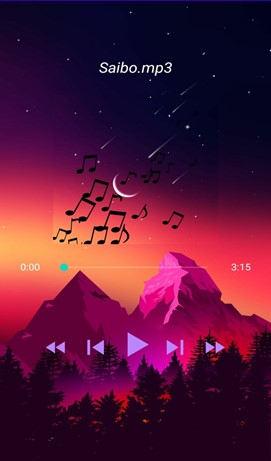
Interface for direct access to a previously created menu item.An Item is returned by calling one of the [Menu.add(int)](https://developer.android.com/reference/android/view/Menu#add(int)) methods.

**android.widget.SeekBar :**

A SeekBar is an extension of ProgressBar that adds a draggable thumb. The user can touch the thumb and drag left or right to set the current progress level or use the arrow keys. Placing focusable widgets to the left or right of a SeekBar is discouraged.Clients of the SeekBar can attach a [SeekBar.OnSeekBarChangeListener](https://developer.android.com/reference/android/widget/SeekBar.OnSeekBarChangeListener) to be notified of the user's actions.

**android.widget.TextView :**

A user interface element that displays text to the user.



###### List.java

This file is working as a blueprint or parent file including the designing part and the required java code for the MoodBooster.java , Workout.java , Study.java , Indian.java , Western.java .

android.media.AudioManager

AudioManager provides access to volume and ringer mode control.

android.media.MediaPlayer;

The MediaPlayer class can be used to control playback of audio/video files and streams.

MediaPlayer is not thread-safe. Creation of and all access to player instances should be on the same thread. If registering [callbacks](https://developer.android.com/reference/android/media/MediaPlayer#Callbacks), the thread must have a Looper.

android.net.Uri;

Immutable URI reference. A URI reference includes a URI and a fragment, the component of the URI following a '#'. Builds and parses URI references which conform to [RFC 2396](http://www.faqs.org/rfcs/rfc2396.html).

In the interest of performance, this class performs little to no validation. Behavior is undefined for invalid input. This class is very forgiving--in the face of invalid input, it will return garbage rather than throw an exception unless otherwise specified.

android.widget.GridView;

A view that shows items in a two-dimensional scrolling grid. The items in the grid come from the [ListAdapter](https://developer.android.com/reference/android/widget/ListAdapter) associated with this view.

android.widget.ListView;

Displays a vertically-scrollable collection of views, where each view is positioned immediately below the previous view in the list.

android.widget.Toast;

A toast is a view containing a quick little message for the user. The toast class helps you create and show those.

When the view is shown to the user, it appears as a floating view over the application. It will never receive focus. The user will probably be in the middle of typing something else. The idea is to be as unobtrusive as possible, while still showing the user the information you want them to see. Two examples are the volume control, and the brief message saying that your settings have been saved.

###### WordAdapter.java

This file code includes that how we set images, the songs , their text display and place them in correct order using ArrayAdapter.

**android.app.Activity :**

An activity is a single, focused thing that the user can do. Almost all activities interact with the user, so the Activity class takes care of creating a window for you in which you can place your UI with setContentView(View). While activities are often presented to the user as full-screen windows, they can also be used in other ways: as floating windows (via a theme with R.attr.windowIsFloating set), Multi-Window mode or embedded into other windows.

**android.view.LayoutInflater :**

Instantiates a layout XML file into its corresponding [View](https://developer.android.com/reference/android/view/View) objects. It is never used directly.

Instead, use [Activity.getLayoutInflater()](https://developer.android.com/reference/android/app/Activity#getLayoutInflater()) or [Context#getSystemService](https://developer.android.com/reference/android/content/Context#getSystemService(java.lang.Class%3CT%3E)) to retrieve a standard LayoutInflater instance that is already hooked up to the current context and correctly configured for the device you are running on.

android.view.View;

This class represents the basic building block for user interface components. A View occupies a rectangular area on the screen and is responsible for drawing and event handling. View is the base class for *widgets*, which are used to create interactive UI components (buttons, text fields, etc.). The [ViewGroup](https://developer.android.com/reference/android/view/ViewGroup) subclass is the base class for *layouts*, which are invisible containers that hold other Views (or other ViewGroups) and define their layout properties.

**android.view.ViewGroup :**

A ViewGroup is a special view that can contain other views (called children.) The view group is the base class for layouts and views containers. This class also defines the

[ViewGroup.LayoutParams](https://developer.android.com/reference/android/view/ViewGroup.LayoutParams) class which serves as the base class for layouts parameters.

android.widget.ArrayAdapter;

You can use this adapter to provide views for an [AdapterView](https://developer.android.com/reference/android/widget/AdapterView), Returns a view for each object in a collection of data objects you provide. By default, the array adapter creates a view by calling [Object#toString()](https://developer.android.com/reference/java/lang/Object#toString()) on each data object in the collection you provide, and places the result in a TextView. You may also customize what type of view is used for the data object in the collection.

#### Features and files

**MoodBoosters.java**

Opens on the 1st option click of the 2nd choice screen containing 5 options

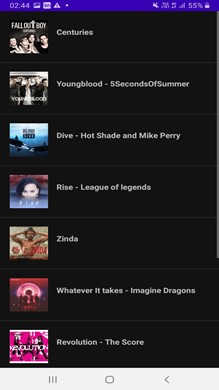
In this activity user would be able to boost up the mood with famous MoodBooster songs.



##### Workout.java

Opens on the 2nd option click of the 2nd choice screen containing 5 options

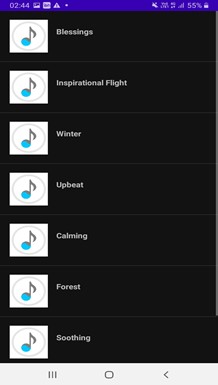
Activity with famous , highly motivating workout songs , help users to get motivated over the period of their workout or whenever feeling low.



##### Study.java

Opens on the 3rd option click of the 2nd choice screen containing 5 options

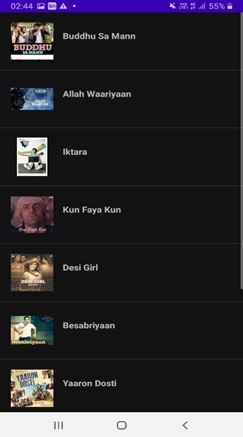
This activity mainly helps the students while studying as it consists of very cool and calm songs , helps in stability of mind while getting in stress during study.



##### Indian.java

Opens on the 4th option click of the 2nd choice screen containing 5 options

This activity consists of famous indian songs including punjabi and party songs , helps to listen most of indian famous songs at a single place.

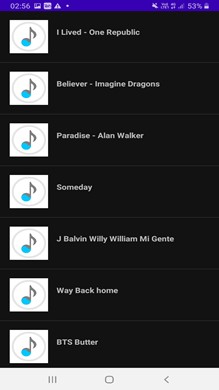


##### Western.java

Opens on the 5th option click of the 2nd choice screen containing

5 options

Perfect place for those having great interest and madness for the western songs , can find famous western songs at a single place with more convenience.



**REQUIREMENTS**

**Software Requirements**

These requirements are separated based on whether you are developing the app or running the app on a device.

**For development:**

Operating System: Windows XP or higher/Mac OS X 15.8 or later/Linux Platform: Android SDK

Framework 16 or Higher Tools: Android Studio 4.2.2

Technologies used: Java, Android, Android Emulator: API level 29 or higher.

**For running on a device:**

Operating System: Android 5 or higher Cellular capabilities.

**Hardware Requirements:**

* For development:

Processor: Intel Pentium IV or higher RAM:256MB

Space on disk: 250 MB (at the least)

* For running on a device:

Device: Phone or tablet running Android 4.0 or higher Disk space: 42 MB(at the least 29)

## IMPLEMENTATION

### Tools and Languages Used

1. **Platform used**

Android

The proposed system is a mobile application. Therefore the android platform was chosen to become the system platform that we used in this system. Android is a famous mobile platform and people are well known about it. It is a linux kernel based mobile platform, the reason to choose android is because android mobile apps can fit in many types of smartphone.

1. **Languages used :**

**JAVA**

The proposed system is in android platform. Therefore, Java will be used in its development. Java is an object-oriented programming language that is used to develop various types of systems.

Nowadays, most of the systems are using Java programming language to develop. The reason to use Java to develop the system is because most of the android applications are developed using Java and using its object-oriented features.

**XML**

Android application requires xml language to design the layout and graphic user interface of the mobile application. The layout size, text, color, background and so on is required to use xml language in order to make changes.

1. **Software Used**

**Android Studio**

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.

Android smartphones(version 11 ) and Android SDK emulator that used to debug and test the functionality of the mobile application.

### Implementation Issues and Challenges

*During the project development, we faced some challenges and issues also. The main issue was that we are doing this project in a team of four members and due to online mode we faced a lot of issues communicating and coordinating with each other though we had many ways to communicate but that was not as effective as we wanted .*

*The first challenge and issue was the ANDROID DEVELOPMENT projects as this was new for us and not yet studied in our syllabus but we kept the enthusiasm to learn and explore things in making projects making sure that it should have user benefits in real world .We searched and did lot of research in learning the android development .Next was with xml language, it is a programming language that we were not so familiar. But xml was included in the development, this is because most of the IDE’s in android development use xml to develop different kinds of apps. So in order to use it, we needed to learn this language in order to design a good user interface.*

*Besides, the second was mobile app bugs, errors not compatible with some mobile*

*devices problems. If we find a bug during the development of apps, we need to spend almost two or three hours looking for the solution to solve the bug. Sometimes for two day, we still cannot find any solution. So, this might be one of the reasons to slow down our project’s progress. In addition, the virtual machine of android that is used to display the outcome is slow. For example, when we finished developing the apps then we wanted to test the apps. After running the apps we need to wait some time to get the output,or the emulator not running properly .*

### MERITS

1.This application is an offline application , the user doesn't need to have an internet connection and can listen to his/her favourite music without any interruption.

1. Users can listen to music depending on the current mood as this application providesa category of music.
2. If User is confused in music taste , can choose the random feature of the applicationand listen to variety of songs randomly
3. Users can play the songs and audio files which are already downloaded in the user'sdevice(phone) in case the user can’t find the exact place of the downloaded songs in the device.

#### DEMERITS

1.This application does not store any data of the user including their personal info like phone no and mail id .

1. This application can’t show the previous status of the user .
2. Since this is an offline application, it has limited songs only.

### FUTURE IMPROVEMENTS

1. Firebase can be attached in order to store user’s data and previous status , it makes this application more flexible to use.
2. Songs can be displayed as per user’s age .
3. Recommendations can be given to the user , depending on the user’s history and their taste.
4. Internet permission can be added in order to explore through a sea of songs.
5. Application look / design can be improved.

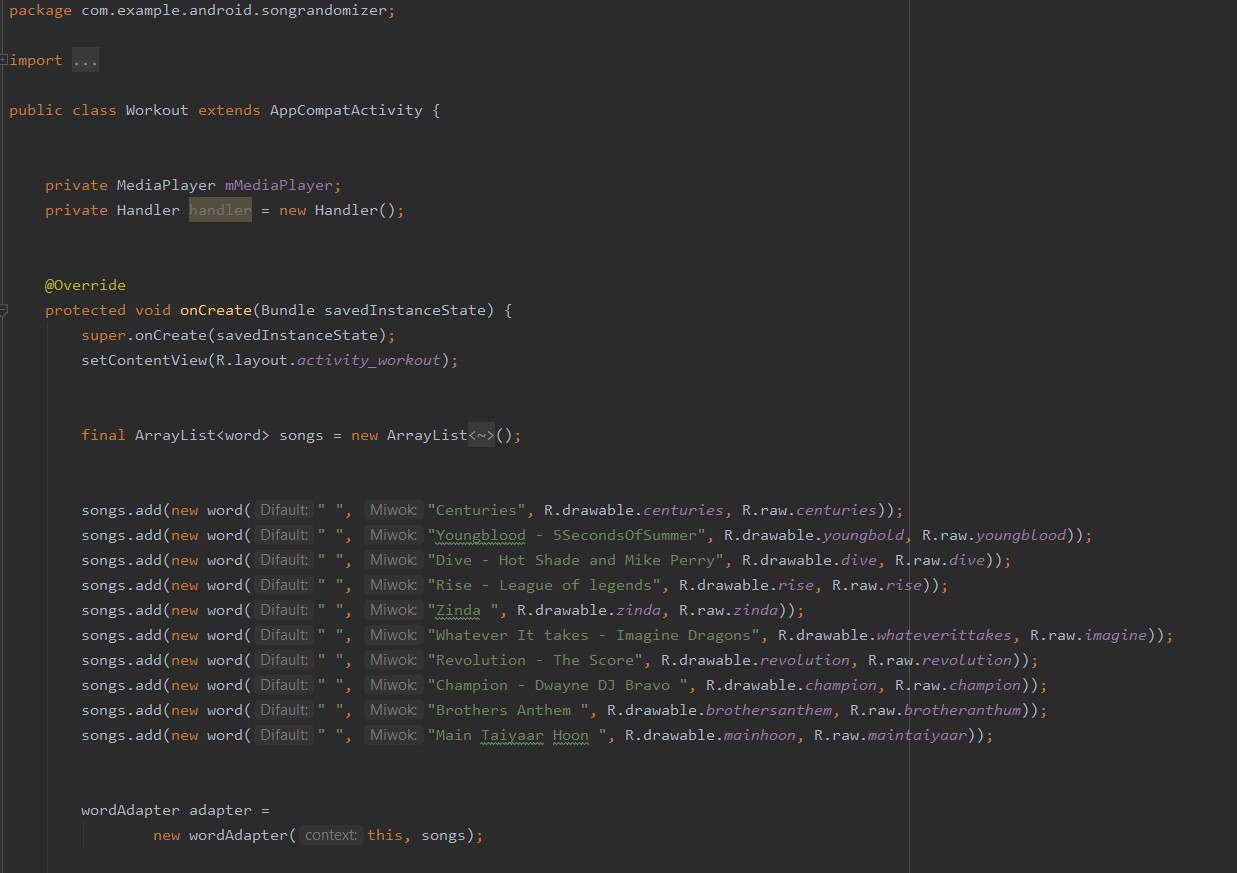
**CONCLUSION**

1. This application is developed for entertainment purposes for the users.
2. No Internet Connection is required.
3. This Application consists of songs in order to cover maximum users with a varietyof songs.
4. This application will collect all your device music files in one single place whichsometimes becomes hectic work to manage.
5. This is an application which not at all directly helps you, but yes, deep down, itwill help you in every work which you do in an indirect way.
6. If a user is confused about the type of song they want to listen, then the randomfeature of the application will be of help for such a user.
7. Users can play the songs and audio files which are already downloaded in theuser's device (phone) in case the user can’t find the exact place of the downloaded songs in the device.

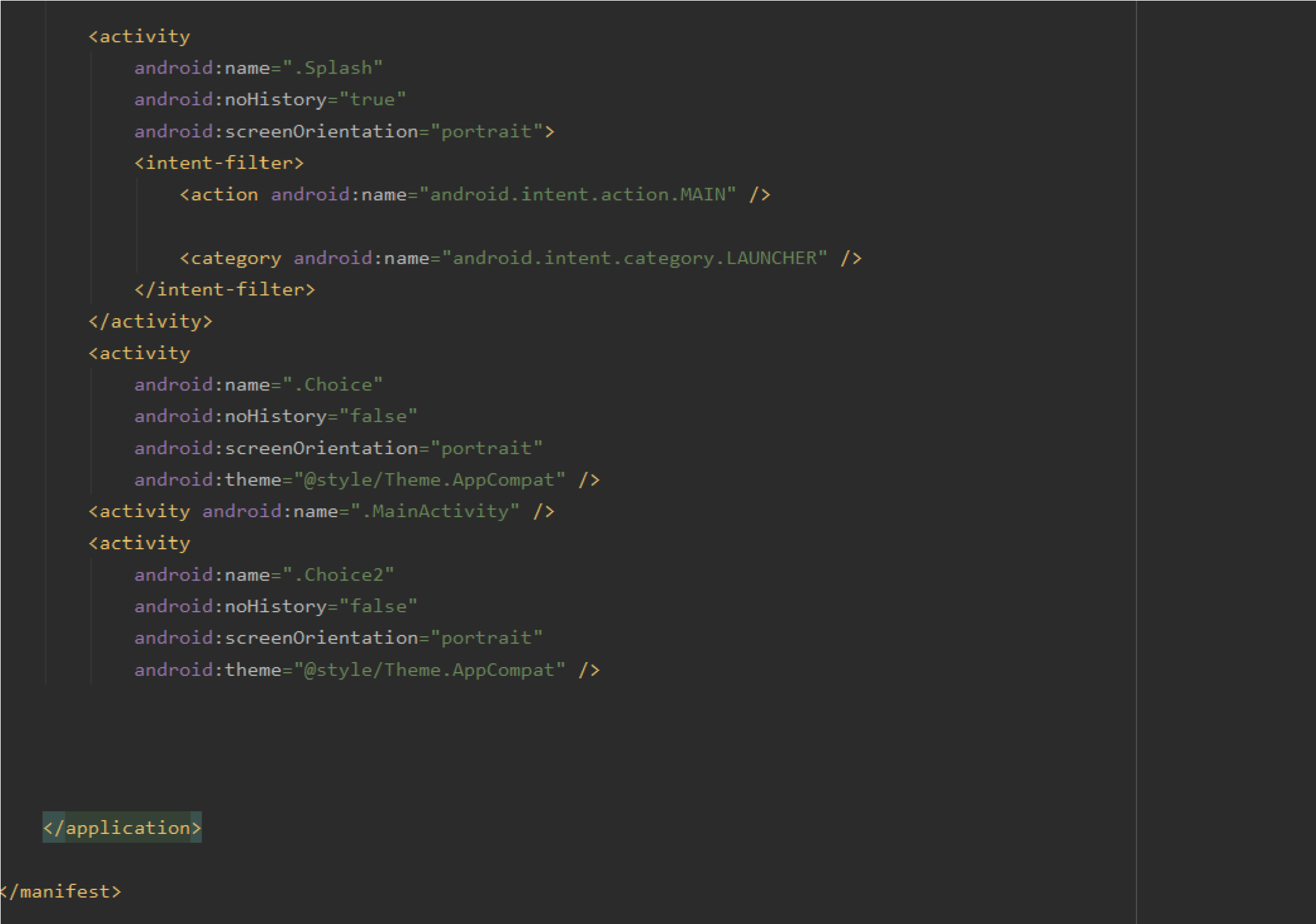
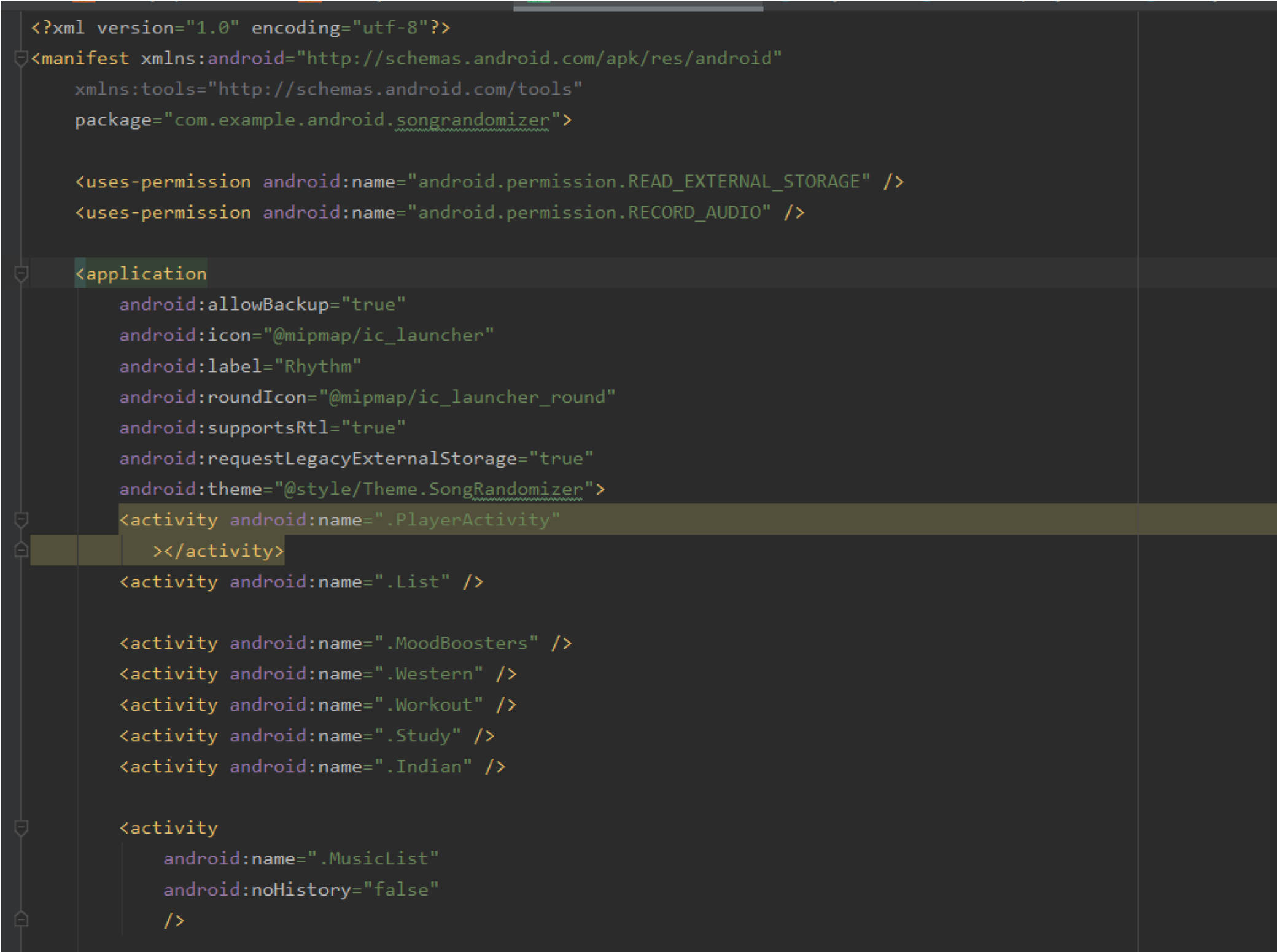
### SUPPLEMENTARY MATERIAL

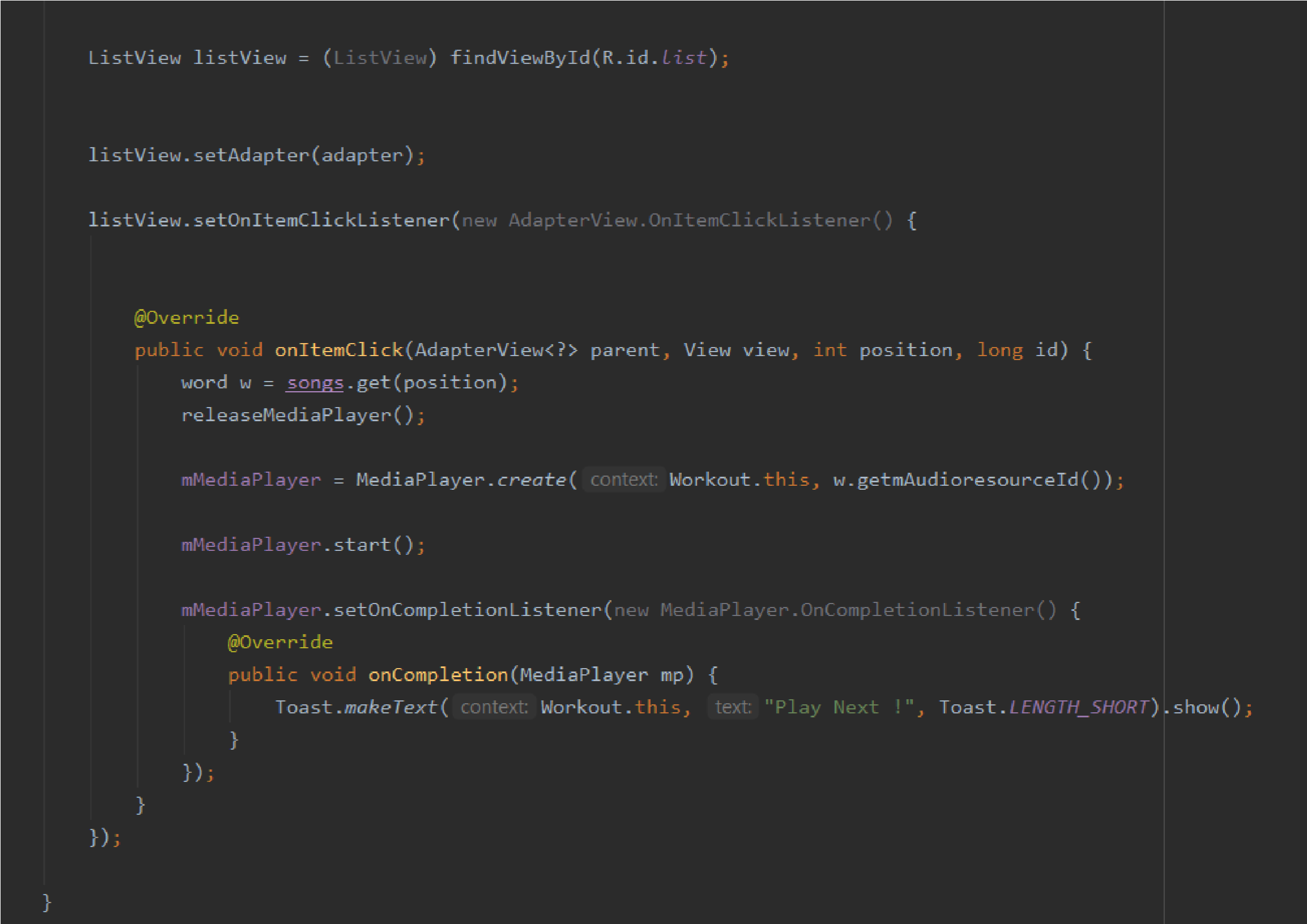
**CODE SNAPSHOTS**

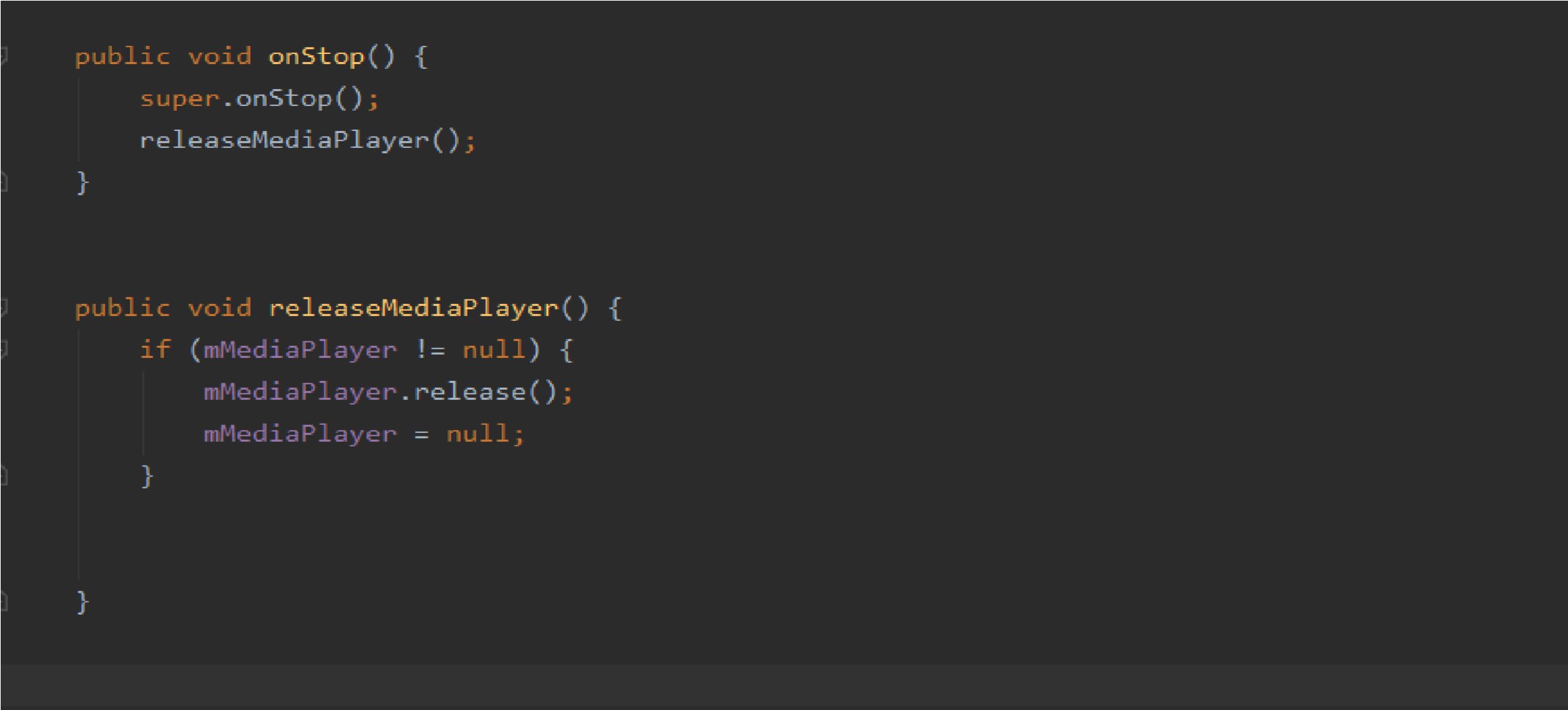
**1.Workout.java**



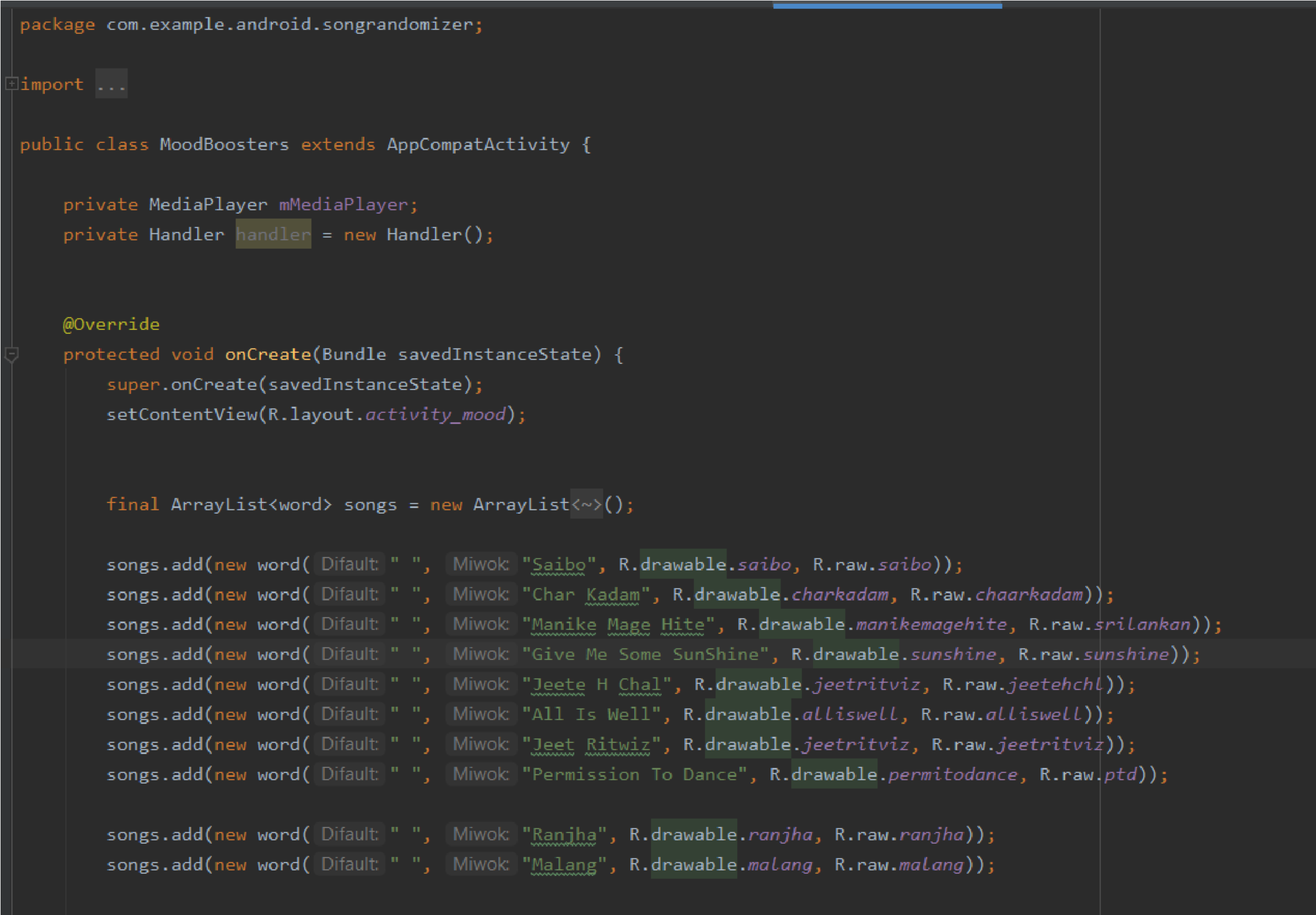
**4. AndroidManifest**

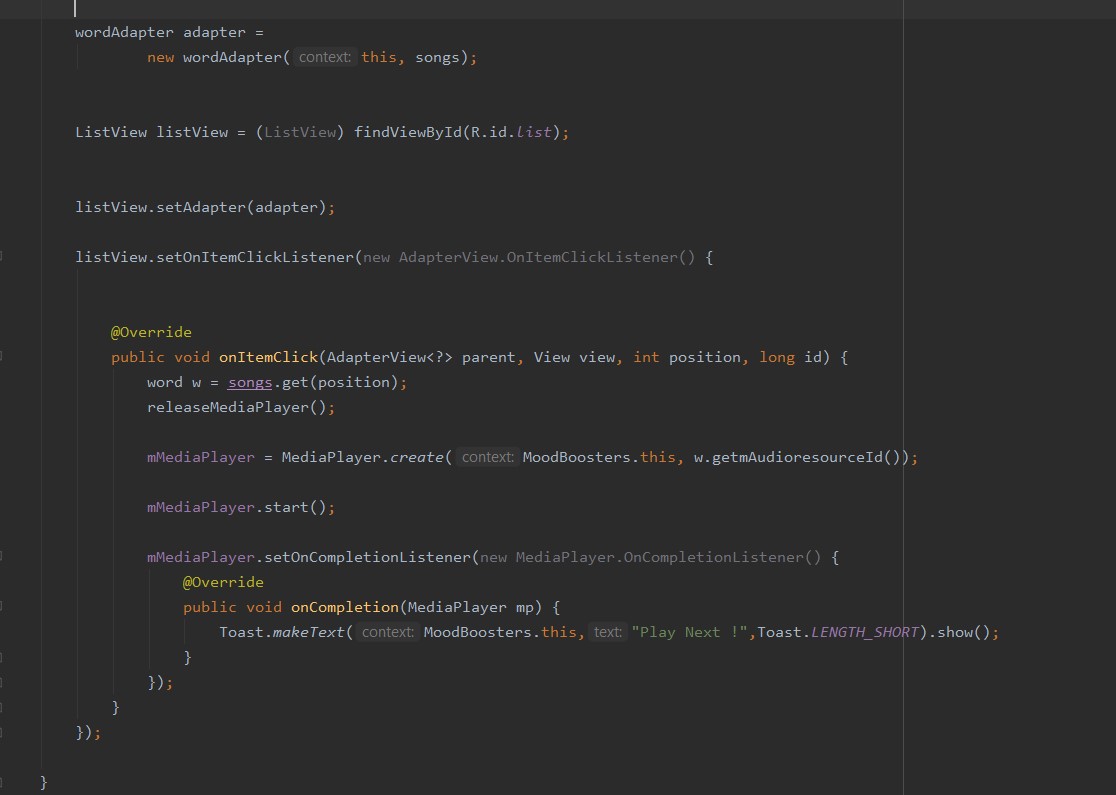




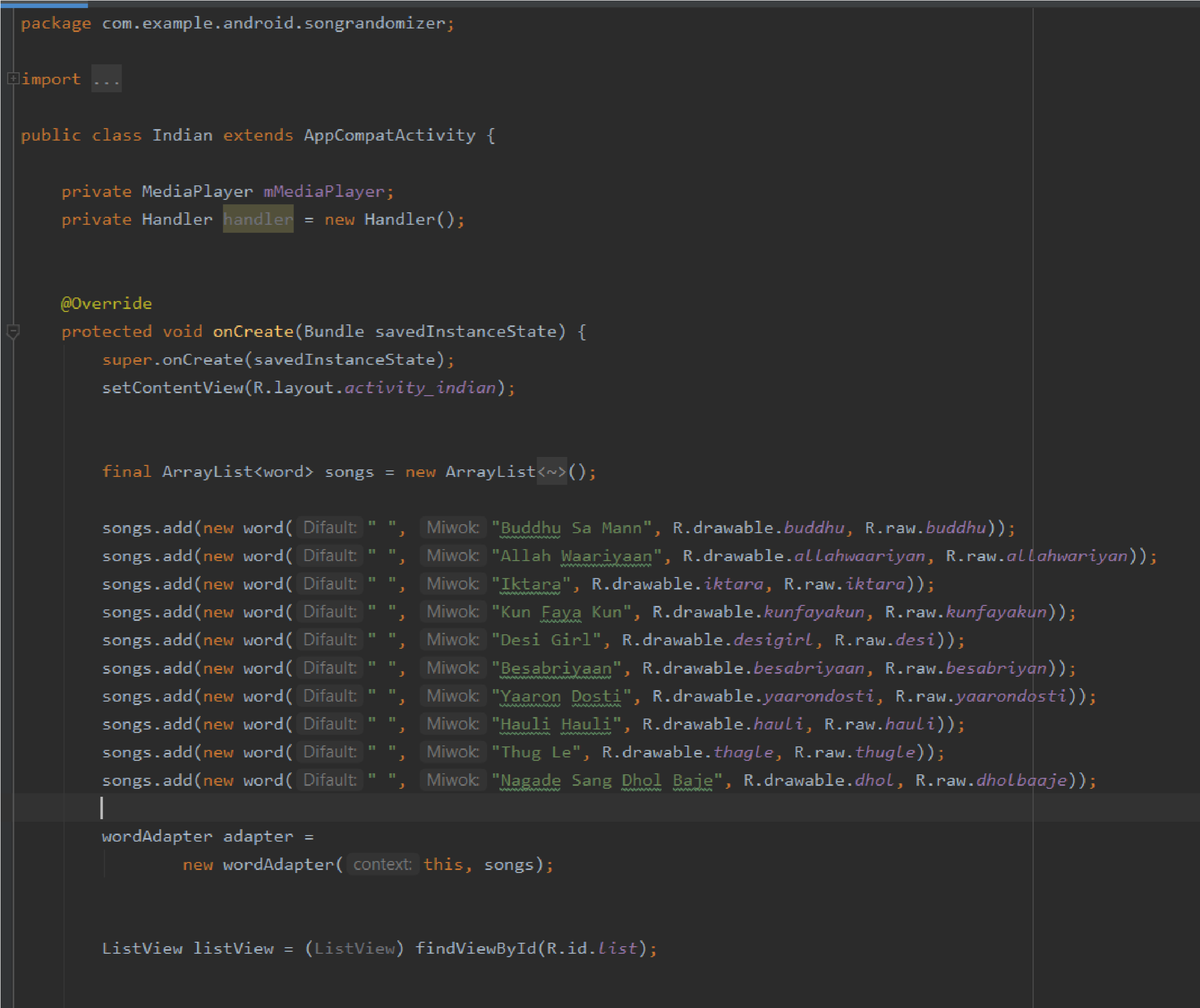


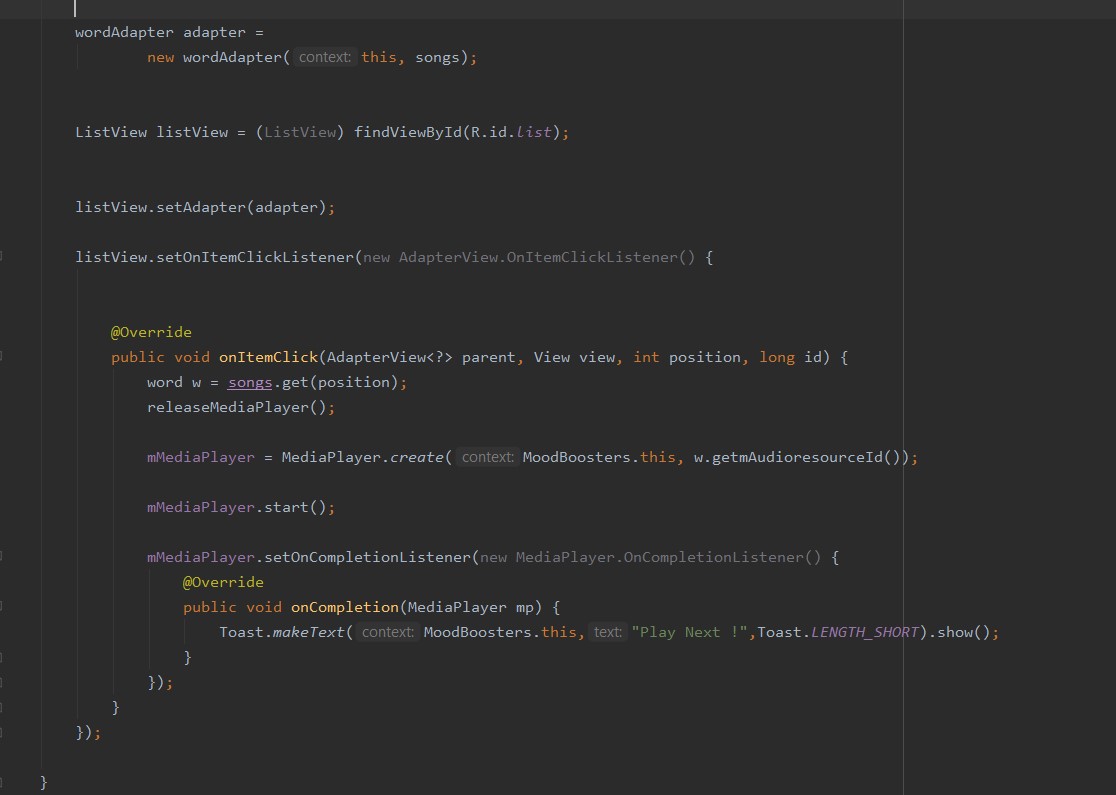
#### 2.MoodBooster.java



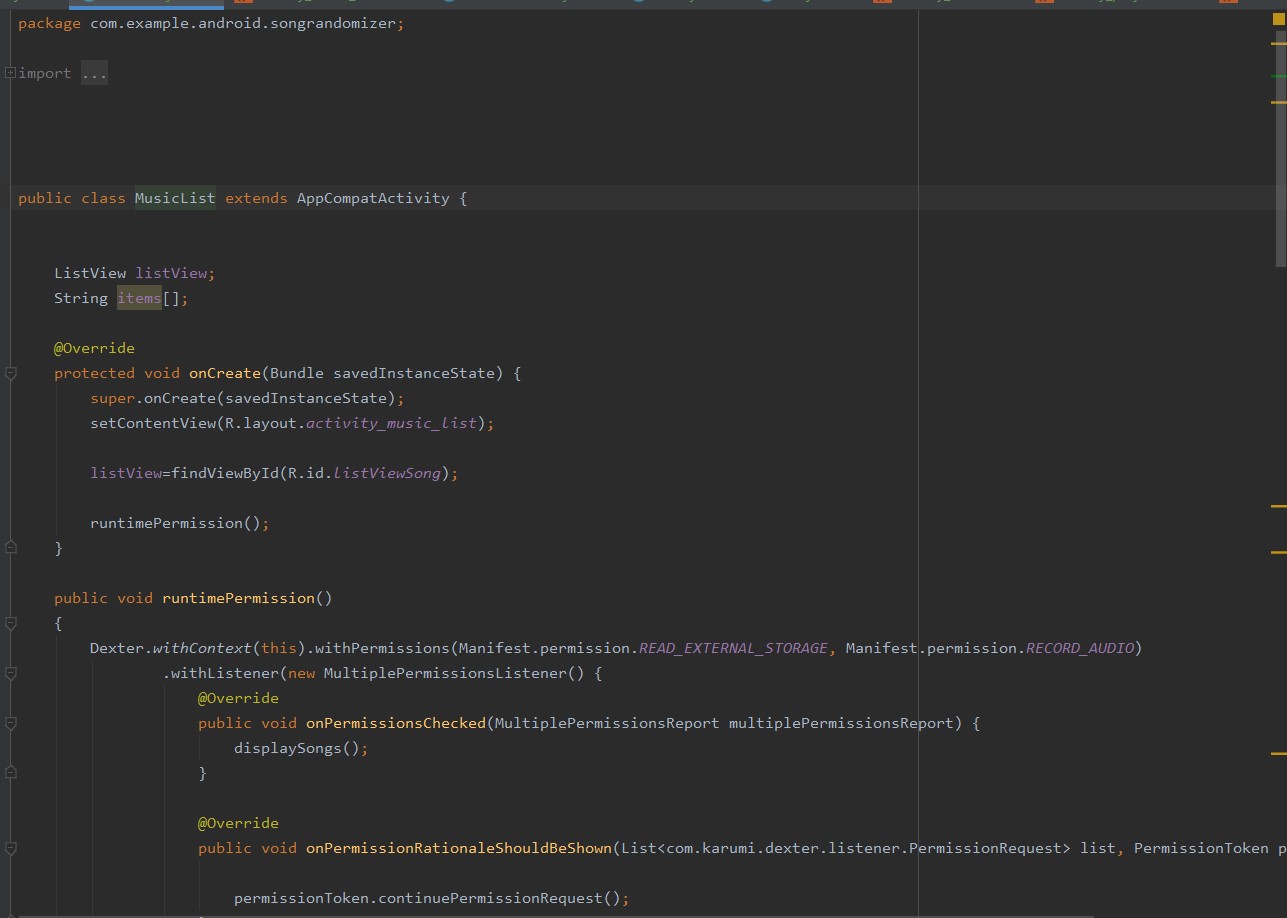


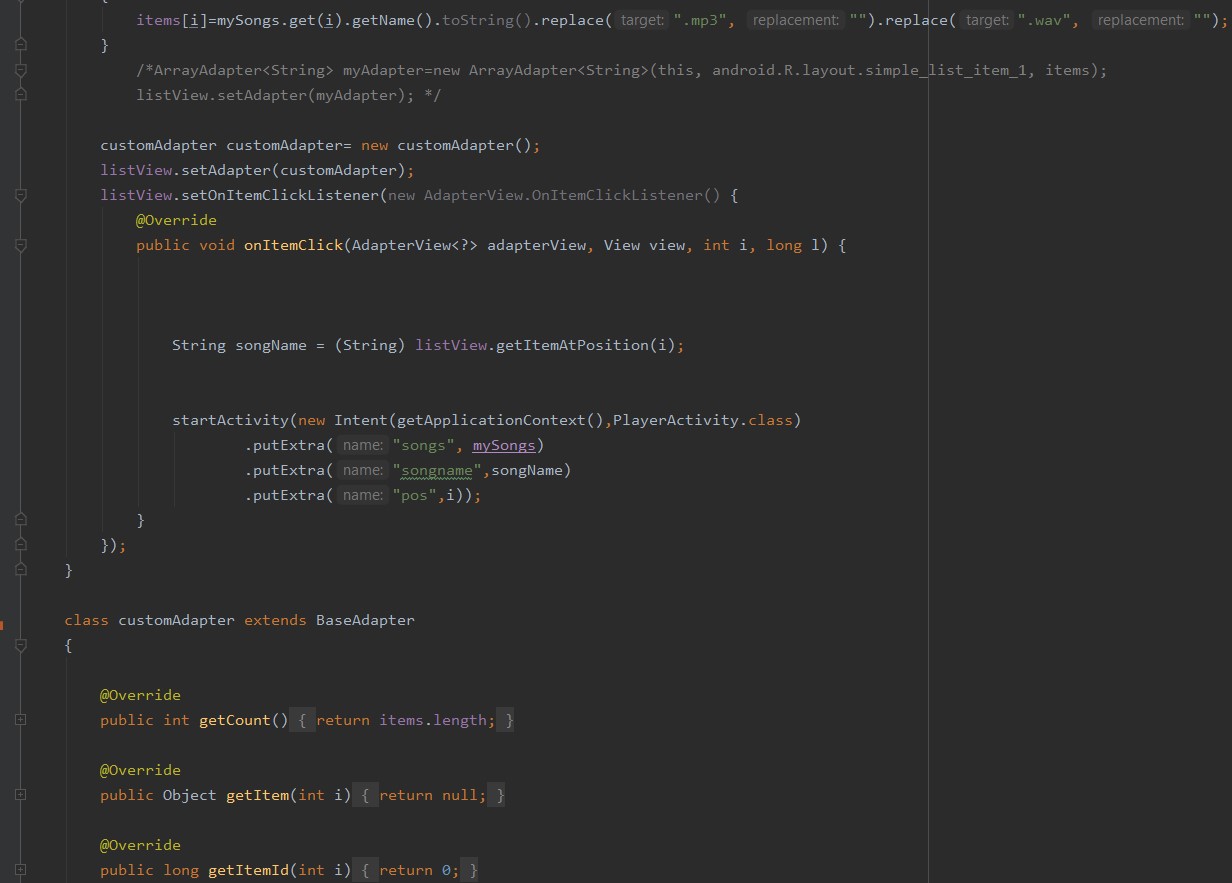
#### 3. Indian.java

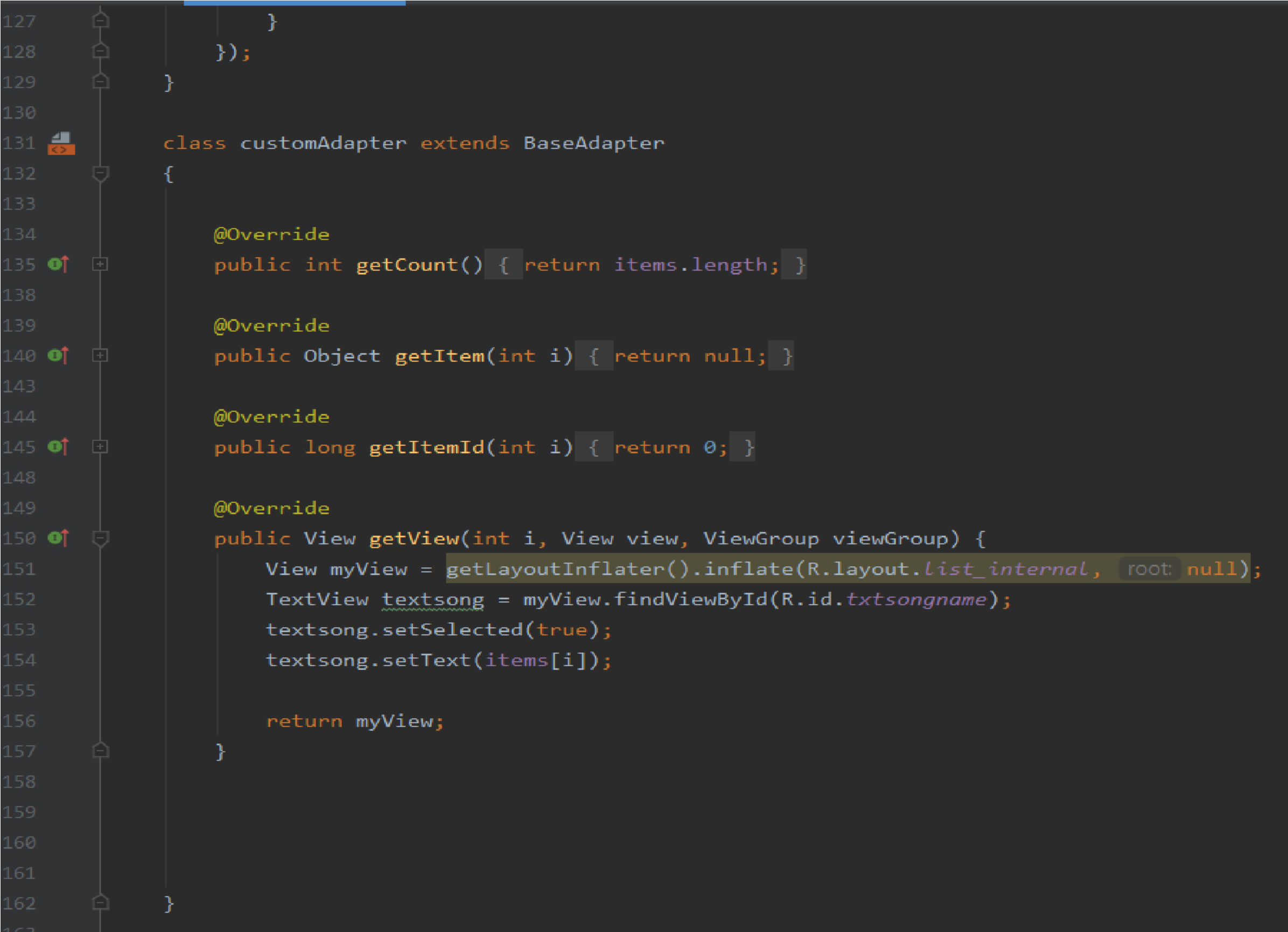




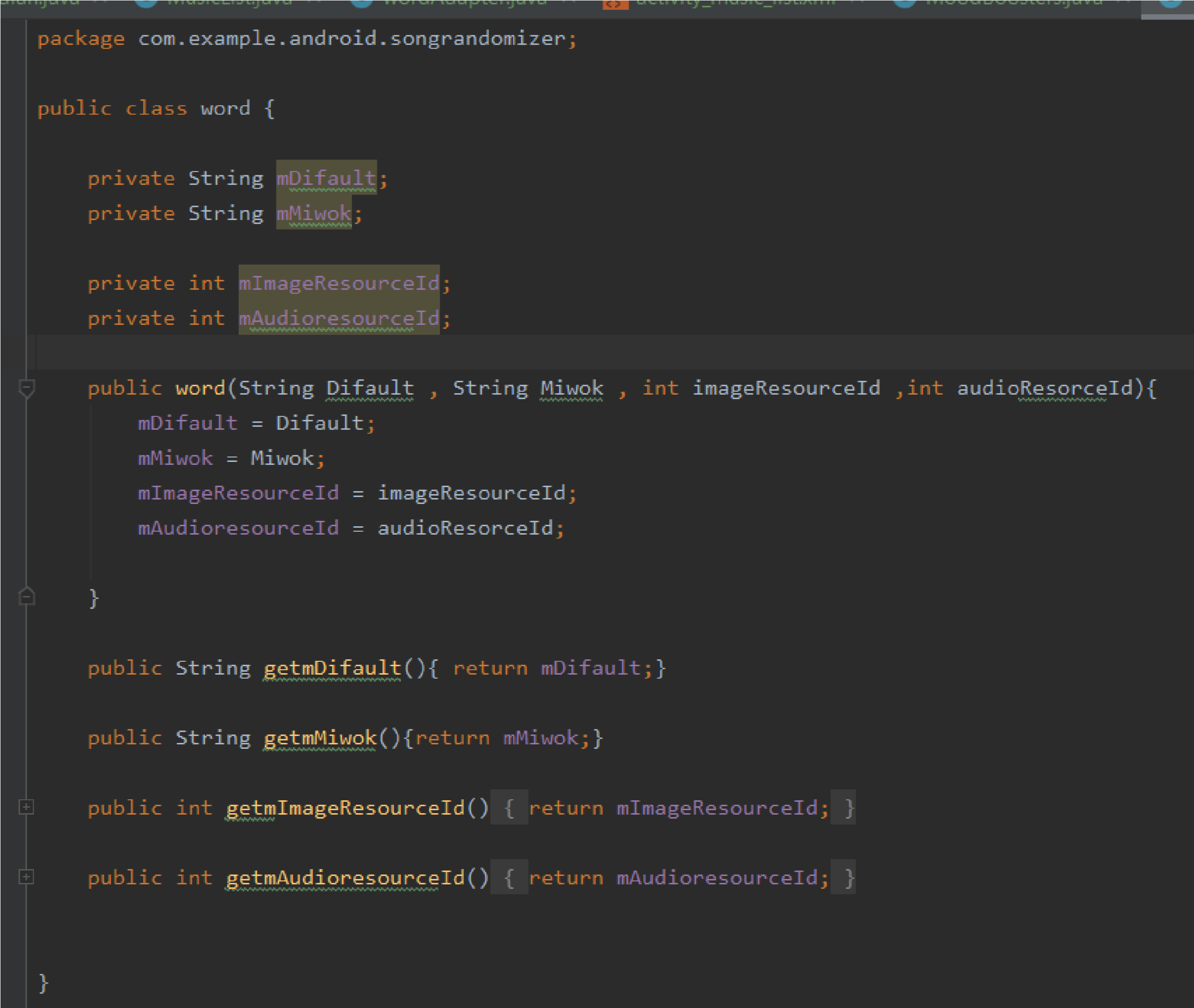
#### 4.javaMusicList



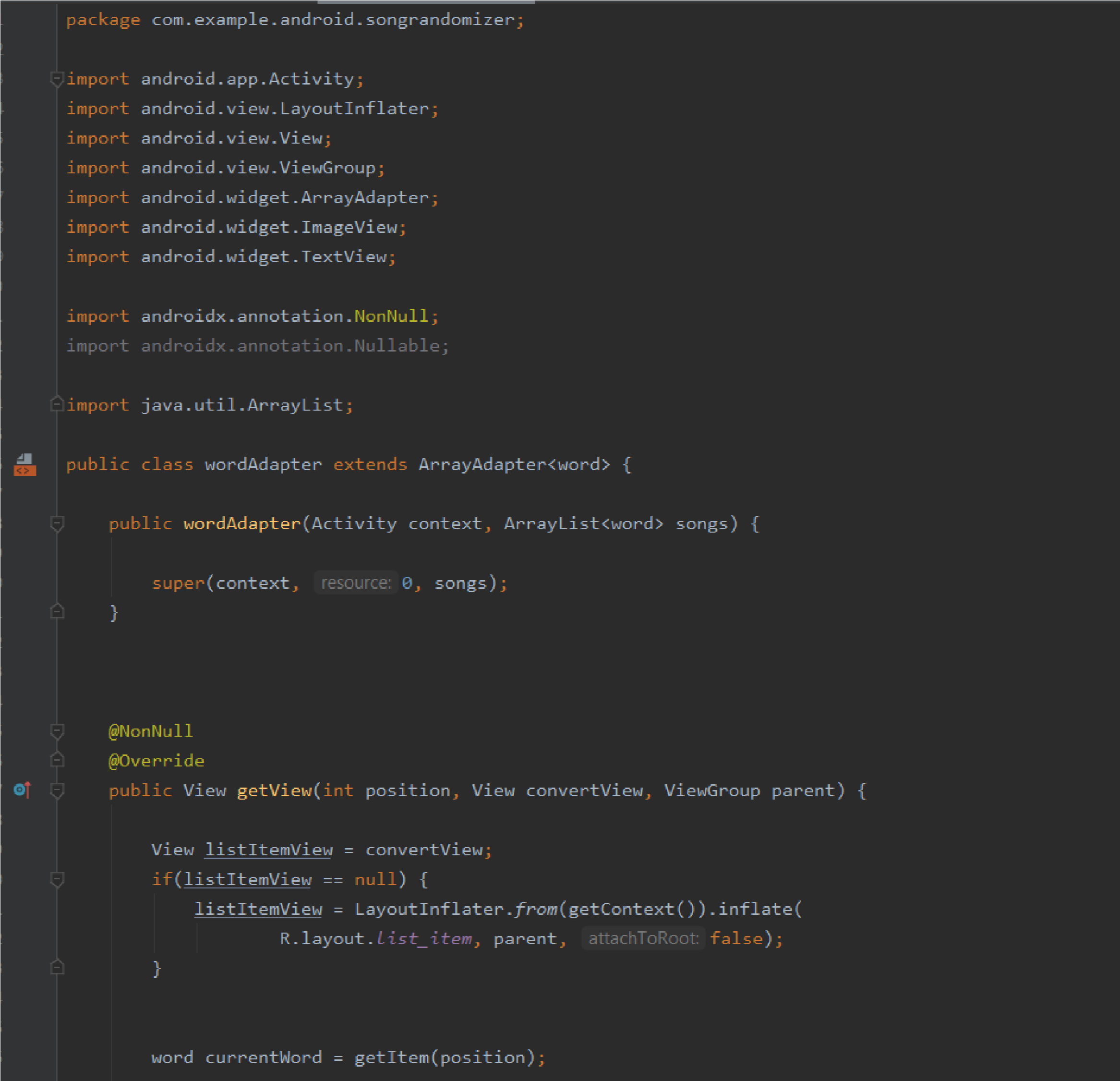


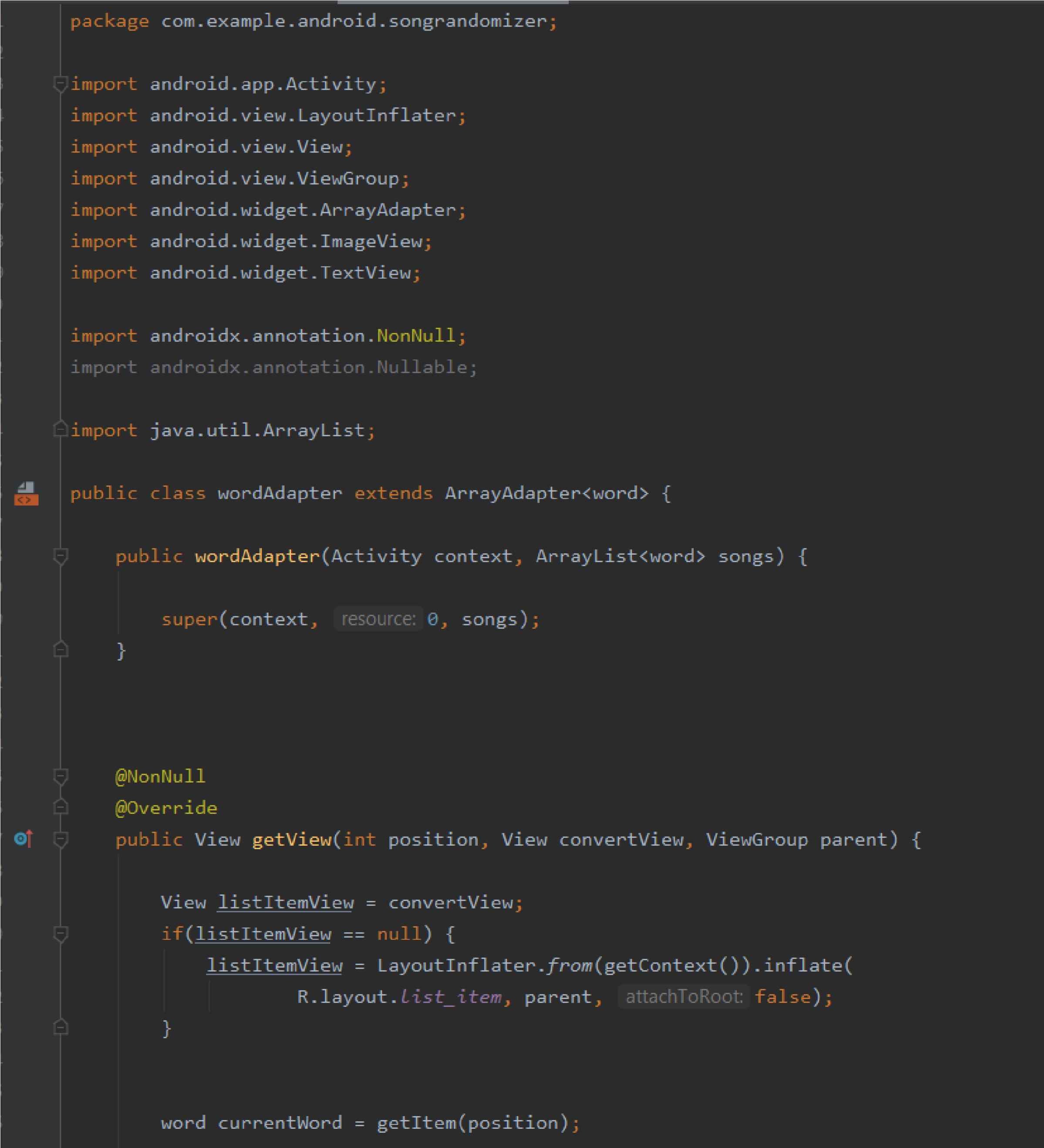


#### 5.Word.java

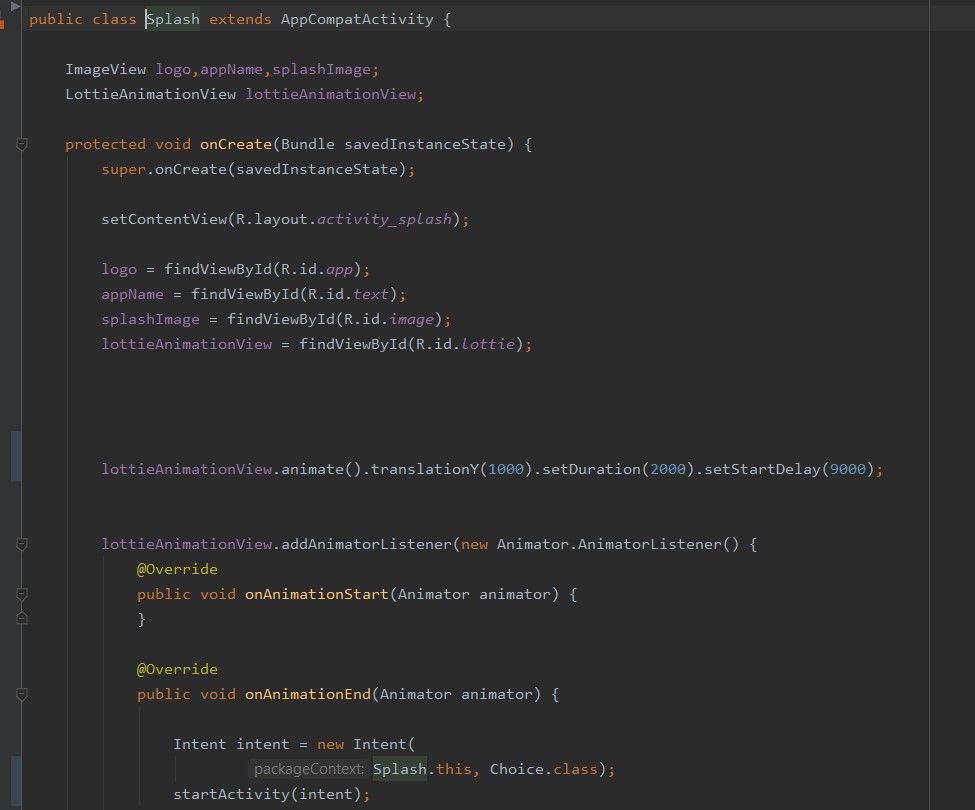


#### 6.WordAdapter.java



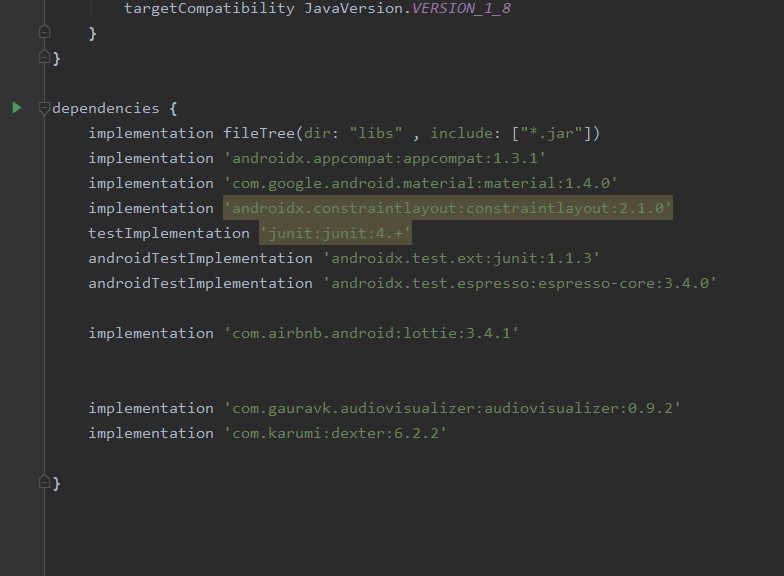


#### 7.Splash.java





**Dependencies in gradle module :**



**REFERENCES**

* [**https://developer.android.com/studio**](https://developer.android.com/studio)

**(To read about everything related to all the functions used in the app)**

* [**https://stackoverflow.com**](https://stackoverflow.com)

**(To clear the doubts and errors and for better implementation of code)**

* [**https://www.udacity.com/**](https://www.udacity.com/)

**(To get the idea of how apps are made, we referred few tutorials)**

* [**https://github.com/**](https://github.com/)

**(To get the idea of music apps already made, for reference)**

* [**https://oyebesmartest.com/s/mobile-amoled-wallpapers**](https://oyebesmartest.com/s/mobile-amoled-wallpapers)

**(For the wallpapers used while making the background of the screens)**

* [**https://www.tailorbrands.com/logo-maker**](https://www.tailorbrands.com/logo-maker)

**(To make the logo of the app)**

**(The end)**