NAME: SAKSHI VINAYAK KITTURKAR
CSU ID: 2860273
HOMEWORK 13

Based on the shake detection code provided, develop an app that stores at least three songs (or any audio files), and on shaking your phone, the app plays the next song on the list of songs.

= I have created an application that is shake detection which when you shake your mobile the next song from the list will play.

In layout: MainActivity.java this code represents a basic unit test for a simple addition operation, ensuring that 2 + 2 equals 4. Unit tests are crucial in software development for verifying that individual units of code (functions or methods) work as expected. They help catch bugs early and provide a safety net for code changes.

This line declares the package to which the ExampleUnitTest class belongs. In this case, it's in the com.example.hw13 package.

```
package com.example.hw13;
```

These lines import the necessary classes for writing JUnit tests. org.junit.Test is used to annotate methods as test methods, and static org.junit.Assert.* imports static assertion methods like assertEquals.

```
import org.junit.Test;
import static org.junit.Assert.*;
```

This line declares the class named ExampleUnitTest. It is a public class, meaning it can be accessed from outside the package.

```
public class ExampleUnitTest {
```

The @Test annotation indicates that the addition_isCorrect method is a test method. The purpose of this specific test is to check if the addition of 2 and 2 equals 4. The assertEquals method is used to assert that the expected result (4) is equal to the actual result (2 + 2).

```
@Test
public void addition_isCorrect() {
   assertEquals(4, 2 + 2);
}
```

In layout: ExampleInstrumentedTest this code represents an instrumented test for an Android application, specifically testing that the package name of the app's context matches the expected value. Instrumented tests are used to test the behavior of an app on a real Android device or emulator, and they are crucial for ensuring the correctness of the application in the Android environment.

These lines import classes and annotations necessary for writing instrumented tests on Android. Notably, android.content.Context is imported to work with Android application context, and InstrumentationRegistry is used to access instrumentation-related features. The @RunWith annotation specifies the test runner, and @Test is used to annotate the test method.

```
import android.content.Context;
import androidx.test.platform.app.InstrumentationRegistry;
import androidx.test.ext.junit.runners.AndroidJUnit4;
import org.junit.Test;
import org.junit.runner.RunWith;
import static org.junit.Assert.*;
```

This line declares the class named ExampleInstrumentedTest and specifies the use of the AndroidJUnit4 test runner.

```
@RunWith(AndroidJUnit4.class)
public class ExampleInstrumentedTest {
```

The @Test annotation indicates that the useAppContext method is a test method. This test method retrieves the context of the app under test using InstrumentationRegistry.getInstrumentation().getTargetContext() and then asserts that the package name of the app's context is equal to "com.example.hw13" using the assertEquals method.

```
@Test
public void useAppContext() {
   // Context of the app under test.
   Context appContext = InstrumentationRegistry.getInstrumentation().getTargetContext();
   assertEquals("com.example.hw13", appContext.getPackageName());
}
```

In layout: ExampleUnitTest this code represents a basic unit test for a simple addition operation, ensuring that 2 + 2 equals 4. Local unit tests are executed on the development machine, and they are an essential part of the software development process to verify that individual units of code (such as methods) behave as expected.

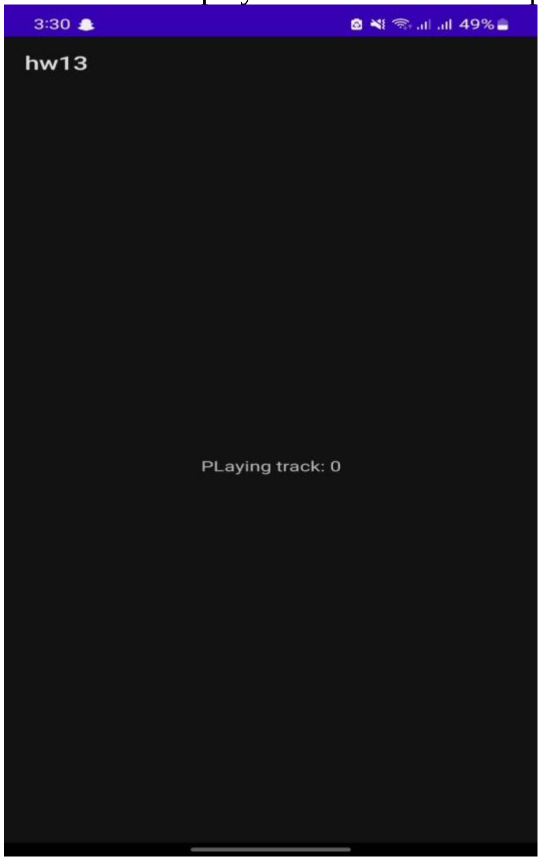
This line declares the class named ExampleUnitTest. It is a public class, meaning it can be accessed from outside the package.

```
public class ExampleUnitTest {
```

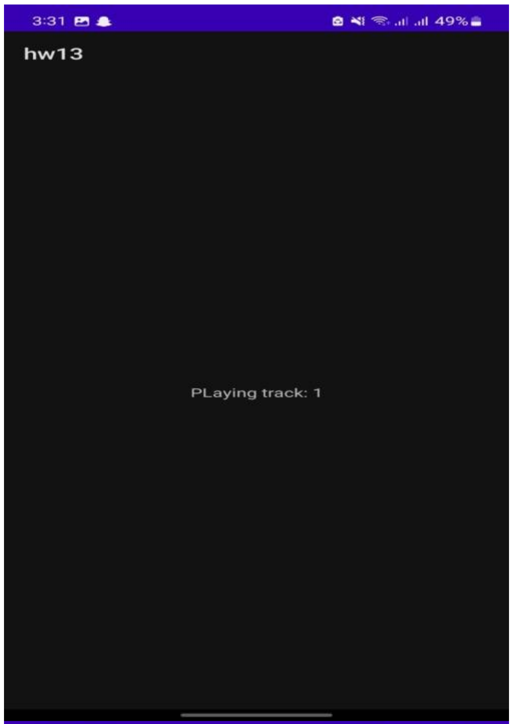
The @Test annotation indicates that the addition_isCorrect method is a test method. The purpose of this specific test is to check if the addition of 2 and 2 equals 4. The assertEquals method is used to assert that the expected result (4) is equal to the actual result (2 + 2).

```
@Test
public void addition_isCorrect() {
    assertEquals(4, 2 + 2);
}
```

The screens display the tracks which are played I have saved the tracks from 0 to 2:

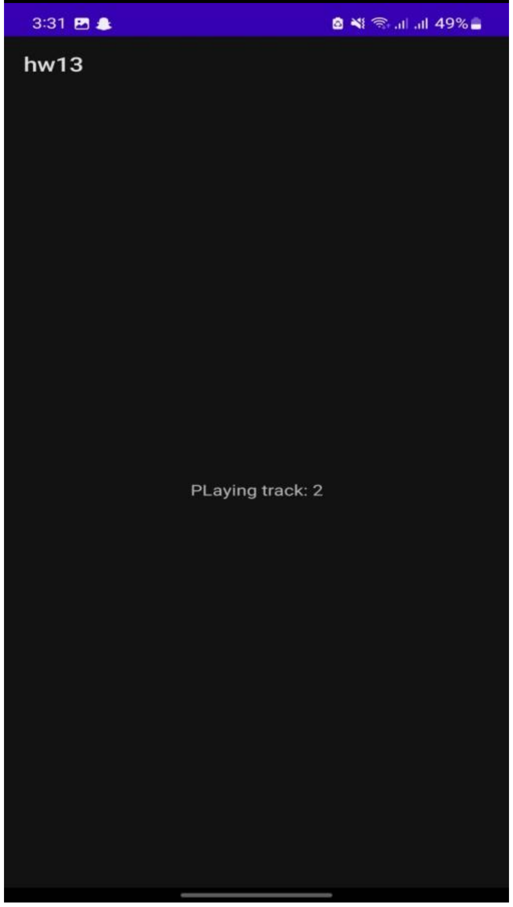


PLaying track: 0



hw13

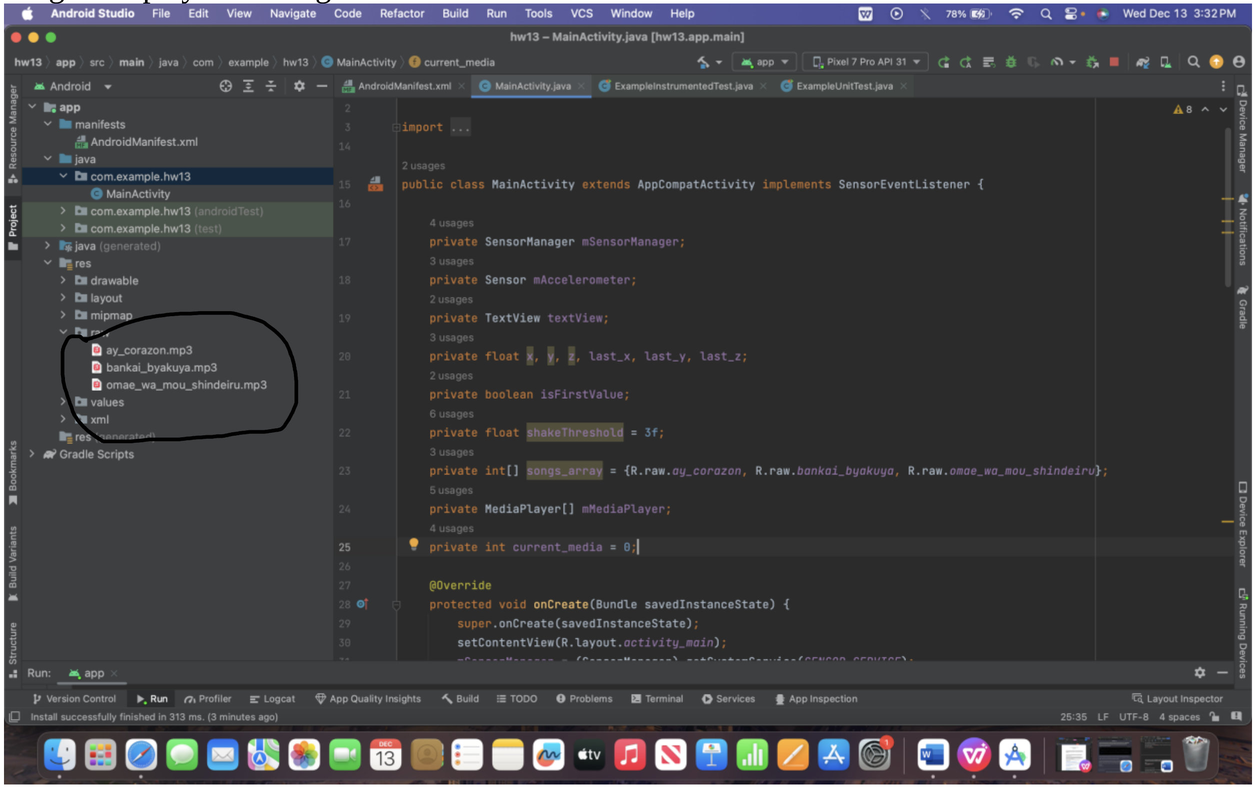
PLaying track: 1



hw13

PLaying track: 2

I have saved the songs in the raw I have saved three song when you run and shake your mobile the songs will play according to the list



hw13 – MainActivity.java [hw13.app.main]

```
app
  manifests
    AndroidManifest.xml
  java
    com.example.hw13
      MainActivity
    com.example.hw13 (androidTest)
    com.example.hw13 (test)
  java (generated)
  res
    drawable
    layout
    mipmap
    raw
      ay_corazon.mp3
      bankai_byakuya.mp3
      omae_wa_mou_shindeiru.mp3
    values
    xml
  res (generated)
Gradle Scripts
```

```java
import ...

2 usages
public class MainActivity extends AppCompatActivity implements SensorEventListener {

    4 usages
    private SensorManager mSensorManager;
    3 usages
    private Sensor mAccelerometer;
    2 usages
    private TextView textView;
    3 usages
    private float x, y, z, last_x, last_y, last_z;
    2 usages
    private boolean isFirstValue;
    6 usages
    private float shakeThreshold = 3f;
    3 usages
    private int[] songs_array = {R.raw.ay_corazon, R.raw.bankai_byakuya, R.raw.omae_wa_mou_shindeiru};
    5 usages
    private MediaPlayer[] mMediaPlayer;
    4 usages
    private int current_media = 0;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
Install successfully finished in 313 ms. (3 minutes ago)
```