# *Lab 4*

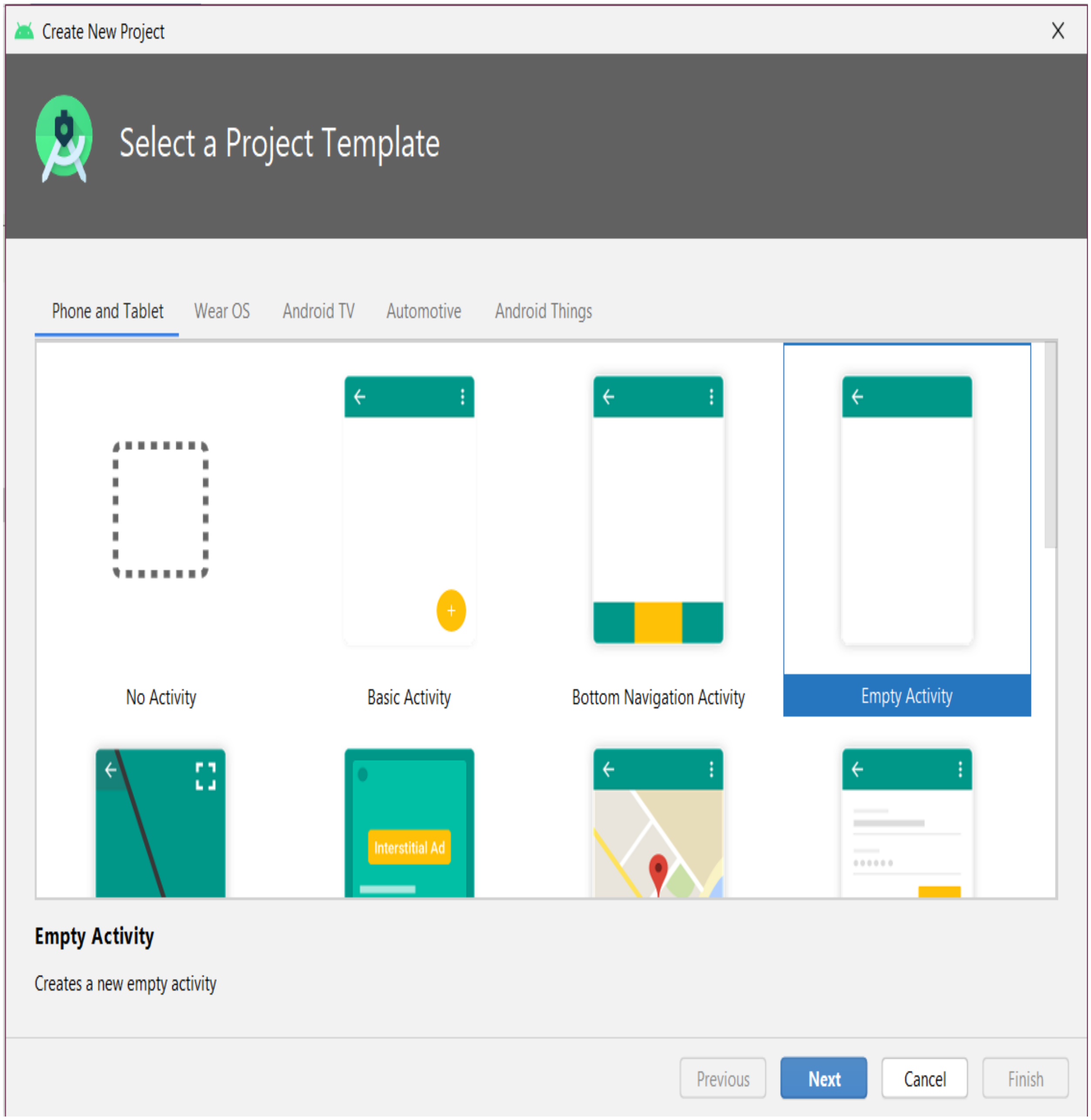
# Mobile Application Development – 2

**Rebecca Moses Dmello**

**300322984**

I am confirming that I have completed this classwork assignment completely based on the requirements and it is working and fully functional.

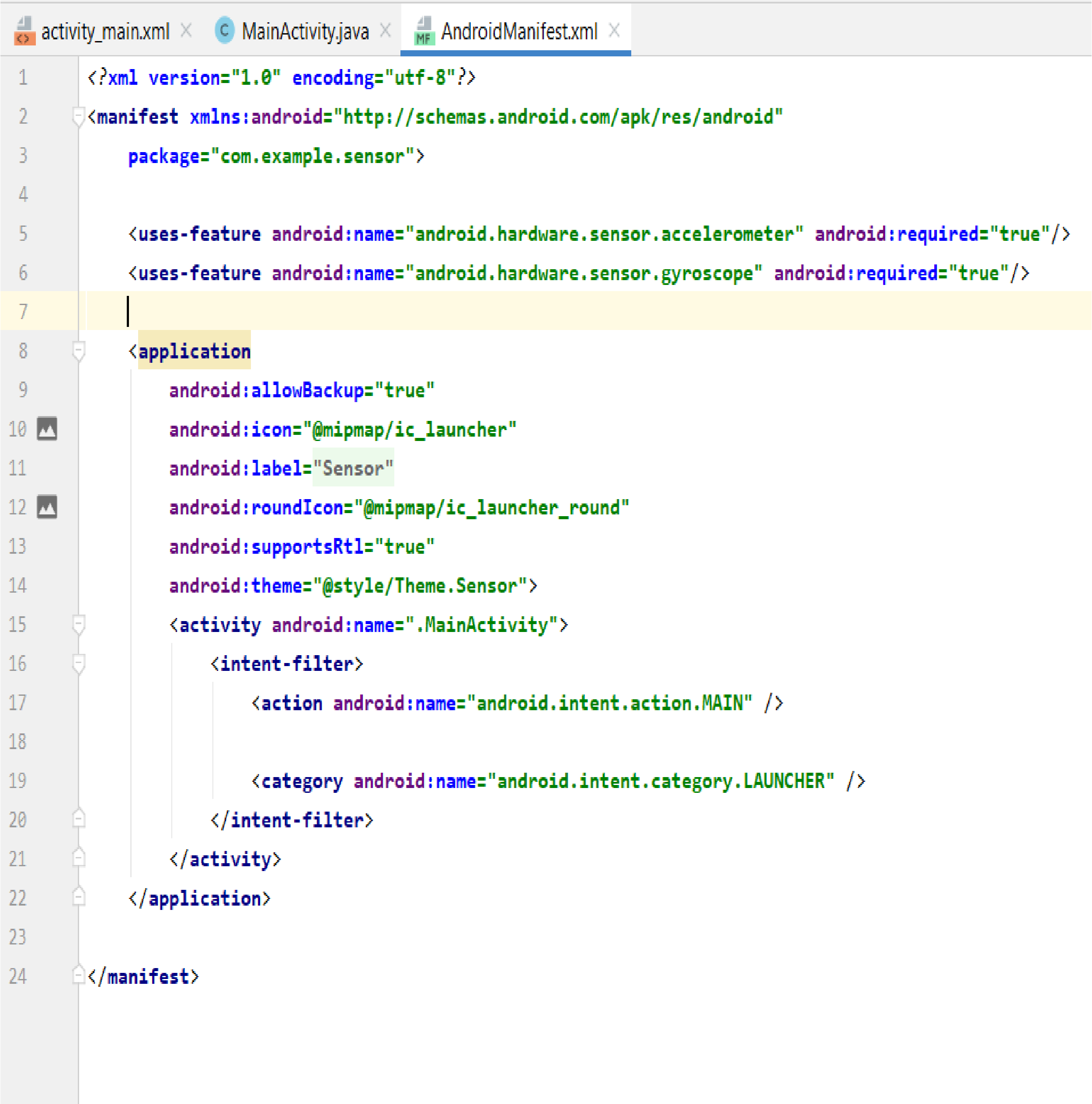
## Step 1: Create a New Project and Select Empty Activity



Empty Activity

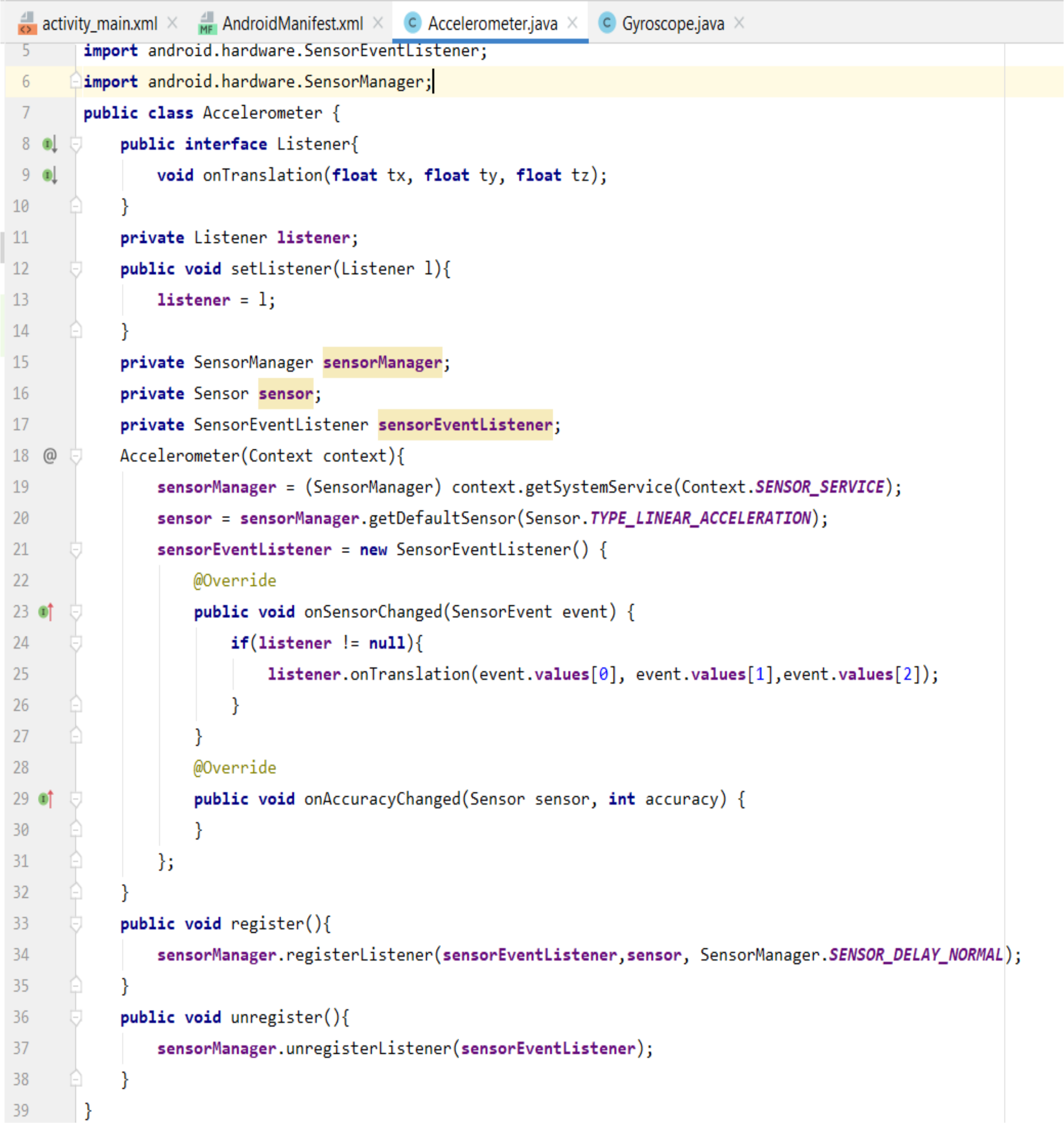
Creates a new empty activity

## Step 2: Open AndroidManifest.xml and  add the uses-feature tag



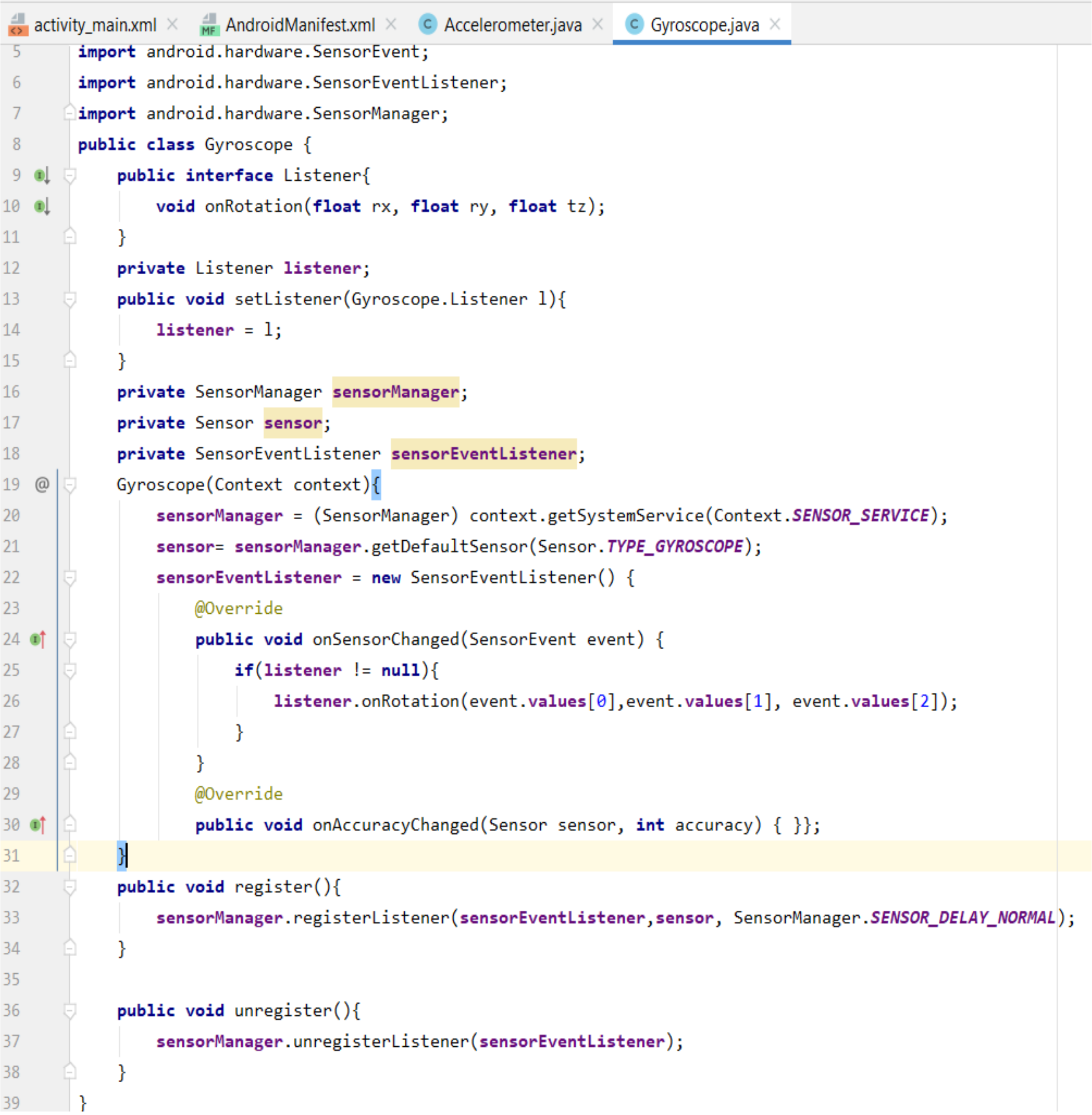
```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.sensor">

    <uses-feature android:name="android.hardware.sensor.accelerometer" android:required="true"/>
    <uses-feature android:name="android.hardware.sensor.gyroscope" android:required="true"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Sensor"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Sensor">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

**Step 3: Create a new Java class by right clicking on the package -> new -> java class and name it as Accelerometer and add the following code in it.**



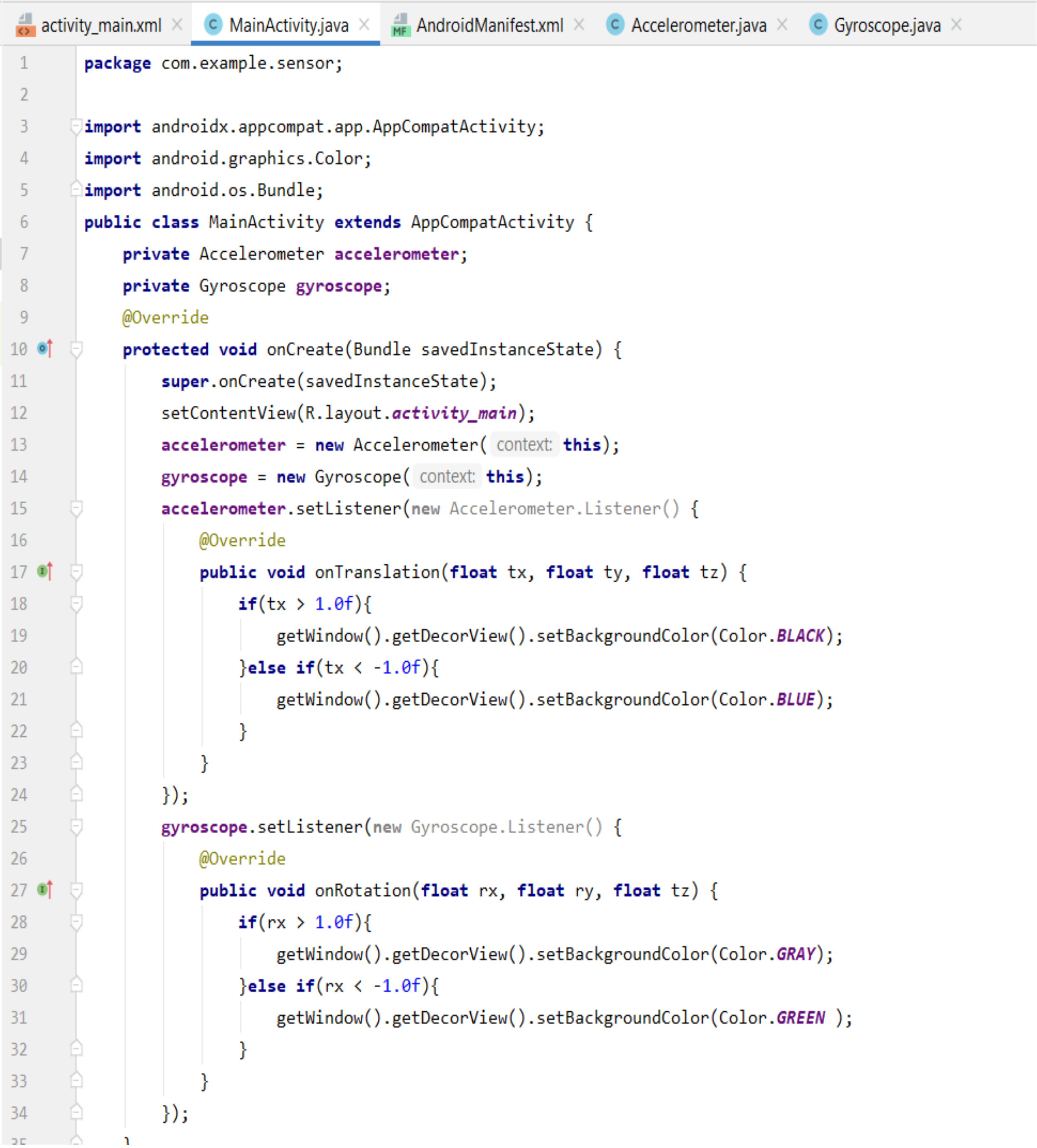
```java
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
public class Accelerometer {
    public interface Listener{
        void onTranslation(float tx, float ty, float tz);
    }
    private Listener listener;
    public void setListener(Listener l){
        listener = l;
    }
    private SensorManager sensorManager;
    private Sensor sensor;
    private SensorEventListener sensorEventListener;
    Accelerometer(Context context){
        sensorManager = (SensorManager) context.getSystemService(Context.SENSOR_SERVICE);
        sensor = sensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);
        sensorEventListener = new SensorEventListener() {
            @Override
            public void onSensorChanged(SensorEvent event) {
                if(listener != null){
                    listener.onTranslation(event.values[0], event.values[1],event.values[2]);
                }
            }
            @Override
            public void onAccuracyChanged(Sensor sensor, int accuracy) {
            }
        };
    }
    public void register(){
        sensorManager.registerListener(sensorEventListener,sensor, SensorManager.SENSOR_DELAY_NORMAL);
    }
    public void unregister(){
        sensorManager.unregisterListener(sensorEventListener);
    }
}
```

### Step 4: Create a new Java class by right clicking on the package -> new -> java class and name it as Gyroscope and add the following code in it.

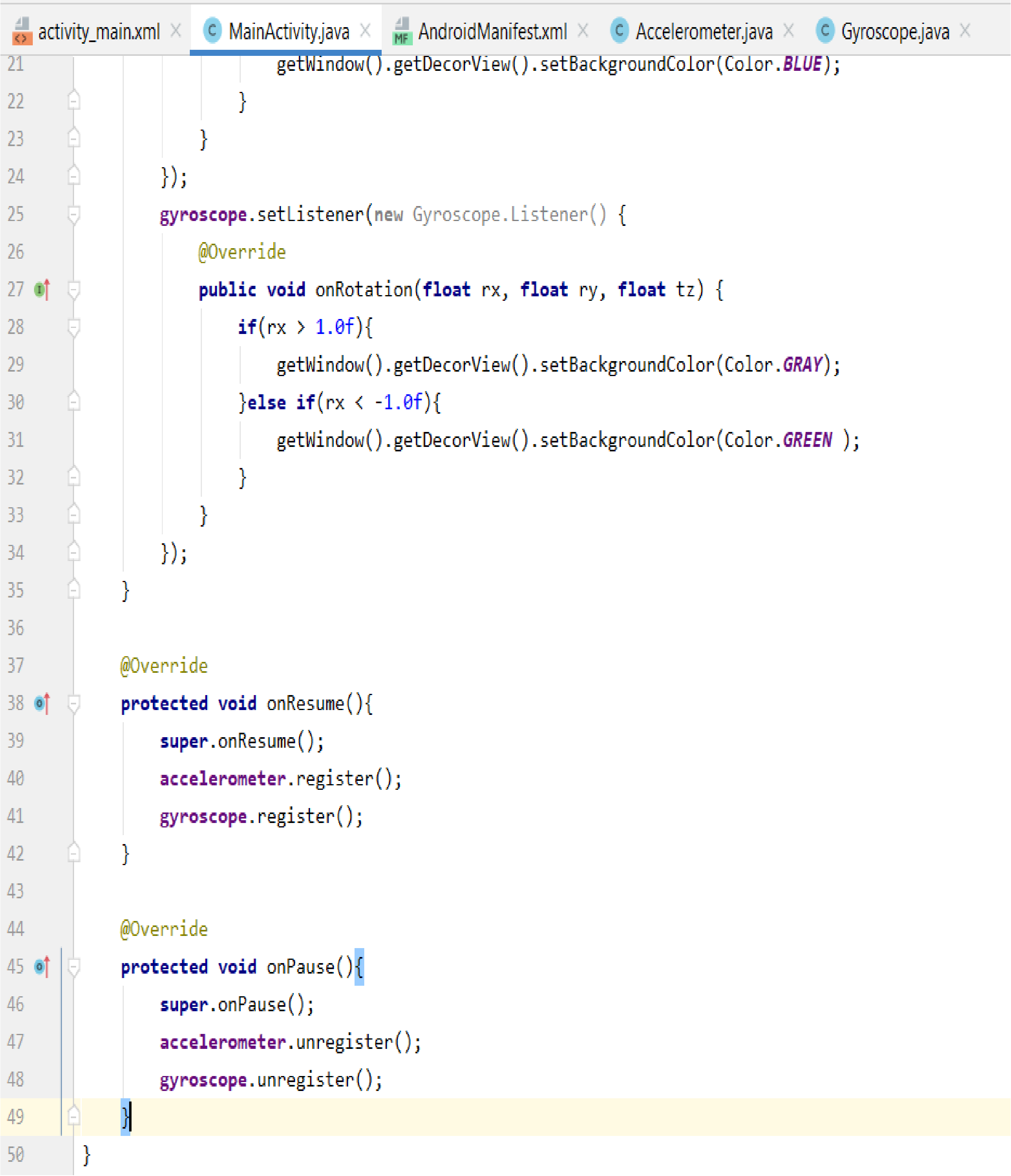


```java
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
public class Gyroscope {
    public interface Listener{
        void onRotation(float rx, float ry, float tz);
    }
    private Listener listener;
    public void setListener(Gyroscope.Listener l){
        listener = l;
    }
    private SensorManager sensorManager;
    private Sensor sensor;
    private SensorEventListener sensorEventListener;
    Gyroscope(Context context){
        sensorManager = (SensorManager) context.getSystemService(Context.SENSOR_SERVICE);
        sensor= sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
        sensorEventListener = new SensorEventListener() {
            @Override
            public void onSensorChanged(SensorEvent event) {
                if(listener != null){
                    listener.onRotation(event.values[0],event.values[1], event.values[2]);
                }
            }
            @Override
            public void onAccuracyChanged(Sensor sensor, int accuracy) { }};
    }
    public void register(){
        sensorManager.registerListener(sensorEventListener,sensor, SensorManager.SENSOR_DELAY_NORMAL);
    }

    public void unregister(){
        sensorManager.unregisterListener(sensorEventListener);
    }
}
```

## Step 5: Go to MainActivity.class and Instantiate an object of Accelerometer and Gyroscope.



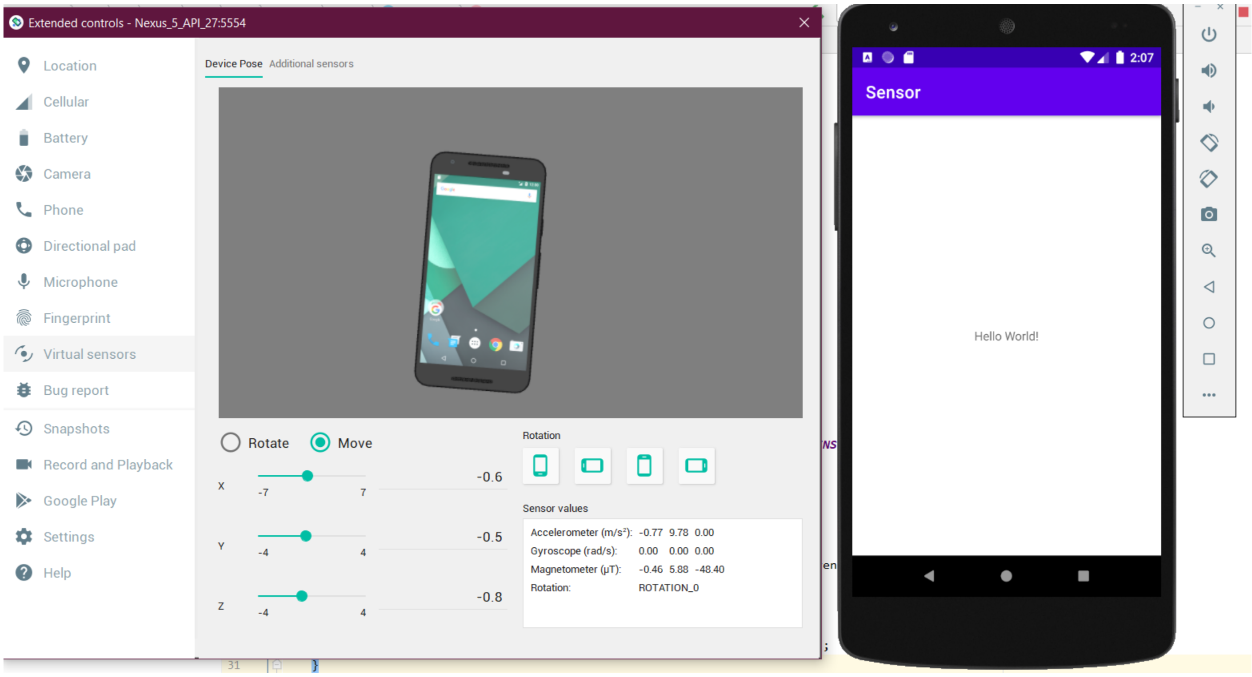
```java
package com.example.sensor;

import androidx.appcompat.app.AppCompatActivity;
import android.graphics.Color;
import android.os.Bundle;
public class MainActivity extends AppCompatActivity {
    private Accelerometer accelerometer;
    private Gyroscope gyroscope;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        accelerometer = new Accelerometer( context: this);
        gyroscope = new Gyroscope( context: this);
        accelerometer.setListener(new Accelerometer.Listener() {
            @Override
            public void onTranslation(float tx, float ty, float tz) {
                if(tx > 1.0f){
                    getWindow().getDecorView().setBackgroundColor(Color.BLACK);
                }else if(tx < -1.0f){
                    getWindow().getDecorView().setBackgroundColor(Color.BLUE);
                }
            }
        });
        gyroscope.setListener(new Gyroscope.Listener() {
            @Override
            public void onRotation(float rx, float ry, float tz) {
                if(rx > 1.0f){
                    getWindow().getDecorView().setBackgroundColor(Color.GRAY);
                }else if(rx < -1.0f){
                    getWindow().getDecorView().setBackgroundColor(Color.GREEN );
                }
            }
        });
    }
```



```java
                    getWindow().getDecorView().setBackgroundColor(Color.BLUE);
                }
            }
        });
        gyroscope.setListener(new Gyroscope.Listener() {
            @Override
            public void onRotation(float rx, float ry, float tz) {
                if(rx > 1.0f){
                    getWindow().getDecorView().setBackgroundColor(Color.GRAY);
                }else if(rx < -1.0f){
                    getWindow().getDecorView().setBackgroundColor(Color.GREEN );
                }
            }
        });
    }

    @Override
    protected void onResume(){
        super.onResume();
        accelerometer.register();
        gyroscope.register();
    }

    @Override
    protected void onPause(){
        super.onPause();
        accelerometer.unregister();
        gyroscope.unregister();
    }
}
```

## Step 6: Run your emulator to observe the animations

## Click on the 3 dots on the right of the emulator and select virtual sensors option



○ Rotate ● Move

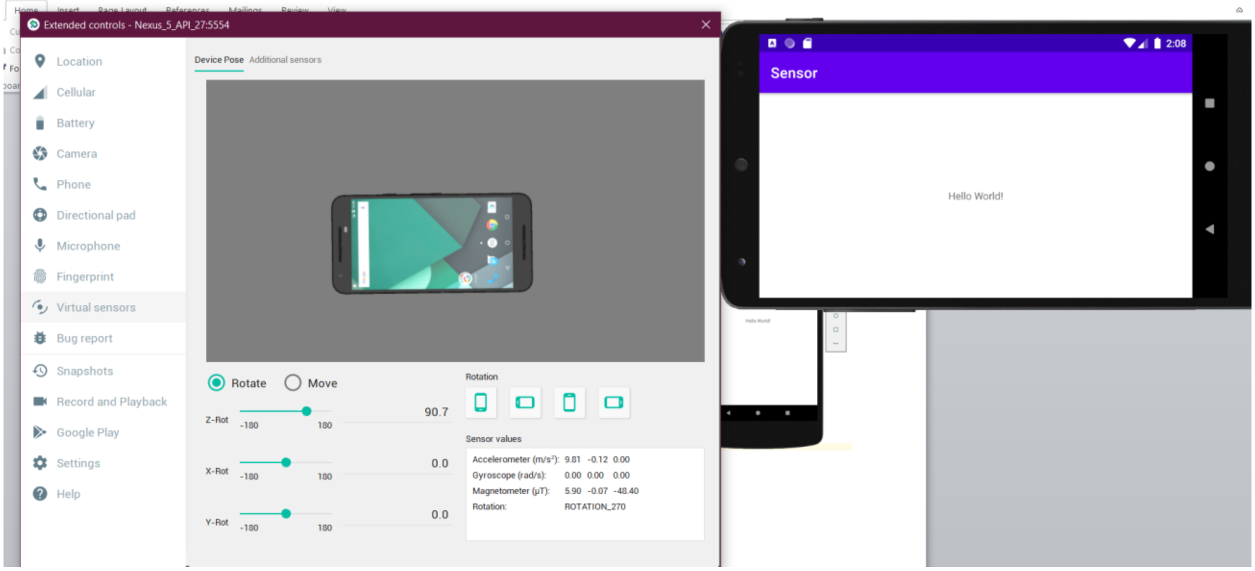
X -0.6 (-7 to 7)

Y -0.5 (-4 to 4)

Z -0.8 (-4 to 4)

Accelerometer (m/s²): -0.77 9.78 0.00
Gyroscope (rad/s): 0.00 0.00 0.00
Magnetometer (µT): -0.46 5.88 -48.40
Rotation: ROTATION_0

## Click on rotate and change the “Z” value



● Rotate ○ Move

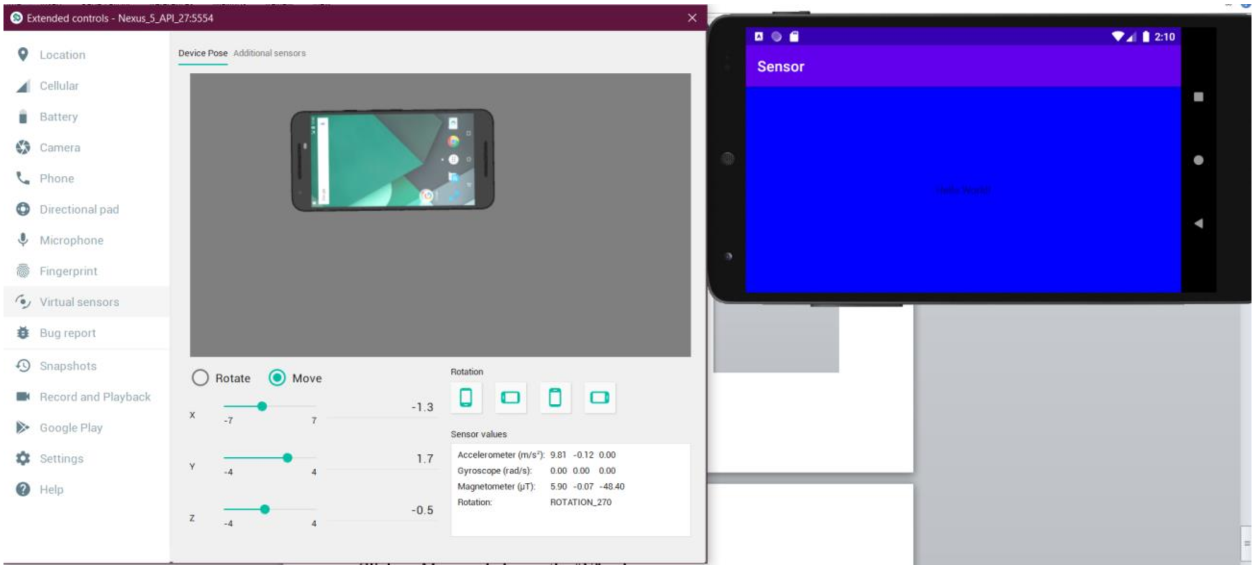
Z-Rot 90.7 (-180 to 180)

X-Rot 0.0 (-180 to 180)

Y-Rot 0.0 (-180 to 180)

Accelerometer (m/s²): 9.81 -0.12 0.00
Gyroscope (rad/s): 0.00 0.00 0.00
Magnetometer (µT): 5.90 -0.07 -48.40
Rotation: ROTATION_270

## Click on Move and change the “Y” value, you will notice the change in color of the background as specified in the code.

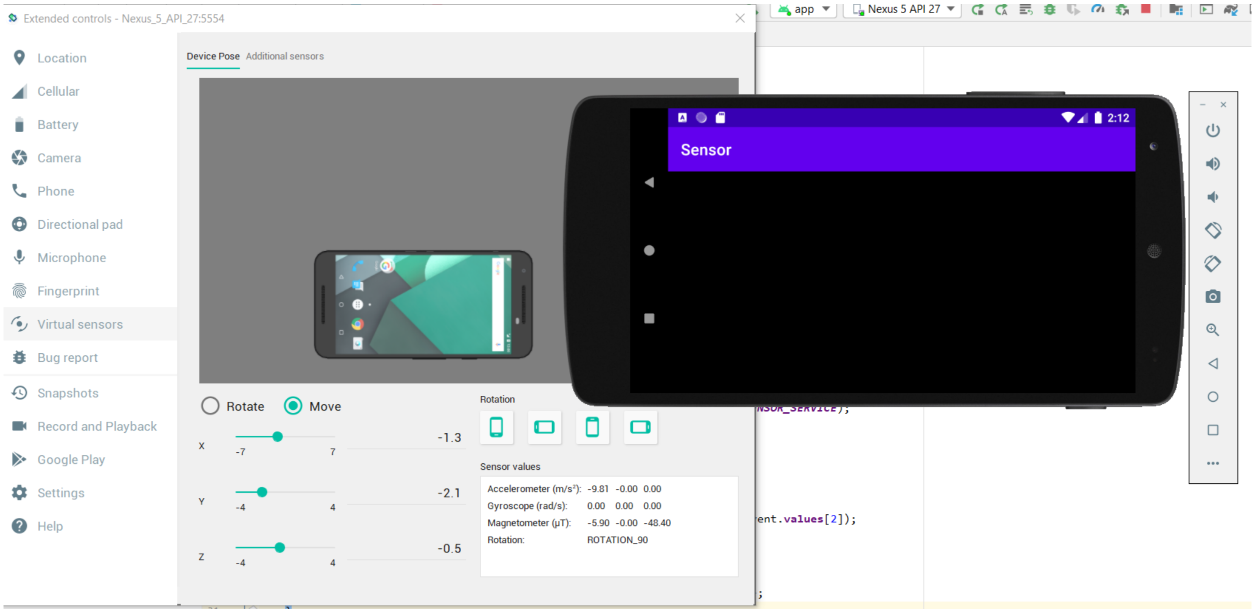


○ Rotate ● Move

X -1.3 (-7 to 7)
Y 1.7 (-4 to 4)
Z -0.5 (-4 to 4)

Sensor values

```
Accelerometer (m/s²):  9.81  -0.12  0.00
Gyroscope (rad/s):     0.00   0.00  0.00
Magnetometer (μT):     5.90  -0.07  -48.40
Rotation:              ROTATION_270
```



○ Rotate ● Move

X -1.3 (-7 to 7)
Y -2.1 (-4 to 4)
Z -0.5 (-4 to 4)

Sensor values

```
Accelerometer (m/s²):  -9.81  -0.00  0.00
Gyroscope (rad/s):      0.00   0.00  0.00
Magnetometer (μT):     -5.90  -0.00  -48.40
Rotation:              ROTATION_90
```

```
NSOR_SERVICE);
ent.values[2]);
;
```

## References:

### Programming Android Motion Sensors

https://www.youtube.com/watch?v=OPsVr44uCb8