```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/txtv"
        android:text="Shake to change the Color!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
package com.example.sensordemo;

import android.app.Activity;
import android.graphics.Color;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends Activity implements SensorEventListener{
    private SensorManager sensorManager;
    private boolean isColor = false;
    private View view;
    private long lastUpdate;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        view = findViewById(R.id.txtv);
        view.setBackgroundColor(Color.LTGRAY);

        sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
```

```
    lastUpdate = System.currentTimeMillis();
}
//overriding two methods of SensorEventListener
@Override
public void onAccuracyChanged(Sensor sensor, int accuracy) {}
@Override
public void onSensorChanged(SensorEvent event) {
    if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
        getAccelerometer(event);
    }

}

private void getAccelerometer(SensorEvent event) {
    float[] values = event.values;
    // Movement
    float x = values[0];
    float y = values[1];
    float z = values[2];

    float accelationSquareRoot = (x * x + y * y + z * z)
            / (SensorManager.GRAVITY_EARTH * SensorManager.GRAVITY_EARTH);

    long actualTime = System.currentTimeMillis();
    Toast.makeText(getApplicationContext(),String.valueOf(accelationSquareRoot)+" "+
            SensorManager.GRAVITY_EARTH,Toast.LENGTH_SHORT).show();

    if (accelationSquareRoot >= 2) //it will be executed if you shuffle
    {

        if (actualTime - lastUpdate < 200) {
            return;
        }
        lastUpdate = actualTime;//updating lastUpdate for next shuffle
        if (isColor) {
            view.setBackgroundColor(Color.BLUE);

        } else {
            view.setBackgroundColor(Color.YELLOW);
        }
        isColor = !isColor;
    }
}

@Override
protected void onResume() {
```

```
        super.onResume();
        // register this class as a listener for the orientation and
        // accelerometer sensors

sensorManager.registerListener(this,sensorManager.getDefaultSensor(Sensor.TYPE_ACCEL
EROMETER),
                SensorManager.SENSOR_DELAY_NORMAL);
    }

    @Override
    protected void onPause() {
        // unregister listener
        super.onPause();
        sensorManager.unregisterListener(this);
    }
}
```