Practical No 22

X.1

- **Program**
- **XML File:**

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:tools="http://schemas.android.com/tools"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 tools:context=".MainActivity" >
 <TextView
 android:id="@+id/textView"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:text="Shake to switch color" />
</RelativeLayout>
```

- **JAVA File:**

```
package com.example.pr_22_1;
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;
```

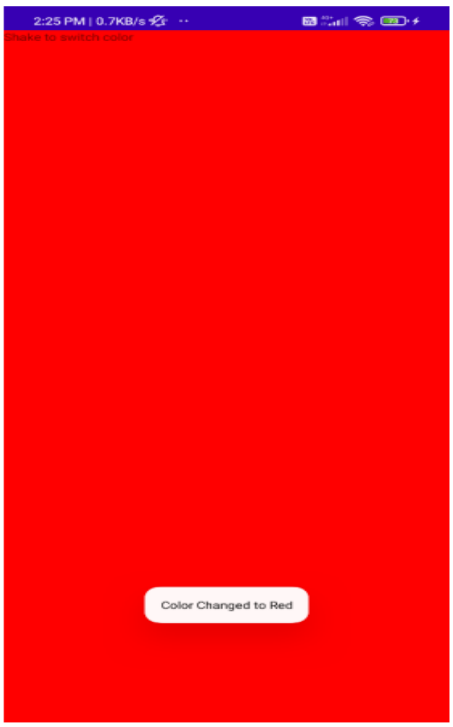
```
public class MainActivity extends Activity implements
SensorEventListener{
 private SensorManager sensorManager;
 private boolean isColor = false;
 private View view;
 private long lastUpdate;
 @Override
 public void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main);
 view = findViewById(R.id.textView);
 view.setBackgroundColor(Color.GREEN);
 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
```

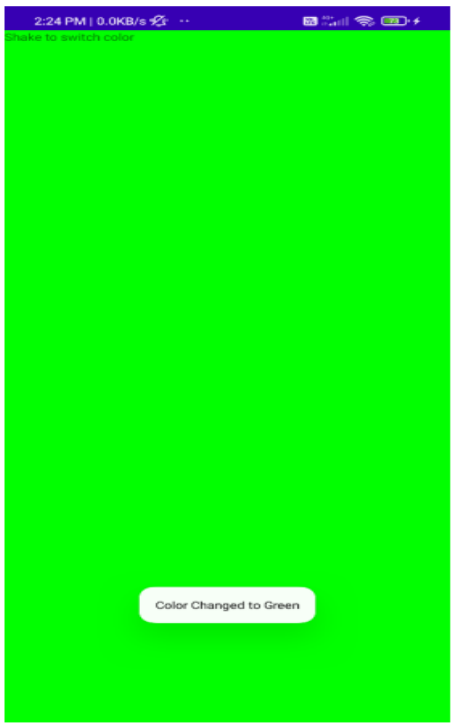
```
 float x = values[0];
 float y = values[1];
 float z = values[2];
 float accelationSquareRoot = (x * x + y * y + z * z)
 / (SensorManager.GRAVITY_EARTH *
SensorManager.GRAVITY_EARTH);
 long actualTime = System.currentTimeMillis();
 if (accelationSquareRoot >= 2) //it will be executed if you shuffle
 {
 if (actualTime - lastUpdate < 200) {
 return;
 }
 lastUpdate = actualTime;//updating lastUpdate for next shuffle
 if (isColor) {
 view.setBackgroundColor(Color.GREEN);
 Toast.makeText(this, "Color Changed to Green",
Toast.LENGTH_SHORT).show();
 } else {
 view.setBackgroundColor(Color.RED);
 Toast.makeText(this, "Color Changed to Red",
Toast.LENGTH_SHORT).show();
 }
 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(Sens
or.T
YPE_ACCELEROMETER),
SensorManager.SENSOR_DELAY_NORMAL);
}
@Override
protected void onPause() {
// unregister listener
super.onPause();
sensorManager.unregisterListener(this);
}
}
```

- **Output:**





X.2

- **Program**
- **XML File:**

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
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">
 <EditText
 android:id="@+id/e1"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:layout_alignParentEnd="true"
 android:layout_alignParentBottom="true"
 android:layout_marginEnd="122dp"
 android:layout_marginBottom="573dp"
 android:ems="10"
 android:hint=" enter number" />
 <Button
 android:id="@+id/t1"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:layout_alignParentEnd="true"
 android:layout_alignParentBottom="true"
 android:layout_marginEnd="151dp"
```

```
 android:layout_marginBottom="500dp"
 android:text="Start Dialer" />
</RelativeLayout>
```

- **JAVA File:**

```
package com.example.pr_22_2;
import android.hardware.Sensor;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import java.util.List;
public class MainActivity extends AppCompatActivity {
 TextView tv;
 @Override
 protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main);
 tv = findViewById(R.id.tv);
 String sensorInfo = "";
 SensorManager sensorManager = (SensorManager)
getSystemService(SENSOR_SERVICE);
 List<Sensor> sensorList =
sensorManager.getSensorList(Sensor.TYPE_ALL);
 for(Sensor s : sensorList) {
 sensorInfo += s.getName() + "\n";
 }
 tv.setText(sensorInfo);
 }
}
```

- **Output:**


2:26 PM | 5.2KB/s
Pr_22_2
bst_bma2x2 Accelerometer Non-wakeup
mmc56x3x Magnetometer Non-wakeup
Rotation Vector Non-wakeup
gyro Gyroscope Non-wakeup
stk_stk3a5x Ambient Light Sensor Non-wakeup
stk_stk3a5x Ambient Light Sensor Wakeup
stk_stk3a5x Proximity Sensor Non-wakeup
stk_stk3a5x Proximity Sensor Wakeup
gravity Non-wakeup
linear_acceleration
Rotation Vector Non-wakeup
mmc56x3x Magnetometer-Uncalibrated Non-wakeup
Game Rotation Vector Non-wakeup
sns_smd Wakeup
pedometer Non-wakeup
pedometer Wakeup
pedometer Non-wakeup
pedometer Wakeup
sns_geomag_rv Non-wakeup
sns_tilt Wakeup
Device Orientation Non-wakeup
Device Orientation Wakeup
stationary_detect
stationary_detect_wakeup
motion_detect
motion_detect_wakeup
bst_bma2x2 Accelerometer-Uncalibrated Non-wakeup
pickup Non-wakeup
pickup Wakeup