Activity\_main.xml

*<?***xml version="1.0" encoding="utf-8"***?>*

<**LinearLayout 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"**

**android:orientation="vertical"**

**tools:context=".MainActivity"**>

<**TextView**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="30dp"**

**android:layout\_marginLeft="20dp"**

**android:textSize="30dp"**

**android:textColor="@color/colorAccent"**

**android:id="@+id/label\_light"**

**android:text="@string/label\_light"**

/>

<**TextView**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="20dp"**

**android:layout\_marginLeft="20dp"**

**android:textSize="30dp"**

**android:textColor="@color/colorAccent"**

**android:id="@+id/label\_proximity"**

**android:text="@string/label\_proximity"**/>

<**TextView**

**android:layout\_width="wrap\_content"**

**android:layout\_height="wrap\_content"**

**android:layout\_marginTop="20dp"**

**android:layout\_marginLeft="20dp"**

**android:textSize="30dp"**

**android:textColor="@color/colorAccent"**

**android:id="@+id/label\_temp"**

**android:text="@string/label\_temp"**/>

</**LinearLayout**>

Strings.xml

<**resources**>

<**string name="app\_name"**>25\_2\_Light\_ProximitySensor</**string**>

<**string name="error\_no\_sensor"**>No Sensor Available on Device</**string**>

<**string name="label\_light"**>Light Sensor: %1$.2f </**string**>

<**string name="label\_proximity"**>Proximity Sensor: %1$.2f </**string**>

<**string name="label\_temp"**>Temperature Sensor: %1$.2f </**string**>

</**resources**>

MainActivity.java

**package** com.example.jevitha.a25\_2\_light\_proximitysensor;

**import** android.support.v7.app.AppCompatActivity;

**import** android.os.Bundle;

**import** android.content.Context;

**import** android.hardware.Sensor;

**import** android.hardware.SensorEvent;

**import** android.hardware.SensorManager;

**import** android.hardware.SensorEventListener;

**import** android.support.v7.app.AppCompatActivity;

**import** android.os.Bundle;

**import** android.widget.TextView;

*/\*\**

*\* SensorListeners demonstrates how to gain access to sensors (here, the light*

*\* and proximity sensors), how to register sensor listeners, and how to*

*\* handle sensor events.*

*\*/*

**public class** MainActivity **extends** AppCompatActivity **implements**

SensorEventListener {

*// System sensor manager instance.*

**private** SensorManager **mSensorManager**;

*// Proximity and light sensors, as retrieved from the sensor manager.*

**private** Sensor **mSensorProximity**;

**private** Sensor **mSensorLight**;

**private** Sensor **mSensorTemp**;

*// TextViews to display current sensor values.*

**private** TextView **mTextSensorLight**;

**private** TextView **mTextSensorProximity**;

**private** TextView **mTextSensorTemp**;

@Override

**protected void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.***activity\_main***);

*// Initialize all view variables.*

**mTextSensorLight** = (TextView) findViewById(R.id.***label\_light***);

**mTextSensorTemp** = (TextView) findViewById(R.id.***label\_temp***);

**mTextSensorProximity** = (TextView) findViewById

(R.id.***label\_proximity***);

*// Get an instance of the sensor manager.*

**mSensorManager** = (SensorManager)

getSystemService(Context.***SENSOR\_SERVICE***);

*// Get light and proximity sensors from the sensor manager.*

*// The getDefaultSensor() method returns null if the sensor is not available on the device.*

**mSensorProximity** = **mSensorManager**.getDefaultSensor

(Sensor.***TYPE\_PROXIMITY***);

**mSensorLight** = **mSensorManager**.getDefaultSensor

(Sensor.***TYPE\_LIGHT***);

**mSensorTemp** = **mSensorManager**.getDefaultSensor

(Sensor.***TYPE\_AMBIENT\_TEMPERATURE***);

*// Get the error message from string resources.*

String sensor\_error = getResources().getString(R.string.***error\_no\_sensor***);

*// If either mSensorLight or mSensorProximity are null, those sensors are not available in the device. Set the text to the error message*

**if** (**mSensorLight** == **null**) {

**mTextSensorLight**.setText(sensor\_error);

}

**if** (**mSensorProximity** == **null**) {

**mTextSensorProximity**.setText(sensor\_error);

}

**if** (**mSensorTemp** == **null**) {

**mTextSensorTemp**.setText(sensor\_error);

}

}

@Override

**protected void** onStart() {

**super**.onStart();

*// Listeners for the sensors are registered in this callback and*

*// can be unregistered in onPause().*

*//*

*// Check to ensure sensors are available before registering listeners.*

*// Both listeners are registered with a "normal" amount of delay*

*// (SENSOR\_DELAY\_NORMAL)*

**if** (**mSensorProximity** != **null**) {

**mSensorManager**.registerListener(**this**,

**mSensorProximity**,

SensorManager.***SENSOR\_DELAY\_NORMAL***);

}

**if** (**mSensorLight** != **null**) {

**mSensorManager**.registerListener(**this**, **mSensorLight**,

SensorManager.***SENSOR\_DELAY\_NORMAL***);

}

**if** (**mSensorTemp** != **null**) {

**mSensorManager**.registerListener(**this**, **mSensorTemp**,

SensorManager.***SENSOR\_DELAY\_NORMAL***);

}

}

@Override

**protected void** onStop() {

**super**.onStop();

*// Unregister all sensor listeners in this callback so they don't*

*// continue to use resources when the app is paused.*

**mSensorManager**.unregisterListener(**this**);

}

@Override

**public void** onSensorChanged(SensorEvent sensorEvent) {

*// The sensor type (as defined in the Sensor class).*

**int** sensorType = sensorEvent.**sensor**.getType();

*// The new data value of the sensor. Both the light and proximity*

*// sensors report one value at a time, which is always the first*

*// element in the values array.*

**float** currentValue = sensorEvent.**values**[0];

**switch** (sensorType) {

*// Event came from the light sensor.*

**case** Sensor.***TYPE\_LIGHT***:

*// Set the light sensor text view to the light sensor string*

*// from the resources, with the placeholder filled in.*

**mTextSensorLight**.setText(getResources().

getString(R.string.***label\_light***,

currentValue));

**break**;

**case** Sensor.***TYPE\_PROXIMITY***:

*// Set the proximity sensor text view to the light sensor*

*// string from the resources, with the placeholder filled in.*

**mTextSensorProximity**.setText(getResources().

getString(R.string.***label\_proximity***,

currentValue));

**break**;

**case** Sensor.***TYPE\_AMBIENT\_TEMPERATURE***:

*// Set the proximity sensor text view to the light sensor*

*// string from the resources, with the placeholder filled in.*

**mTextSensorTemp**.setText(getResources().

getString(R.string.***label\_temp***,

currentValue));

**break**;

**default**:

*// do nothing*

}

}

*/\*\**

*\* Abstract method in SensorEventListener.*

*\*Called if the sensor's accuracy changes, so your app can react to that change.*

*\* Most sensors, including the light and proximity sensors, do not report accuracy changes*

*\* In this app, you leave onAccuracyChanged() empty.*

*\*/*

@Override

**public void** onAccuracyChanged(Sensor sensor, **int** i) {

}

}