sensor

센서의 종류

1. Motion sensor (가속력 및 회전력)
   : accelerometers, gravity sensors, gyroscopes, rotational vector sensors
2. Environmental sensors (기온 기압 조도 습도 등)
   : barometers, photometers, thermometers
3. Position Sensor
   : orientation sensors, magnetometers.

센서 사용을 위한 클래스 및 인터페이스

1. SensorManager
2. Sensor
3. SensorEvent
4. SensorEventListener

https://developer.android.com/guide/topics/sensors?hl=ko

SensorManager 클래스

- 안드로이드 센서를 관리하고 이용할 수 있도록 돕는 클래스로,
  어떤 센서가 있고 그 센서가 제공하는 값의 범위가 어디에서 어디까지인지 알려준다.
  가속도 센서와 같은 센서 값을 받아올려고할 때 값을 받아올 수 있게 한다.
  (센서 이벤트 리스너 등록을 센서매니저를 통해 하게되면 센서값을 받아올 수 있게 된다.)

- SensorManager sm = (SensorManager)getSystemService(Context.SENSOR_SERVICE);
  여기서 SENSOR_SERVICE는 자기가 원하는 구체적인 센서로 설정해서 쓴다.
  ex) LOCATION_SERVICE

Sensor 클래스

- 센서매니저 클래스의 메소드를 활용하여 접근하고자 하는 센서의 객체를 선언할 수 있다.
- 위에서 만들어놓은 sm.을 이용해
- Sensor accel = sm.getDefaultSensor(Sensor.TYPE_ACCELEROMETER); //사용예시

- Sensor에서 제공하는 클래스 메소드는 다음과 같다.

- float getMaximumRange() 센서값의최대범위

  int getMinDelay() 두개의센서이벤트사이의최소딜레이 (ms단위) 혹은0 (측정데이터에변화가 있을때만값을주는경우)

  String getName()

  float getPower() 사용중에소모한파워(mA 단위)

  String getStringType()

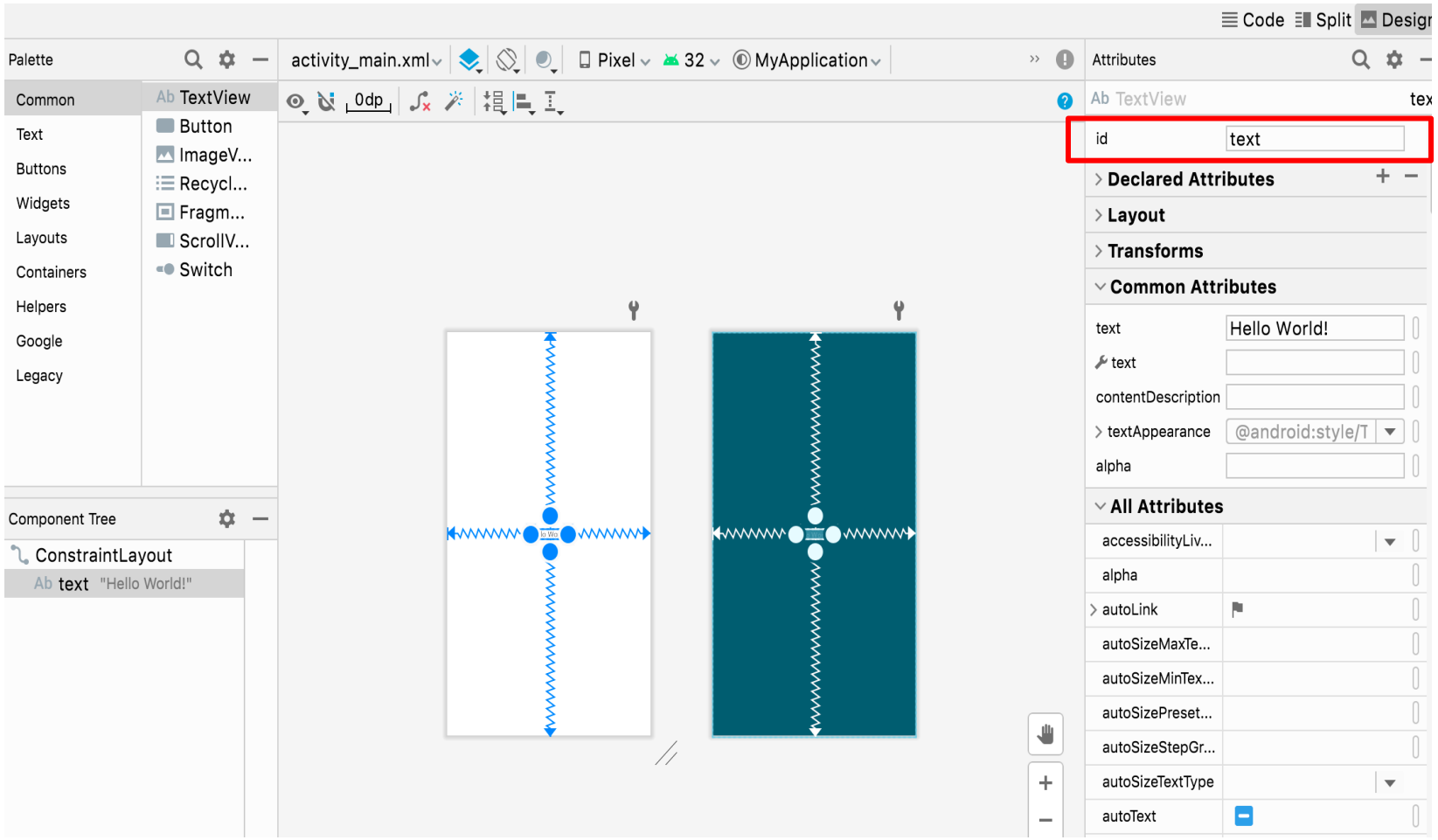
  int getType()

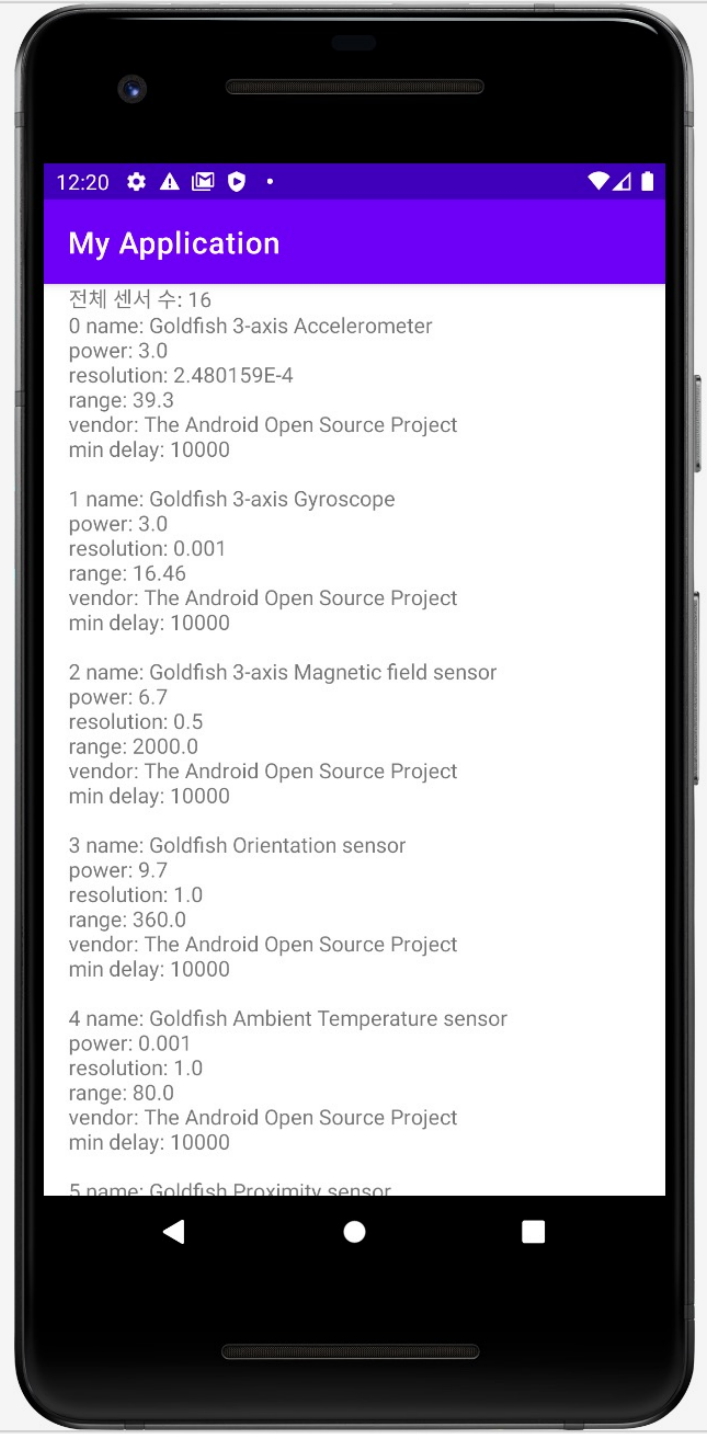
  String getVendor()

  int getVersion()

- 센서타입
- 이는 sensor 클래스에 정의된 상수이므로 가져다가 쓰면 된다.
- 실제로 내 안드로이드 폰에 내장된 센서가 제공하는 정보를 확인해보자.

상수	타입	설명	단위
TYPE_ALL		모든 센서	
TYPE_ACCELEROMETER	HW	가속도 센서	m/sec^2
TYPE_GRAVITY	SW or HW	중력 센서	m/sec^2
TYPE_GYROSCOPE	HW	자이로스코프	
TYPE_ORIENTATION	SW	방향 센서	도 (degree)
TYPE_LIGHT	HW	조도 센서	Lux
TYPE_LINEAR_ACCELERATION	SW or HW	선형 가속도 센서	
TYPE_MAGNETIC_FIELD	HW	지자기 센서	마이크로테슬라 (uT)
TYPE_PRESSURE	HW	압력 센서	
TYPE_PROXIMITY	HW	근접 센서	미터
TYPE_ROTATION_VECTOR	SW or HW	회전 벡터 센서	
TYPE_AMBIENT_TEMPERATURE	HW	온도 센서	


Code
Split
Design
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageV...
Recycl...
Fragm...
ScrollV...
Switch
activity_main.xml
Pixel
32
MyApplication
0dp
Component Tree
ConstraintLayout
text "Hello World!"
Attributes
TextView
id
text
Declared Attributes
Layout
Transforms
Common Attributes
text
Hello World!
text
contentDescription
textAppearance
@android:style/T
alpha
All Attributes
accessibilityLiv...
alpha
autoLink
autoSizeMaxTe...
autoSizeMinTex...
autoSizePreset...
autoSizeStepGr...
autoSizeTextType
autoText


12:20
My Application
전체 센서 수: 16
0 name: Goldfish 3-axis Accelerometer
power: 3.0
resolution: 2.480159E-4
range: 39.3
vendor: The Android Open Source Project
min delay: 10000
1 name: Goldfish 3-axis Gyroscope
power: 3.0
resolution: 0.001
range: 16.46
vendor: The Android Open Source Project
min delay: 10000
2 name: Goldfish 3-axis Magnetic field sensor
power: 6.7
resolution: 0.5
range: 2000.0
vendor: The Android Open Source Project
min delay: 10000
3 name: Goldfish Orientation sensor
power: 9.7
resolution: 1.0
range: 360.0
vendor: The Android Open Source Project
min delay: 10000
4 name: Goldfish Ambient Temperature sensor
power: 0.001
resolution: 1.0
range: 80.0
vendor: The Android Open Source Project
min delay: 10000

activity_main.xml | MainActivity.java

```java
package com.example.myapplication;

import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import java.util.List;

public class MainActivity extends AppCompatActivity {

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

        String list = "";
        // SensorManager 객체를 getSystemService 메소드를 통해 얻음
        SensorManager sm = (SensorManager)getSystemService(Context.SENSOR_SERVICE);
        // 모든 타입의 센서 목록을 얻음
        List<Sensor> sensors = sm.getSensorList(Sensor.TYPE_ALL);

```

```
26
27          list += "전체 센서 수: " + sensors.size() + "\n";
28          int i = 0;
29          for(Sensor s: sensors) {
30              list += "" + i++ + " name: " + s.getName() + "\n" + "power: " + s.getPower() + "\n"
31                      + "resolution: " + s.getResolution() + "\n" + "range: " + s.getMaximumRange() + "\n"
32                      + "vendor: " + s.getVendor() + "\n" + "min delay: " + s.getMinDelay() + "\n\n";
33          }
34
35          TextView text = (TextView)findViewById(R.id.text);
36          // TextView에 텍스트 내용이 화면 크기를 넘어서 들어갈 때 스크롤 가능하게 만들기 위한 메소드 호출
37          text.setMovementMethod(new ScrollingMovementMethod());
38          text.setText(list);
39      }
40  }
```

SensorEventListener 인터페이스

- 센서 값이 변경되었을 때 SensorManager한테 이벤트형태로 값을 전달받을 수 있도록 해줍니다.
  등록방법) SensorManager에 있는 registerListener메소드를 활용하여 등록합니다.
  boolean registerListener(SensorEventListener listener, Sensor sensor, int samplingPeriodUs)

- listener: SensorEventListener객체
  sensor: 데이터를받고싶은센서객체
- samplingPeriodUs는얼마나 자주 변경된 데이터를 받을지 값을 설정하는 것인데

- SENSOR_DELAY_NORMAL
  SENSOR_DELAY_UI
  SENSOR_DELAY_GAME
  SENSOR_DELAY_FASTEST

- 4가지 상수로 SensorManager에 정의되어 있다. 이는 밑으로 내려갈수록 빠르며, 기기마다 조금씩 차이 날 수 있다.

구현

SensorEventListener 인터페이스를 구현하기 위해서는 두개의 메소드를 구현해야 함

public void onAccuracyChanged(Sensor sensor, intaccuracy)

//등록된 센서의 정확도가 변경됐을때 호출

public void onSensorChanged(SensorEventevent)

//센서값이 변했을 때 호출.

- SensorEvent객체로데이터를전달받게됨

```
public void onAccuracyChanged(Sensor sensor, int accuracy) {

    }

    public void onSensorChanged(SensorEvent event) {
        if(event.sensor.getType() == Sensor.TYPE_ORIENTATION) {
            mText.setText("방향 센서값\n\n방위각: " + event.values[0]
                    + "\n피치: " + event.values[1] + "\n롤: " + event.values[2]);
        }
    }
```

해제

센서 데이터 업데이트가 필요하지 않으면 반드시 이를 해제시켜주어야한다.

public void unregisterListener(SensorEventListenerlistener, Sensor sensor) public void

unregisterListener(SensorEventListenerlistener)

```
@Override
  protected void onResume() {
      super.onResume();
      // SensorEventListener 등록
      mSensorManger.registerListener(this, mOrientation, SensorManager.SENSOR_DELAY_UI);
  }//this인 이유는 implements로 SensorEventListener 해놓았기때문~

  @Override
  protected void onPause() {
      super.onPause();
      // SensorEventListener 해제
      mSensorManger.unregisterListener(this);
  }
```

SensorEvent 클래스

- Sensor객체 내에는 센서에 대한 정보(센서타입, 시간, 데이터 등)을 담고 있으므로 이를 사용하면 된다.
- public intaccuracy 이 이벤트의 정확도

  public Sensor sensor 이 이벤트를 발생한 센서

  public long timestamp 이벤트가 발생한 시각

  public final float[] values 센서 데이터를 담고 있는 배열.

  센서 타입에따라 길이나 그 내용은 달라짐->예)

  조도센서일경우 int 값 하나라서 values[0]만 쓰이겠지만

  ex) Sensor.TYPE_LIGHT -> values[0]

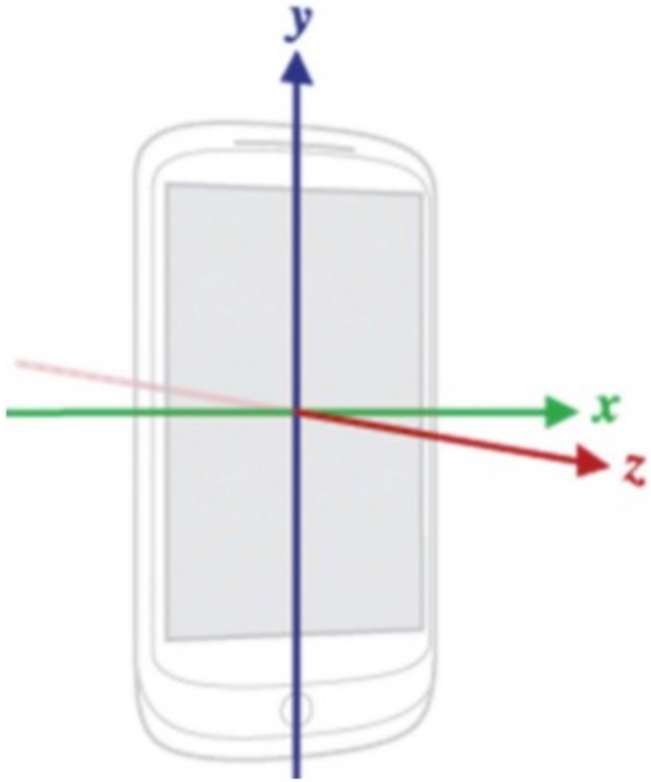
  가속도나 자이로센서 등은 값이 3개씩(x, y, z) 들어가있어서 values[0][1][2]값을 쓴다.

  ex) Sensor.TYPE_ORIENTATION ->

values[0]: 방위각(Azimuth) 북쪽과y축사이각도(0 to 359). z축중심회전각0=North, 90=East, 180=South, 270=West

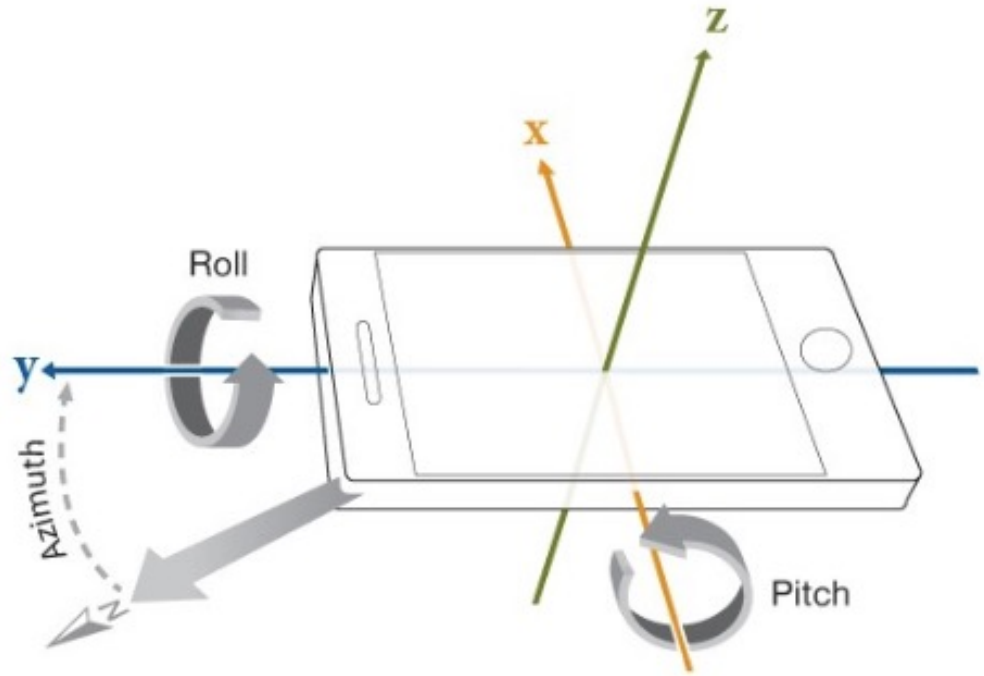
values[1]: 피치(Pitch) x축중심회전각(-180 to 180)

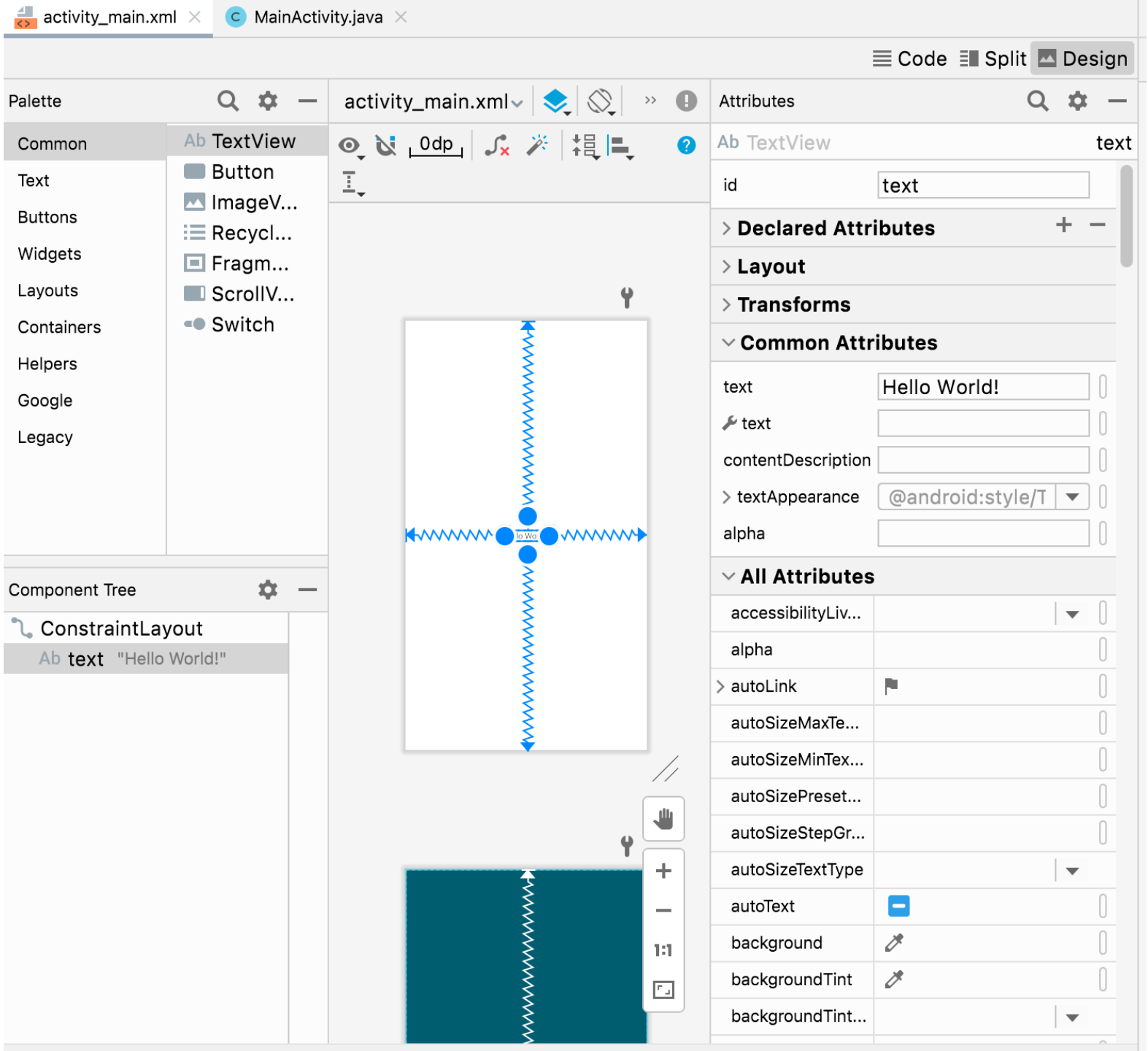
values[2]: 롤(Roll) y축중심회전각(-90 to 90).



지자기 센서(MAGNETIC)

나침반센서라고도 불리며 주변의 자기장의 변화나, 지구의 자기를 측정합니다.

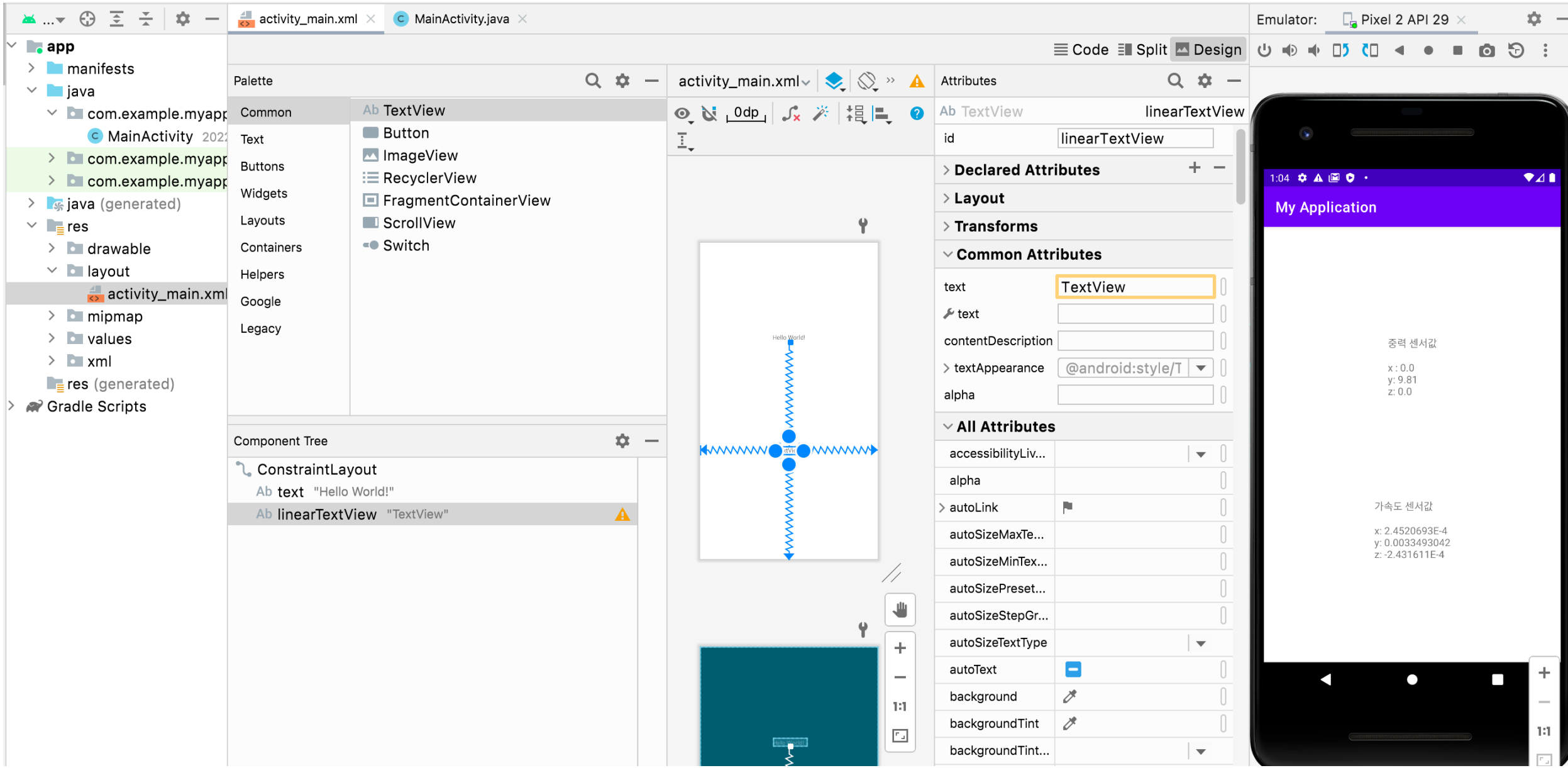



activity_main.xml
MainActivity.java
Code
Split
Design
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageV...
Recycl...
Fragm...
ScrollV...
Switch
activity_main.xml
0dp
Component Tree
ConstraintLayout
text "Hello World!"
Attributes
TextView
text
id
Declared Attributes
Layout
Transforms
Common Attributes
text
Hello World!
contentDescription
textAppearance
@android:style/T
alpha
All Attributes
accessibilityLiv...
autoLink
autoSizeMaxTe...
autoSizeMinTex...
autoSizePreset...
autoSizeStepGr...
autoSizeTextType
autoText
background
backgroundTint
backgroundTint...
1:1


12:55
My Application
방향 센서값
방위각: 0.0
피치: 0.0
롤: 0.0

```java
package com.example.myapplication;

import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity implements SensorEventListener {
    private SensorManager mSensorManger;
    private Sensor mOrientation;
    TextView mText;

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

        mSensorManger = (SensorManager)getSystemService(Context.SENSOR_SERVICE);
        mOrientation = mSensorManger.getDefaultSensor(Sensor.TYPE_ORIENTATION);
        mText = (TextView)findViewById(R.id.text);
    }

```

```
27
28      @Override
29      protected void onResume() {
30          super.onResume();
31          // SensorEventListener 등록
32          mSensorManger.registerListener( listener: this, mOrientation, SensorManager.SENSOR_DELAY_UI);
33      }//this인 이유는 implements로 SensorEventListener 해놓았기때문~
34
35      @Override
36      protected void onPause() {
37          super.onPause();
38          // SensorEventListener 해제
39          mSensorManger.unregisterListener(this);
40      }
41
42      public void onAccuracyChanged(Sensor sensor, int accuracy) {
43
44      }
45
46      public void onSensorChanged(SensorEvent event) {
47          if(event.sensor.getType() == Sensor.TYPE_ORIENTATION) {
48              mText.setText("방향 센서값\n\n방위각: " + event.values[0]
49                      + "\n피치: " + event.values[1] + "\n롤: " + event.values[2]);
50          }
51      }
52  }
```


activity_main.xml
MainActivity.java
Emulator: Pixel 2 API 29
Code
Split
Design
app
manifests
java
com.example.myapp
MainActivity
java (generated)
res
drawable
layout
activity_main.xml
mipmap
values
xml
res (generated)
Gradle Scripts
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageView
RecyclerView
FragmentContainerView
ScrollView
Switch
Component Tree
ConstraintLayout
text "Hello World!"
linearTextView "TextView"
Attributes
TextView
linearTextView
id
Declared Attributes
Layout
Transforms
Common Attributes
text
contentDescription
textAppearance
alpha
All Attributes
accessibilityLiv...
autoLink
autoSizeMaxTe...
autoSizeMinTex...
autoSizePreset...
autoSizeStepGr...
autoSizeTextType
autoText
background
backgroundTint
backgroundTint...
My Application
중력 센서값
x : 0.0
y: 9.81
z: 0.0
가속도 센서값
x: 2.4520693E-4
y: 0.0033493042
z: -2.431611E-4

```java
package com.example.myapplication;

import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity implements SensorEventListener {
    private SensorManager mSensorManger;
    private Sensor mOrientation;
    private Sensor linearSensor;
    TextView mText;
    TextView linearText;

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

        mSensorManger = (SensorManager)getSystemService(Context.SENSOR_SERVICE);
        mOrientation = mSensorManger.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        linearSensor = mSensorManger.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);
        mText = (TextView)findViewById(R.id.text);
        linearText = (TextView)findViewById(R.id.linearTextView);
    }
```

```
31
32      @Override
33      protected void onResume() {
34          super.onResume();
35          // SensorEventListener 등록
36          mSensorManger.registerListener( listener: this, mOrientation, SensorManager.SENSOR_DELAY_UI);
37          mSensorManger.registerListener( listener: this, linearSensor, SensorManager.SENSOR_DELAY_UI);
38      }//this인 이유는 implements로 SensorEventListener 해놓았기때문~
39
40      @Override
41      protected void onPause() {
42          super.onPause();
43          // SensorEventListener 해제
44          mSensorManger.unregisterListener(this);
45      }
46
47      public void onAccuracyChanged(Sensor sensor, int accuracy) {
48
49      }
50
51      public void onSensorChanged(SensorEvent event) {
52          if(event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
53              mText.setText("중력 센서값\n\nx : " + event.values[0]
54                      + "\ny: " + event.values[1] + "\nz: " + event.values[2]);
55          }else if(event.sensor.getType() == Sensor.TYPE_LINEAR_ACCELERATION){
56              linearText.setText("가속도 센서값\n\nx: " + event.values[0]
57                      + "\ny: " + event.values[1] + "\nz: " + event.values[2]);
58          }
59      }
60  }
```

Sensor Framework

- SensorManager

  SensorManager sensorManager;

  이름대로 센서들을 관리 하는 역할의 모듈입니다.

  각각의 센서를 선언 할 수 있으며, 센서값을 내보냅니다.

  또한, 센서를 보정하기위한 상수를 제공

- Sensor

  Sensor sensor;

  특정 센서의 인스턴스를 만들기위해 해당 클래스를 사용합니다.

  이를 통해 샌서의 기능을 선택할 수 있습니다.

- SensorEvent/Listener

```
public class MainActivity extends AppCompatActivity implements SensorEventListener
```

센서 이벤트와 리스너의 경우. 센서값이 변하는등의 이벤트가 발생한다. 해당 센서의 값을 읽어들일 수 있다

onCreate에서

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

sensor = sensorManager.getDefaultSensor(Sensor.TYPE_PRESSURE);
```

위와 같은 형식으로. 설정을 해줍니다.

getDefaultSensor에서는 사용할 센서들을 선택할 수 있다.

onResum에서

```
sensorManager.registerListener(this, sensor, SensorManager.SENSOR_DELAY_UI);
```

위와 같이 설정.

첫번째 인자값은 이벤트 리스너를 설정하는데, 메인액티비티에 implement를 하였기 때문에 this로

sensor에서 어떠한 센서를 사용하였는지 설정을 해두었기 때문에 sensor을 두번째 인자값으로

마지막은 표현하자면 센서값을 갱신하는 속도에 관한 인자값.

```
@Override
public void onSensorChanged(SensorEvent event) {
    Sensorvalue = event.values[0];
}
```

위와 같이 센서값이 변할때의 이벤트가 발생하는데. 위의경우, 하나의 데이터만 읽어들였기 때문에.

바로 event의 values를 가져와 읽을 수 있다.

다수의 데이터를 읽어들이기 위해서는

```
@Override
public void onSensorChanged(SensorEvent event) {
    if(event.sensor == sensor){
        Sensorvalue = event.values[0];
    }
}
```

다음과 같이 해당 센서의 값이 맞는지 확인.

한두개가 아니거나 훨씬 많아질때는 switch case를 사용하여 구분.