

Week11.  
**위치센서**



# 개발환경 구축 절차

2

주 차	수업 내용
1	수업 소개
2	개발 환경 구축과 맛보기 프로젝트
3	텍스트 출력과 레이아웃
4	이미지의 출력
5	이벤트 처리와 액티비티 간 이동
6	오디오 재생
7	비디오 재생
8	중간고사
9	애니메이션
10	사물인터넷과 센서 – 터치 센서, 모션 센서
11	사물인터넷과 센서 – <b>위치 센서</b> , 환경 센서
12	NFC 활용
13	공공 DB 오픈 API 활용
14	구글 맵과 위치 추적
15	기말 고사

# 강의 자료 - <https://github.com/hopypark>

Lecture2018/AndroidApp at master · GitHub

Branch: master

Lecture2018 / AndroidApp /

hopypark Add files via upload

Latest commit adde5e8 14 minutes ago

..	Add files via upload	15 days ago
resources	Add files via upload	a month ago
sources	Add files via upload	3 months ago
Week01.강의 소개.pdf	Add files via upload	2 months ago
Week02.Chap02.앱 개발환경 구축.pdf	Add files via upload	2 months ago
Week02.Chap03.앱 프로젝트 구조와 실행원리.pdf	Add files via upload	2 months ago
Week03.Chap06.텍스트 출력과 레이아웃.pdf	Add files via upload	2 months ago
Week04.Chap07.이미지의 출력.pdf	Add files via upload	2 months ago
Week05.Chap08.이벤트 처리와 액티비티 간 이동.pdf	Add files via upload	2 months ago
Week06.Chap09.오디오 재생.pdf	Add files via upload	a month ago
Week07.Chap10.비디오 재생.pdf	Add files via upload	a month ago
Week08.Chap11.애니메이션.pdf	Add files via upload	22 days ago
Week09.Chap12.터치сен서.pdf	Add files via upload	13 days ago
Week10.Chap13.모션센서.pdf	Add files via upload	6 days ago
Week11.Chap14.위치센서.pdf	Add files via upload	14 minutes ago
readme.md	Create readme.md	3 months ago

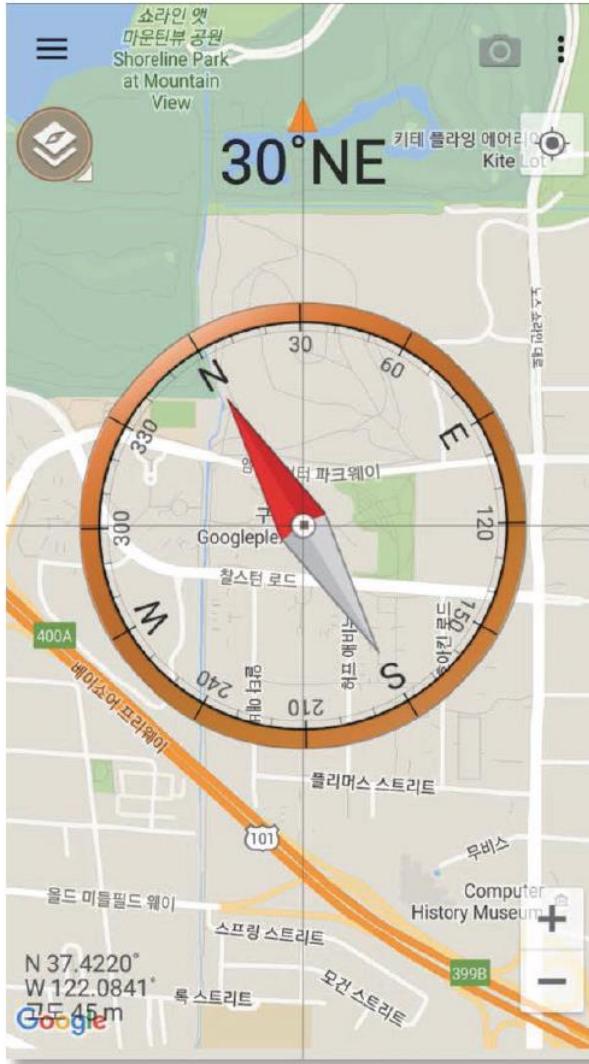
# 위치 센서를 이용한 앱의 예

6

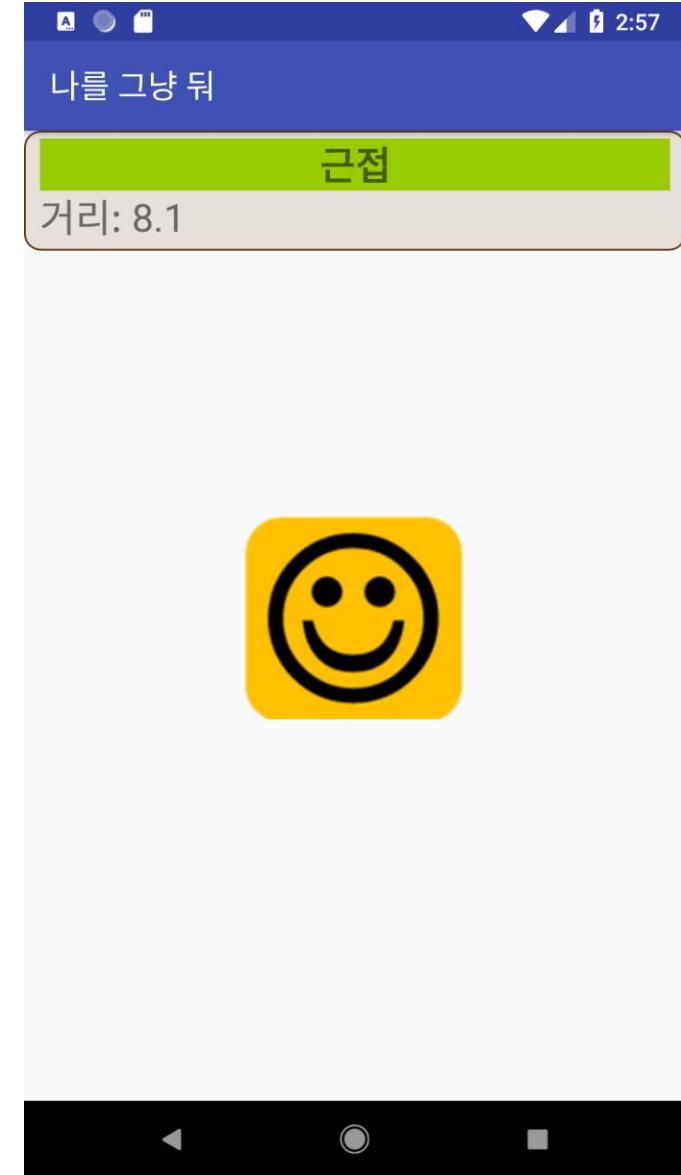


(a) 카메라 화면과 방향

• Smart Compass 앱

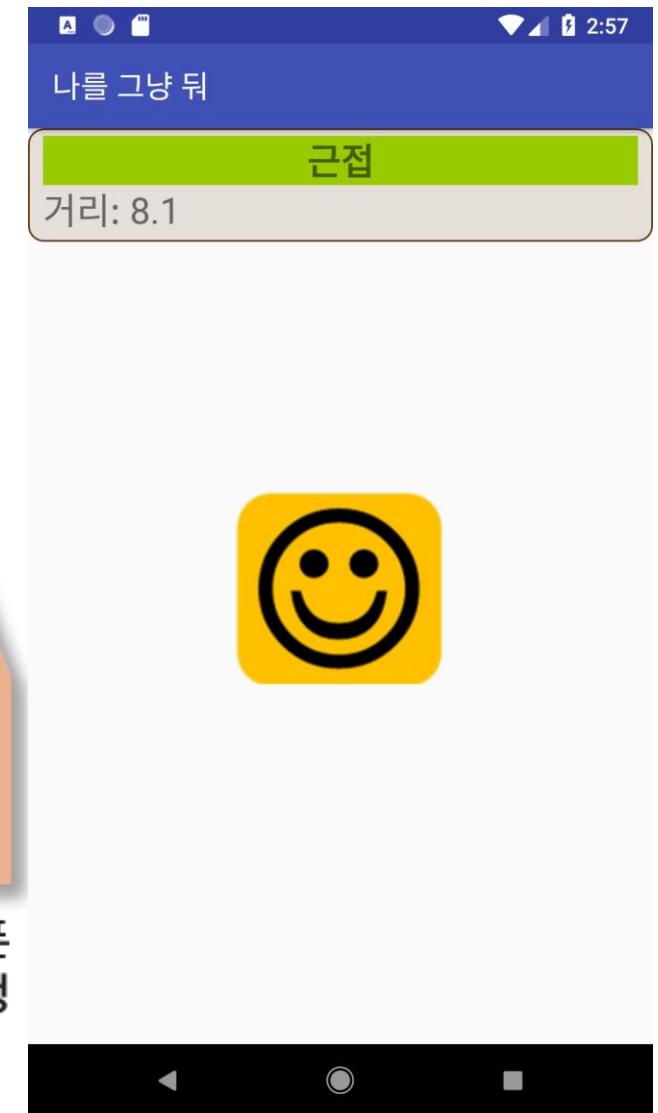
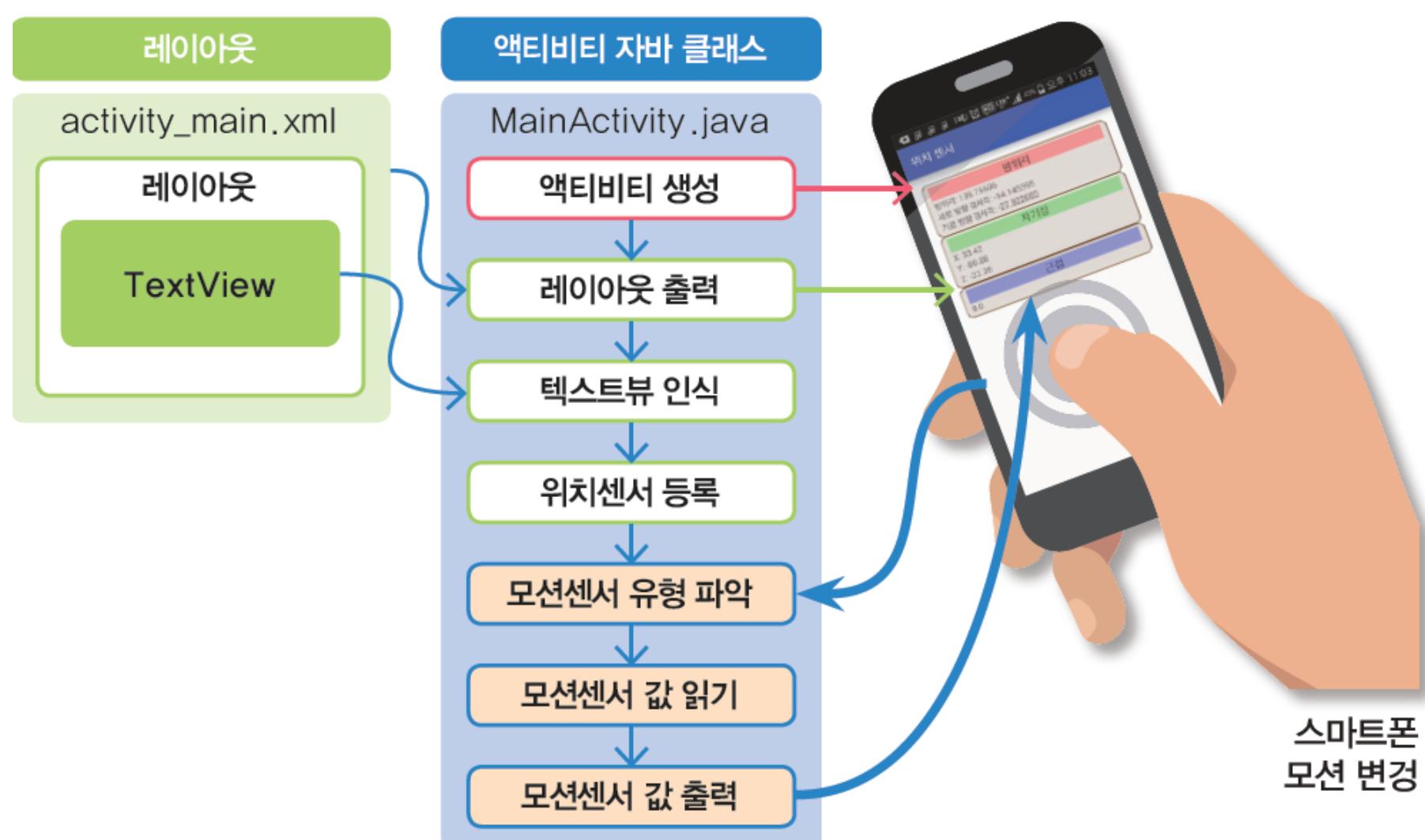


(b) 구글맵 상의 위치와 방향



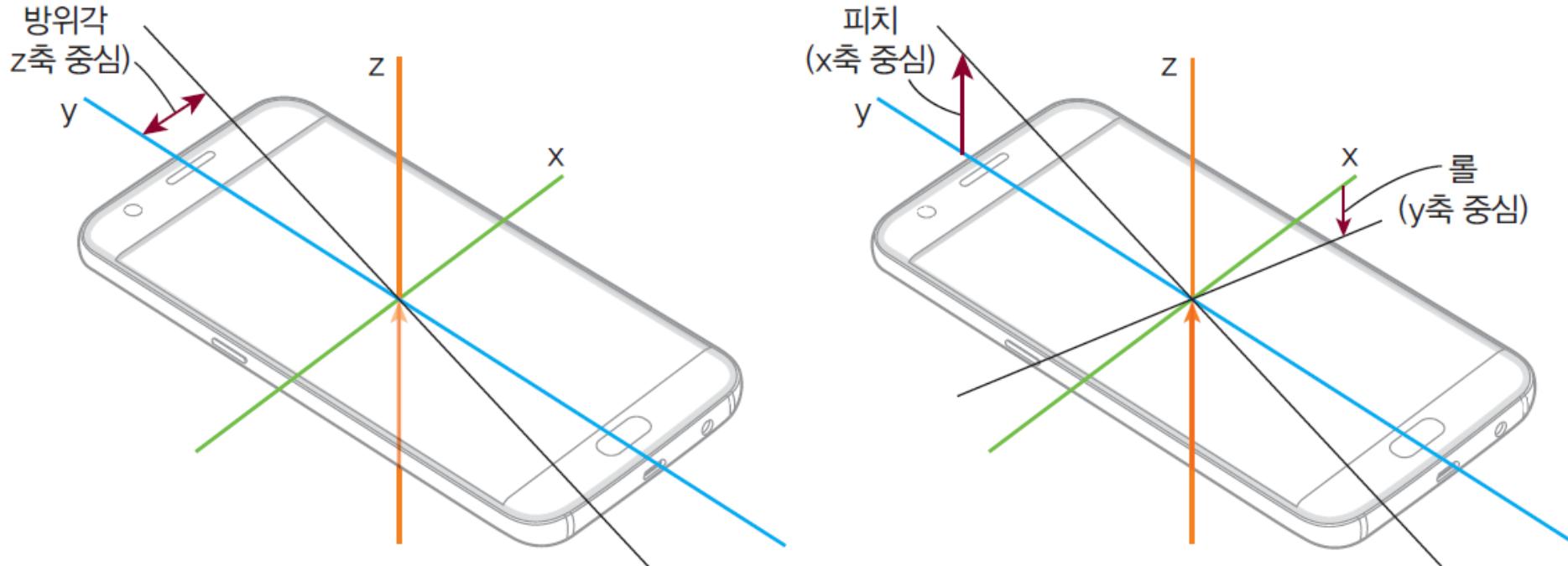
# 모션 센서 원리

7



# 모션 센서는 센서 값을 표현하기 위해 세가지 축 사용

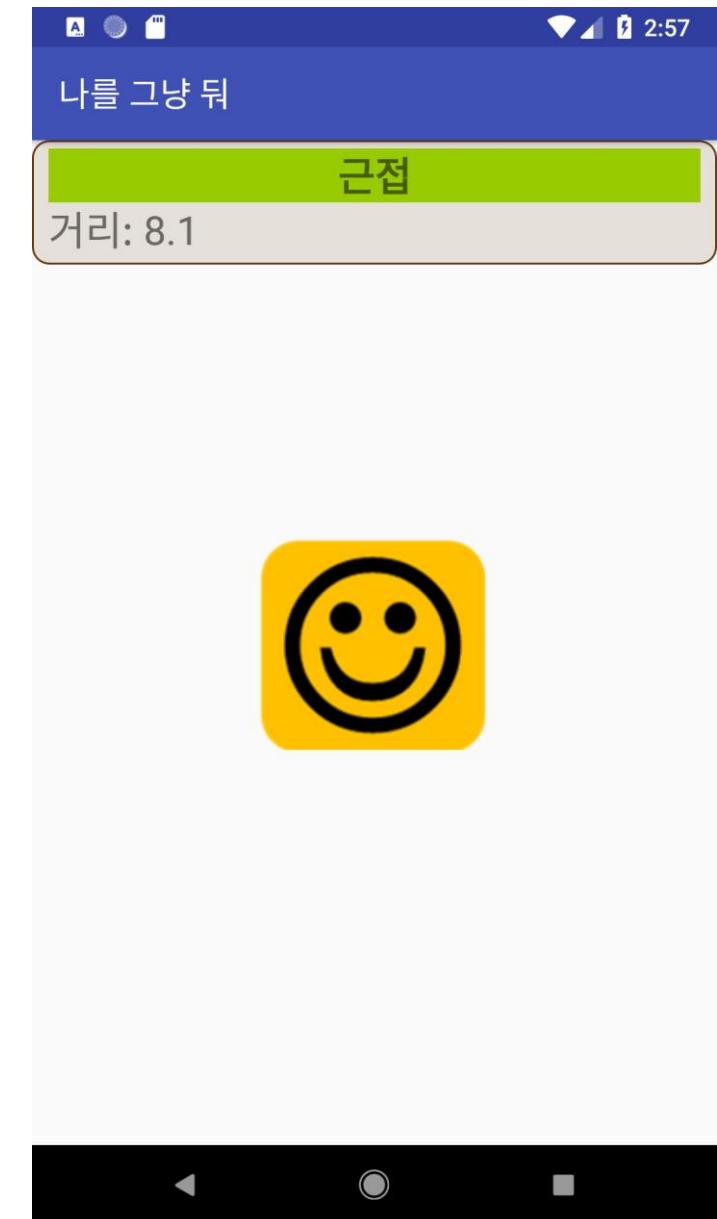
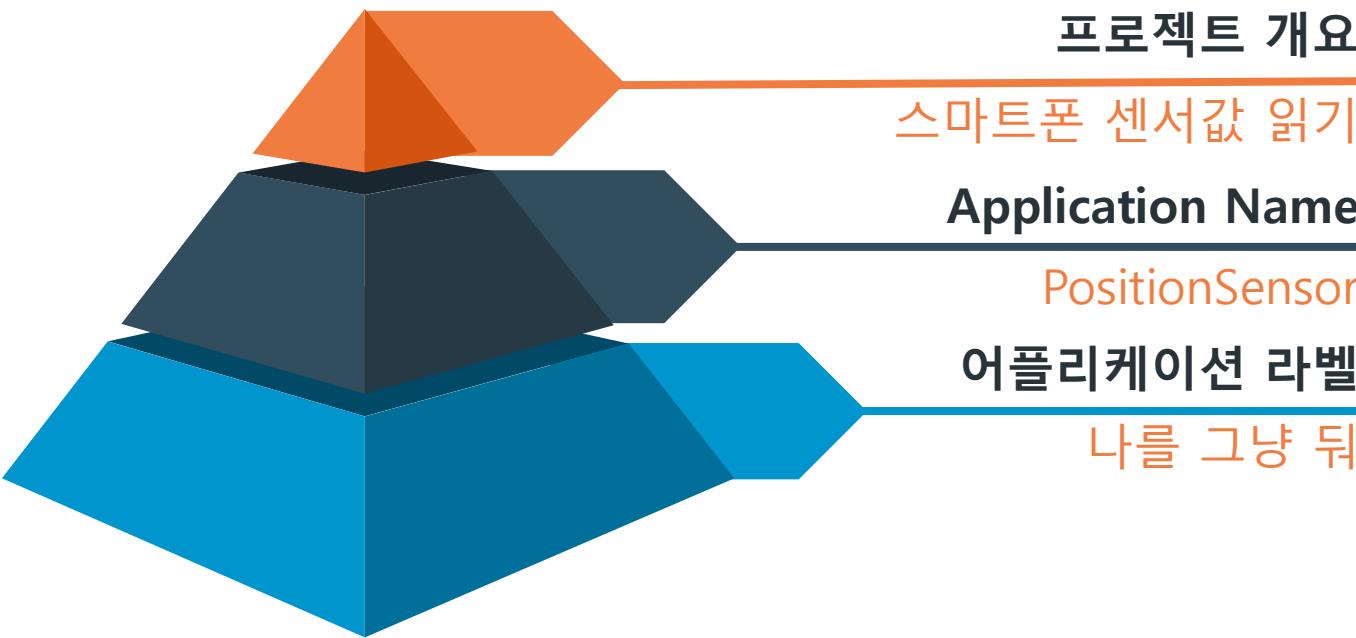
8



속성	설명
롤 Roll(왼쪽-오른쪽)	평평한 상태는 0도이고, 왼쪽으로 기울면 90도까지 증가한다. 반대로 오른쪽으로 기울이면 -90도까지 감소한다.
피치 pitch(위-아래)	평평한 상태는 0도로, 핸드폰의 윗부분을 땅 쪽으로 기울이면 90도까지 증가하고, 더 기울여 뒤집어지면 180도까지 증가한다. 반대로 핸드폰의 아랫부분을 땅 쪽으로 기울이면 -90도까지 감소하며, 더 기울여 뒤집어지면 -180도까지 감소한다.
아지무스 Azimuth(방위각)	폰의 위가 북쪽을 가리키면 0도, 동쪽을 가리키면 90도, 남쪽을 가리키면 180도, 서쪽을 가리키면 270도이다.

# Step 0. 프로젝트 개요

9



# Create Project – PositionSensor

10

Create New Project X

## Create Android Project

Application name  
PositionSensor

Company domain  
user.example.com

Project location  
C:\Users\Kyungtae\AndroidStudioProjects\user\PositionSensor ...

Package name  
com.example.user.positionsensor Edit

**Include C++ support**

**Include Kotlin support**

Previous Next Cancel Finish

Create New Project X

## Target Android Devices

**Select the form factors and minimum SDK**

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

Phone and Tablet

API 27: Android 8.1 (Oreo) ▼

By targeting **API 27 and later**, your app will run on < 1% of devices. [Help me choose](#)

Include Android Instant App support

Wear

API 21: Android 5.0 (Lollipop) ▼

TV

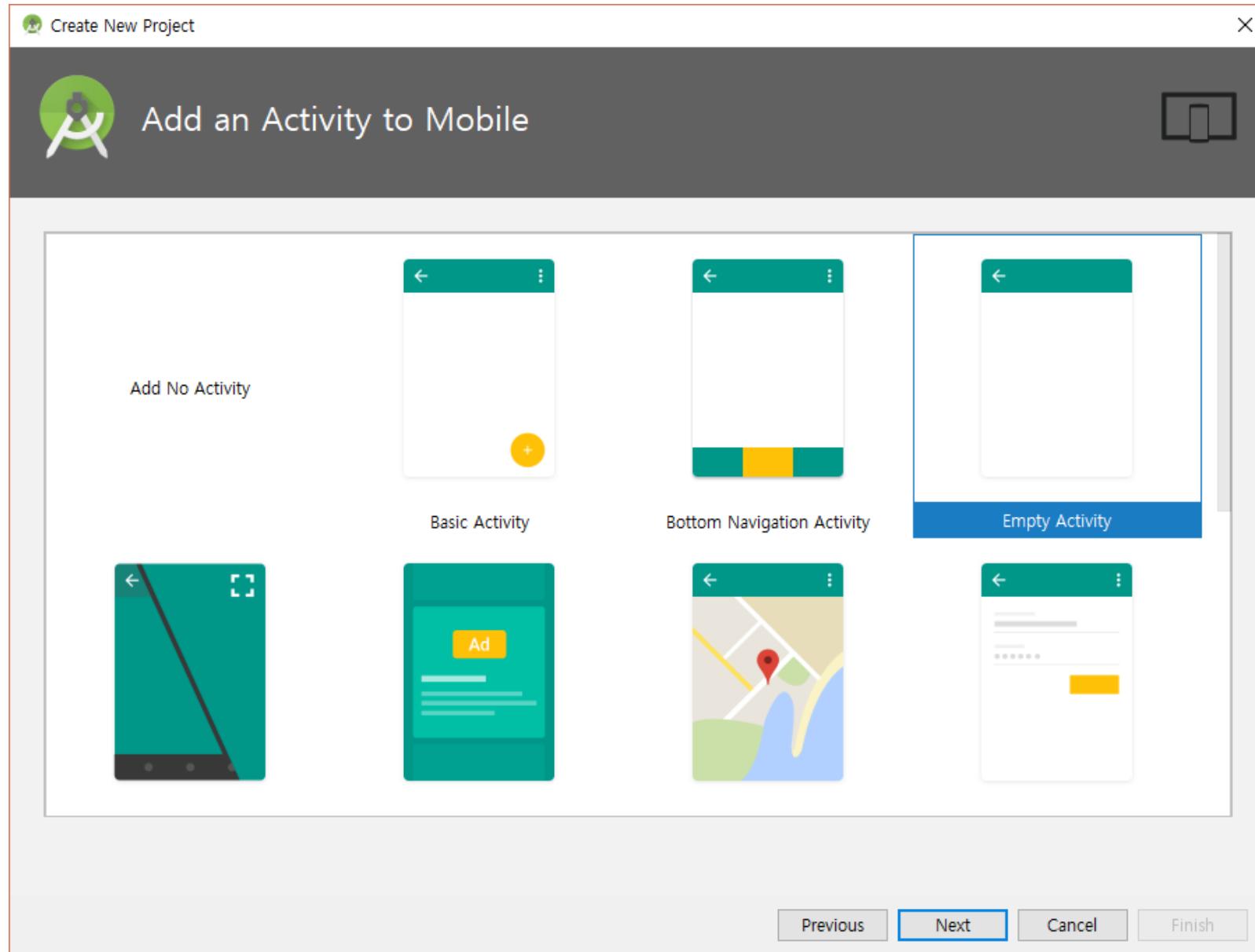
API 21: Android 5.0 (Lollipop) ▼

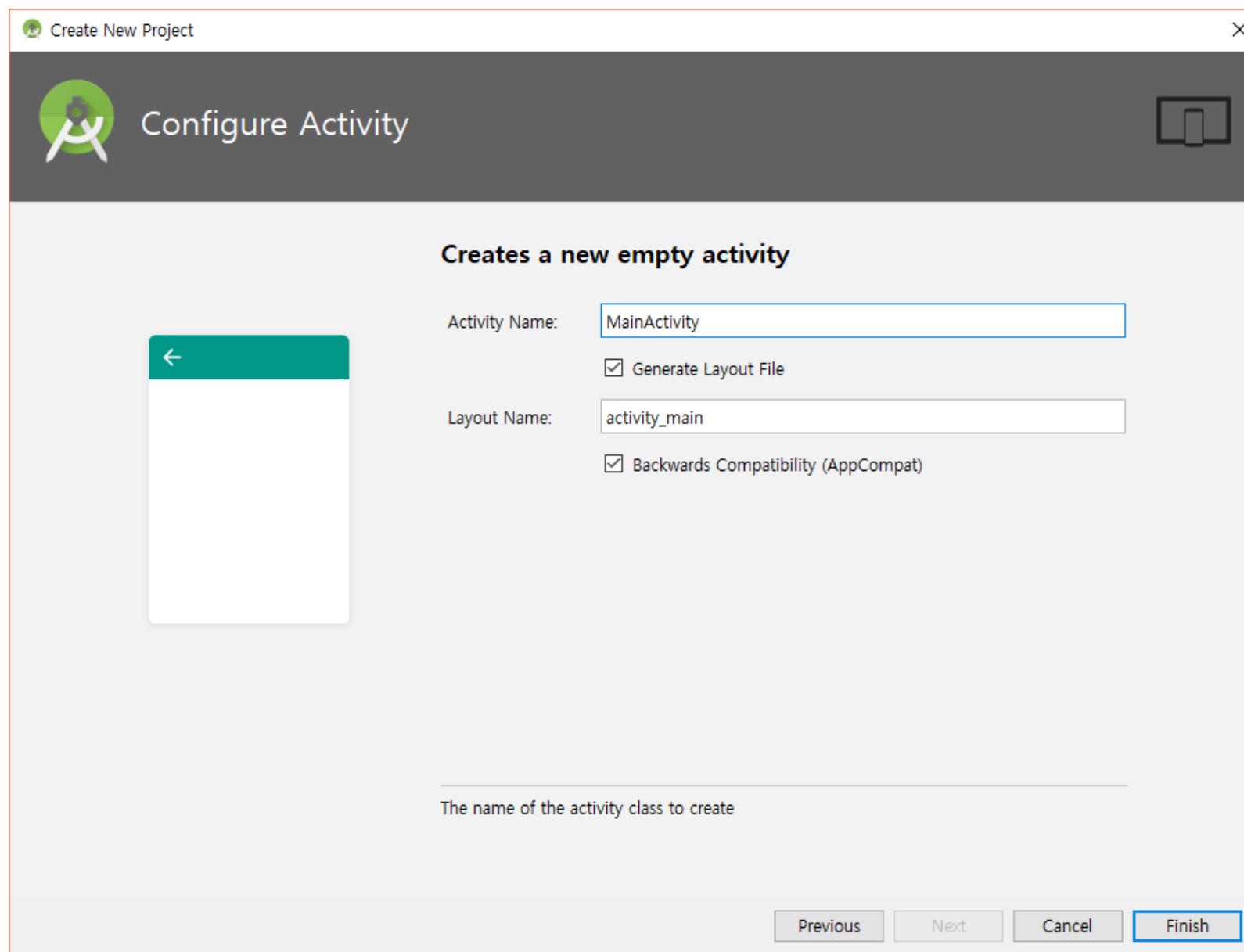
Android Auto

Android Things

API 24: Android 7.0 (Nougat) ▼

[Previous](#) Next [Cancel](#) [Finish](#)





# Step 1. 프로젝트 생성

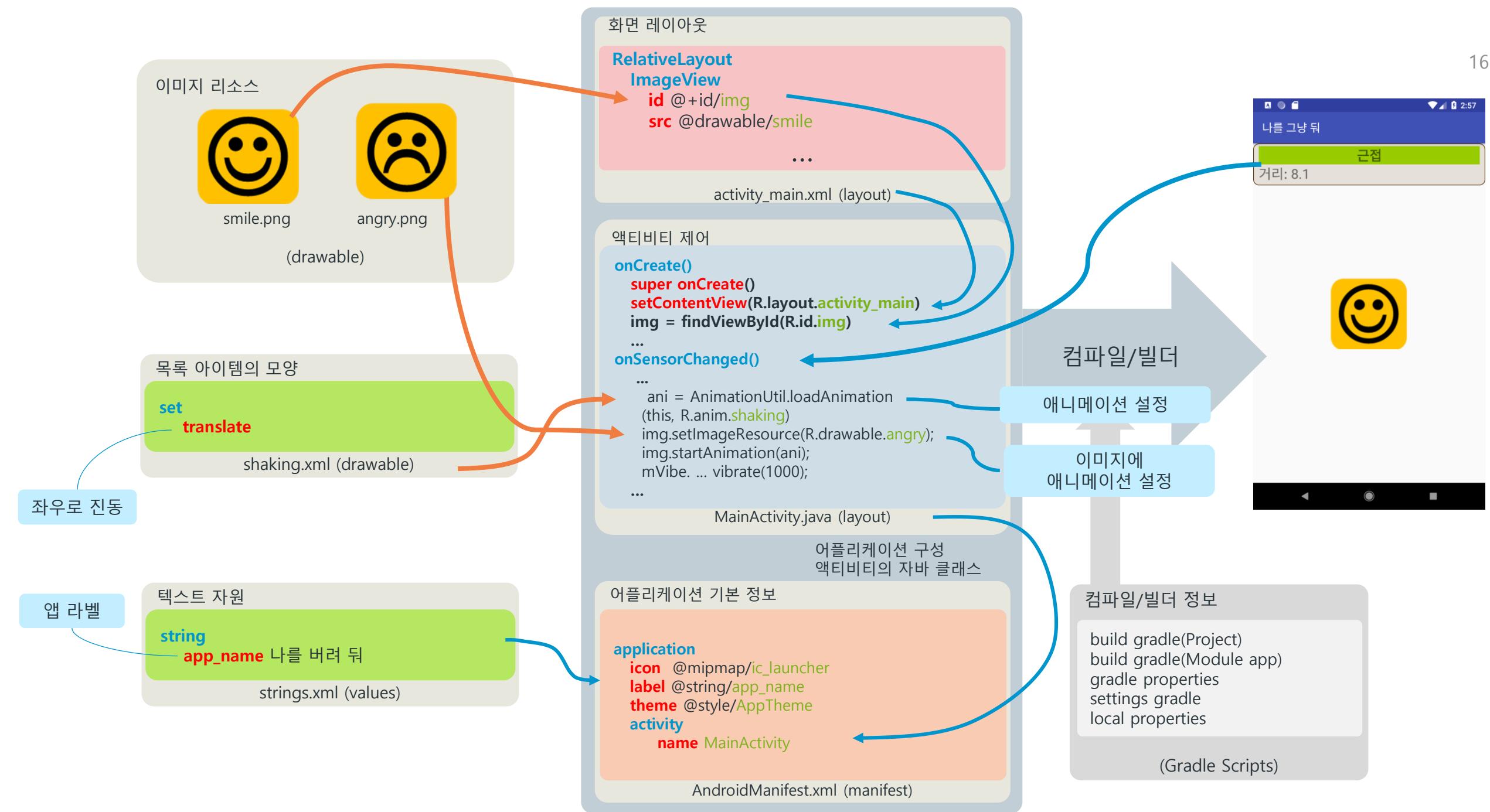
14

절차	내용
① 프로젝트 시작	메뉴에서 'File → New Project' 클릭
② 프로젝트 구성	Application Name: <b>PositionSensor</b>
	Company Domain: <b>kyungtae.example.com</b> (디폴트 사용)
	Project Location: <b>~/AndroidStudioProject/ktpark/PositionSensor</b>
③ 제품형태	Phone and Tablet(사용할 안드로이드 버전 지정: <b>Android 8.1 Oreo</b> )
④ 액티비티 유형	<b>Empty Activity</b>
⑤ 파일 옵션	Activity Name: <b>MainActivity</b> (디폴트 사용)
	Layout Name: <b>activity_main</b> (디폴트 사용)

# Step 2. 파일 편집

15

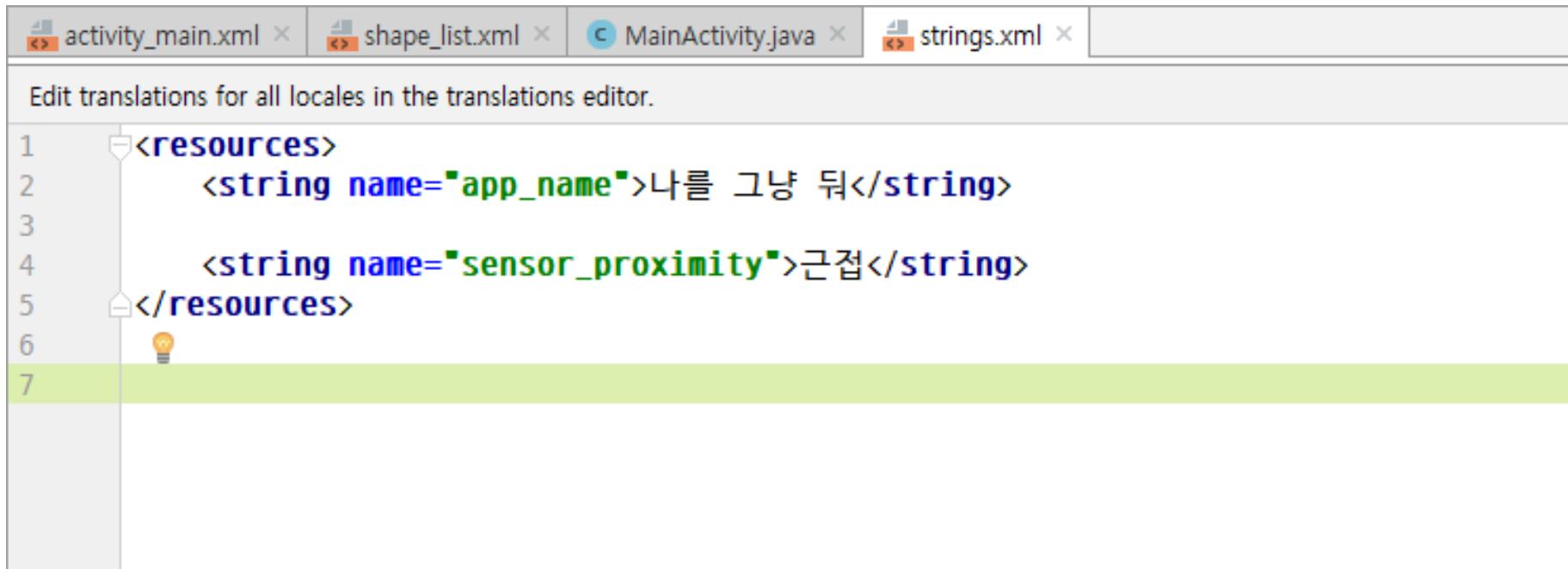
모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	
java	com.example.kyungtae.video1	MainActivity.java	<ul style="list-style-type: none"><li>• 센서 등록</li><li>• 센서 종류 확인 및 값 변경 확인</li><li>• 근접이면 이미지 변경 및 진동</li></ul>
res	anim	shaking.xml	<ul style="list-style-type: none"><li>• 아이콘 이미지의 진동 애니메이션</li></ul>
	drawable	shape_list	<ul style="list-style-type: none"><li>• 출력모양 설계(배경색)</li></ul>
	layout	activity_main.xml	<ul style="list-style-type: none"><li>• 이미지의 화면 중앙 배치</li></ul>
	mipmap	ic_launcher.png	
	values	colors.xml	
		dimens.xml	
		strings.xml	<ul style="list-style-type: none"><li>• 어플리케이션 라벨 수정</li><li>• “나를 그냥 둬”의 문자열 추가</li></ul>
		styles.xml	



# Step 2.1 텍스트 자원의 편집

17

- strings.xml

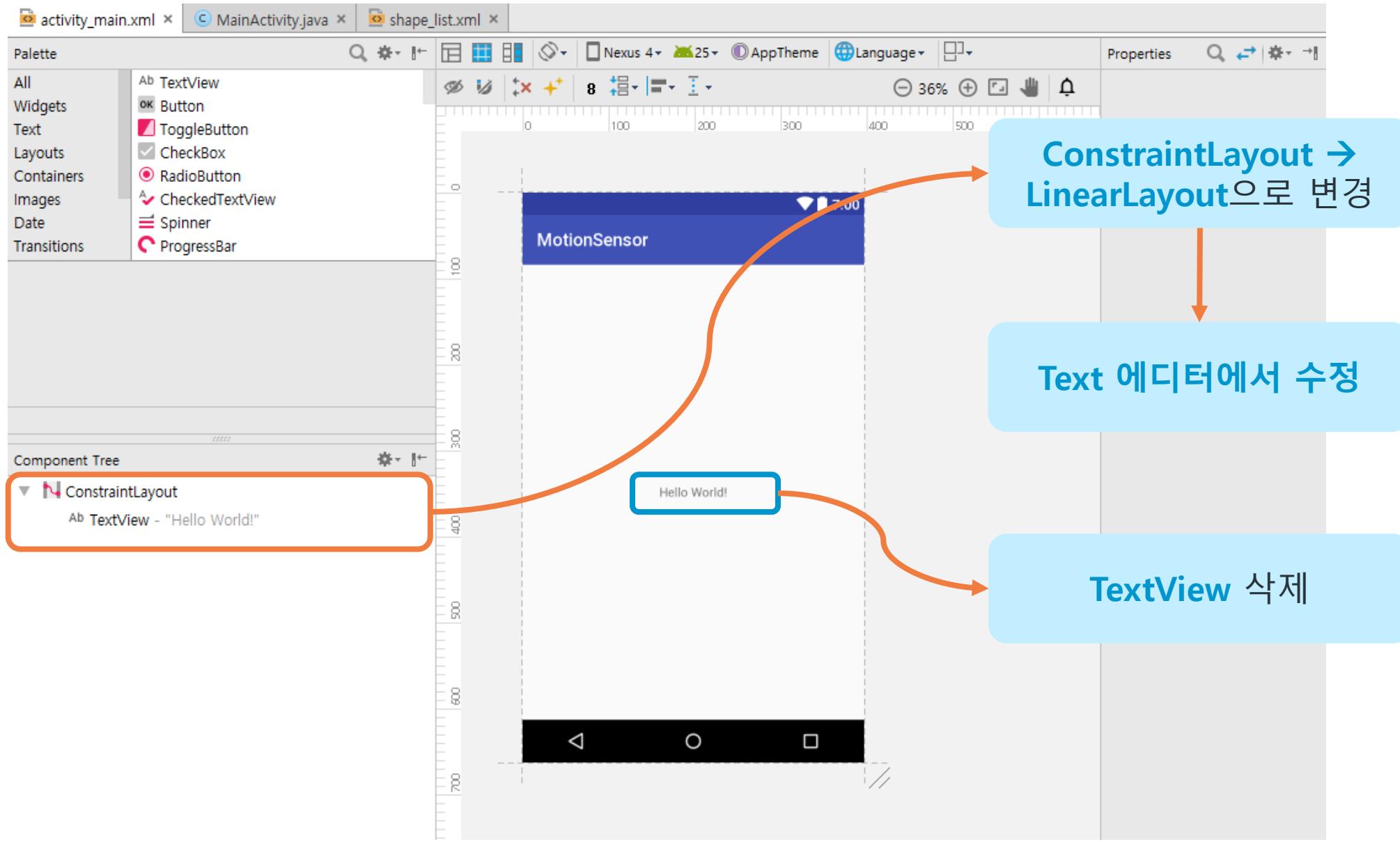


The screenshot shows the Android Studio code editor with the tabs bar at the top containing activity\_main.xml, shape\_list.xml, MainActivity.java, and strings.xml. The strings.xml tab is active. The editor displays the following XML code:

```
1 <resources>
2   <string name="app_name">나를 그냥 둬</string>
3
4   <string name="sensor_proximity">근접</string>
5 </resources>
6
7
```

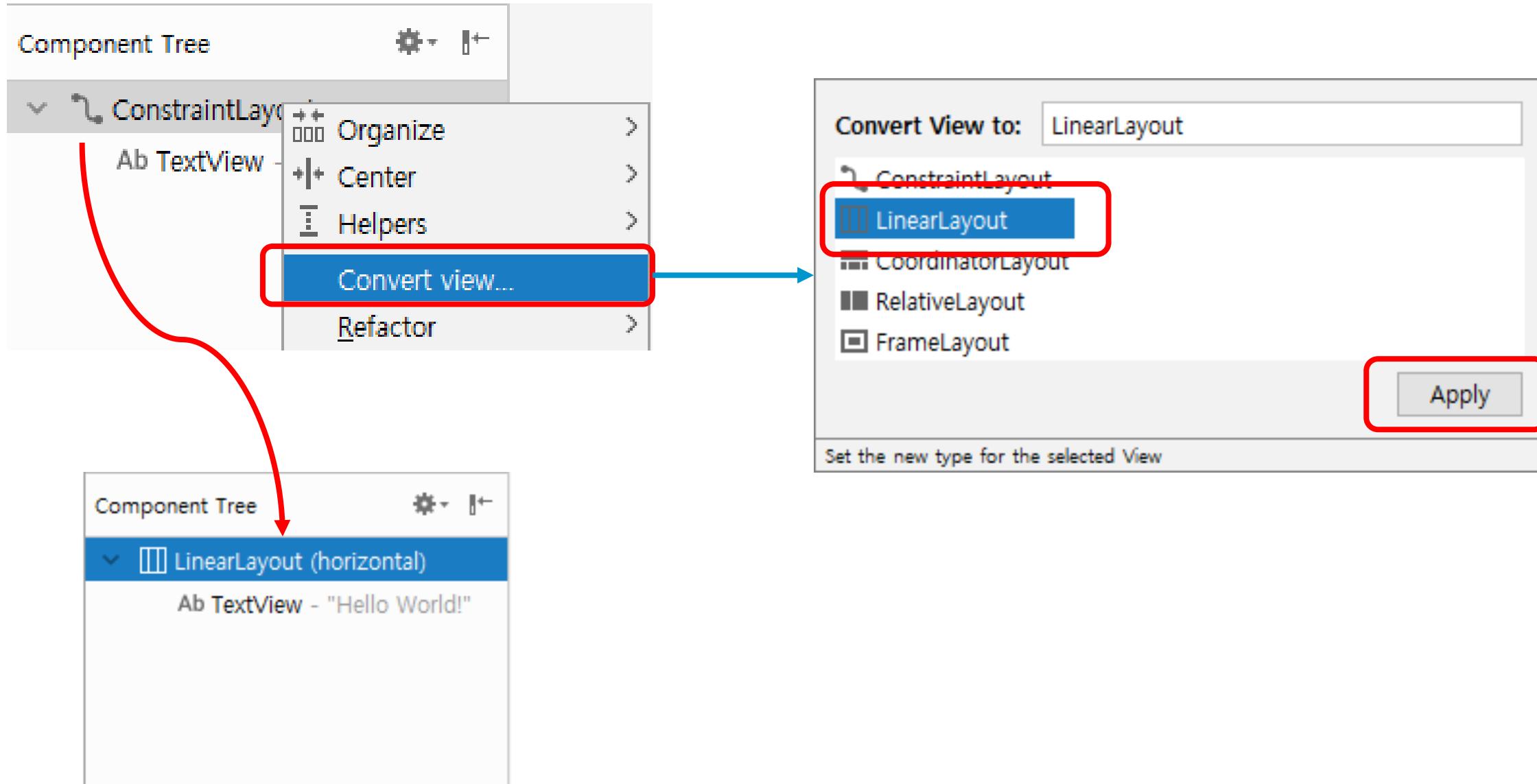
The code is color-coded: blue for tags like <resources>, <string>, and </resources>, green for attribute values like "app\_name" and "sensor\_proximity", and black for the Korean text. Line 7 is highlighted with a light green background.

## 2.2 화면 설계



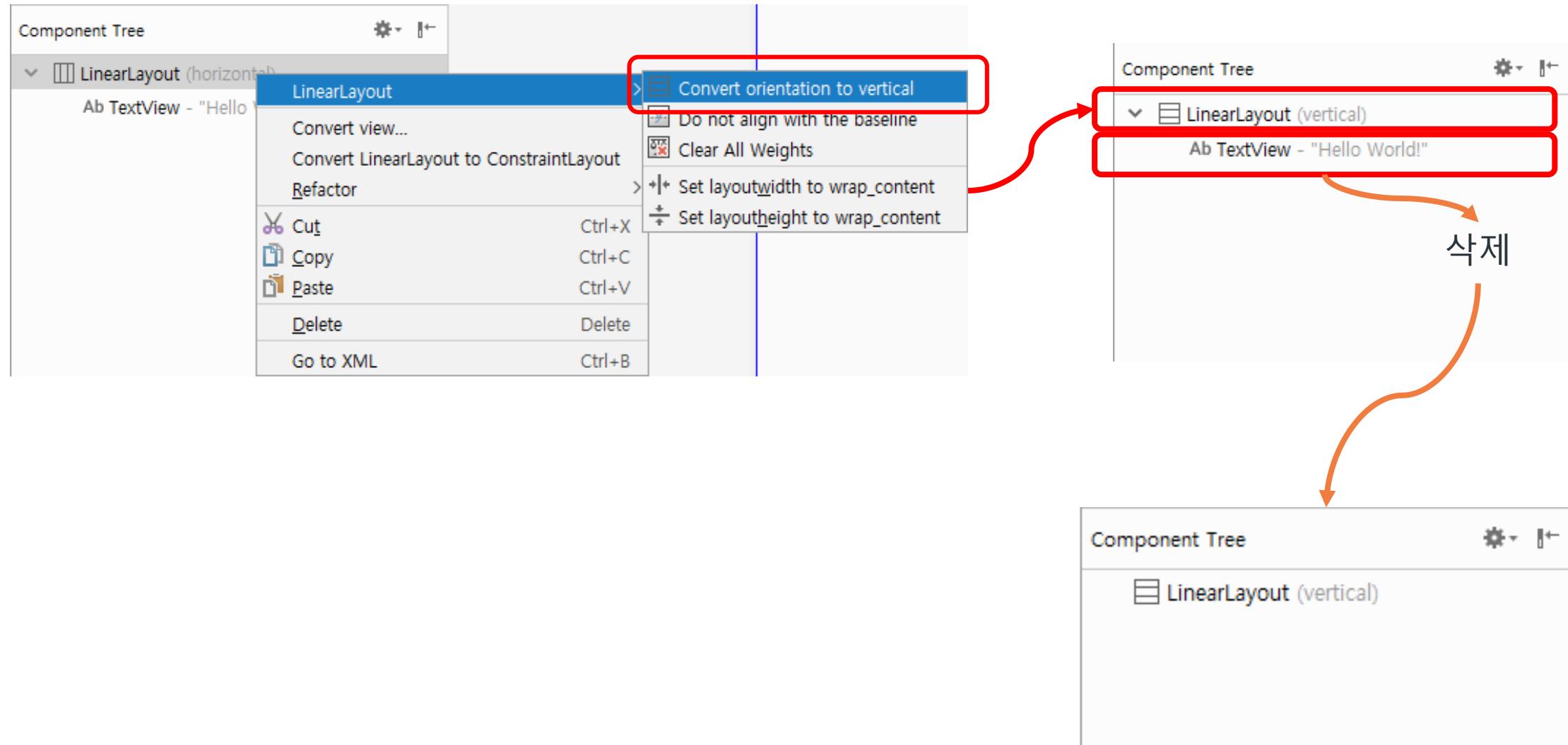
# ConstraintLayout을 LinearLayout로 바꾸기

19



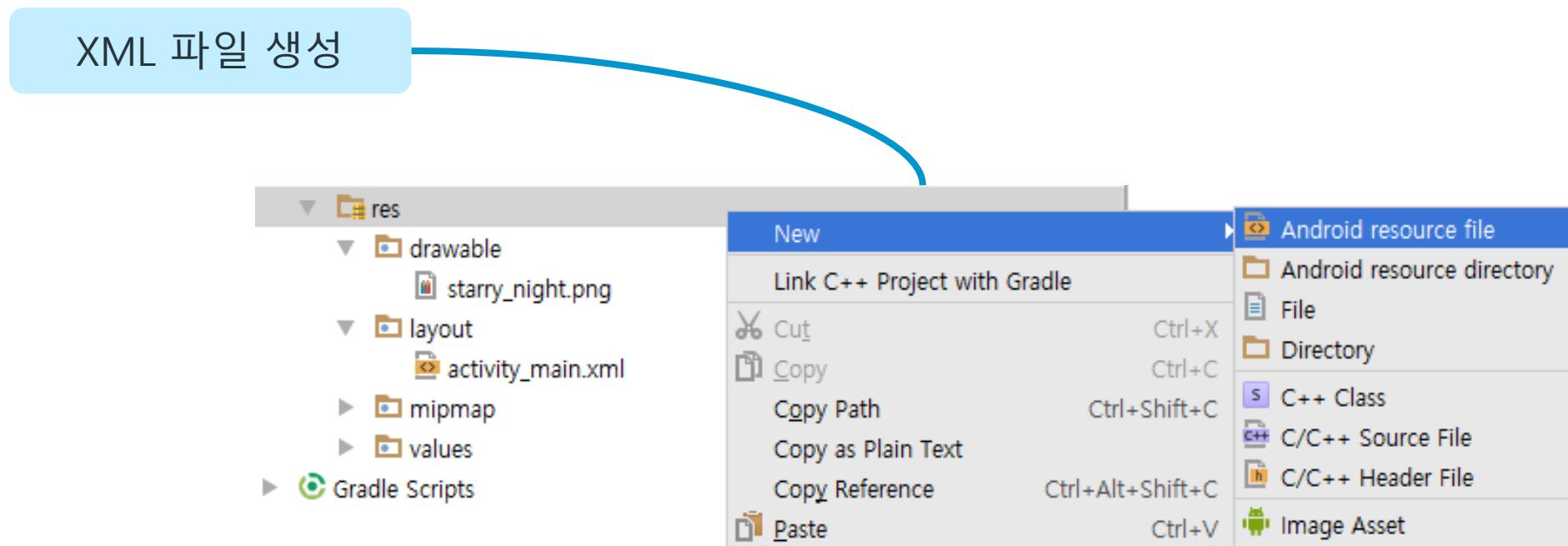
# LinearLayout의 방향을 Horizontal → Vertical로 변경하기

20



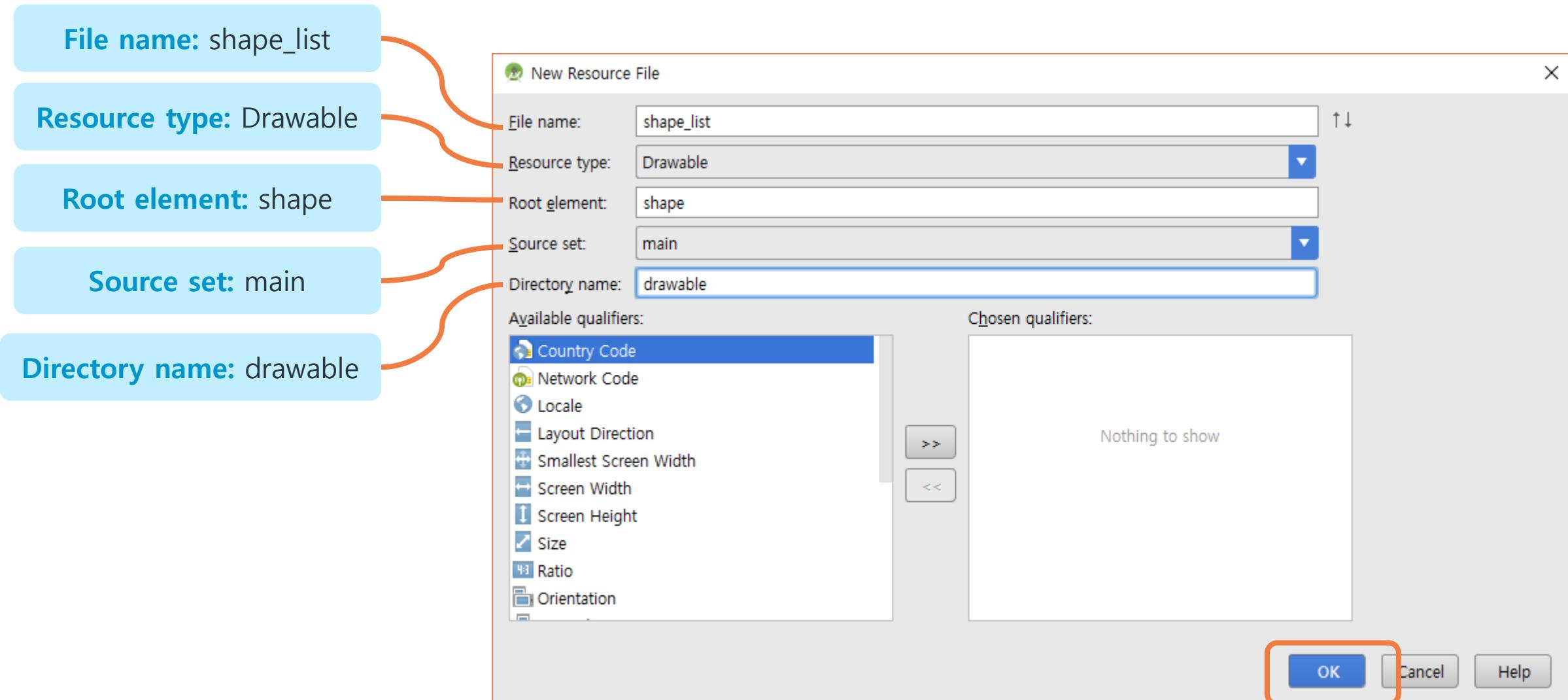
## 2.3 drawable 리소스 - shape\_list.xml 추가

- **shape\_list.xml** 생성(res/drawable 폴더)
  - drawable resource를 이용한 그림 출력



- Set New Resource File - shape\_list.xml

26



## • shape\_list.xml

The screenshot shows the Android Studio interface with three tabs: activity\_main.xml, MainActivity.java, and shape\_list.xml. The shape\_list.xml tab is active, displaying the following XML code:

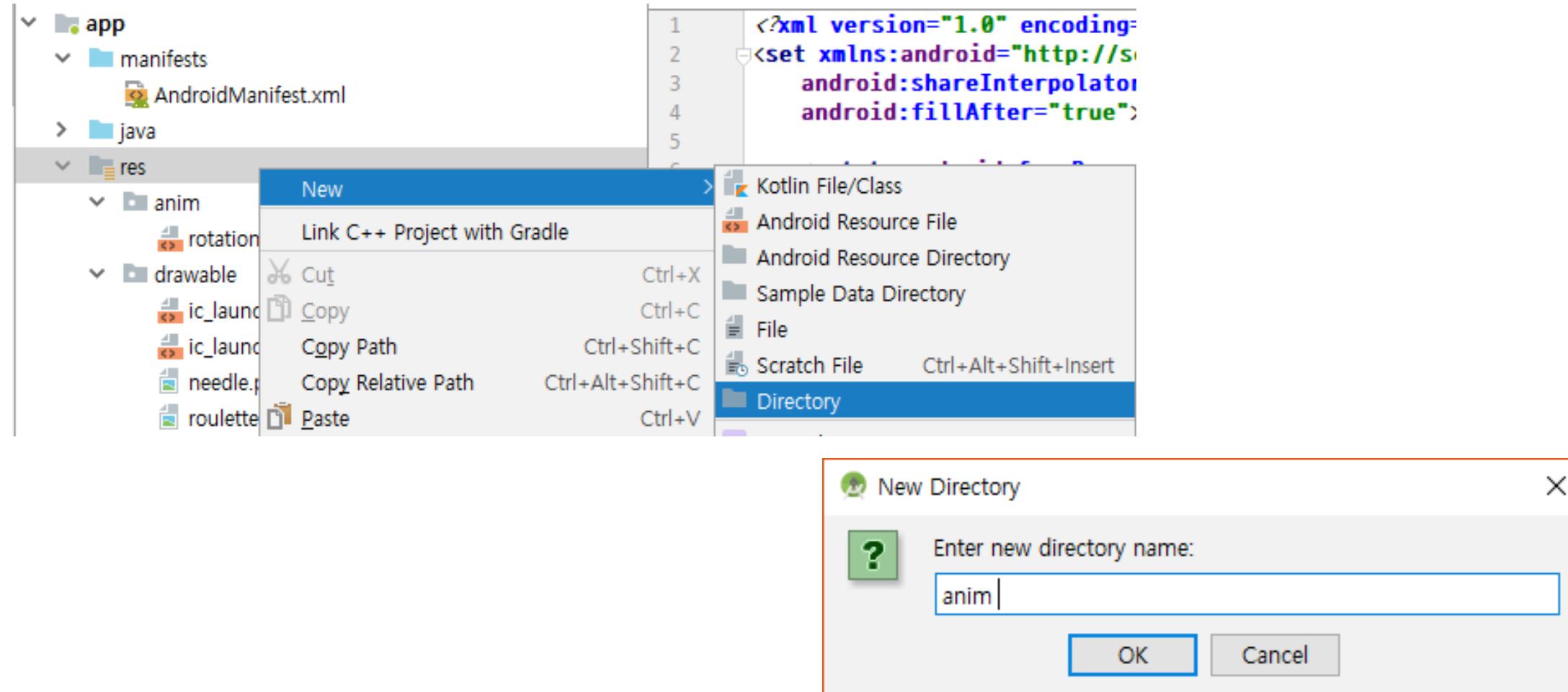
```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <solid android:color="#3061380B"/>
    <stroke android:width="1dp" android:color="#61380B"/>
    <padding
        android:top="2dp"
        android:bottom="2dp"
        android:left="10dp"
        android:right="10dp">
    </padding>
    <corners android:radius="5dp"></corners>
</shape>
```

On the right, the Preview window shows a brown rectangular box with a thin black border, rounded corners, and internal padding. Four callout boxes point to specific XML attributes:

- 출력모양을 내부의 색 (Output shape to inner color) - Points to the <solid> element.
- 출력모양을 테두리의 색 (Output shape to border color) - Points to the <stroke> element.
- 내부 패딩 정보 (Internal padding information) - Points to the <padding> block.
- 출력모양 모서리를 둥근 모양으로 지정(반지름은 5dp) (Specify rounded output shape (radius is 5dp)) - Points to the <corners> element.

# Animation 객체 추가 – res/anim 폴더 만들기

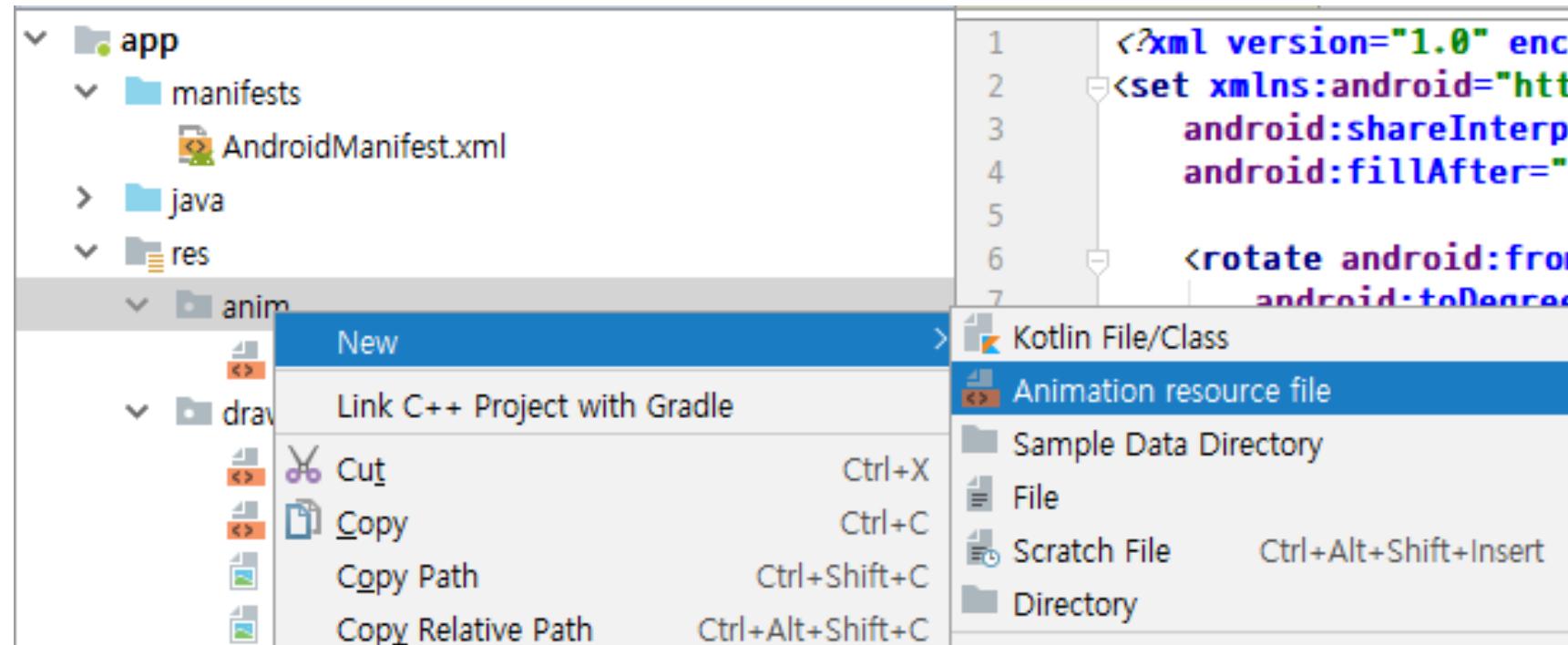
- 애니메이션 설정을 위한 xml 파일 생성



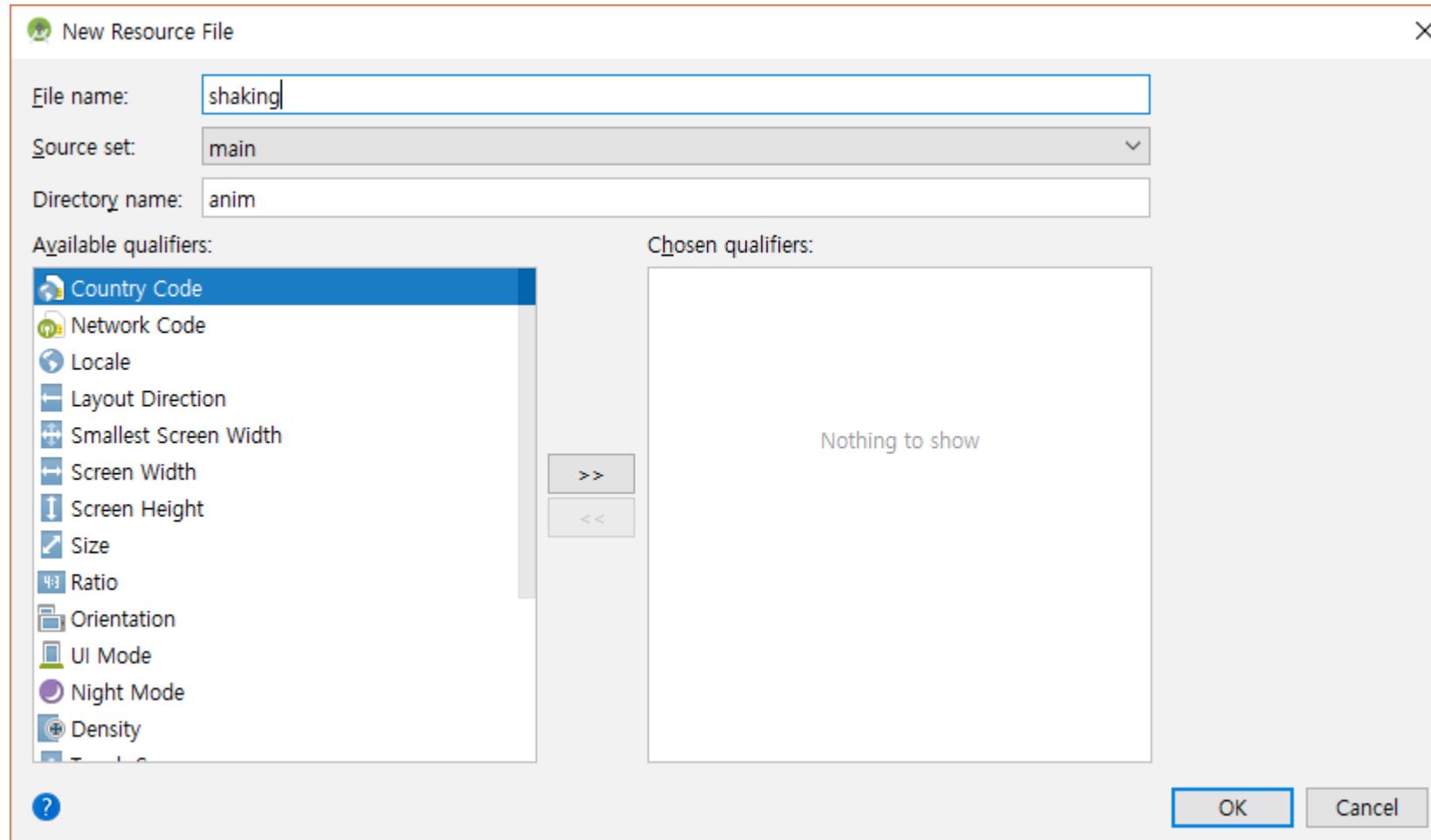
# Animation 객체 추가 – rotation.xml 파일 만들기

31

- 애니메이션 설정을 위한 xml 파일 생성



# shaking.xml 파일 만들기

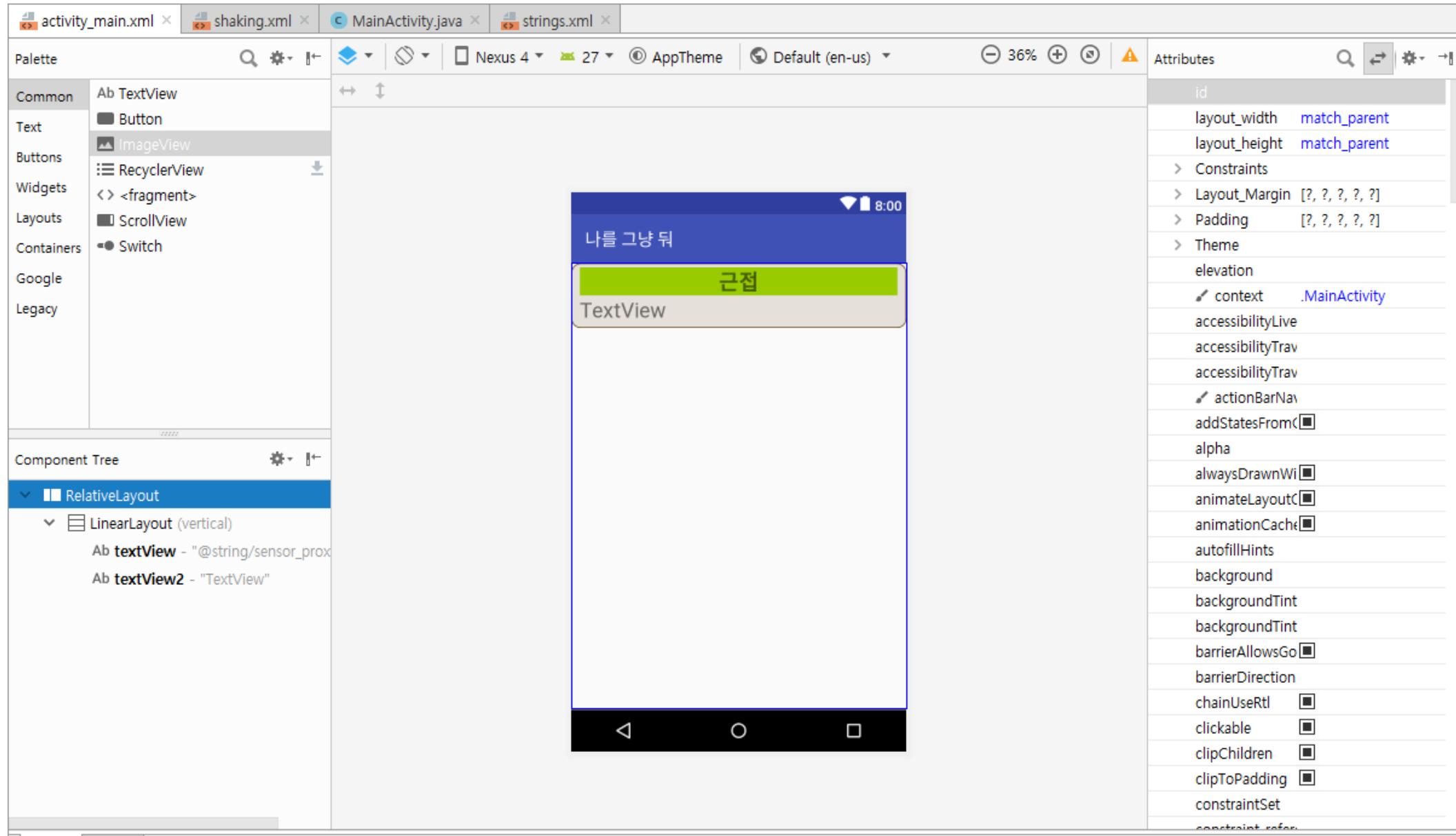


# shaking.xml 파일

```
activity_main.xml × shaking.xml × MainActivity.java × strings.xml ×
1 <?xml version="1.0" encoding="utf-8"?>
2 <set xmlns:android="http://schemas.android.com/apk/res/android">
3     <!-- 이동위치 변화:
4         x방향으로 이미지너비의 2% 크기만큼을 100msec 초 동안 10회 반복 --&gt;
5     &lt;translate
6         android:fromXDelta="-2%"
7         android:toXDelta="2%"
8         android:duration="100"
9         android:repeatCount="10"&gt;
10    &lt;/translate&gt;
11
12 &lt;/set&gt;
13
14</pre>
```

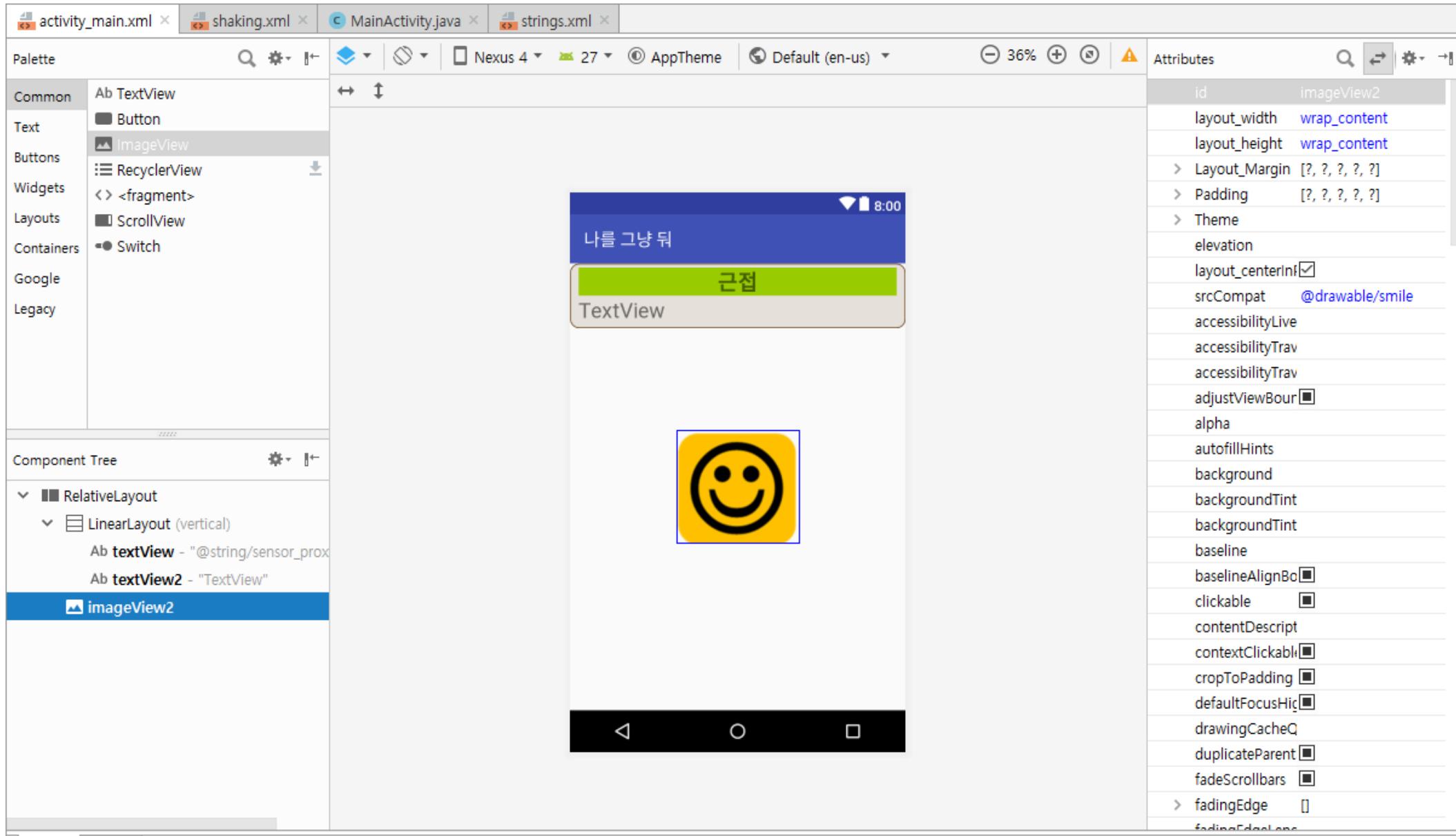
# 근접 센서 정보 – TextView

34



# 애니메이션용 이미지 – ImageView

35



# 2.5 Activity 제어(MainActivity.java)

- 센서이벤트 처리를 위한 액티비티 인터페이스 추가

36

```
activity_main.xml × shaking.xml × MainActivity.java ×  
1 package com.example.user.positionsensor;  
2  
3 import android.support.v7.app.AppCompatActivity;  
4 import android.os.Bundle;  
5  
6 public class MainActivity extends AppCompatActivity {  
7  
8     @Override  
9     protected void onCreate(Bundle savedInstanceState) {  
10         super.onCreate(savedInstanceState);  
11         setContentView(R.layout.activity_main);  
12     }  
13 }  
14 }
```

센서값 변화에 따른 이벤트 처리를  
위한 클래스

```
activity_main.xml × shaking.xml × MainActivity.java ×  
1 package com.example.user.positionsensor;  
2  
3 import android.hardware.SensorEventListener;  
4 import android.support.v7.app.AppCompatActivity;  
5 import android.os.Bundle;  
6  
7 public class MainActivity extends AppCompatActivity implements SensorEventListener {  
8  
9     @Override  
10    protected void onCreate(Bundle savedInstanceState) {  
11        super.onCreate(savedInstanceState);  
12        setContentView(R.layout.activity_main);  
13    }  
14 }
```

## • 센서값 처리를 위한 매소드 구현(@override)

The screenshot shows the Android Studio interface with three tabs at the top: activity\_main.xml, shaking.xml, and MainActivity.java. The MainActivity.java tab is active, displaying Java code:

```

1 package com.example.user.positionsensor;
2
3 import android.hardware.SensorEventListener;
4 import android.support.v7.app.AppCompatActivity;
5 import android.os.Bundle;
6
7 public class MainActivity extends AppCompatActivity implements SensorEventListener {
8
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13
14        SensorManager sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
15        Sensor sensor = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
16
17        if (sensor != null) {
18            sensorManager.registerListener(this, sensor, SensorManager.SENSOR_DELAY_NORMAL);
19        }
20    }
21
22    @Override
23    public void onAccuracyChanged(Sensor sensor, int accuracy) {
24    }
25
26    @Override
27    public void onSensorChanged(SensorEvent event) {
28    }
29}

```

A blue arrow points from the code editor to the 'Implement methods' option in the context menu that appears when hovering over the 'implements' keyword. The context menu also includes options like 'Create Test', 'Create subclass', 'Unimplement Interface', and 'Annotate interface 'SensorEventListener' as @Deprecated'.

To the right, a 'Select Methods to Implement' dialog box is open, listing the methods from the SensorEventListener interface:

- android.hardware.SensorEventListener
  - onAccuracyChanged(sensor:Sensor, accuracy:int):void
  - onSensorChanged(event:SensorEvent):void

At the bottom of the dialog, there are checkboxes for 'Copy JavaDoc' (unchecked) and 'Insert @Override' (checked), along with 'OK' and 'Cancel' buttons.

activity\_main.xml

shaking.xml

MainActivity.java

```
1 package com.example.user.positionsensor;  
2  
3 import android.hardware.Sensor;  
4 import android.hardware.SensorEvent;  
5 import android.hardware.SensorEventListener;  
6 import android.support.v7.app.AppCompatActivity;  
7 import android.os.Bundle;  
8  
9 public class MainActivity extends AppCompatActivity implements SensorEventListener {  
10  
11     @Override  
12     protected void onCreate(Bundle savedInstanceState) {  
13         super.onCreate(savedInstanceState);  
14         setContentView(R.layout.activity_main);  
15     }  
16  
17     @Override  
18     public void onSensorChanged(SensorEvent event) {  
19     }  
20  
21     @Override  
22     public void onAccuracyChanged(Sensor sensor, int accuracy) {  
23     }  
24  
25 }  
26  
27 }
```

센서 값이 변할 때 호출

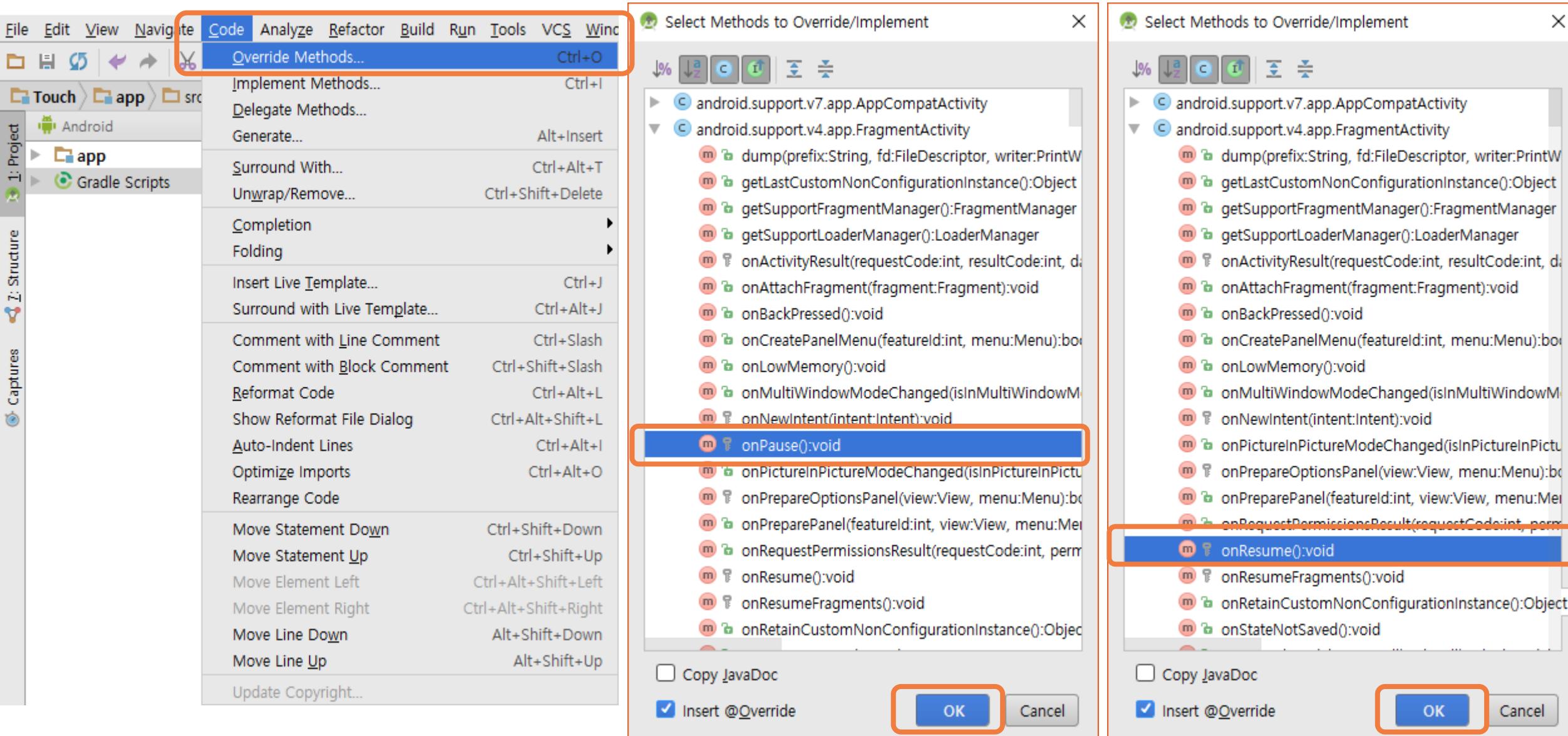
등록된 센서의 정확도가 변할 때 호출

- 센서와 진동을 처리하기 위한 변수 선언 (빨간상자만 입력)

```
17 public class MainActivity extends AppCompatActivity implements SensorEventListener{  
18  
19     ImageView img;  
20     TextView textView;  
21  
22     SensorManager sm;  
23     Sensor sensor_proximity;  
24  
25     Vibrator mVibe;  
26  
27     @Override  
28     protected void onCreate(Bundle savedInstanceState) {  
29         super.onCreate(savedInstanceState);  
30         setContentView(R.layout.activity_main);  
31  
32         img = (ImageView) findViewById(R.id.imgSmile);  
33         textView = (TextView) findViewById(R.id.textPosition);  
34  
35         sm = (SensorManager) getSystemService(SENSOR_SERVICE);  
36         sensor_proximity = sm.getDefaultSensor(Sensor.TYPE_PROXIMITY);  
37  
38         mVibe = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);  
39     }  
40 }
```

# onPause()/onResume() 메소드 재정의(Override)

40



- 재정의를 위한 매소드 추가

```
40  
41 ↗ @Override  
42     protected void onPause() {  
43         super.onPause();  
44     }  
45  
46 ↗ @Override  
47     protected void onResume() {  
48         super.onResume();  
49     }
```

화면에 표시되는 상태에서 사용자와 상호 작용하지 않을 때

액티비티가 일시정지(pause)상태에서 복귀할 때 호출

# (빨간상자만 입력)

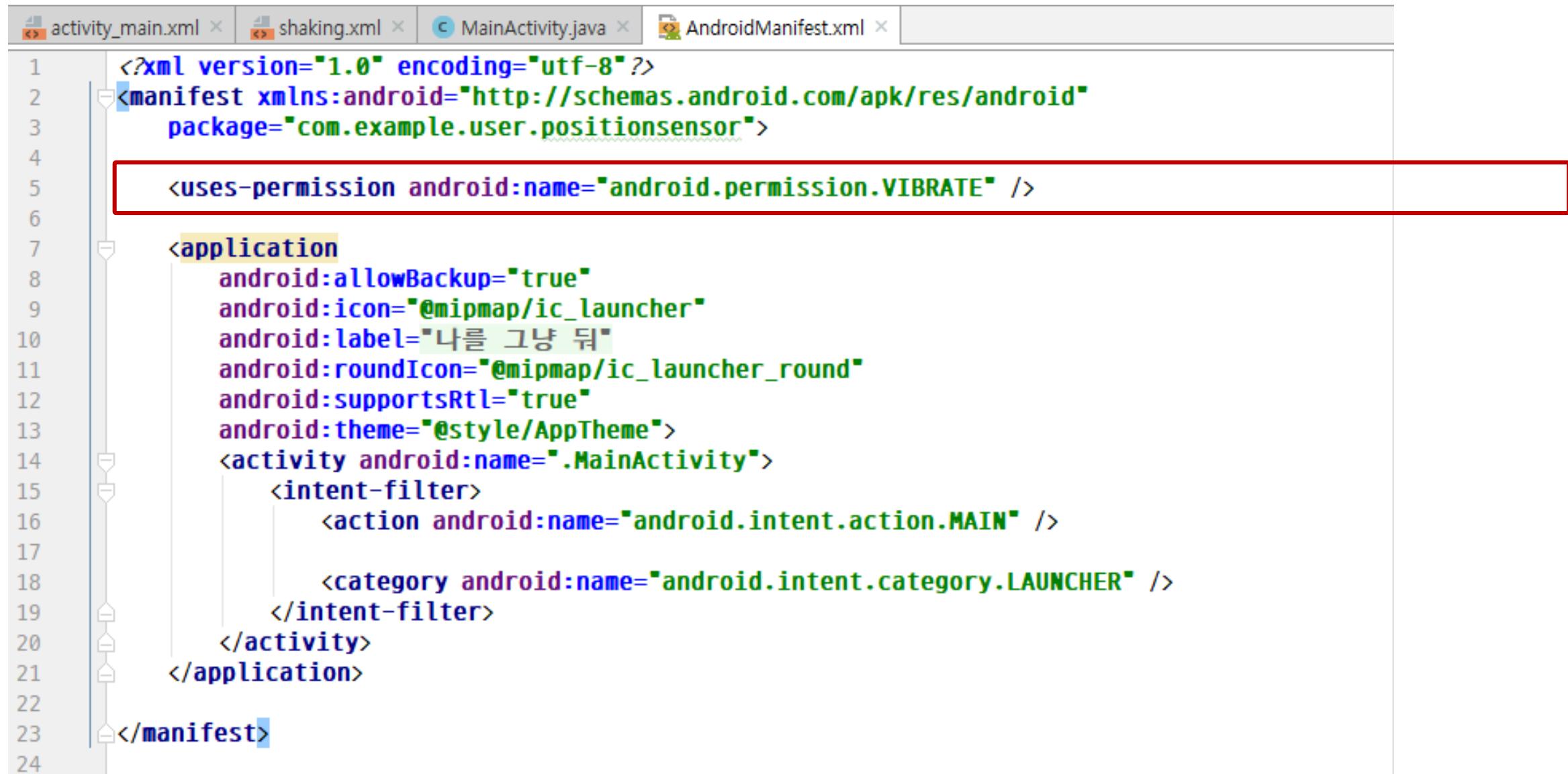
```
45      @Override  
46  ⬆  protected void onPause() {  
47      super.onPause();  
48      sm.unregisterListener(this);  
49  }  
50  
51      @Override  
52  ⬆  protected void onResume() {  
53      super.onResume();  
54  
55      sm.registerListener( listener: this, sensor_proximity, SensorManager.SENSOR_DELAY_NORMAL);  
56  }
```

## • 근접 센서의 값 읽어서 처리하기(빨간상자만 입력)

```
41 @Override  
42 public void onSensorChanged(SensorEvent event) {  
43  
44     if (event.sensor.getType() == Sensor.TYPE_PROXIMITY){  
45  
46         textView.setText("거리: " + event.values[0]);  
47  
48         if(event.values[0] < 5){ // min=0, max=10  
49             Animation ani = AnimationUtils.loadAnimation(context: this, R.anim.shaking);  
50  
51             img.setImageResource(R.drawable.angry);  
52             img.startAnimation(ani);  
53  
54             // 1000 : Vibrate for 1 sec  
55             // VibrationEffect.DEFAULT_AMPLITUDE - would perform vibration at full strength  
56             VibrationEffect effect = VibrationEffect.createOneShot(milliseconds: 1000, VibrationEffect.DEFAULT_AMPLITUDE);  
57             mVibe.vibrate(effect);  
58         }else{  
59             img.setImageResource(R.drawable.smile);  
60         }  
61     }  
62 }  
63  
64 }
```

빨간줄 처리하기  
(진동 사용에 대한 허가가 필요)

## • 진동 사용 허가 주기-AndoridManifest.xml (빨간상자만 입력)



```
activity_main.xml × shaking.xml × MainActivity.java × AndroidManifest.xml ×
1 <?xml version="1.0" encoding="utf-8"?
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3   package="com.example.user.positionsensor">
4
5   <uses-permission android:name="android.permission.VIBRATE" />
6
7   <application
8     android:allowBackup="true"
9     android:icon="@mipmap/ic_launcher"
10    android:label="나를 그냥 둬"
11    android:roundIcon="@mipmap/ic_launcher_round"
12    android:supportsRtl="true"
13    android:theme="@style/AppTheme">
14     <activity android:name=".MainActivity">
15       <intent-filter>
16         <action android:name="android.intent.action.MAIN" />
17
18         <category android:name="android.intent.category.LAUNCHER" />
19       </intent-filter>
20     </activity>
21   </application>
22
23 </manifest>
24
```

# Activity LifeCycle

메소드	설명	다음 메소드
onCreate()	액티비티가 생성될 때 호출되며 사용자 인터페이스 초기화에 사용 됨.	onStart()
onRestart()	액티비티가 멈췄다가 다시 시작되기 바로 전에 호출됨.	onStart()
onStart()	액티비티가 사용자에게 보여지기 바로 직전에 호출됨.	onResume() 또는 onStop()
onResume()	액티비티가 사용자와 상호작용하기 바로 전에 호출됨.	onPause()
onPause()	다른 액티비티가 보여질 때 호출됨. 데이터 저장, 스레드 중지 등의 처리를 하기에 적당한 메소드.	onResume() 또는 onStop()
onStop()	액티비티가 더이상 사용자에게 보여지지 않을 때 호출됨. 메모리가 부족할 경우에는 onStop() 메소드가 호출되지 않을 수도 있음.	onRestart() 또는 onDestroy()
onDestroy()	액티비티가 소멸될 때 호출됨. finish() 메소드가 호출되거나 시스템이 메모리 확보를 위해 액티비티를 제거할 때 호출됨.	없음

# 클래스와 속성/메소드

## • 클래스

클래스/인터페이스	설명
Sensor	센서 표현을 위한 클래스
SensorEvent	센서 이벤트를 표현하는 클래스
SensorEventListener	센서 값이 변할 때 센서 매니저로부터 공지를 받는 데 사용
SensorManager	디바이스의 센서에 접근할 수 있도록 함.

## • 상수

클래스	상수	설명
Context	String <code>SENSOR_SERVICE</code>	센서 이용을 위한 센서 매니저를 추출하기 위해 <code>getSystemService(Class)</code> 와 함께 사용
Sensor	int <code>TYPE_GRAVITY</code>	중력센서 타입을 기술하는 상수
	int <code>TYPE_ACCELEROMETER</code>	가속도계 타입을 기술하는 상수
	int <code>TYPE_LINEAR_ACCELERATION</code>	직선 가속도 센서 타입을 기술하는 상수
	int <code>TYPE_GYROSCOPE</code>	자이로스코프 센서 타입을 기술하는 상수
SensorManager	int <code>SENSOR_DELAY_NORMAL</code>	스크린 방향 변화에 적당한 비율

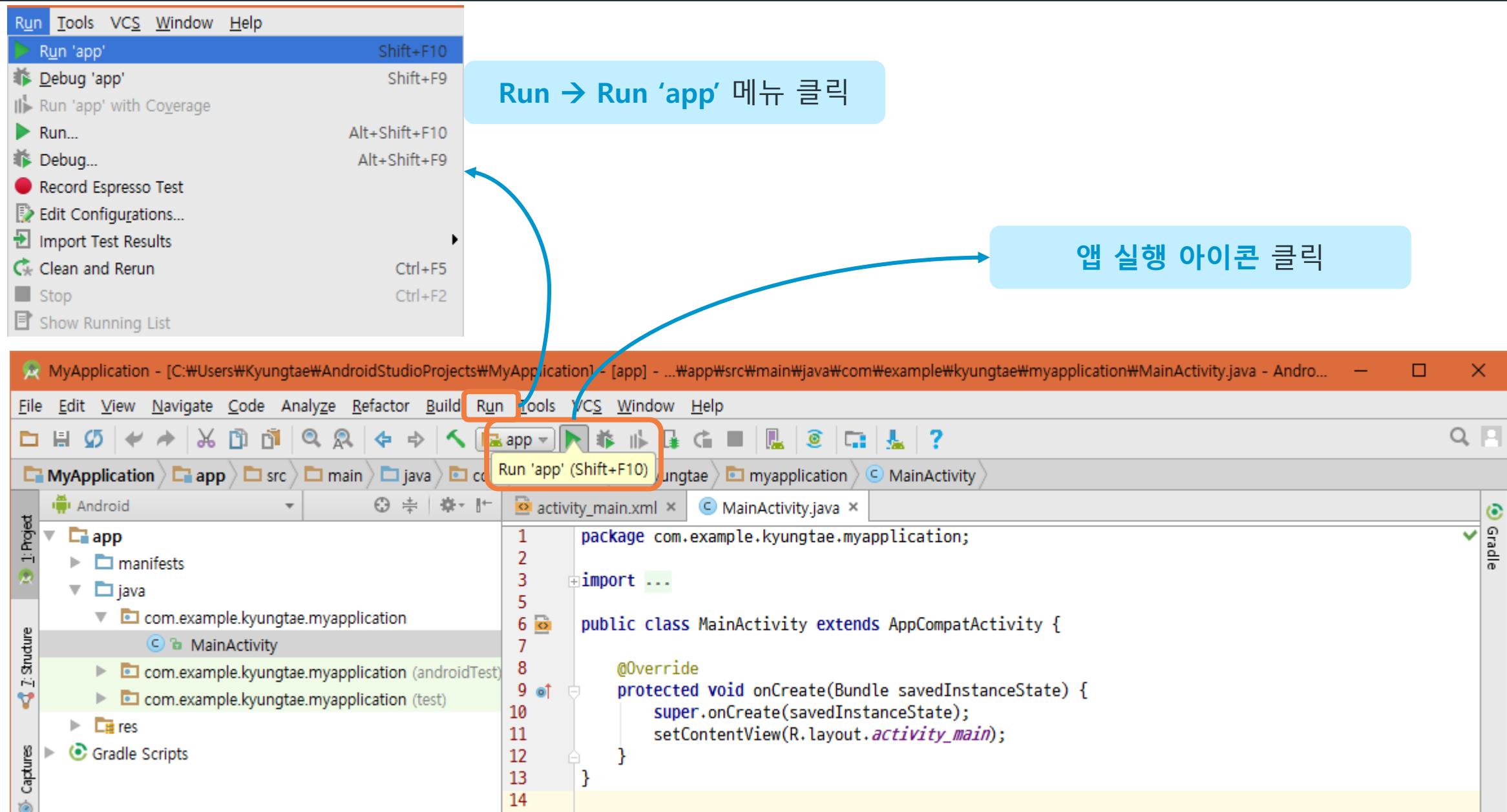
# 클래스와 속성/메소드

## • 메소드

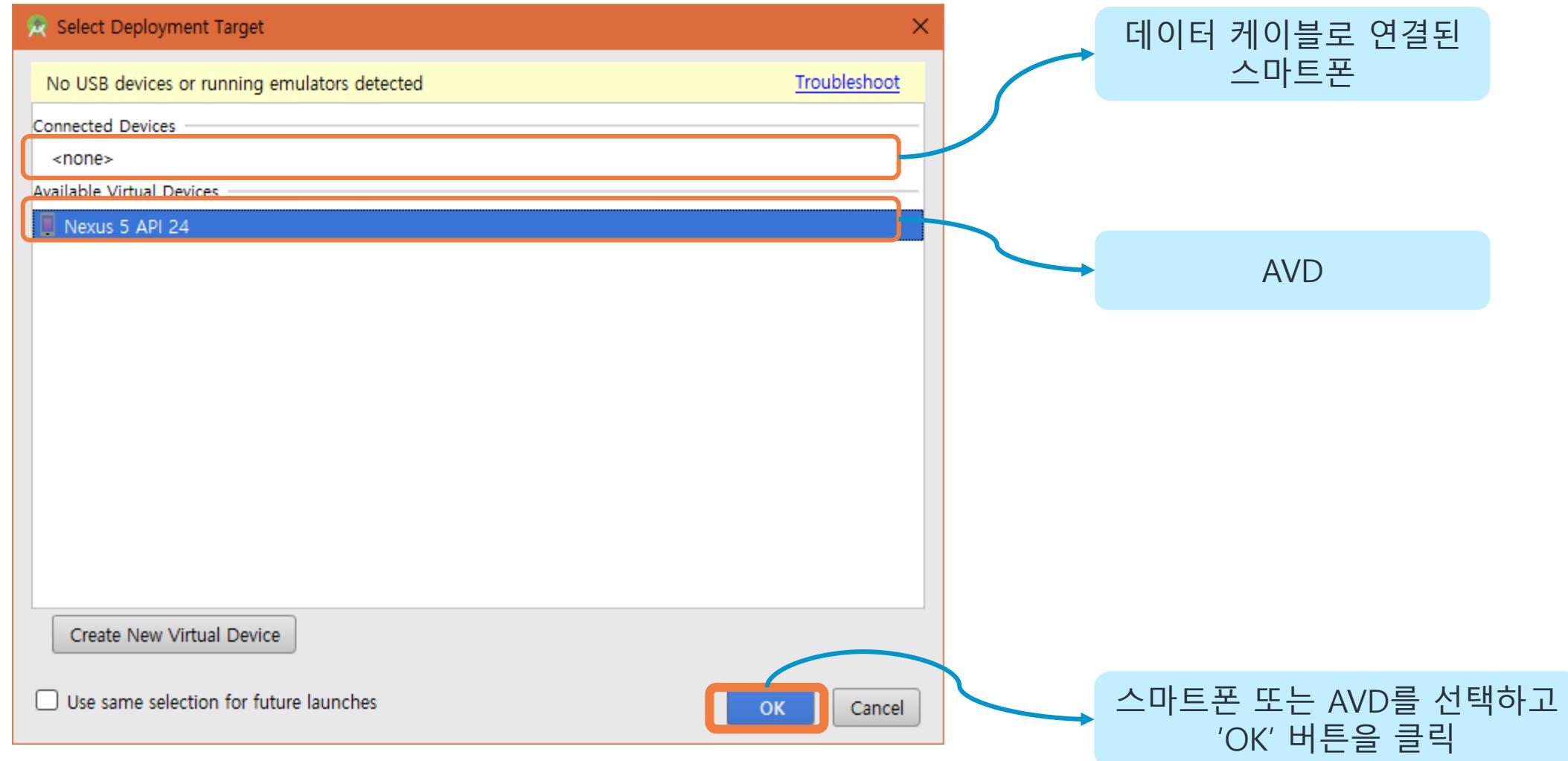
클래스	메소드	설명
Sensor	int <code>getType()</code>	센서 타입을 반환함
SensorEvent	public final float[] <code>values</code>	센서가 인식한 값들을 저장하는 배열
SensorEventListener	abstract void <code>onAccuracyChanged(Sensor, int accuracy)</code>	등록된 센서의 정확도가 변할 때 호출됨
	abstract void <code>onSensorChanged(SenseorEvent event)</code>	센서 값이 변할 때 호출됨
SensorManager	Sensor <code>getDefaultSensor(int type)</code>	주어진 type을 위한 디폴트 센서를 얻기 위해 사용
	Boolean <code>registerListener(SensorEventListener listener, Sensor sensor, int samplingPeriodUs)</code>	주어진 샘플링 주파수에서 주어진 센서를 위한 SensorEventListener를 등록함
	void <code>unregisterListener(SensorEventListener listener)</code>	모든 센서에 대한 리스너를 해제함

# Step 3. 프로젝트 실행

55



## • AVD 장비 선택하기



## • 실행 결과

57

Android Emulator - Nexus\_5X\_API\_24:5554

나를 그냥 뒤  
근접  
거리: 8.1

거리: 8.1

Extended controls - Nexus\_5X\_API\_27\_Oreo\_8.1:5554

Location  
Cellular  
Battery  
Camera  
Phone  
Directional pad  
Microphone  
Fingerprint  
Virtual sensors  
Bug report  
Screen record  
Google Play  
Settings  
Help

Accelerometer Additional sensors

Ambient temperature (°C) 0.0  
-273.1 100

Magnetic field (North-East-Up μT) 22.00 5.90 43.10

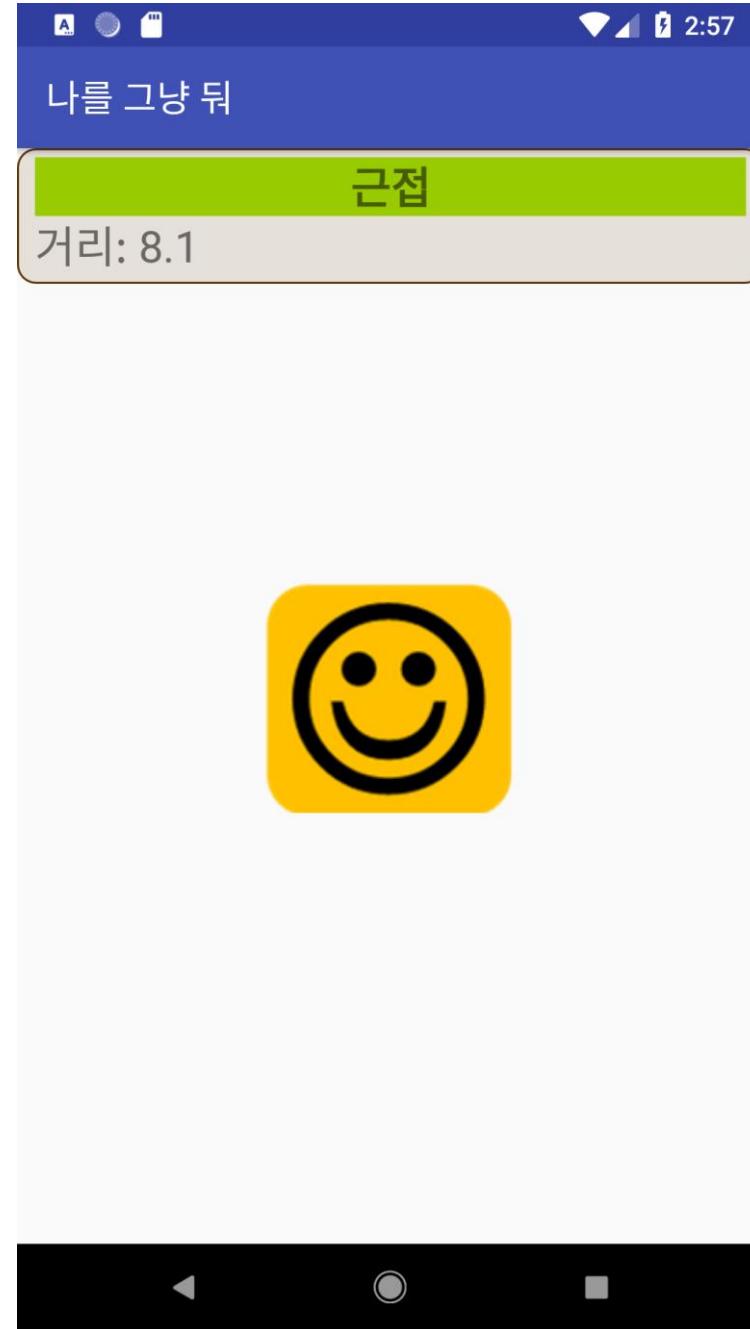
Proximity (cm) 8.1  
0 10

Light (lux) 0.0  
0 40000

Pressure (hPa) 0.0  
0 1100

Relative humidity (%) 0.0  
0 100

# O outputs





question

&



answer

