

Week10.

모션센서

개발환경 구축 절차

주 차	수 업 내 용
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 – 위치 센서, 환경 센서
12	NFC 활용
13	공공 DB 오픈 API 활용
14	구글 맵과 위치 추적
15	기말 고사



센서의 이해

- 안드로이드에서 지원하는 모션, 환경, 위치 센서

구분	기능	센서 종류
모션 센서	스마트폰의 세 축에 따른 가속도와 회전력 측정	중력 센서, 자이로스코프, 회전 벡터 센서
환경 센서	온도, 기압, 조도, 습도와 같은 환경 변수 측정	기압계, 광도계, 온도계
위치 센서	디바이스의 물리적인 위치 측정	방향 센서, 자력계

스마트폰에 내장된 센서들...

5



(앞면)

삼성(갤럭시 S7)



(앞면)



(앞면)

LG(G5)



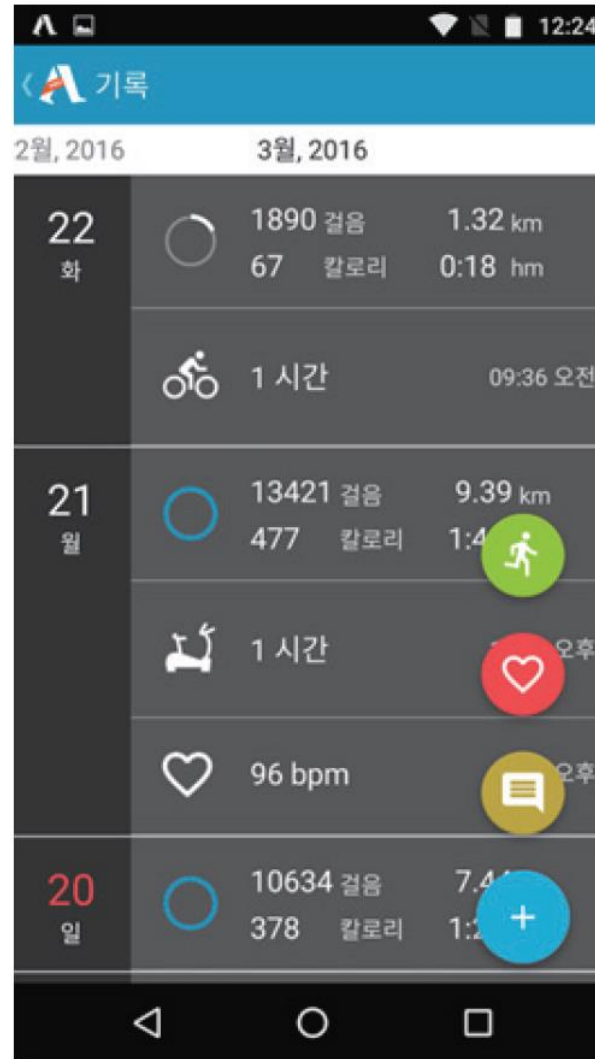
(뒷면)

모션센서를 이용한 앱의 예

6



현재 상황



일별 기록현황

● 애큐페도 만보기

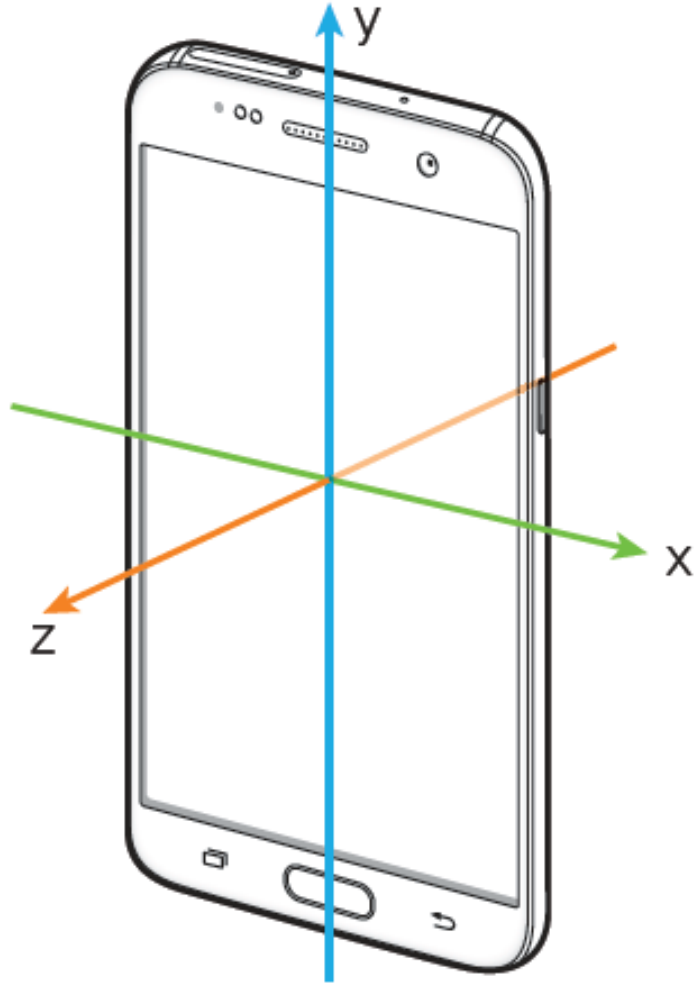


모션 센서 원리

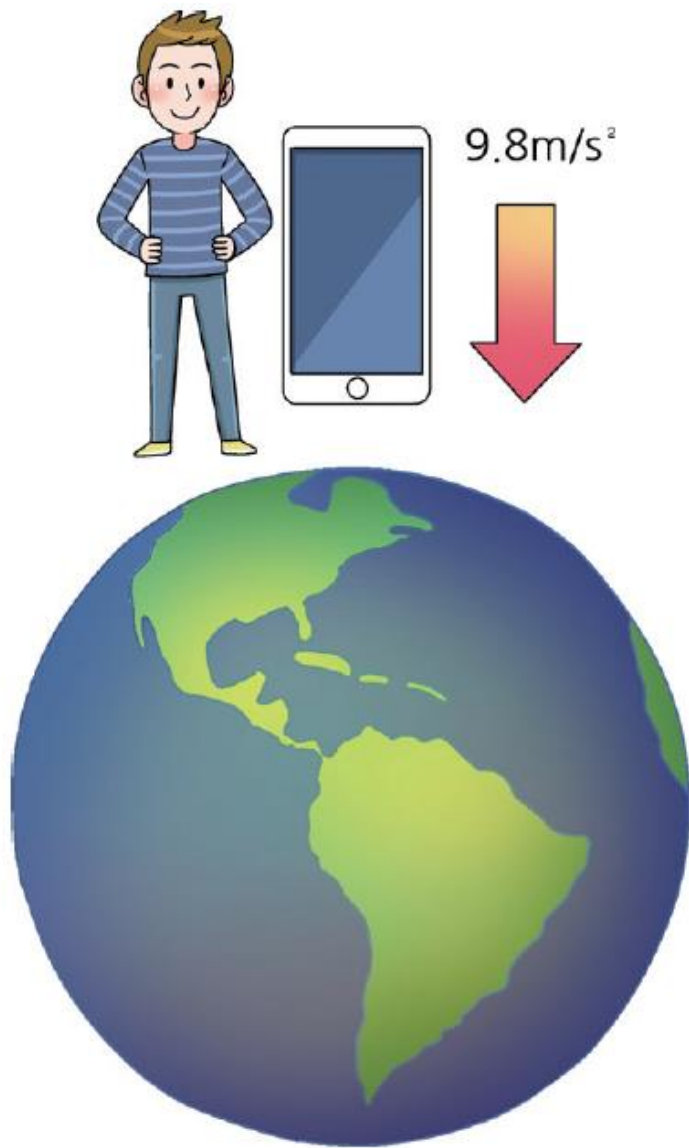


모션 센서는 센서 값을 표현하기 위해 세가지 축 사용

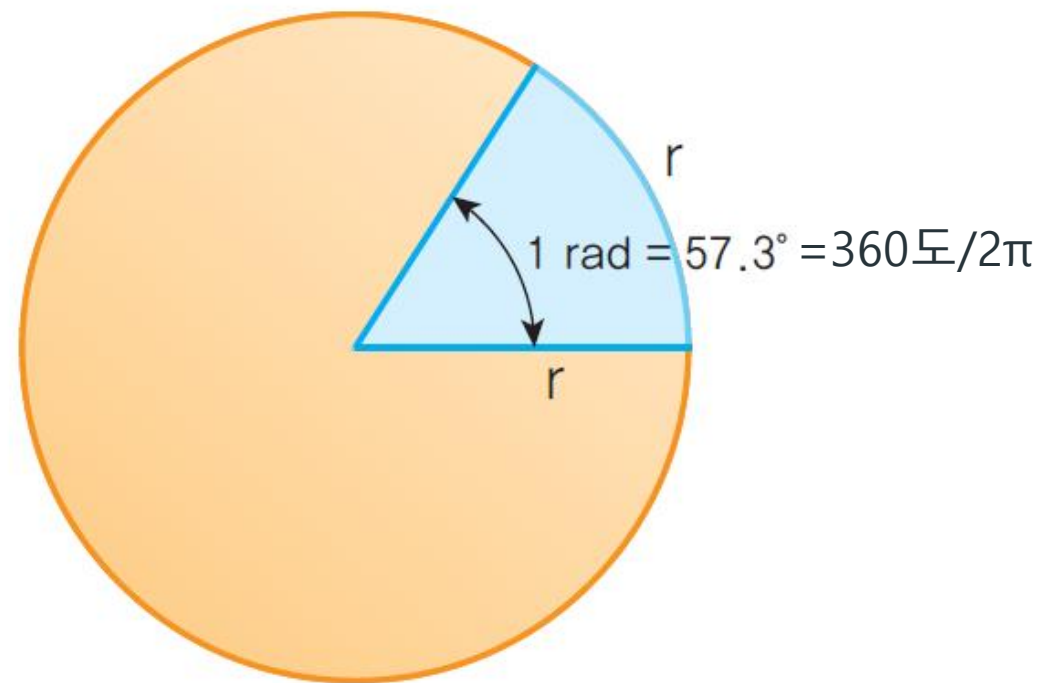
8



- 중력
 - 디바이스에 가해지는 중력의 크기
 - 지표면에서 9.8 m/s^2
 - 고도가 높을 수록 작아짐.
- 가속도
 - 중력을 포함해서 디바이스에 가해지는 가속도를 더하거나 뺀 값
 - 중력과 반대 방향으로 가속도가 작용하면 중력에 가속도를 더한 값이며, 같은 방향이면 가속도를 뺀 값으로 측정
- 직선 가속도
 - 중력을 제외하고 디바이스에 가해지는 가속도의 크기를 측정한 값
- 자이로스코프
 - x, y, z축 방향에 대한 회전율을 측정(rad/s)
 - 1라디안은 원의 반지름과 같은 원호에 대한 중심 각(57.3도)



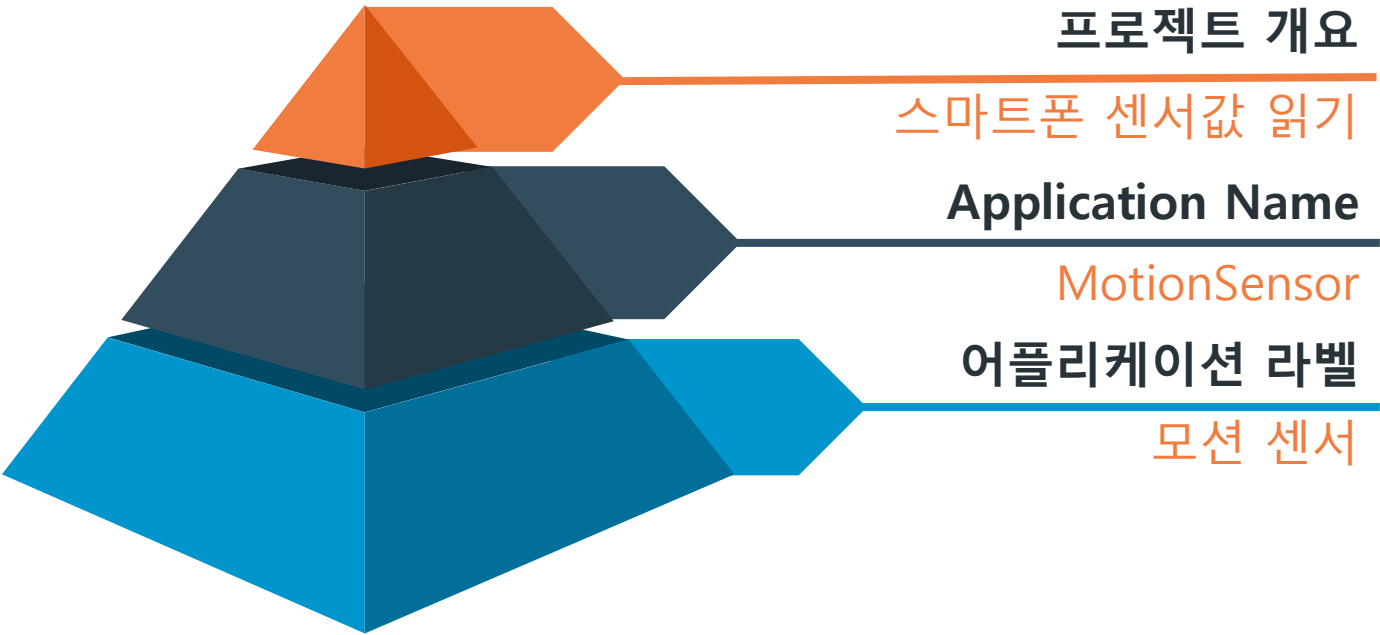
지표면의 중력



라디안(radian)

Step 0.프로젝트 개요

10



모션센서	
중력	X: -8.7678285E-7 Y: 9.80665 Z: 5.845219E-7
가속도	X: 0.0 Y: 9.81 Z: 0.0
직선 가속도	X: 8.7678285E-7 Y: 0.0033502579 Z: -5.845219E-7
자이로스코프	X: 0.0 Y: 0.0 Z: 0.0
자기장	X: 22.0 Y: 5.9 Z: 43.1

Create New Project

Create Android Project

Application name

MotionSensor

Company domain

user.example.com

Project location

C:\Users\Kyungtae\AndroidStudioProjects\user\MotionSensor

Package name

com.example.user.motionsensor


Edit

☐ Include C++ support

☐ Include Kotlin support

Previous Next Cancel Finish

Create New Project

 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**

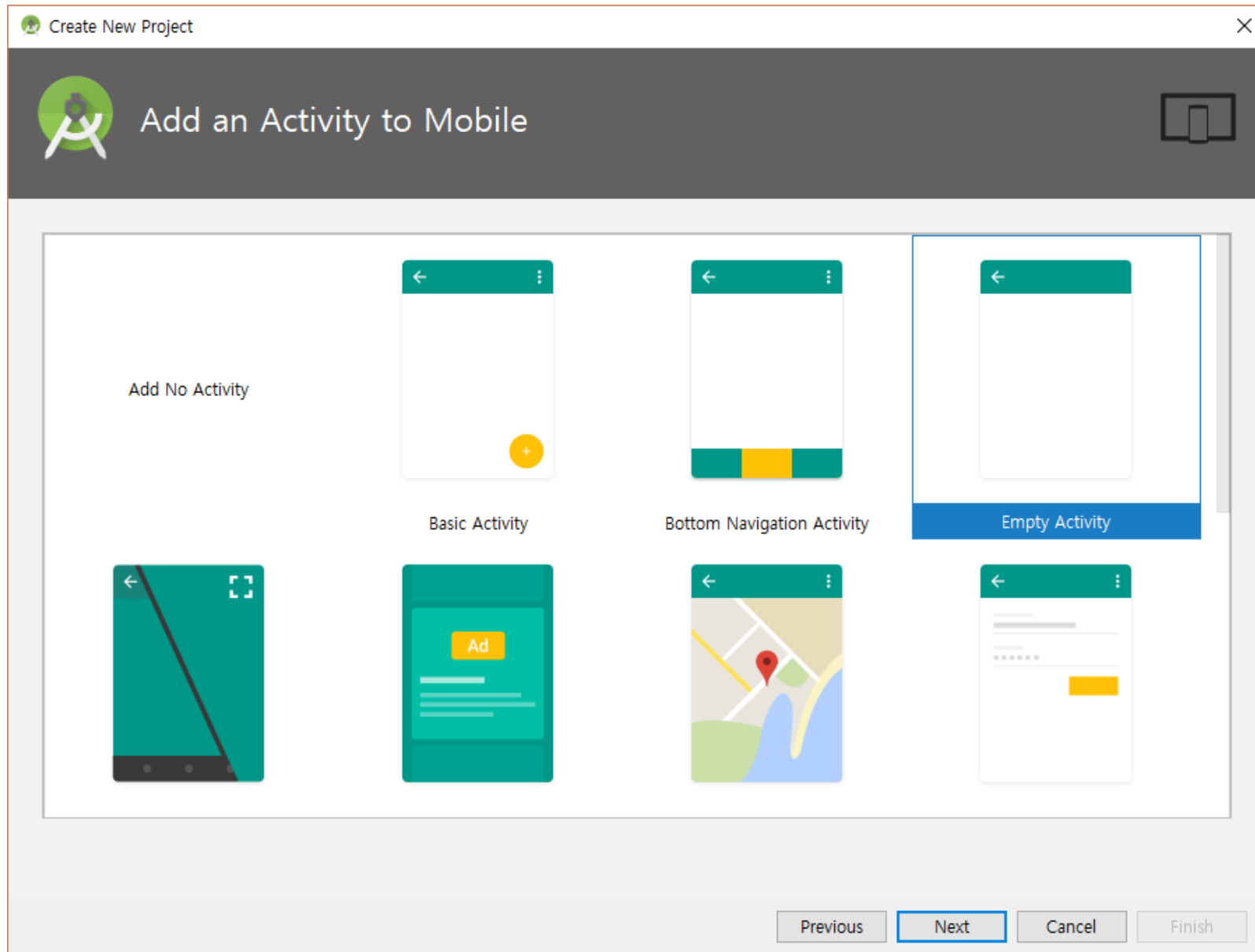
API 24: Android 7.0 (Nougat)

Previous



Next

Cancel

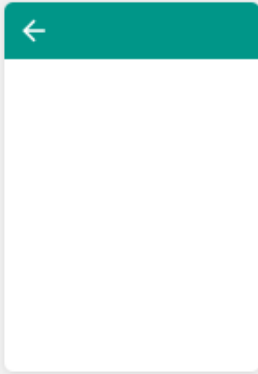
Finish



Create New Project

 Configure Activity 

Creates a new empty activity



Activity Name:

☒ Generate Layout File

Layout Name:

☒ Backwards Compatibility (AppCompat)

The name of the activity class to create

Previous

Next

Cancel

Finish

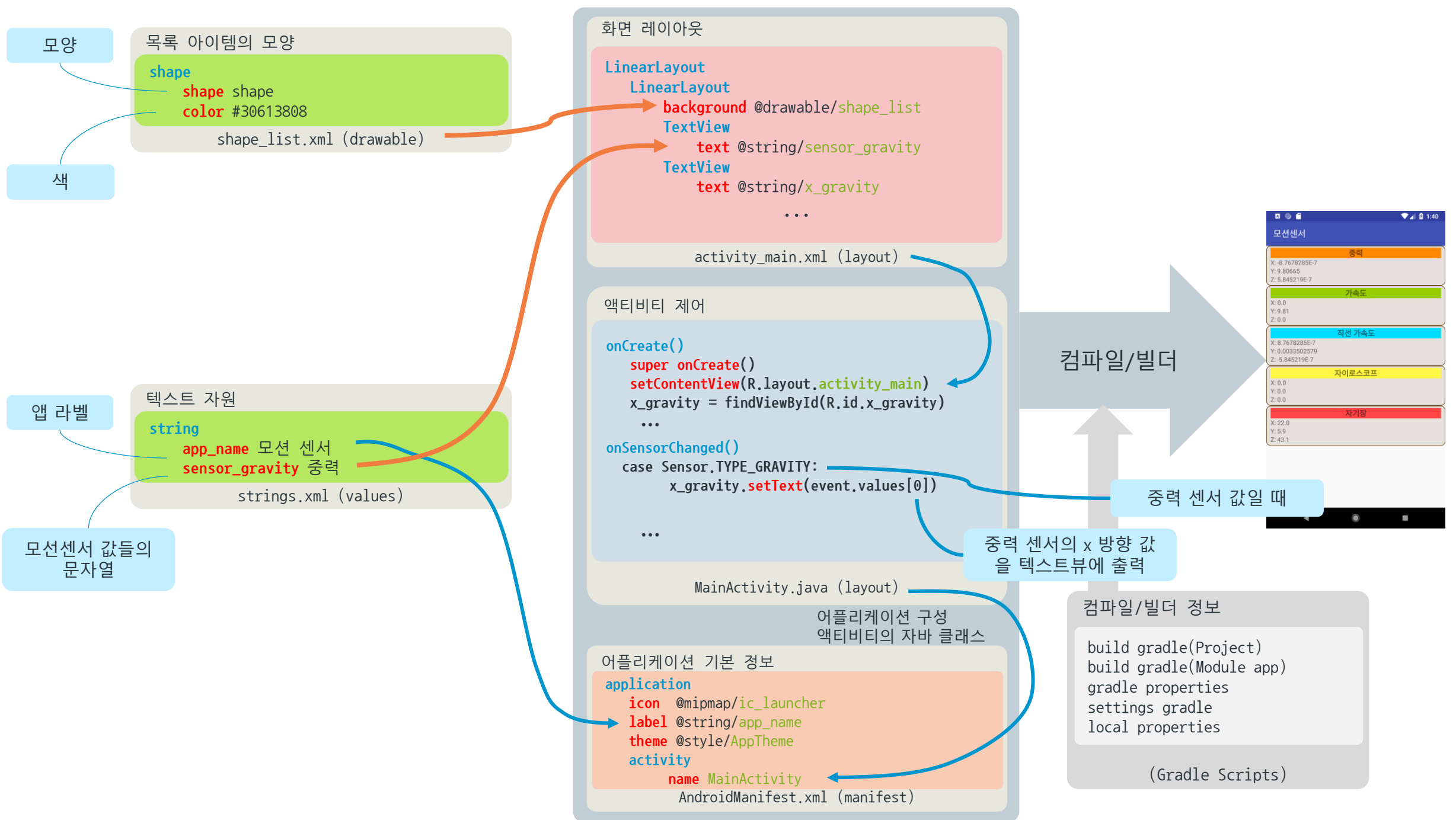
Step 1. 프로젝트 생성

15

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

Step 2. 파일 편집

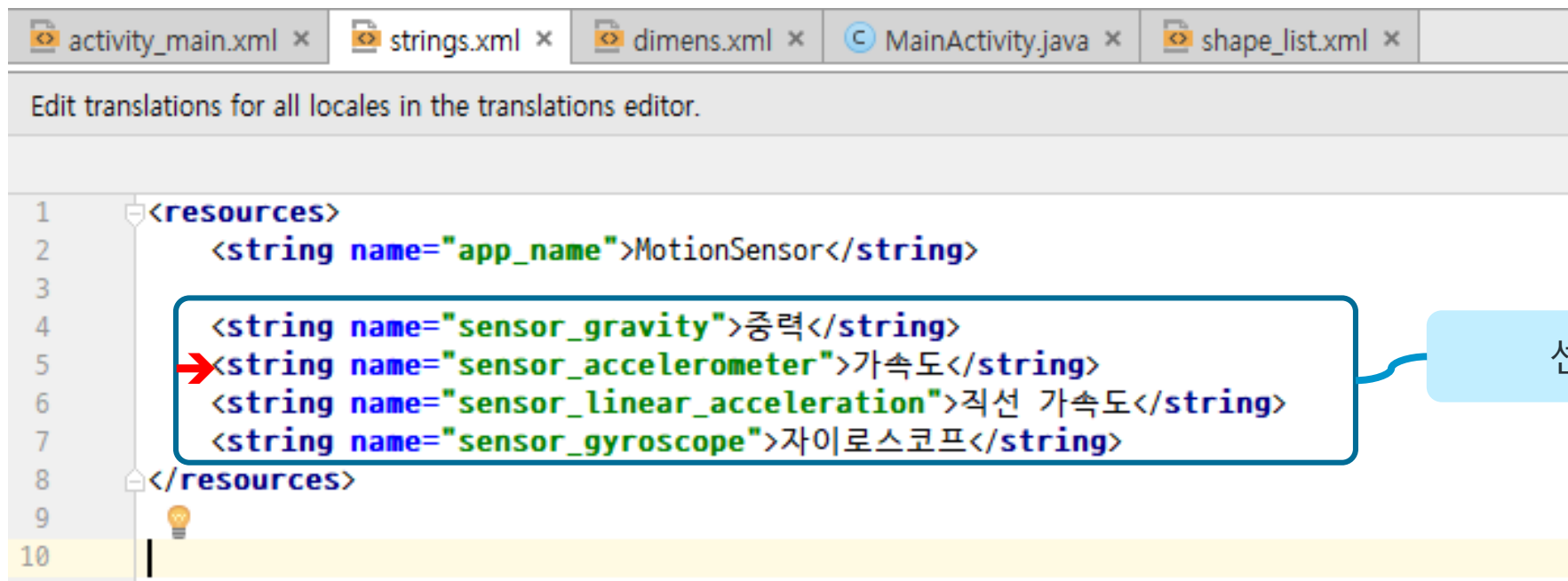
모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	
java	com.example.kyungtae.video1	MainActivity.java	<ul style="list-style-type: none">• 센서 등록• 센서 값 변경 확인• 센서 값 출력
res	drawable	shape_list	<ul style="list-style-type: none">• 출력모양 설계(배경색)
	layout	activity_main.xml	<ul style="list-style-type: none">• 모션센서 측정값 배치(Textview)
	mipmap	ic_launcher.png	
	values	colors.xml	
		dimens.xml	
		strings.xml	<ul style="list-style-type: none">• 어플리케이션 라벨 수정• 모션센서 이름의 문자열 추가
		styles.xml	



Step 2.1 텍스트 자원의 편집

18

- strings.xml

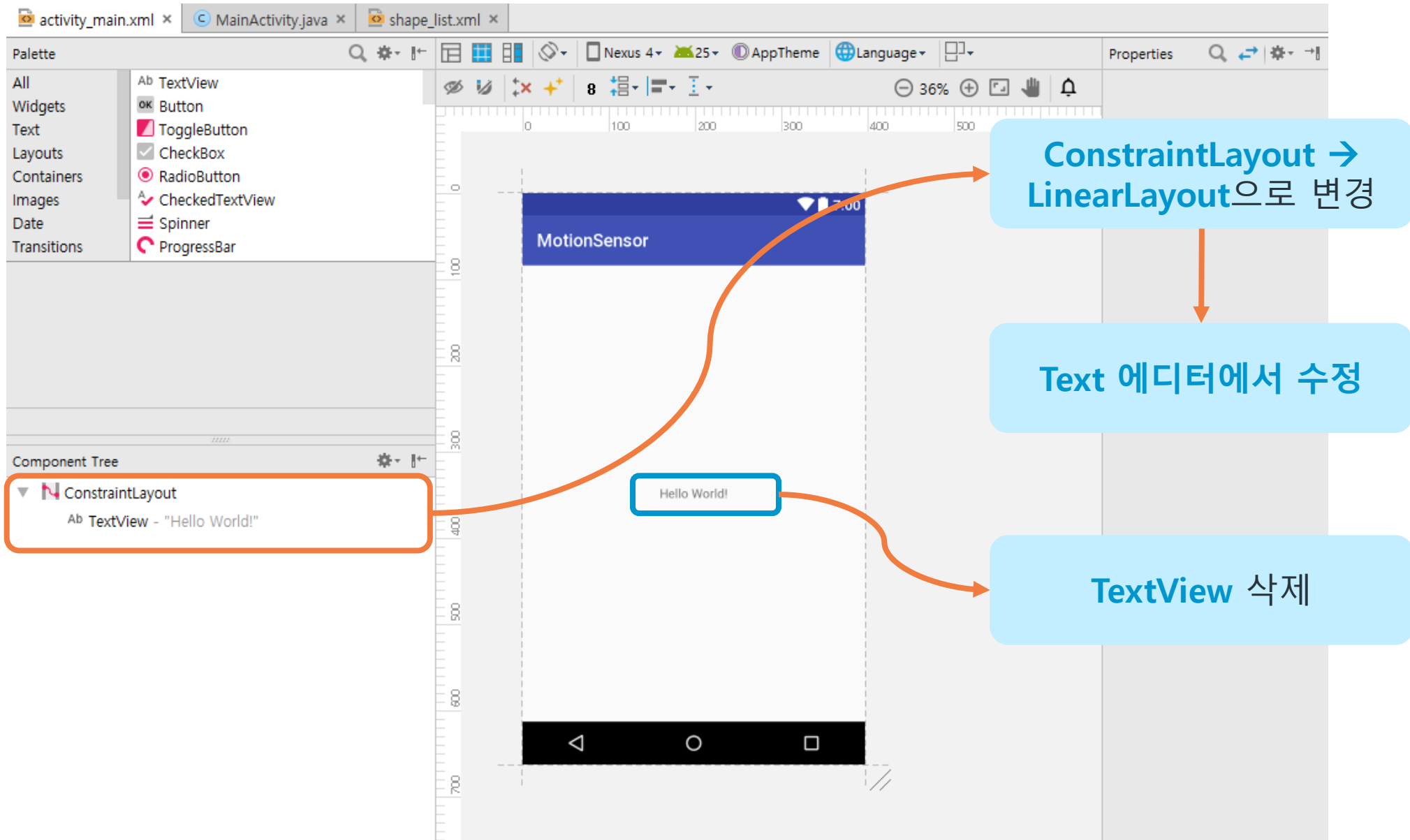


```
activity_main.xml x strings.xml x dimens.xml x MainActivity.java x shape_list.xml x
Edit translations for all locales in the translations editor.

1 <resources>
2   <string name="app_name">MotionSensor</string>
3
4   <string name="sensor_gravity">중력</string>
5   <string name="sensor_accelerometer">가속도</string>
6   <string name="sensor_linear_acceleration">직선 가속도</string>
7   <string name="sensor_gyroscope">자이로스코프</string>
8 </resources>
9
10
```

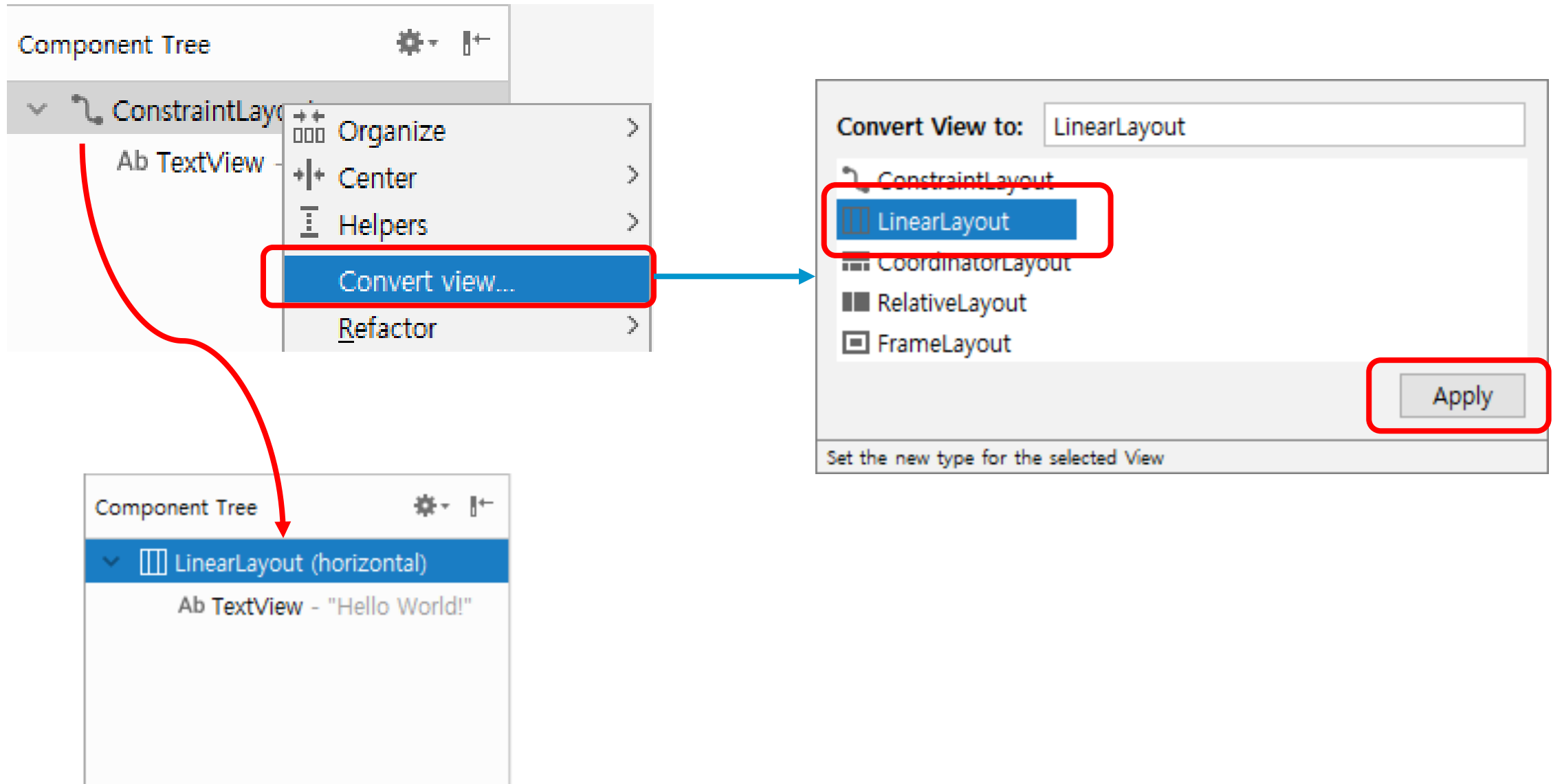
센서 값들의 제목

2.2 화면 설계



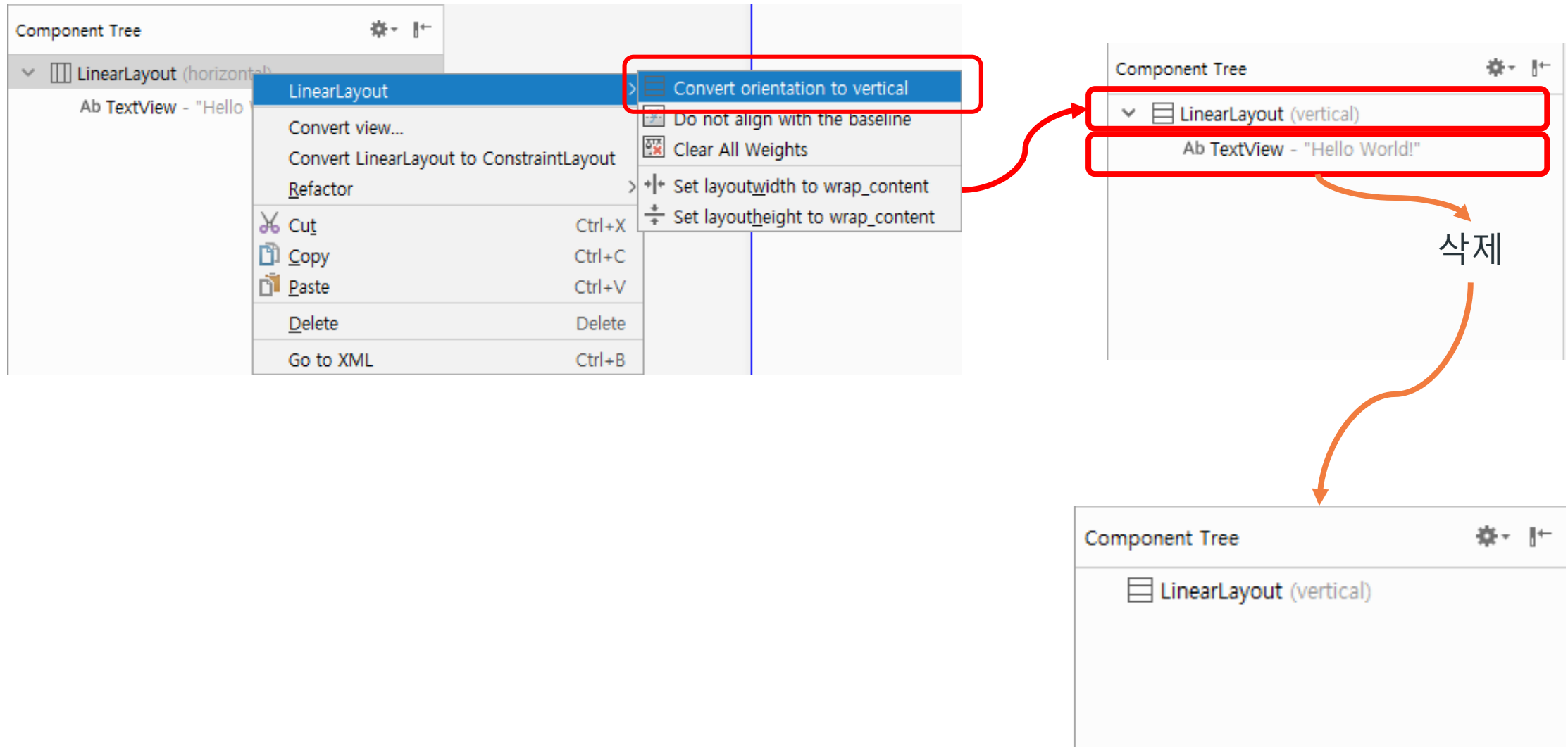
ConstraintLayout을 LinearLayout로 바꾸기

20



LinearLayout의 방향을 Horizontal → Vertical로 변경하기

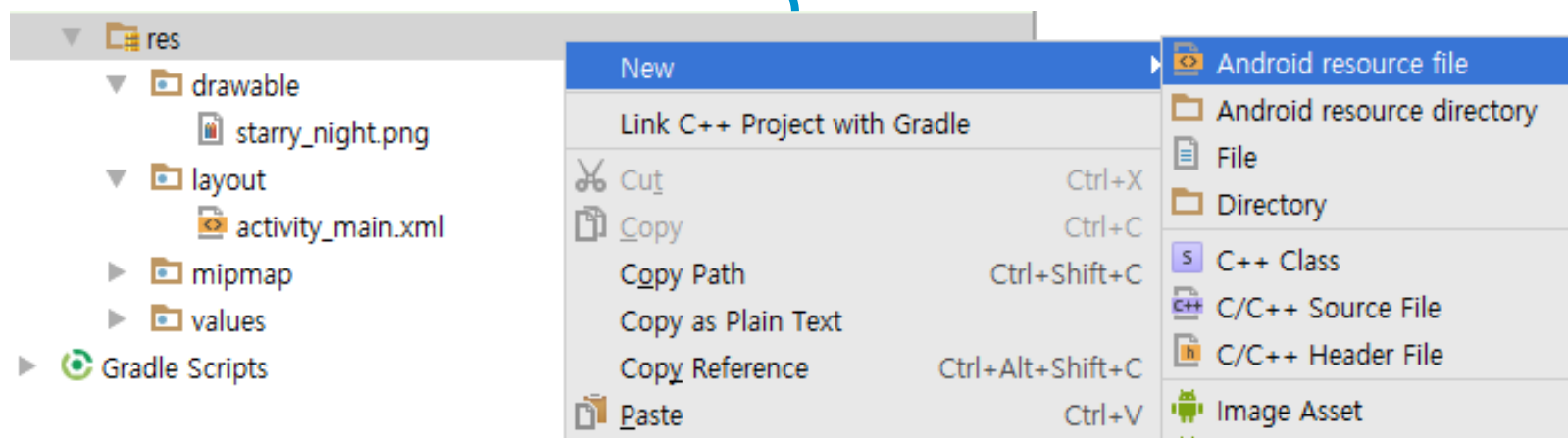
21



2.3 drawable 리소스 – shape_list.xml추가

- **shape_list.xml** 생성(res/drawable 폴더)
 - drawable resource를 이용한 그림 출력

XML 파일 생성



- Set New Resource File - shape_list.xml

26

File name: shape_list

Resource type: Drawable

Root element: shape

Source set: main

Directory name: drawable

The screenshot shows the 'New Resource File' dialog box in Android Studio. The fields are filled as follows:

- File name:** shape_list
- Resource type:** Drawable
- Root element:** shape
- Source set:** main
- Directory name:** drawable

Below these fields are two sections: 'Available qualifiers' and 'Chosen qualifiers'. The 'Available qualifiers' list includes Country Code, Network Code, Locale, Layout Direction, Smallest Screen Width, Screen Width, Screen Height, Size, Ratio, and Orientation. The 'Chosen qualifiers' section is empty, displaying 'Nothing to show'. At the bottom right, there are three buttons: 'OK', 'Cancel', and 'Help'. The 'OK' button is highlighted with an orange box.

• shape_list.xml

27

The image shows the Android Studio interface with the `shape_list.xml` file open. The XML code is as follows:

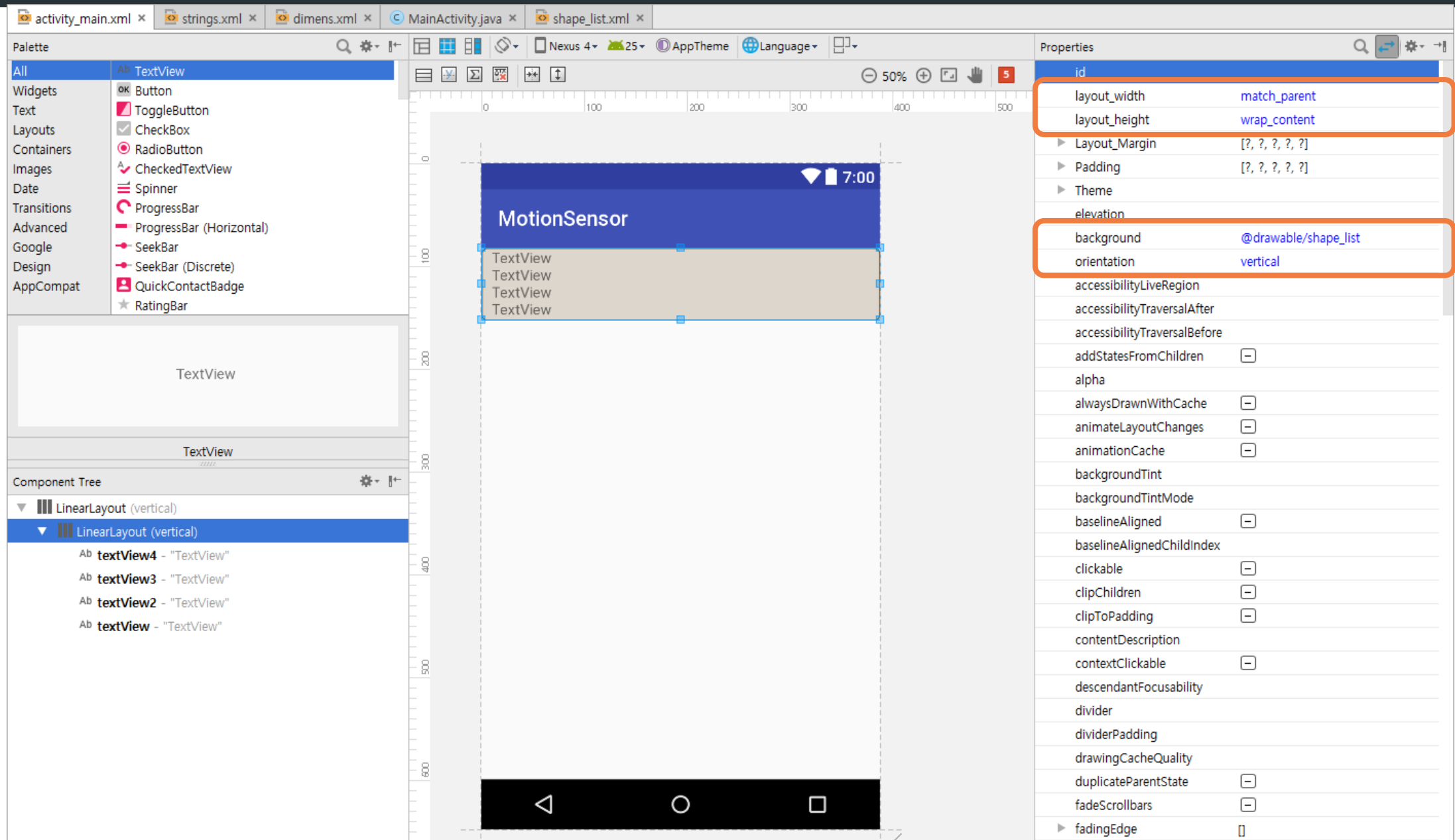
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <shape xmlns:android="http://schemas.android.com/apk/res/android"
3     android:shape="rectangle">
4     <solid android:color="#3061380B"/>
5     <stroke android:width="1dp" android:color="#61380B"/>
6     <padding
7         android:top="2dp"
8         android:bottom="2dp"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

Annotations in Korean explain the attributes:

- `android:color="#3061380B"`: 출력모양을 내부의 색 (Output shape internal color)
- `android:width="1dp" android:color="#61380B"`: 출력모양을 테두리의 색 (Output shape border color)
- `padding` attributes: 내부 패딩 정보 (Internal padding information)
- `android:radius="5dp"`: 출력모양 모서리를 둥근 모양으로 지정(반지름은 5dp) (Specify output shape corners as rounded (radius is 5dp))

The preview window on the right shows a light gray rounded rectangle with a brown border, representing the visual output of the XML code.

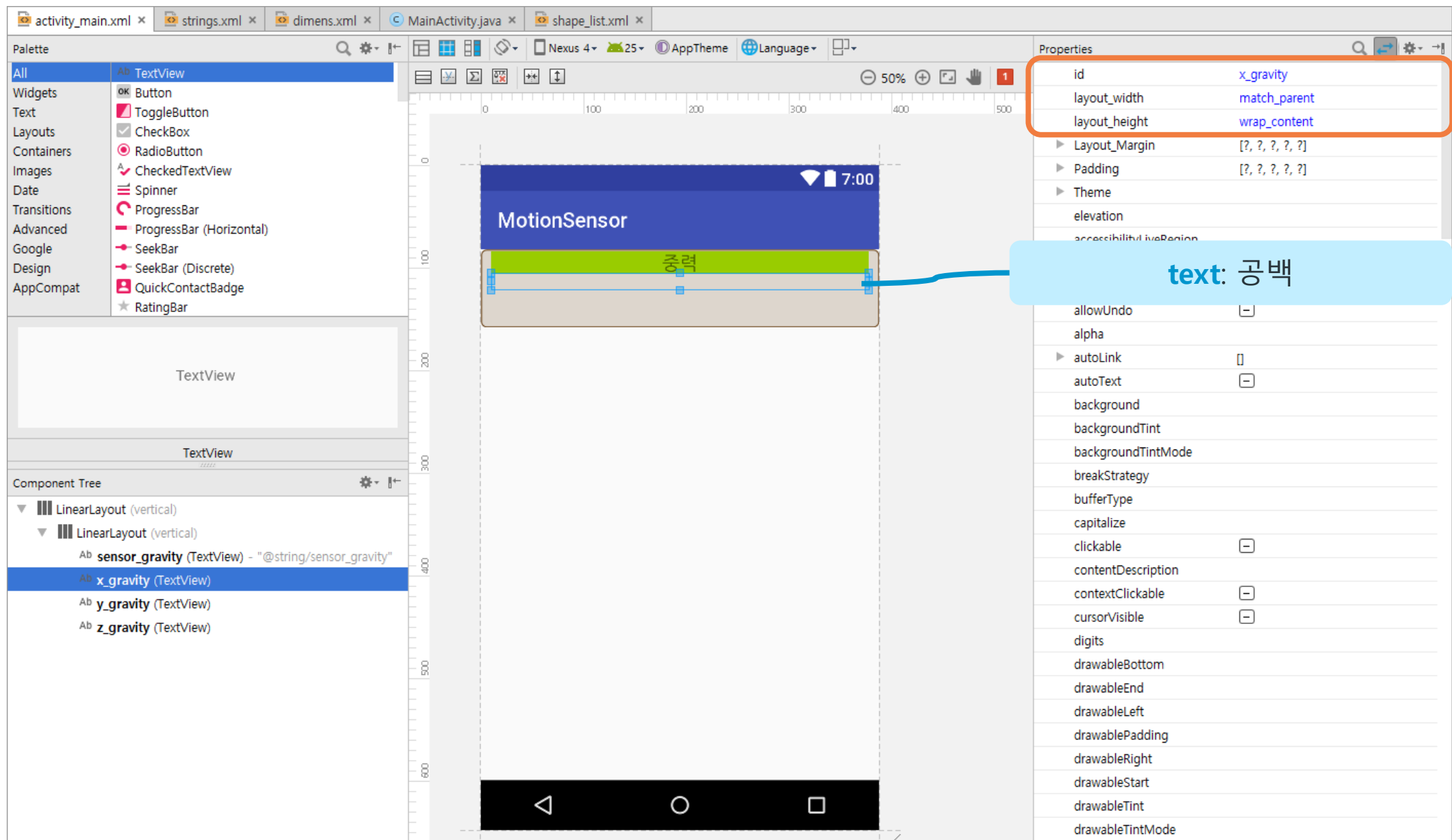
2.4 화면 설계-중력

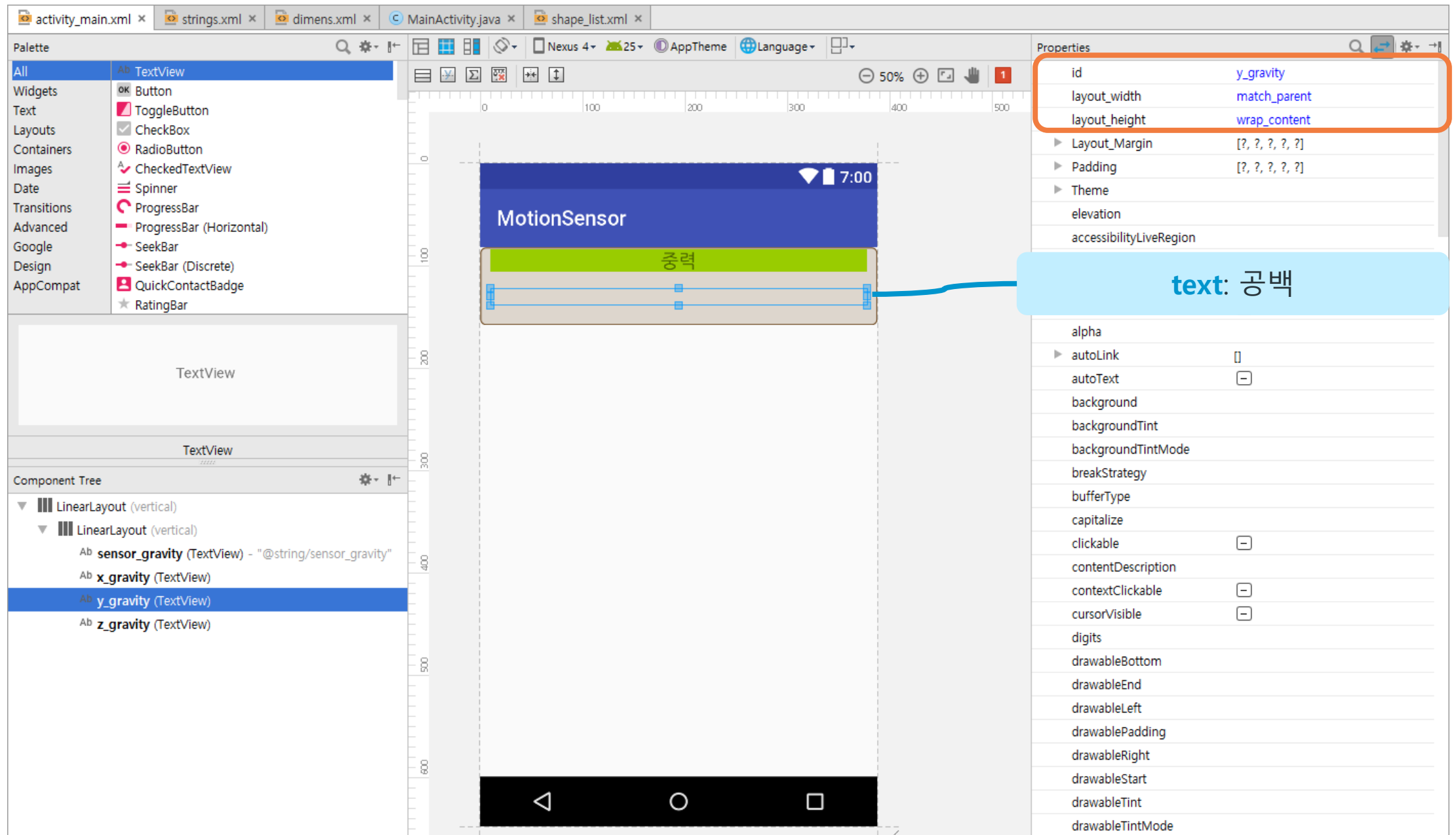


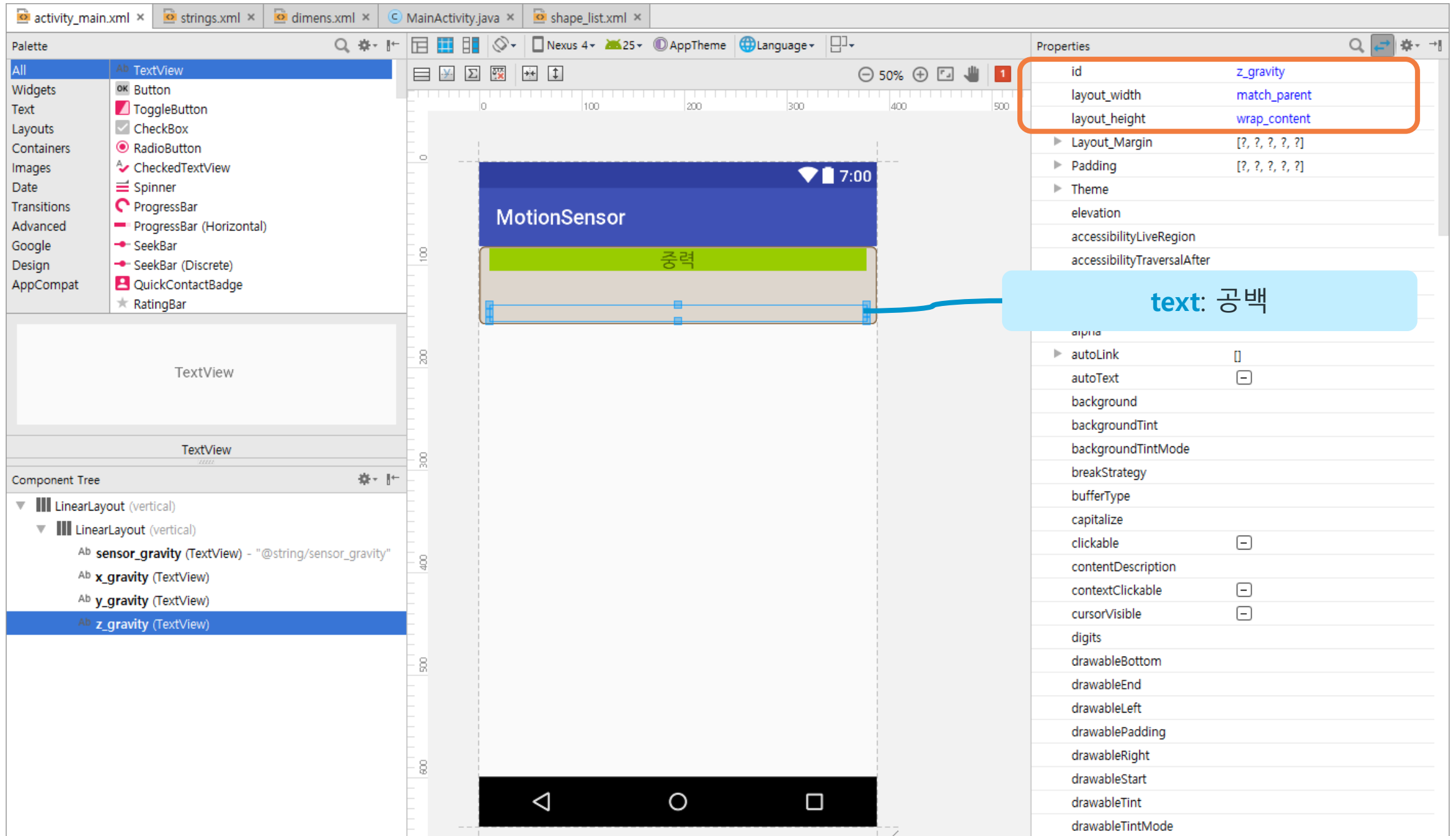
The screenshot displays the Android Studio IDE with the following components:

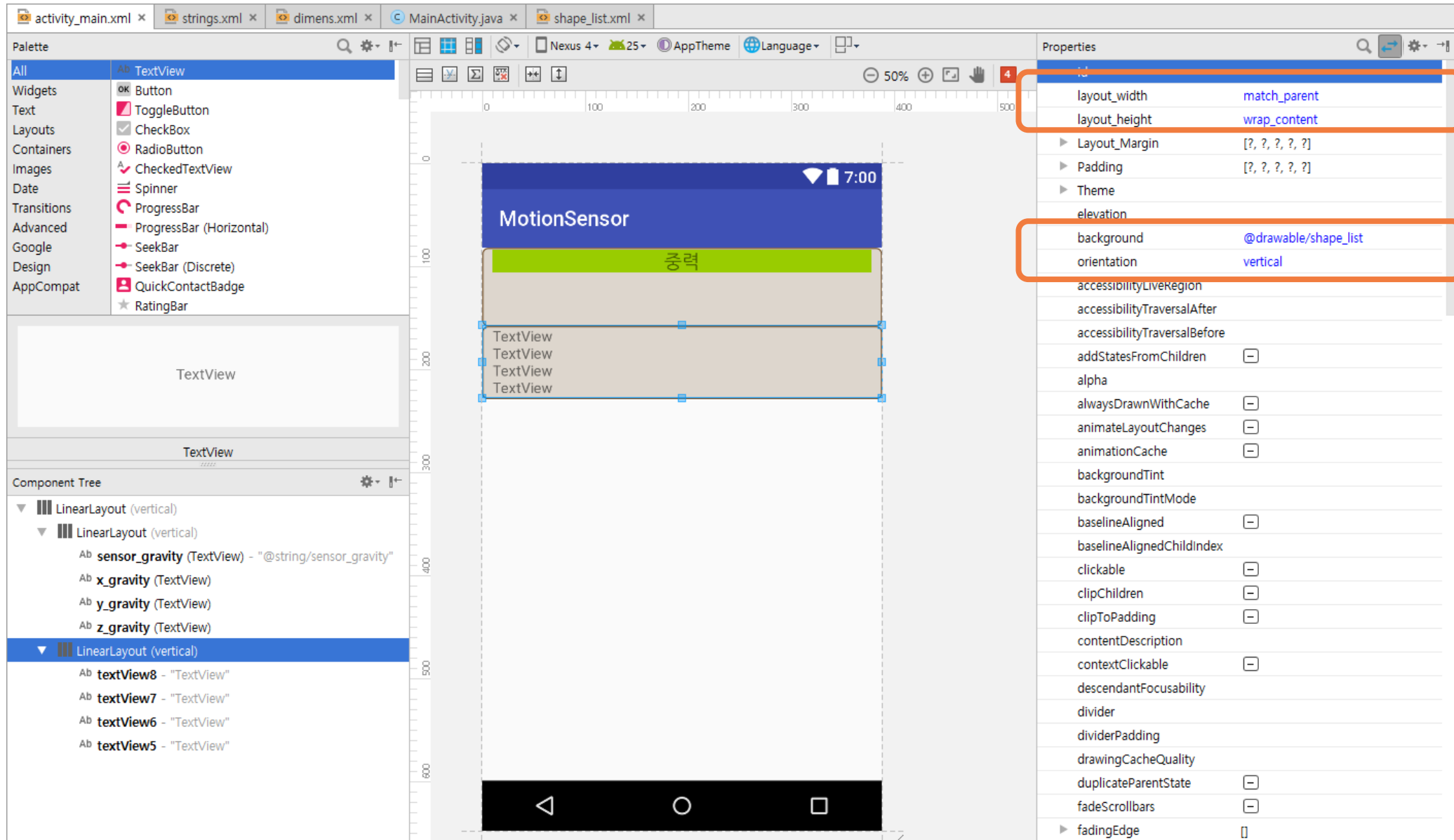
- Top Bar:** Tabs for `activity_main.xml`, `strings.xml`, `dimens.xml`, `MainActivity.java`, and `shape_list.xml`. The toolbar shows the Nexus 4 device, API level 25, AppTheme, and Language settings.
- Palette:** A list of widgets including TextView, Button, ToggleButton, CheckBox, RadioButton, CheckedTextView, Spinner, ProgressBar, SeekBar, and QuickContactBadge.
- Design Canvas:** A visual representation of the app layout. It features a blue header with the text "MotionSensor", a green bar below it, and a light gray background. The canvas is overlaid with a grid and a ruler.
- Properties Panel:** A table of properties for the selected `TextView1` widget. The properties are as follows:

id	TextView1
layout_width	match_parent
layout_height	wrap_content
Layout_Margin	[?, ?, ?, ?]
Padding	[?, ?, ?, ?]
Theme	
elevation	
background	@android:color/holo_green_light
gravity	[center]
text	@string/sensor_gravity
textSize	18sp
accessibilityLiveRegion	
accessibilityTraversalAfter	
accessibilityTraversalBefore	
allowUndo	-
alpha	
autoLink	[]
autoText	-
backgroundTint	
backgroundTintMode	
breakStrategy	
bufferType	
capitalize	
clickable	-
contentDescription	
contextClickable	-
cursorVisible	-
digits	
drawableBottom	
drawableEnd	
drawableLeft	
drawablePadding	
drawableRight	







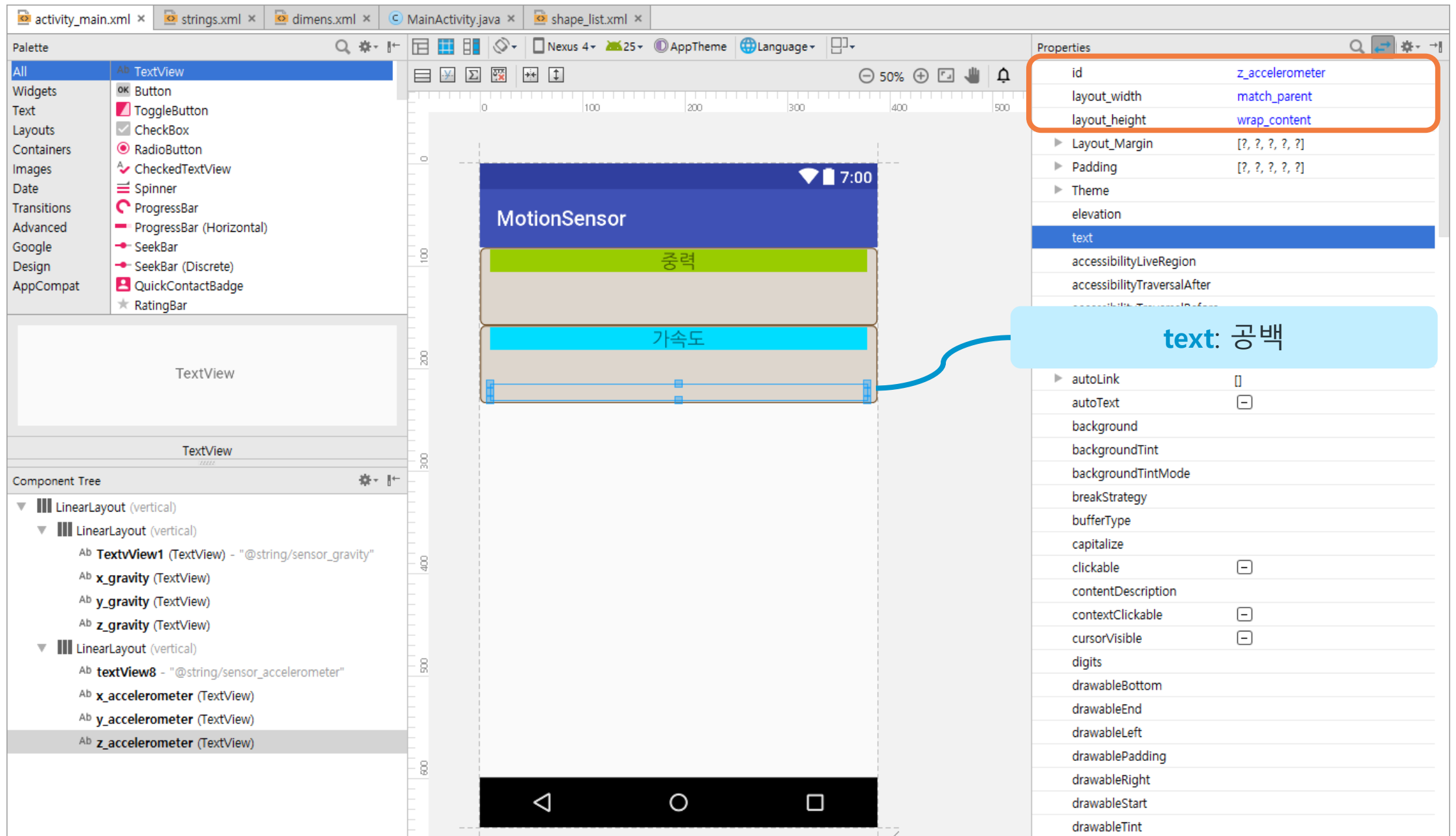


The screenshot displays the Android Studio IDE with the following components:

- Palette:** Shows various Android widgets categorized by type (All, Widgets, Text, Layouts, Containers, Images, Date, Transitions, Advanced, Google, Design, AppCompat). The **TextView** widget is selected.
- Component Tree:** Displays the hierarchy of the UI components:
 - LinearLayout (vertical)
 - LinearLayout (vertical)
 - TextView1 (TextView) - "@string/sensor_gravity"
 - x_gravity (TextView)
 - y_gravity (TextView)
 - z_gravity (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_accelerometer"
 - x_accelerometer (TextView)** (Selected)
 - textView6 - "TextView"
 - textView5 - "TextView"
- Design Canvas:** Shows a visual representation of the app's layout. It includes a status bar at the top with the time 7:00, a blue header labeled "MotionSensor", and several colored bars representing different sensor data (green for gravity, blue for acceleration). A blue callout box points to a text view within the acceleration bar, indicating its text content.
- Properties Panel:** Shows the properties of the selected **x_accelerometer (TextView)** widget. The properties listed are:
 - id: x_accelerometer
 - layout_width: match_parent
 - layout_height: wrap_content
 - Layout_Margin: [?, ?, ?, ?]
 - Padding: [?, ?, ?, ?]
 - Theme: elevation
 - text: 공백 (Highlighted by a blue callout box)
 - accessibilityLiveRegion
 - alpha
 - autoLink: []
 - autoText: [-]
 - background
 - backgroundTint
 - backgroundTintMode
 - breakStrategy
 - bufferType
 - capitalize
 - clickable: [-]
 - contentDescription
 - contextClickable: [-]
 - cursorVisible: [-]
 - digits
 - drawableBottom
 - drawableEnd
 - drawableLeft
 - drawablePadding
 - drawableRight
 - drawableStart
 - drawableTint

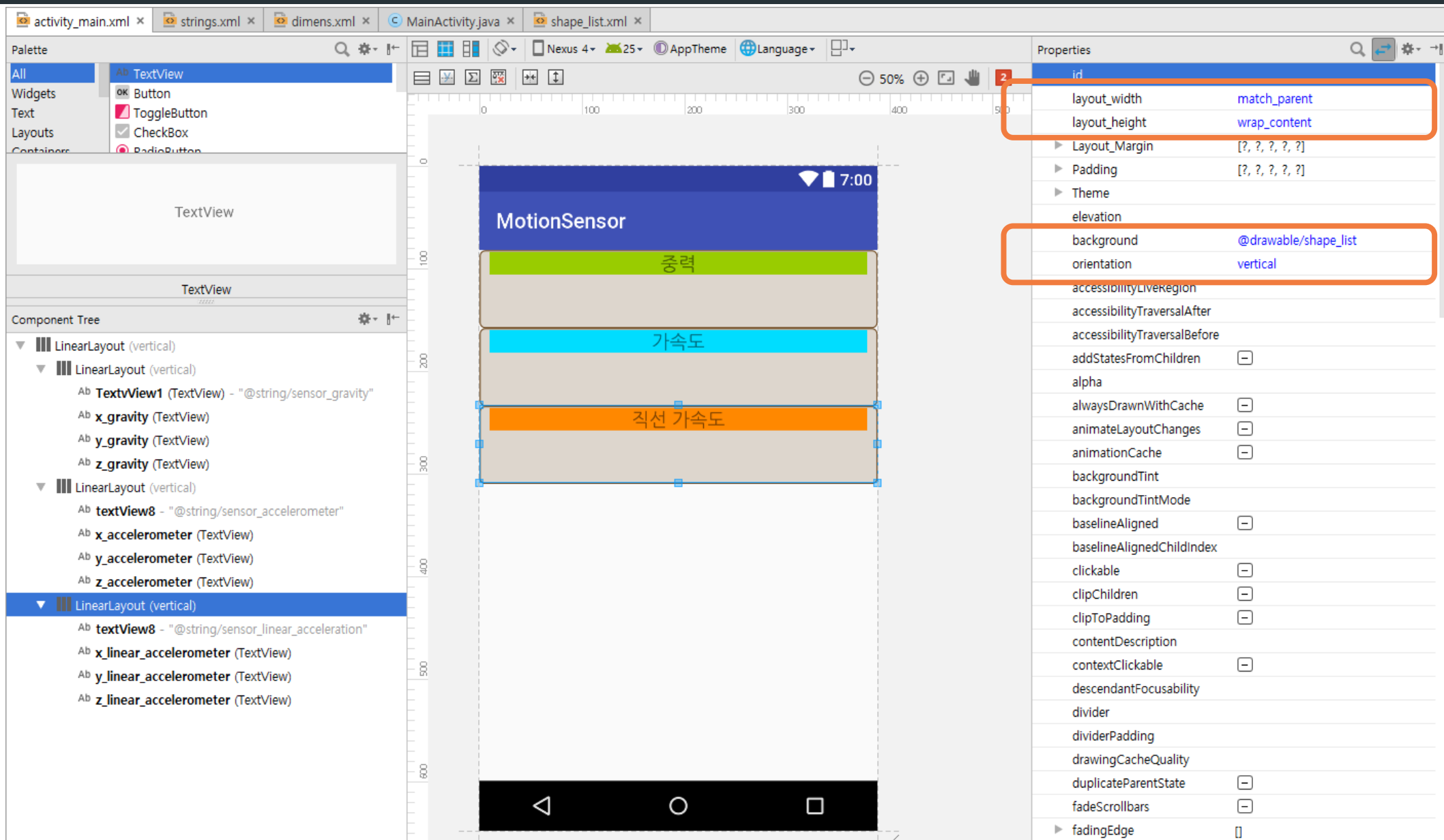
The screenshot displays the Android Studio IDE with the following components:

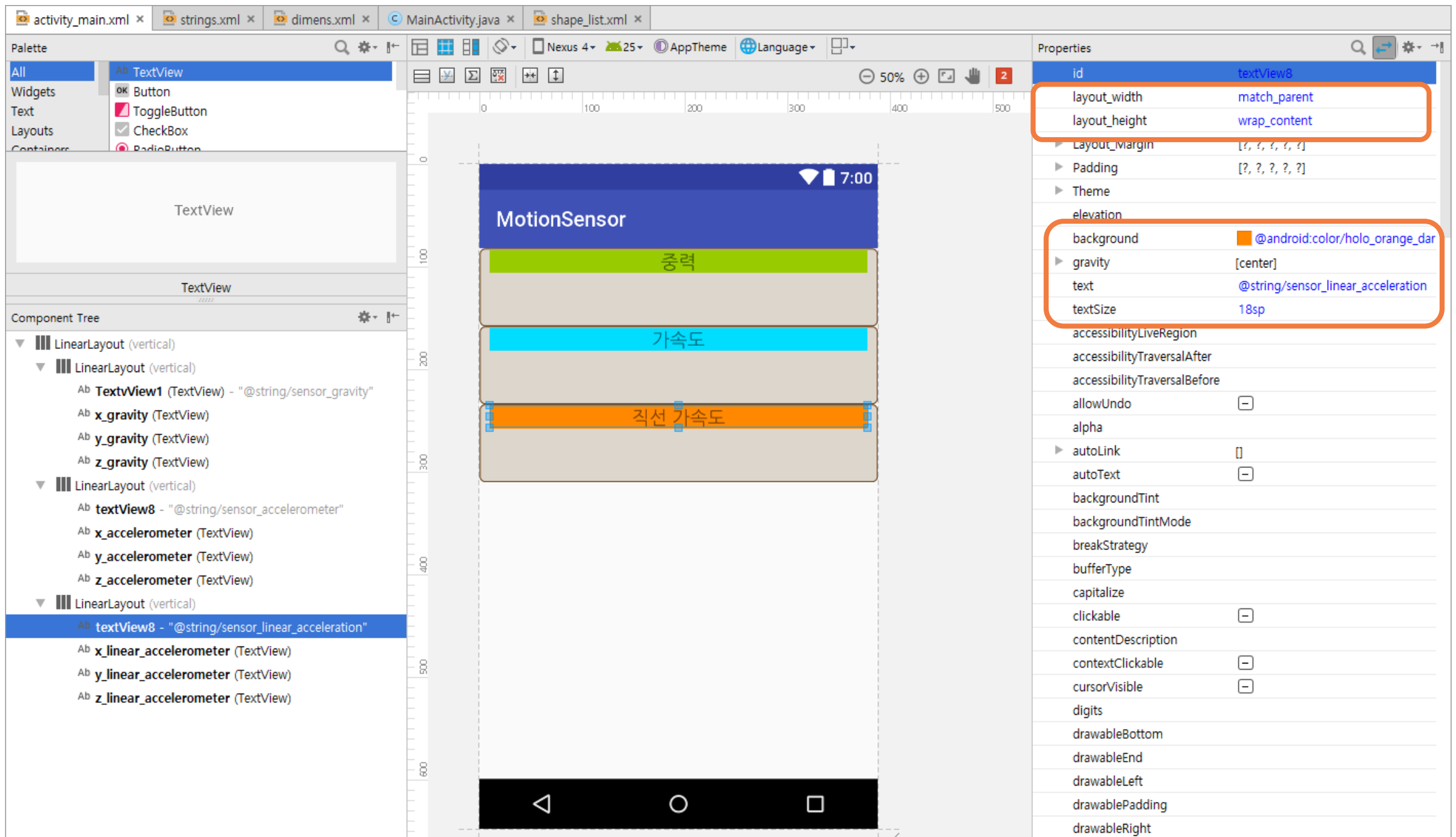
- Palette:** A list of UI widgets including TextView, Button, ToggleButton, CheckBox, RadioButton, CheckedTextView, Spinner, ProgressBar, SeekBar, and RatingBar.
- Component Tree:** A hierarchical view of the app's layout structure, showing nested LinearLayouts and individual TextViews.
- Properties:** A panel on the right showing the attributes of the selected TextView. A red box highlights the 'id' (y_accelerometer), 'layout_width' (match_parent), and 'layout_height' (wrap_content) properties.
- Design Canvas:** A visual representation of the app's UI. It features a 'MotionSensor' header, a green bar for '중력' (Gravity), and a blue bar for '가속도' (Acceleration). Below these are two horizontal sliders. A blue callout box points to the bottom slider with the text 'text: 공백' (text: blank).



화면설계-직선가속도 (입력하지 않음)

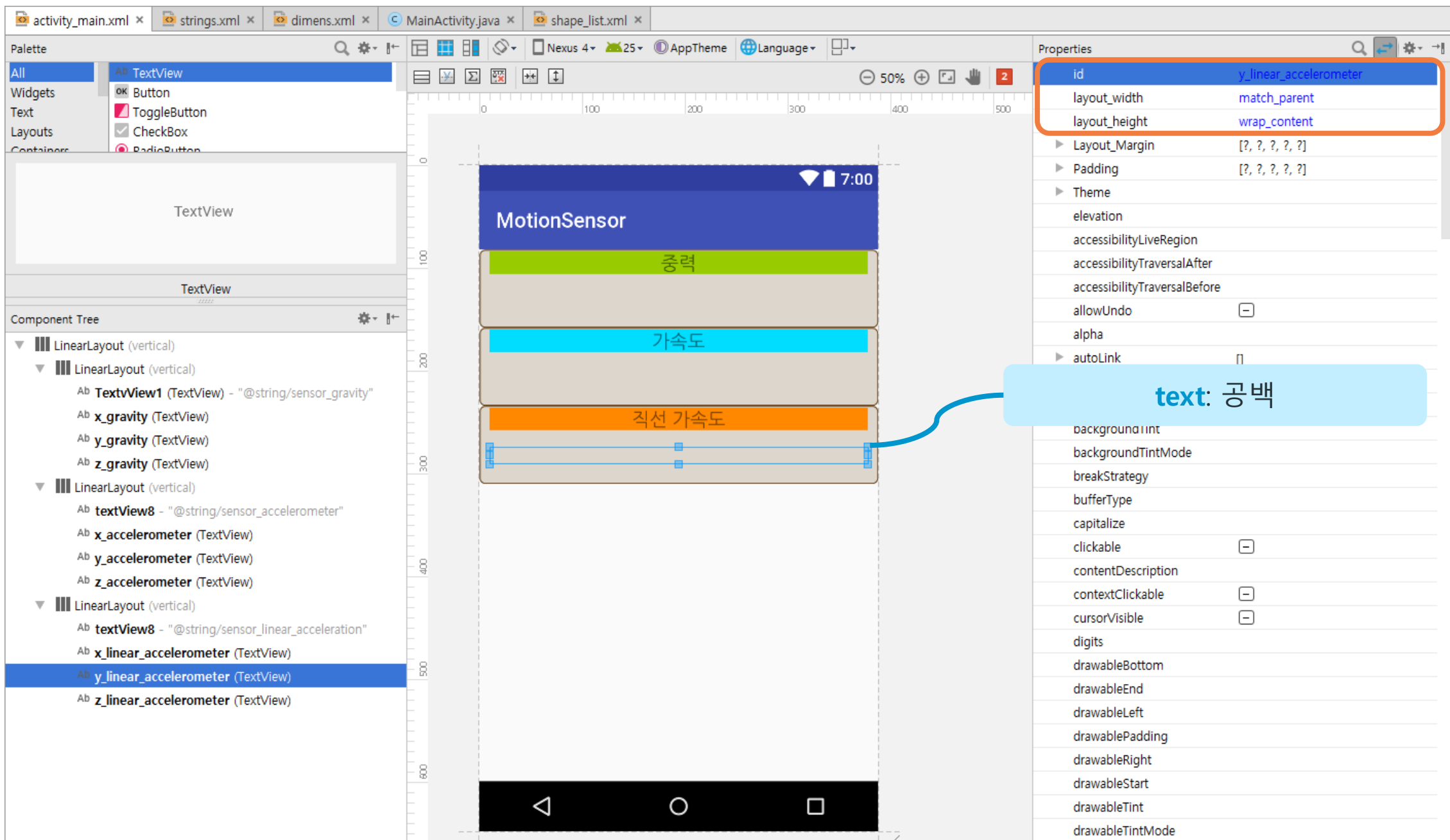
40

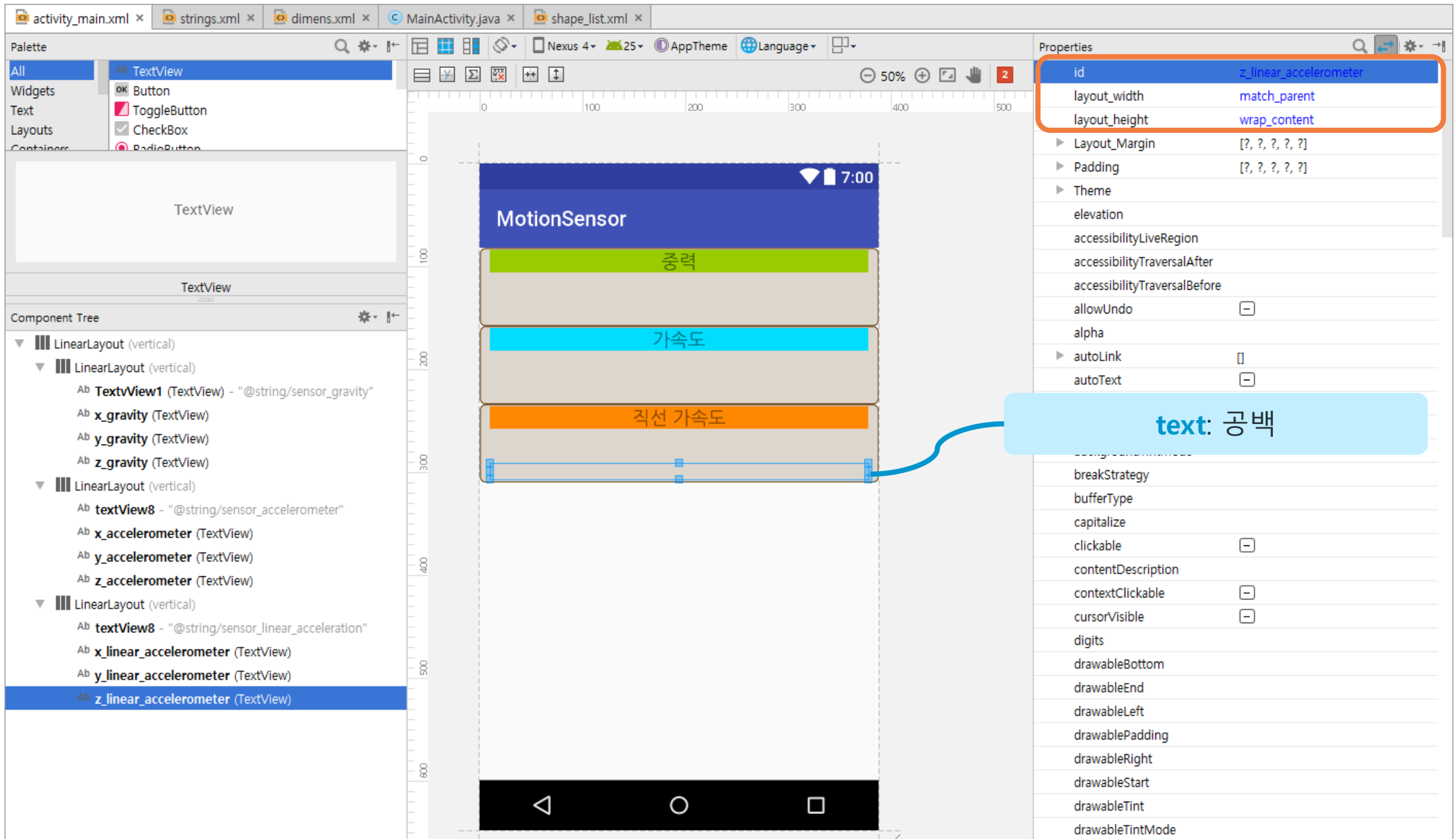




The screenshot displays the Android Studio IDE with the following components:

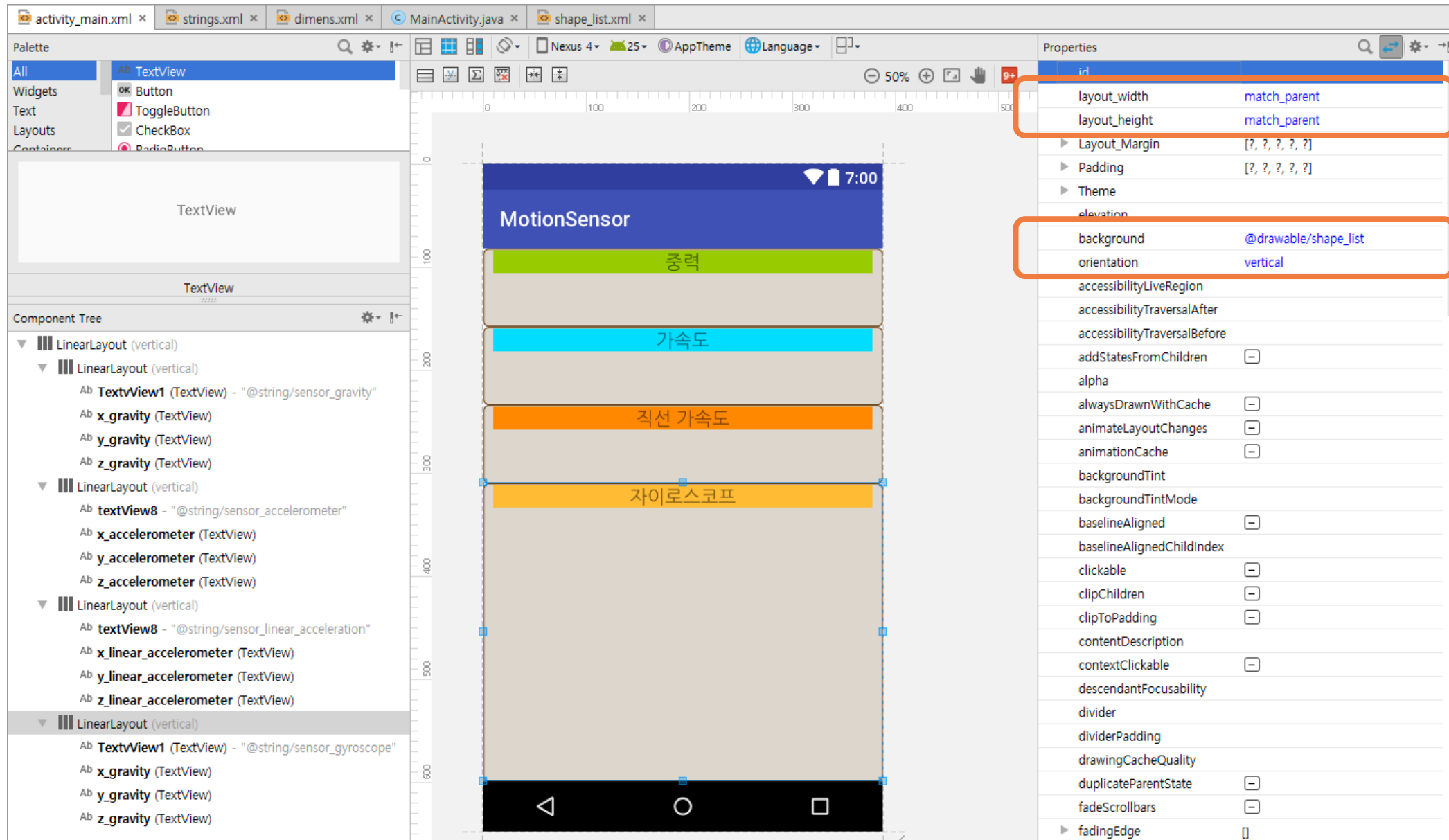
- Palette:** Shows various widgets including TextView, Button, ToggleButton, CheckBox, and RadioButton.
- Component Tree:** Displays the hierarchy of the UI components:
 - LinearLayout (vertical)
 - LinearLayout (vertical)
 - TextView1 (TextView) - "@string/sensor_gravity"
 - x_gravity (TextView)
 - y_gravity (TextView)
 - z_gravity (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_accelerometer"
 - x_accelerometer (TextView)
 - y_accelerometer (TextView)
 - z_accelerometer (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_linear_acceleration"
 - x_linear_accelerometer (TextView)** (Selected)
 - y_linear_accelerometer (TextView)
 - z_linear_accelerometer (TextView)
- Properties Panel:** Shows the properties for the selected **x_linear_accelerometer** TextView, including id, layout_width, layout_height, Layout_Margin, Padding, Theme, elevation, accessibilityLiveRegion, accessibilityTraversalAfter, accessibilityTraversalBefore, allowUndo, alpha, and autolink. A blue callout bubble points to the **text: 공백** property.
- Design Canvas:** Shows a visual representation of the app's UI. It features a blue header with the text "MotionSensor", a green bar with "중력", a cyan bar with "가속도", and an orange bar with "직선 가속도". The bottom of the canvas shows a black navigation bar with three icons.

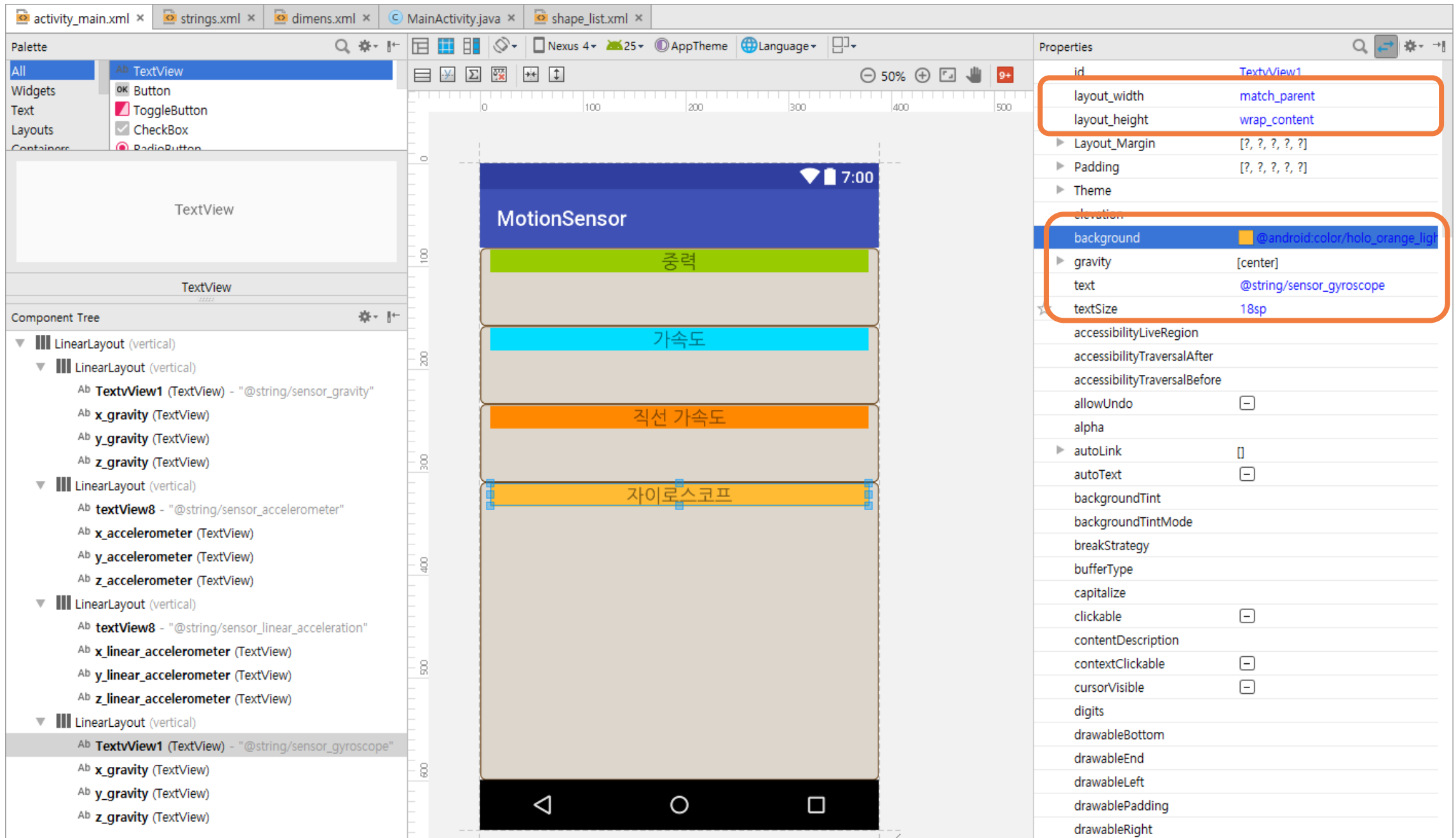




화면설계-자이로스코프 [입력하지 않음]

45





activity_main.xml x strings.xml x dimens.xml x MainActivity.java x shape_list.xml x

Palette

All

Widgets

Text

Layouts

Containers

TextView

Button

ToggleButton

CheckBox

RadioButton

Component Tree

- LinearLayout (vertical)
 - LinearLayout (vertical)
 - TextView1 (TextView) - "@string/sensor_gravity"
 - x_gravity (TextView)
 - y_gravity (TextView)
 - z_gravity (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_accelerometer"
 - x_accelerometer (TextView)
 - y_accelerometer (TextView)
 - z_accelerometer (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_linear_acceleration"
 - x_linear_accelerometer (TextView)
 - y_linear_accelerometer (TextView)
 - z_linear_accelerometer (TextView)
 - LinearLayout (vertical)
 - TextView1 (TextView) - "@string/sensor_gyroscope"
 - x_gyroscope (TextView)**
 - y_gyroscope (TextView)
 - z_gyroscope (TextView)

Properties

id	x_gyroscope
layout_width	match_parent
layout_height	wrap_content
Layout_margin	[?, ?, ?, ?]
Padding	[?, ?, ?, ?]
Theme	
elevation	
accessibilityLiveRegion	
accessibilityTraversalAfter	
accessibilityTraversalBefore	
allowUndo	<input type="checkbox"/>
alpha	
autoLink	<input type="checkbox"/>
autoText	<input type="checkbox"/>
background	
backgroundTint	
bufferType	
capitalize	
clickable	<input type="checkbox"/>
contentDescription	
contextClickable	<input type="checkbox"/>
cursorVisible	<input type="checkbox"/>
digits	
drawableBottom	
drawableEnd	
drawableLeft	
drawablePadding	
drawableRight	
drawableStart	
drawableTint	
drawableTintMode	

text: 공백

MotionSensor

중력

가속도

직선 가속도

자이로스코프

activity_main.xml x strings.xml x dimens.xml x MainActivity.java x shape_list.xml x

Palette

- All
- Widgets
- Text
 - TextView
- Layouts
- Containers

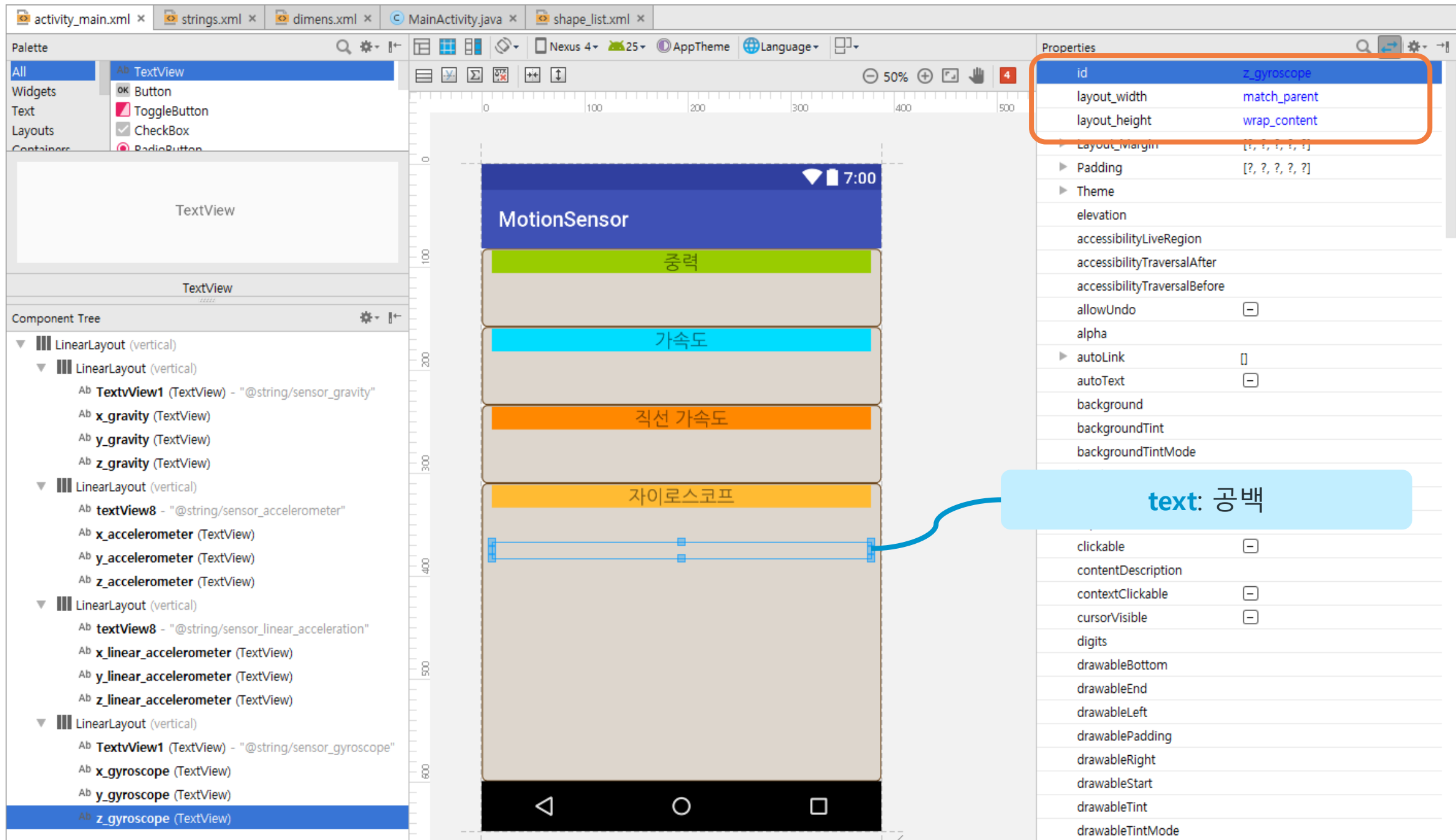
Component Tree

- LinearLayout (vertical)
 - LinearLayout (vertical)
 - TextView1 (TextView) - "@string/sensor_gravity"
 - x_gravity (TextView)
 - y_gravity (TextView)
 - z_gravity (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_accelerometer"
 - x_accelerometer (TextView)
 - y_accelerometer (TextView)
 - z_accelerometer (TextView)
 - LinearLayout (vertical)
 - textView8 - "@string/sensor_linear_acceleration"
 - x_linear_accelerometer (TextView)
 - y_linear_accelerometer (TextView)
 - z_linear_accelerometer (TextView)
 - LinearLayout (vertical)
 - TextView1 (TextView) - "@string/sensor_gyroscope"
 - x_gyroscope (TextView)
 - y_gyroscope (TextView)
 - z_gyroscope (TextView)

Properties

id	y_gyroscope
layout_width	match_parent
layout_height	wrap_content
Layout_Margin	[?, ?, ?, ?]
Padding	[?, ?, ?, ?]
Theme	
elevation	
accessibilityLiveRegion	
accessibilityTraversalAfter	
accessibilityTraversalBefore	
allowUndo	<input type="checkbox"/>
alpha	
autoLink	<input type="checkbox"/>
autoText	<input type="checkbox"/>
background	
backgroundTint	
capitalize	
clickable	<input type="checkbox"/>
contentDescription	
contextClickable	<input type="checkbox"/>
cursorVisible	<input type="checkbox"/>
digits	
drawableBottom	
drawableEnd	
drawableLeft	
drawablePadding	
drawableRight	
drawableStart	
drawableTint	
drawableTintMode	

text: 공백



2.5 Activity 제어(MainActivity.java)

50

- 센서이벤트 처리를 위한 액티비티 인터페이스 추가

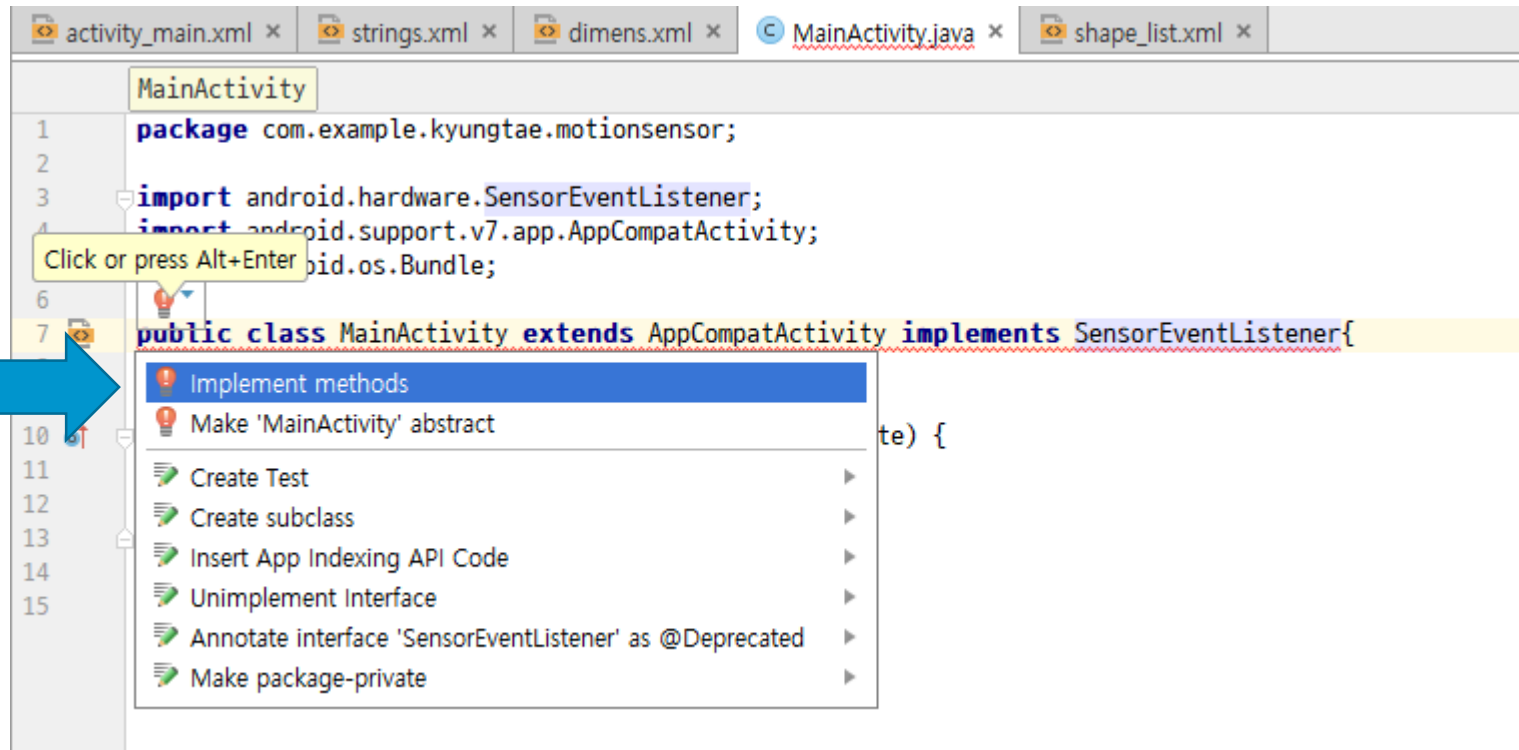
```
activity_main.xml x strings.xml x dimens.xml x MainActivity.java x shape_list.xml x
1 package com.example.kyungtae.motionsensor;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 public class MainActivity extends AppCompatActivity {
7
8     @Override
9     protected 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 x strings.xml x dimens.xml x MainActivity.java x shape_list.xml x
MainActivity
1 package com.example.kyungtae.motionsensor;
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 }
15
```

• 센서값 처리를 위한 매소드 구현(@override)

51

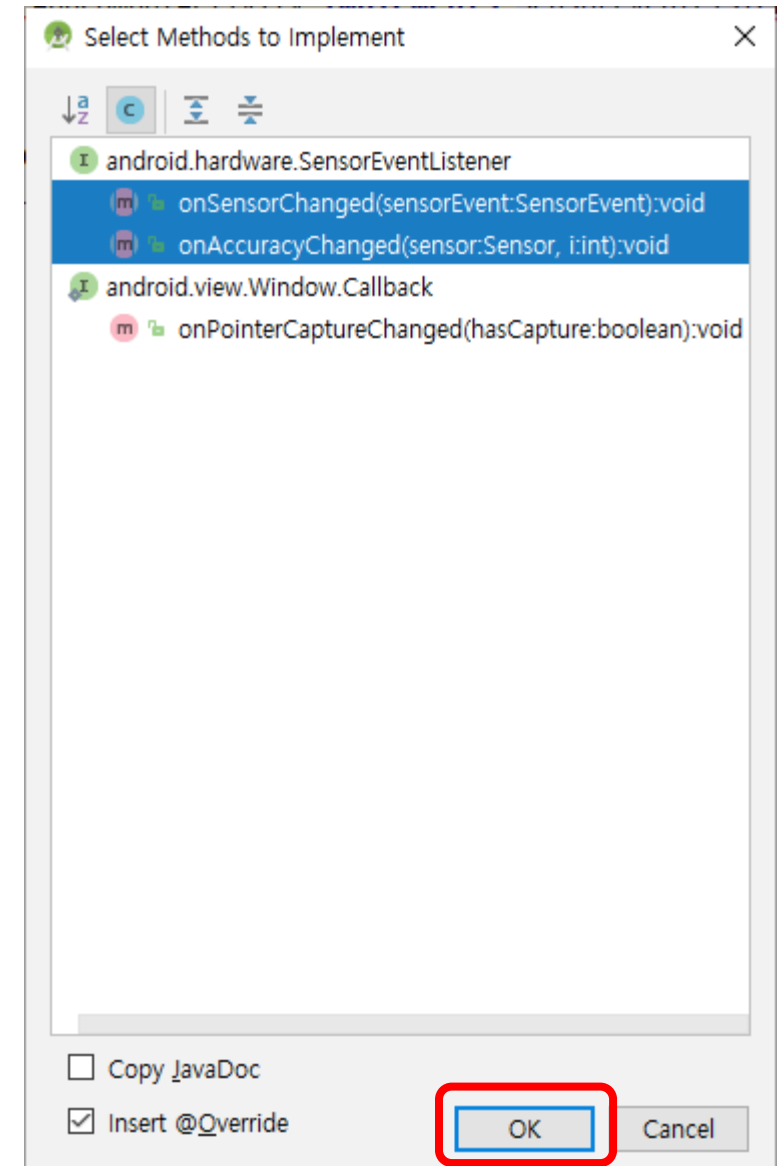


The screenshot shows an IDE with several tabs: activity_main.xml, strings.xml, dimens.xml, MainActivity.java, and shape_list.xml. The MainActivity.java file is open, showing the following code:

```
1 package com.example.kyungtae.motionsensor;  
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  
10  
11  
12  
13  
14  
15
```

A context menu is open over the `implements SensorEventListener` line. The menu options are:

- Implement methods (highlighted with a blue arrow)
- Make 'MainActivity' abstract
- Create Test
- Create subclass
- Insert App Indexing API Code
- Unimplement Interface
- Annotate interface 'SensorEventListener' as @Deprecated
- Make package-private



```
activity_main.xml x strings.xml x dimens.xml x MainActivity.java x shape_list.xml x

1 package com.example.kyungtae.motionsensor;
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
22     @Override
23     public void onAccuracyChanged(Sensor sensor, int accuracy) {
24
25     }
26 }
27
```

센서 값이 변할 때 호출

등록된 센서의 정확도가 변할 때 호출

- 센서와 센서 값을 표시하기 위한 변수 선언 (빨간상자만 입력)

```
11 public class MainActivity extends AppCompatActivity implements SensorEventListener{
12
13     // gravity
14     TextView x_gravity;
15     TextView y_gravity;
16     TextView z_gravity;
17     // accelerometer
18     TextView x_accelerometer;
19     TextView y_accelerometer;
20     TextView z_accelerometer;
21     // linear acceleration
22     TextView x_linear_acceleration;
23     TextView y_linear_acceleration;
24     TextView z_linear_acceleration;
25     // gyroscope
26     TextView x_gyroscope;
27     TextView y_gyroscope;
28     TextView z_gyroscope;
29     // magnetic field
30     TextView x_magnetic_field;
31     TextView y_magnetic_field;
32     TextView z_magnetic_field;
33
34     // Sensor Manager
35     SensorManager sm;
36     // Sensor
37     Sensor sensor_gravity;
38     Sensor sensor_accelerometer;
39     Sensor sensor_linear_acceleration;
40     Sensor sensor_gyroscope;
41     Sensor sensor_magnetic_field;
```


• 센서값 출력을 위한 텍스트뷰 인식하기 (빨간상자만 입력)

@Override

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

```
    x_gravity = (TextView) findViewById(R.id.x_gravity);  
    y_gravity = (TextView) findViewById(R.id.y_gravity);  
    z_gravity = (TextView) findViewById(R.id.z_gravity);
```

```
    x_accelerometer = (TextView) findViewById(R.id.x_accelerometer);  
    y_accelerometer = (TextView) findViewById(R.id.y_accelerometer);  
    z_accelerometer = (TextView) findViewById(R.id.z_accelerometer);
```

```
    x_linear_acceleration = (TextView) findViewById(R.id.x_linear_acceleration);  
    y_linear_acceleration = (TextView) findViewById(R.id.y_linear_acceleration);  
    z_linear_acceleration = (TextView) findViewById(R.id.z_linear_acceleration);
```

```
    x_gyroscope = (TextView) findViewById(R.id.x_gyroscope);  
    y_gyroscope = (TextView) findViewById(R.id.y_gyroscope);  
    z_gyroscope = (TextView) findViewById(R.id.z_gyroscope);
```

```
    x_magnetic_field = (TextView) findViewById(R.id.x_magnetic);  
    y_magnetic_field = (TextView) findViewById(R.id.y_magnetic);  
    z_magnetic_field = (TextView) findViewById(R.id.z_magnetic);
```

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

```
    sensor_gravity = sm.getDefaultSensor(Sensor.TYPE_GRAVITY);  
    sensor_accelerometer = sm.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);  
    sensor_linear_acceleration = sm.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);  
    sensor_gyroscope = sm.getDefaultSensor(Sensor.TYPE_GYROSCOPE);  
    sensor_magnetic_field = sm.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
```

```
}
```

디바이스 센서 접근을 위한 객체 생성

중력 센서 측정을 위한 객체 생성

중력가속도 센서 측정을 위한 객체 생성

직선 가속도 센서 측정을 위한 객체 생성

자이로스코프 센서 측정을 위한 객체 생성

자기장 센서 측정을 위한 객체 생성

• 센서값이 변할 때 호출

```
80      @Override
81      public void onSensorChanged(SensorEvent event) {
82          switch (event.sensor.getType()){
83              case Sensor.TYPE_GRAVITY:
84                  x_gravity.setText("X: " + event.values[0]);
85                  y_gravity.setText("Y: " + event.values[1]);
86                  z_gravity.setText("Z: " + event.values[2]);
87                  break;
88              case Sensor.TYPE_ACCELEROMETER:
89                  x_accelerometer.setText("X: " + event.values[0]);
90                  y_accelerometer.setText("Y: " + event.values[1]);
91                  z_accelerometer.setText("Z: " + event.values[2]);
92                  break;
93              case Sensor.TYPE_LINEAR_ACCELERATION:
94                  x_linear_acceleration.setText("X: " + event.values[0]);
95                  y_linear_acceleration.setText("Y: " + event.values[1]);
96                  z_linear_acceleration.setText("Z: " + event.values[2]);
97                  break;
98              case Sensor.TYPE_GYROSCOPE:
99                  x_gyroscope.setText("X: " + event.values[0]);
100                 y_gyroscope.setText("Y: " + event.values[1]);
101                 z_gyroscope.setText("Z: " + event.values[2]);
102                 break;
103              case Sensor.TYPE_MAGNETIC_FIELD:
104                  x_magnetic_field.setText("X: " + event.values[0]);
105                  y_magnetic_field.setText("Y: " + event.values[1]);
106                  z_magnetic_field.setText("Z: " + event.values[2]);
107                  break;
108          }
109      }
```

중력 센서 값 출력

중력가속도 센서 값 출력

직선가속도 센서 값 출력

자이로스코프 센서 값 출력

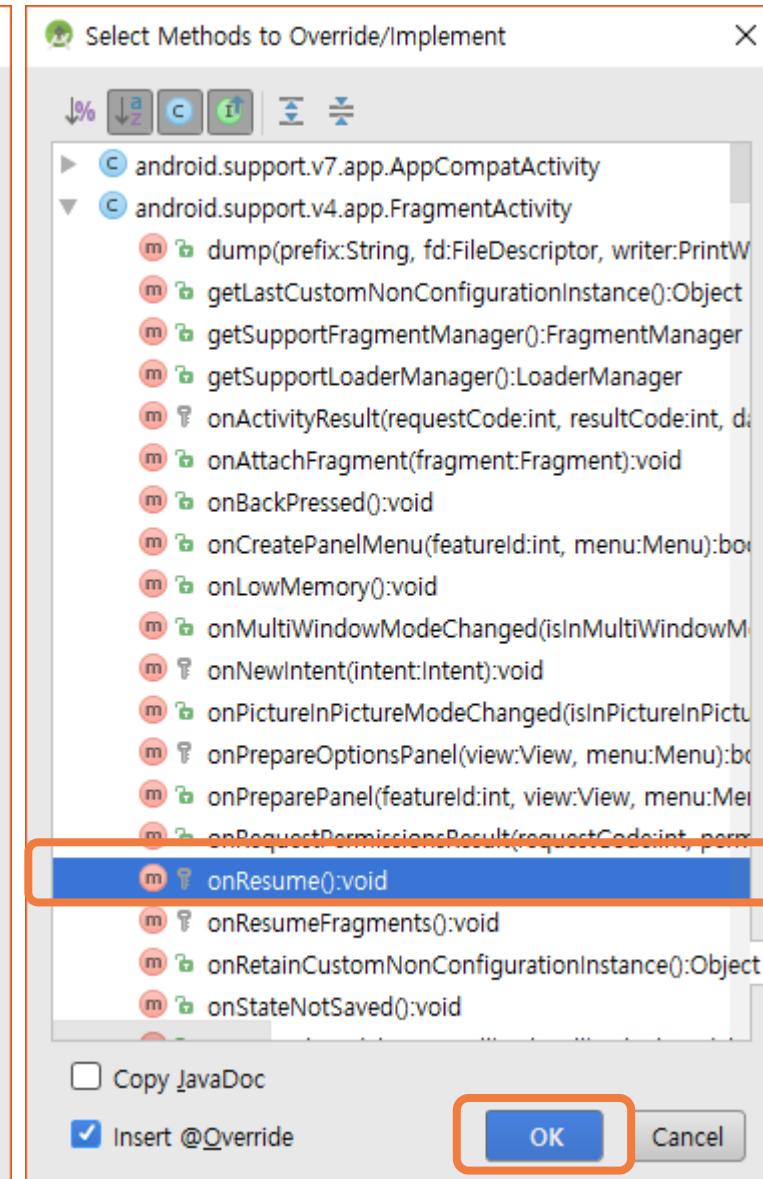
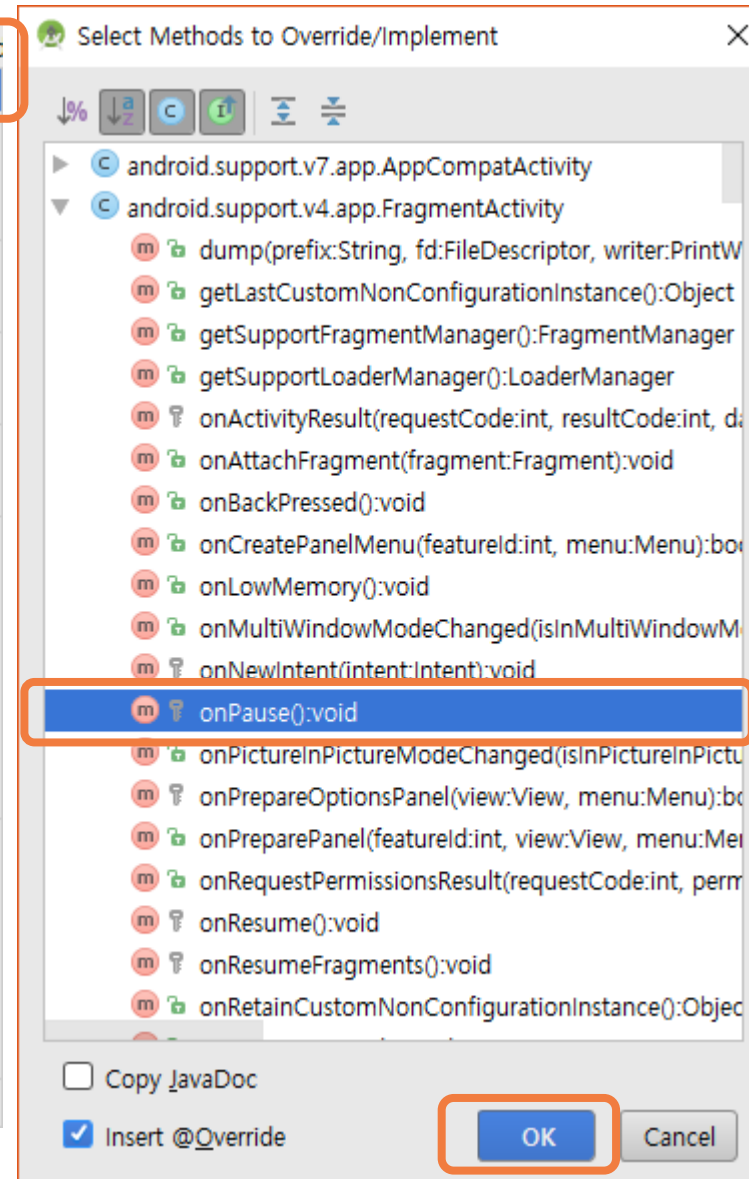
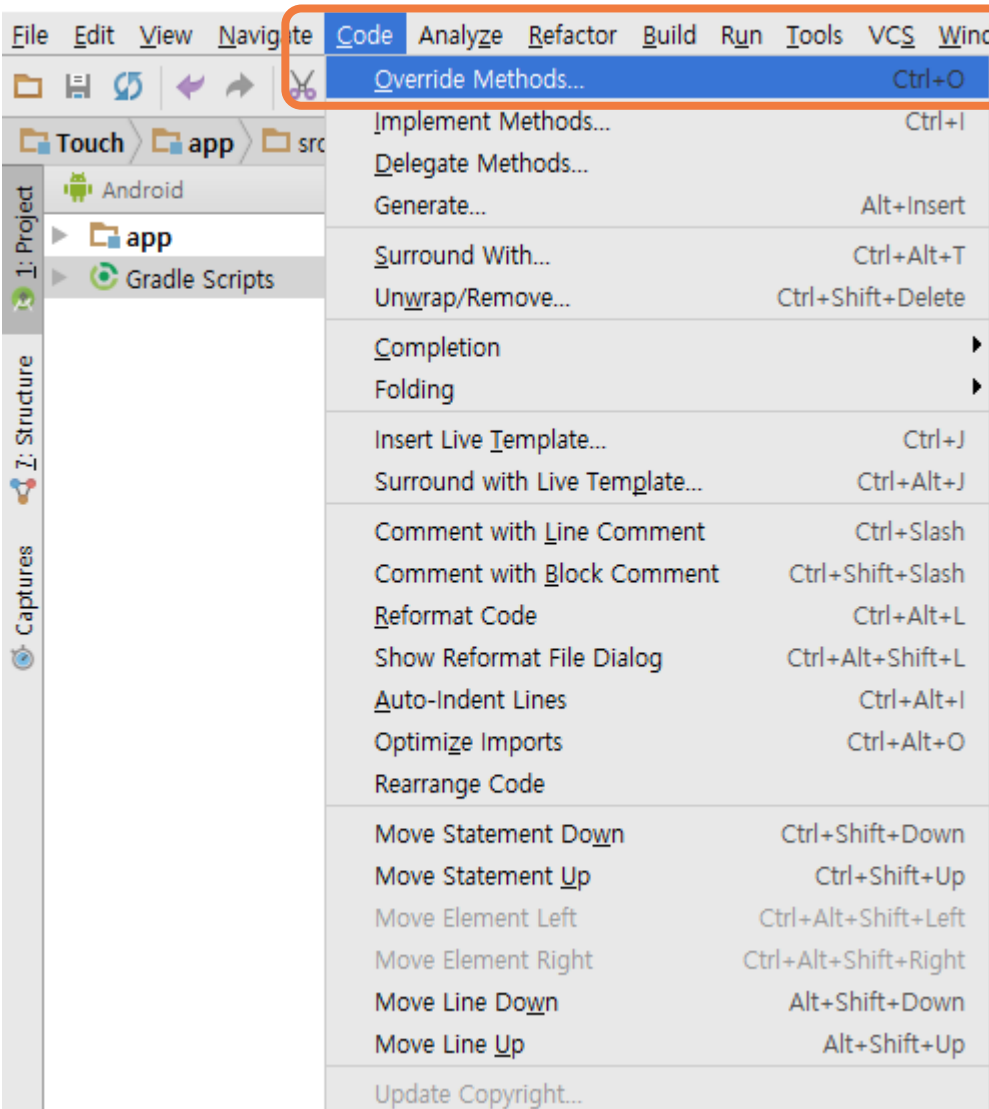
자기장 센서 값 출력

Activity LifeCycle





메소드	설명	다음 메소드
onCreate()	액티비티가 생성될 때 호출되며 사용자 인터페이스 초기화에 사용됨.	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

onPause()/onResume() 매소드 재정의(Override)

58



• 재정의를 위한 매소드 추가

```
40  
41    
42  
43  
44  
45  
46    
47  
48  
49
```

```
@Override  
protected void onPause() {  
    super.onPause();  
}
```

화면에 표시되는 상태에서 사용자와 상호 작용하지 않을 때

```
@Override  
protected void onResume() {  
    super.onResume();  
}
```

액티비티가 일시정지(pause)상태에서 복 귀할 때 호출

• onPause()/onResume() 매소드 재정의 (빨간상자만 추가입력)

```
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
```

```
@Override
protected void onPause() {
    super.onPause();
    sm.unregisterListener(this);
}

@Override
protected void onResume() {
    super.onResume();

    sm.registerListener( listener: this, sensor_gravity, SensorManager.SENSOR_DELAY_NORMAL);
    sm.registerListener( listener: this, sensor_accelerometer, SensorManager.SENSOR_DELAY_NORMAL);
    sm.registerListener( listener: this, sensor_linear_acceleration, SensorManager.SENSOR_DELAY_NORMAL);
    sm.registerListener( listener: this, sensor_gyroscope, SensorManager.SENSOR_DELAY_NORMAL);
    sm.registerListener( listener: this, sensor_magnetic_field, SensorManager.SENSOR_DELAY_NORMAL);
}
```

등록된 센서 리스너 해제

중력 센서 값 변화에 대한 이벤트를 감지하기 위한 리스너 등록

중력가속도 센서 값 변화에 대한 이벤트를 감지하기 위한 리스너 등록

자이로스코프 센서 값 변화에 대한 이벤트를 감지하기 위한 리스너 등록

자기장 센서 값 변화에 대한 이벤트를 감지하기 위한 리스너 등록

직선가속도 센서 값 변화에 대한 이벤트를 감지하기 위한 리스너 등록

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

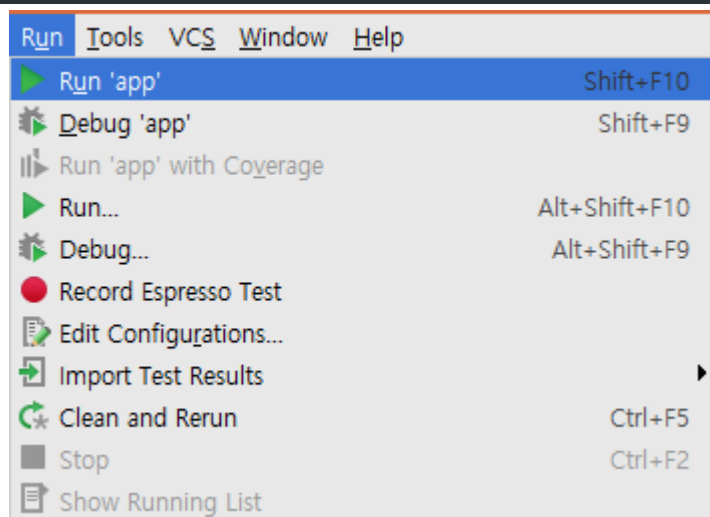
클래스와 속성/메소드

• 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</code> (Sensor, int accuracy)	등록된 센서의 정확도가 변할 때 호출됨
	abstract void <code>onSensorChanged</code> (SensorEvent event)	센서 값이 변할 때 호출됨
SensorManager	Sensor <code>getDefaultSensor</code> (int type)	주어진 type을 위한 디폴트 센서를 얻기 위해 사용
	Boolean <code>registerListener</code> (SensorEventListener listener, Sensor sensor, int samplingPeriodUs)	주어진 샘플링 주파수에서 주어진 센서를 위한 SensorEventListener를 등록함
	void <code>unregisterListener</code> (SensorEventListener listener)	모든 센서에 대한 리스너를 해제함

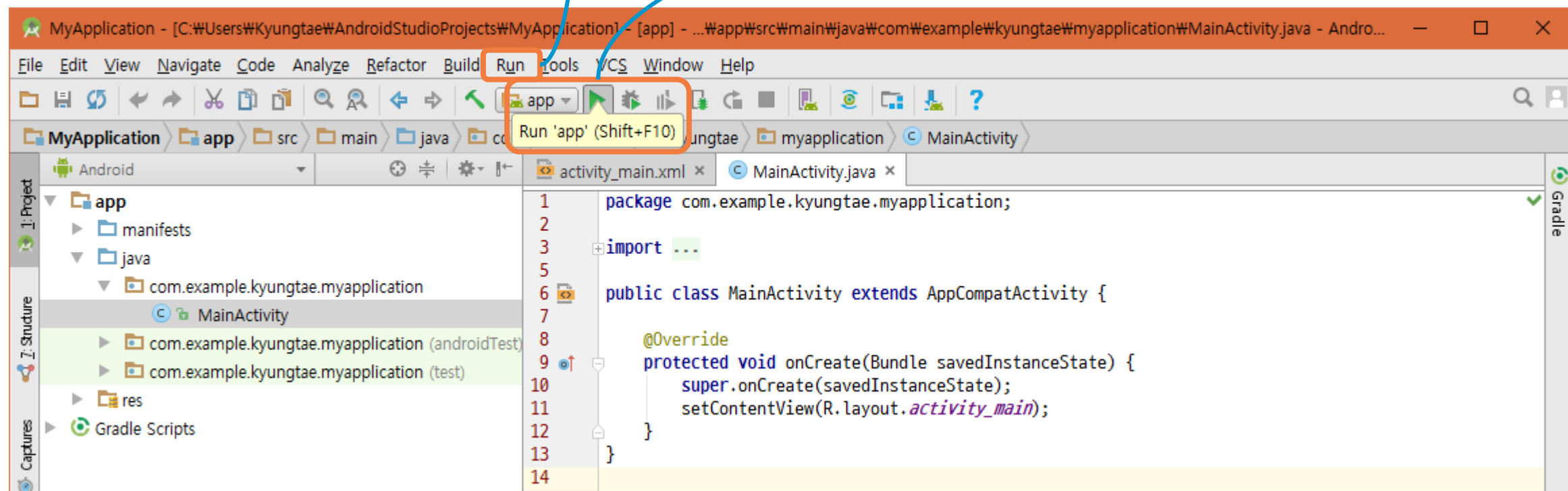
Step 3. 프로젝트 실행

63



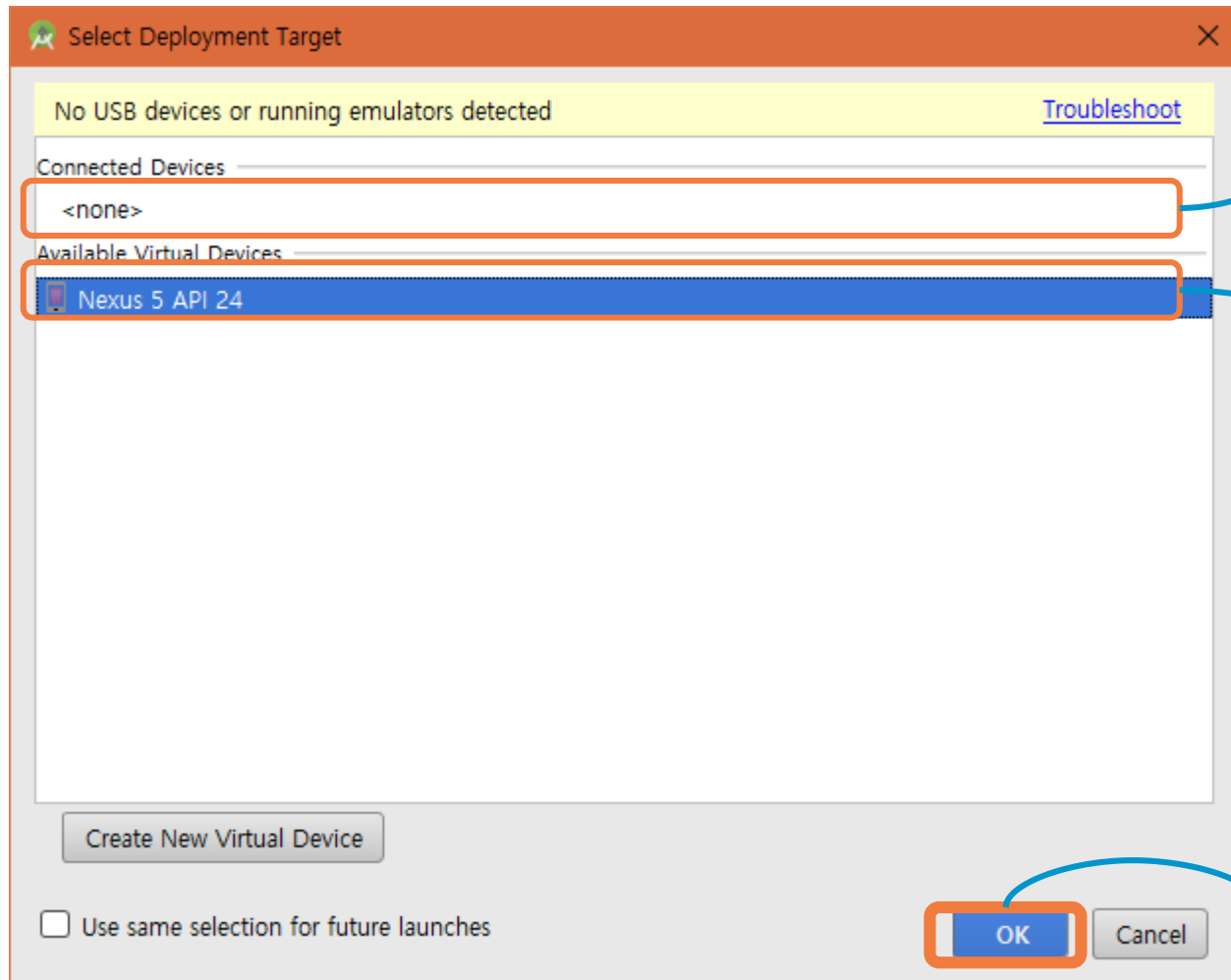
Run → Run 'app' 메뉴 클릭

앱 실행 아이콘 클릭



• AVD 장비 선택하기

64




데이터 케이블로 연결된
스마트폰

AVD

스마트폰 또는 AVD를 선택하고
'OK' 버튼을 클릭

• 실행 결과

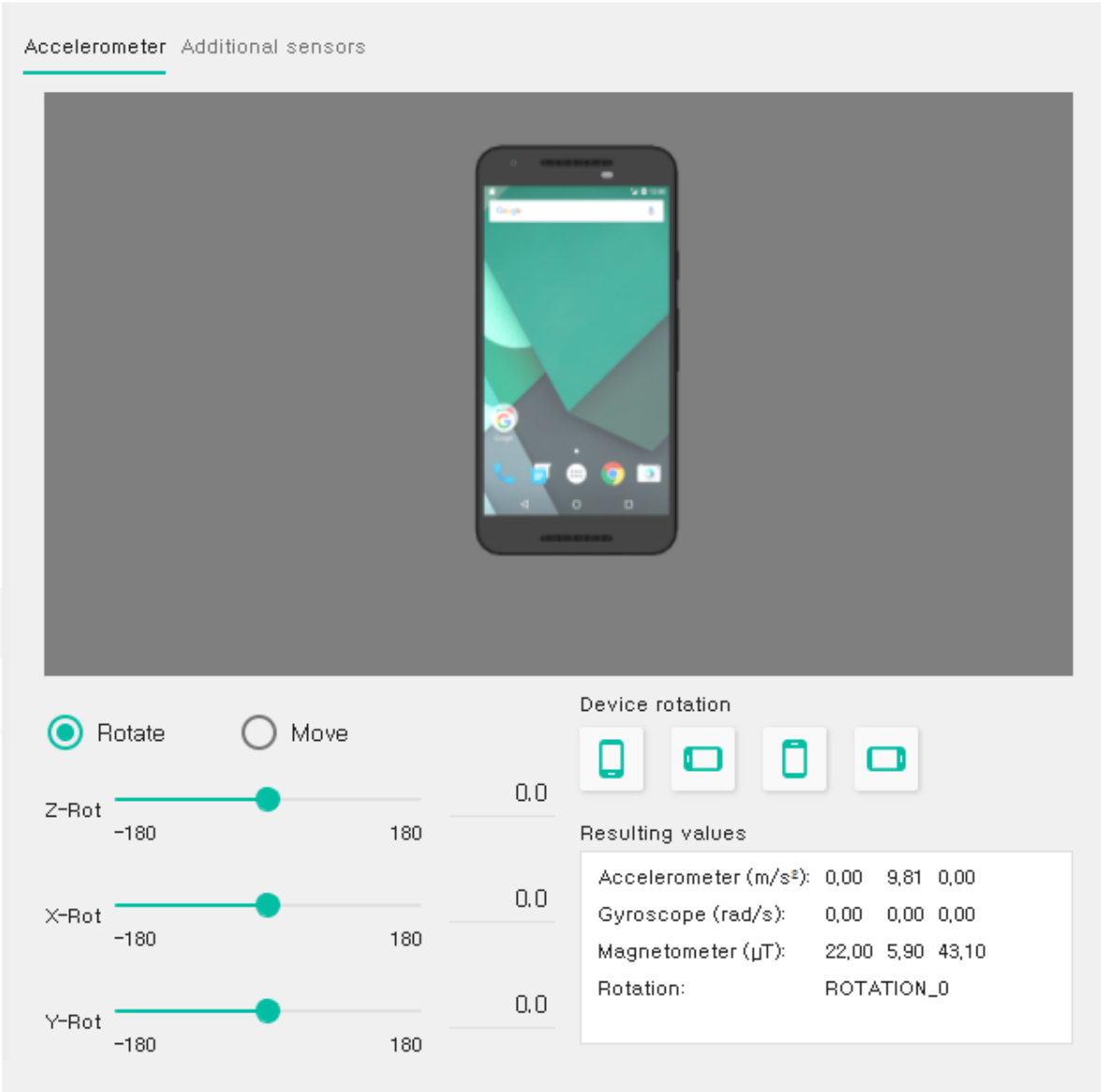
Android Emulator - Nexus_5_API_24:5554



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



☒ Rotate ☐ Move

Z-Rot -180 180 0.0

X-Rot -180 180 0.0

Y-Rot -180 180 0.0

Device rotation

Resulting values

Accelerometer (m/s ²):	0,00	9,81	0,00
Gyroscope (rad/s):	0,00	0,00	0,00
Magnetometer (μT):	22,00	5,90	43,10
Rotation:	ROTATION_0		

O utputs



Q & A

uestion
nswer

67

