



14. 위치 센서

Contents title

14.1 위치 센서의 이해

14.1.1 위치 센서를 이용한 앱의 예

14.1.2 위치 센서 원리

14.2 위치 센서값 읽기: Position Sensor(위치 센서)

14.2.1 프로젝트 개요

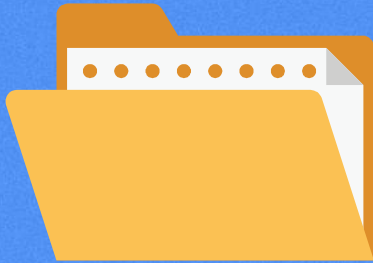
14.2.2 프로젝트 개발

14.3 근접 센서를 이용한 이미지 떨림: Shaking 프로젝트(나를 그냥 뒤)

14.3.1 프로젝트 개요

14.3.2 프로젝트 개발



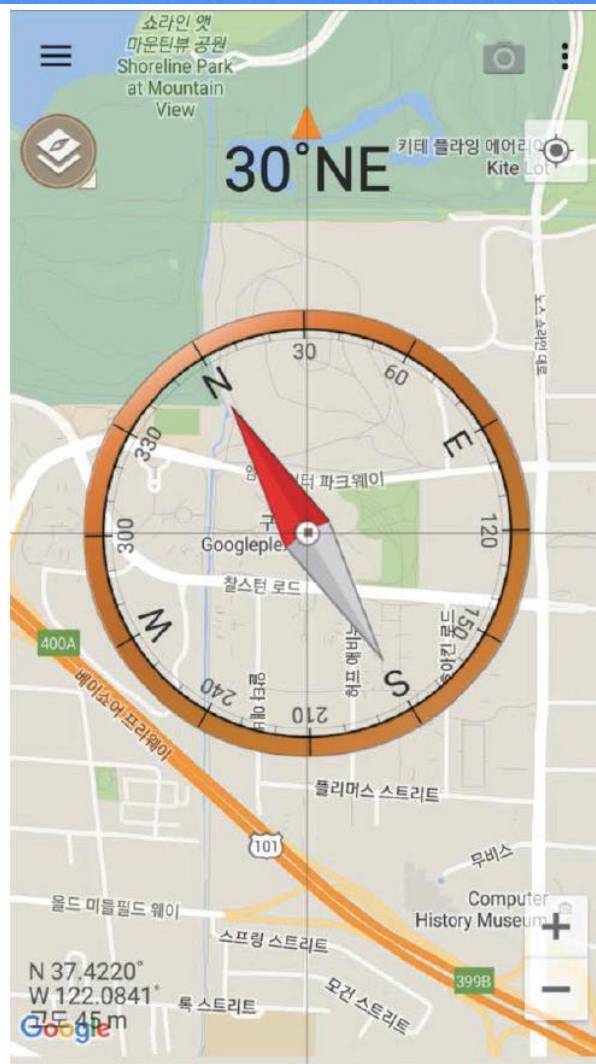


14.1 위치 센서의 이해





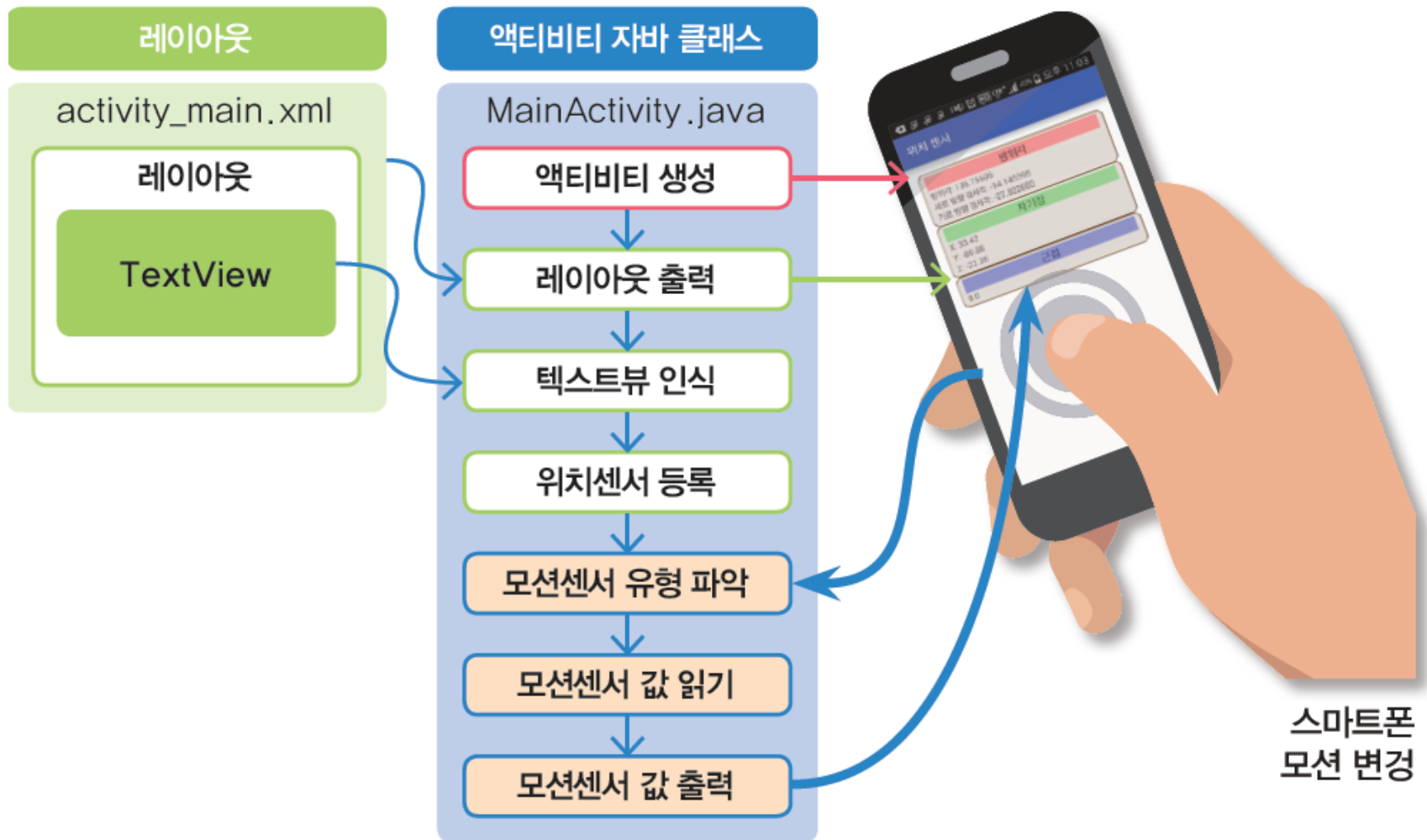
(a) 카메라 화면과 방향



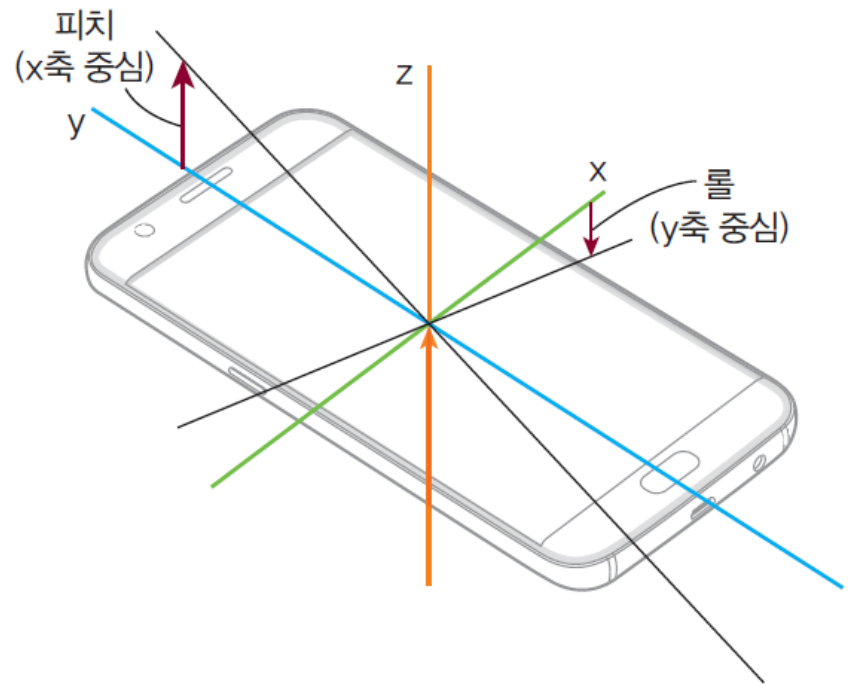
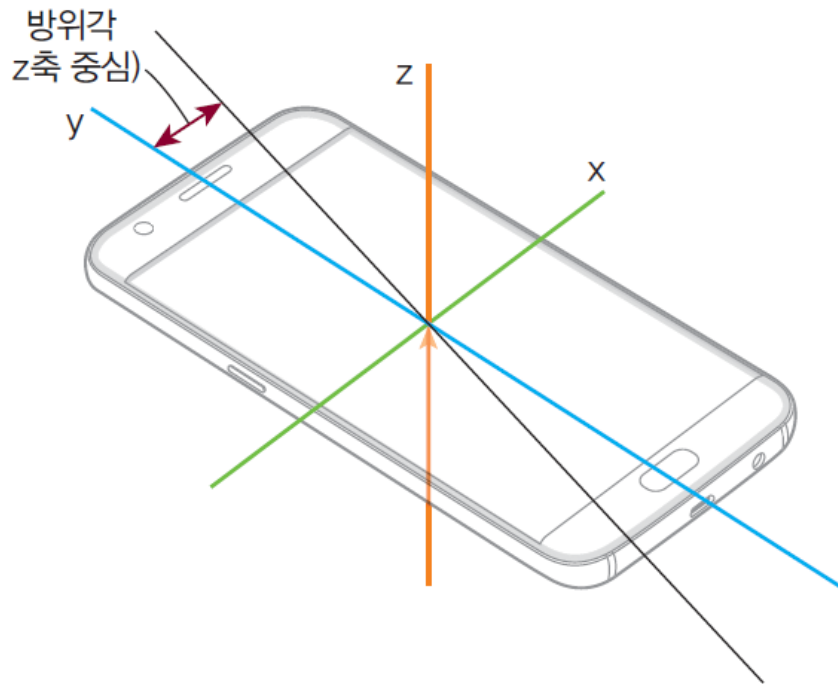
(b) 구글맵 상의 위치와 방향

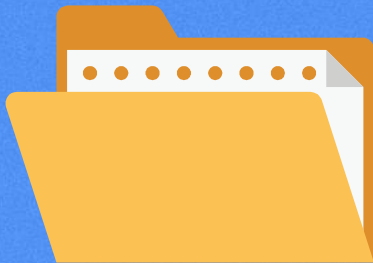
- Smart Compass 앱

14.1.2 위치 센서 원리



위치 센서에는 방향 센서와 자력계





14.2 위치 센서값 읽기: Position Sensor(위치센서)

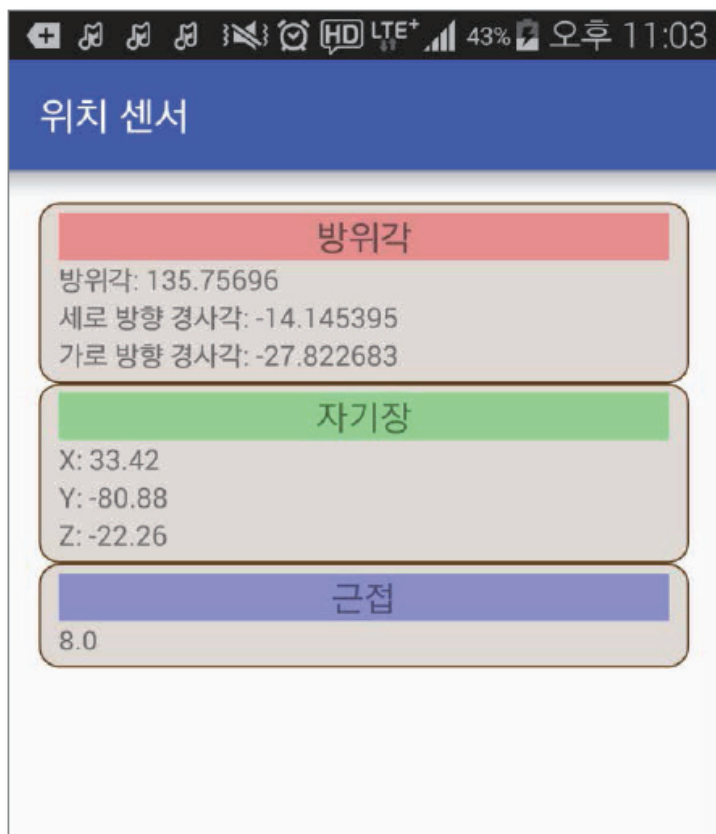


14.2.1 프로젝트 개요

프로젝트 개요: 위치 센서의 측정

Application Name: Position Sensor

어플리케이션 라벨: 위치 센서



방위각, 자기장, 근접 센서의 측정 값을 출력함. 근접 센서의 경우, cm 단위의 거리로 측정되지만, 디바이스에 따라 가까운 거리인지 아니면 먼 거리인지만 구분하여 반환함(가까운 거리이면 0)

STEP 1

프로젝트 생성

절차	내용
① 프로젝트 시작	메뉴에서 'File → New Project' 클릭
② 프로젝트 구성	Application Name: Position Sensor Company Domain: yschang.example.com
③ 제품형태	Phone and Tablet
④ 액티비티 유형	Empty Activity
⑤ 파일 옵션	디폴트 값으로 설정

STEP 2

파일 편집

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

■ 수정 ■ 추가

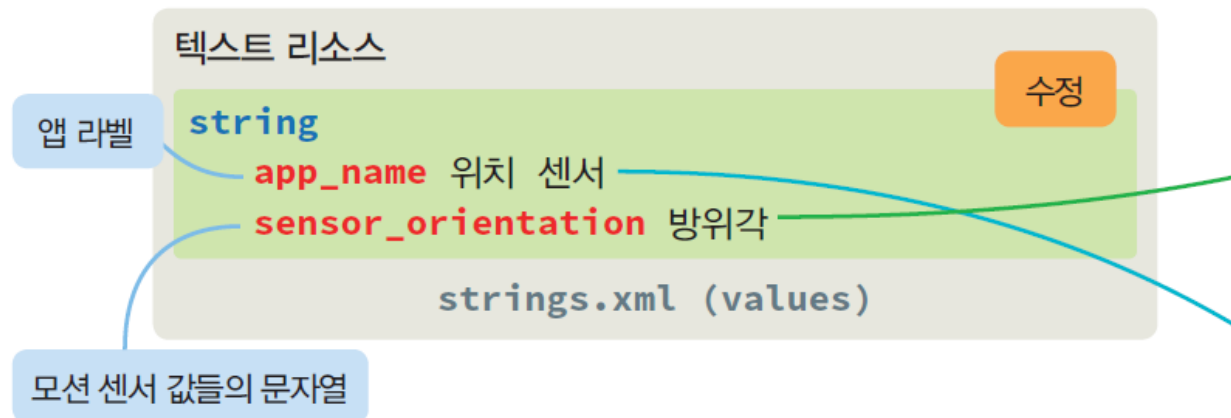
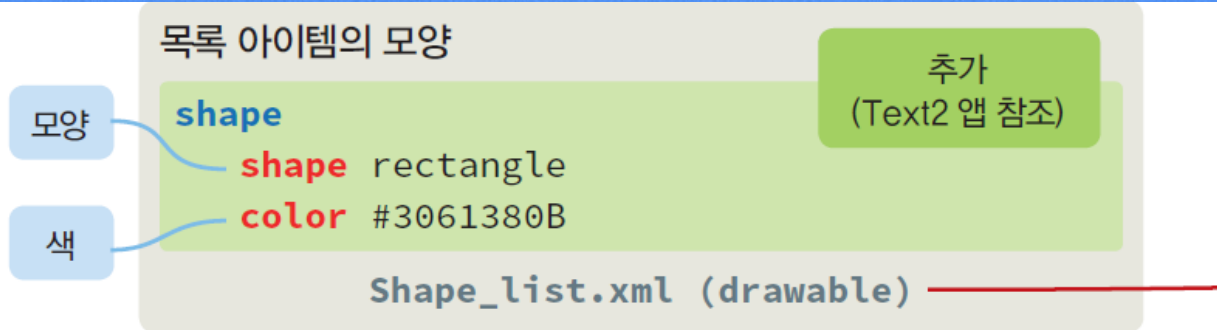
● 파일 간의 연관관계

strings.xml에는 초기치로 설정되어 있는 어플리케이션 라벨을 '위치 센서'로 수정하고, 화면에 출력할 문자열들(sensor_orientation 등)을 추가한다.

목록의 아이템에 대한 출력모양을 정의하는 **shape_list**를 작성한다.

activity_main.xml에는 위치 센서 값들을 출력할 텍스트뷰를 배치한다.

MainActivity.java에는 스마트폰이 움직일 때 인지하는 방위 센서 값들을 출력하는 메소드들(onSensorChanged() 등)을 구현한다.



화면 레이아웃

```
LinearLayout
  LinearLayout
    background @drawable/shape_list
    TextView
      text @string/sensor_orientation
    TextView
      id @+id/azimuth
    ...
```

activity_main.xml (layout)

수정

텍스트뷰 인식

센서의 움직임이
감지될 때

액티비티 제어

```
onCreate()
  super.onCreate()
  setContentView(R.layout.activity_main)
  azimuth = findViewById(R.id.azimuth)
onSensorChanged()
  case Sensor.TYPE_ORIENTATION
    azimuth.setText(event.values[0])
    ...
```

MainActivity.java (java)

방위각 센서 값일때

방위각 센서의 x 방향 값

텍스트뷰에
방위각 센서 값 출력

어플리케이션 구성
액티비티의 자바 클래스

어플리케이션 기본 정보

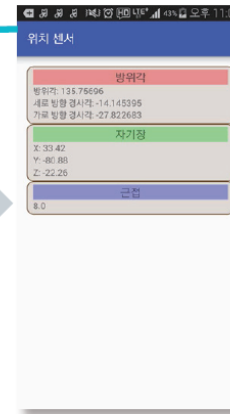
```
application
  icon @mipmap/ic_launcher
  label @string/app_name
  theme @style/AppTheme
  activity
    name MainActivity
```

AndroidManifest.xml (manifest)



사용자

움직임



스마트폰

컴파일/빌더

컴파일/빌더 정보

```
build gradle(Project)
build gradle(Module app)
gradle properties
settings gradle
local properties
```

(Gradle Scripts)

● 편집

1 텍스트 자원의 편집

소스 | strings.xml

```
01 <resources>
02     <string name="app_name"> 위치 센서</string>
03
04     <string name="sensor_orientation">방위각</string>
05     <string name="sensor_magnetic_field">자기장</string>
06     <string name="sensor_proximity">근접</string>
07 </resources>
```

app_name의 데이터를 '위치 센서'로 수정

A

센서 값들의 제목

2 출력모양 편집

모듈	폴더	소스 파일	편집 내용
res	drawable	shape_list.xml	• Text2 App과 동일(6.3절) → B

3 화면 설계

소스 | activity_main.xml → C

```
11 <LinearLayout
12     android:layout_width="match_parent"
13     android:layout_height="wrap_content"
14     android:orientation="vertical"
15     android:background="@drawable/shake_list">
16
17     <TextView
18         android:text="@string/sensor_orientation"
19         android:layout_width="match_parent"
20         android:layout_height="wrap_content"
21         android:textSize="18sp"
22         android:gravity="center"
23         android:background="#55ff0000"/>
24
25     <TextView
26         android:id="@+id/azimuth"
27         android:layout_width="match_parent"
28         android:layout_height="wrap_content" />
29
30     <TextView
31         android:id="@+id/pitch"
32         android:layout_width="match_parent"
33         android:layout_height="wrap_content" />
34
35     <TextView
36         android:id="@+id/roll"
37         android:layout_width="match_parent"
38         android:layout_height="wrap_content" />
39 </LinearLayout>
```

방위각 센서 값 출력 영역

방위각 센서 제목 출력을 위한 텍스트뷰

방위각 센서 값 출력

방위각 출력을 위한 텍스트뷰

세로 방향 기울기 출력을 위한 텍스트뷰

가로 방향 기울기 출력을 위한 텍스트뷰

4 액티비티 제어

소스 | MainActivity.java

```
28  @Override
29  protected void onCreate(Bundle savedInstanceState) {
30      super.onCreate(savedInstanceState);
31      setContentView(R.layout.activity_main);
32
33      azimuth = (TextView) findViewById(R.id.azimuth);
34      pitch = (TextView) findViewById(R.id.pitch);
35      roll = (TextView) findViewById(R.id.roll);
36
37      x_magnetic_field = (TextView) findViewById(R.id.x_magnetic_field);
38      y_magnetic_field = (TextView) findViewById(R.id.y_magnetic_field);
39      z_magnetic_field = (TextView) findViewById(R.id.z_magnetic_field);
40
41      proximity = (TextView) findViewById(R.id.proximity);
42  }
```

방위각과 기울기 출력을
위한 텍스트뷰 인식

자기장 센서 값 출력을
위한 텍스트뷰 인식

근접 센서 값 출력을
위한 텍스트뷰 인식

디바이스 센서 접근을 위한 객체 생성

```
43      sm = (SensorManager) getSystemService(SENSOR_SERVICE);
44
45      sensor_orientation = sm.getDefaultSensor(Sensor.TYPE_ORIENTATION);
46      sensor_magnetic_field = sm.getDefaultSensor
47                               (Sensor.TYPE_MAGNETIC_FIELD);
48      sensor_proximity = sm.getDefaultSensor(Sensor.TYPE_PROXIMITY);
49  }
```

방위각, 자기장, 근접 센서 객체 생성

방위각, 자기장, 근접 센서 객체 생성

액티비티가 일시정지(pause) 상태에서
복귀될 때 호출(센서 리스너 등록)

`@Override`

```
protected void onResume() {  
    super.onResume();  
  
    sm.registerListener(this, sensor_orientation,  
                        SensorManager.SENSOR_DELAY_NORMAL);  
    sm.registerListener(this, sensor_magnetic_field,  
                        SensorManager.SENSOR_DELAY_NORMAL);  
    sm.registerListener(this, sensor_proximity,  
                        SensorManager.SENSOR_DELAY_NORMAL);  
}
```

화면에 표시되는 상태에서 사용자와
상호작용하지 않을 때(센서 리스너 해제)

`@Override`

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

등록된 센서의 정확도가 변할 때 호출

`@Override`

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


69

70 @Override

센서 값이 변할 때 호출

71 public void onSensorChanged(SensorEvent event) {

72

센서 값 유형에 따른 실행

73 switch(event.sensor.getType()) {

74 case Sensor.**TYPE_ORIENTATION**:

75 azimuth.setText("방위각: " + event.values[0]);

76 pitch.setText("세로 방향 경사각: " + event.values[1]);

77 roll.setText("가로 방향 경사각: " + event.values[2]);

78 break;

센서 값 유형이
방위각일 때,
방위각 출력

79

80 case Sensor.**TYPE_MAGNETIC_FIELD**:

81 x_magnetic_field.setText("X: " + event.values[0]);

82 y_magnetic_field.setText("Y: " + event.values[1]);

83 z_magnetic_field.setText("Z: " + event.values[2]);

84 break;

센서 값 유형이
자기장일 때,
자기장 값 출력

85

86 case Sensor.**TYPE_PROXIMITY**:

87 proximity.setText("" + event.values[0]);

88 break;

센서 값 유형이
근접 센서일 때,
근접 센서 값 출력

89 }

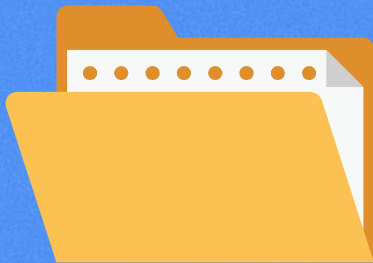
90 }

91 }

STEP 3

프로젝트 실행

절차	내용
실행메뉴 선택	'Run' 메뉴에서 'Run app' 클릭(또는 'Run app' 아이콘 클릭)
디바이스 선택	스마트폰 디바이스를 선택하고 'OK' 버튼 클릭



14.3 근접 센서를 이용한 이미지 떨림: Shaking 프 로젝트 (나를 그냥 뒤)



14.3.1

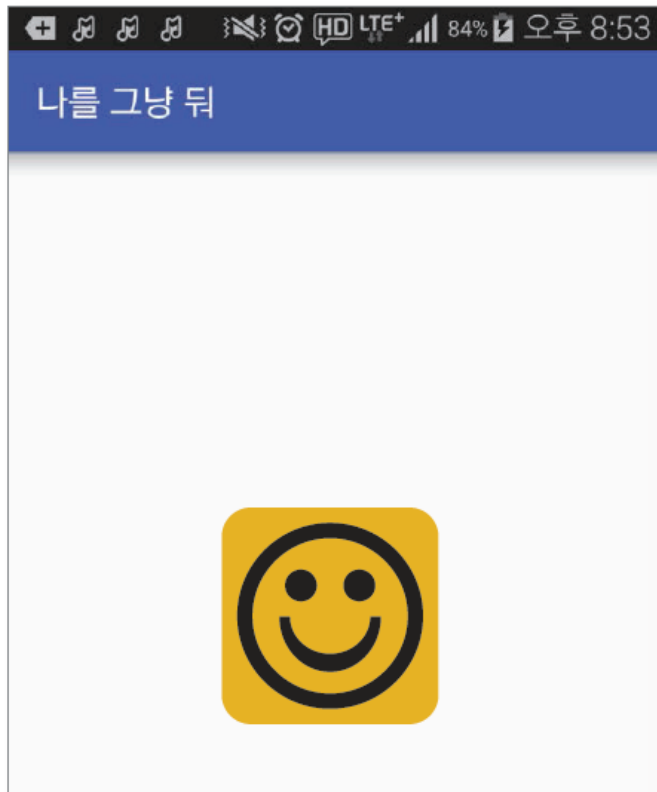
프로젝트 개요

프로젝트 개요: 스마트폰에 가까이 가면 진동과 아이콘이 떨림

Application Name: Shaking

어플리케이션 라벨: 나를 그냥 뒀

초기 화면



근접 센서에 가까이 갈 때



14.3.2 프로젝트 개발

STEP 1 프로젝트 생성

절차	내용
① 프로젝트 시작	메뉴에서 'File → New Project' 클릭
② 프로젝트 구성	Application Name: Shaking Company Domain: yschang.example.com
③ 제품형태	Phone and Tablet
④ 액티비티 유형	Empty Activity
⑤ 파일 옵션	디폴트 값으로 설정

STEP 2

파일 편집

모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	• 진동 허용
java	com.example. yschang.shaking	MainActivity.java	<ul style="list-style-type: none"> • 센서 등록 • 센서 값 변경 확인 • 근접이면 이미지 변경 및 진동
res	anim	shaking.xml	• 아이콘 이미지의 진동 애니메이션
	drawable	smile.png	• 아이콘 이미지
		angry.png	• 아이콘 이미지
	layout	activity_main.xml	• 이미지의 화면 중앙 배치
	mipmap	ic_launcher.png	
	values	dimens.xml	
		strings.xml	• 어플리케이션 라벨 수정
		styles.xml	

● 파일 간의 연관관계

strings.xml에는 초기치로 설정되어 있는 어플리케이션 라벨을 '나를 그냥 뒤'로 수정한다.

activity_main.xml에는 아이콘 이미지를 출력할 이미지를 배치한다.

MainActivity.java에는 손을 가까이 대면 아이콘이 바뀌면서 좌우로 흔들고 진동이 일어나도록 한다.

AndroidManifest.xml에는 앱이 진동 기능을 사용할 수 있도록 허용한다.

이미지 리소스

추가



smile.png

이미지뷰의
입력 소스



angry.png

애니메이션
리소스

(drawable)

목록 아이템의 모양

추가

set

translate

좌우로 진동

shaking.xml (drawable)

애니메이션
리소스

텍스트 리소스

수정

string

app_name 나를 그냥 뒤

앱 라벨

strings.xml (values)

화면 레이아웃

```
RelativeLayout  
ImageView
```

```
id @+id/img
```

```
src @drawable/smile
```

```
...
```

activity_main.xml (layout)

수정

이미지뷰 인식

액티비티 제어

```
onCreate()
```

```
super.onCreate()
```

```
setContentView(R.layout.activity_main)
```

```
img = findViewById(R.id.img)
```

```
onAccuracyChanged()
```

```
...
```

```
ani = AnimationUtils.loadAnimation
```

```
(this, R.anim.shaking)
```

```
img.setImageResource(R.drawable.angry);
```

```
img.startAnimation(ani);
```

```
mVibe.vibrate(1000);
```

MainActivity.java (java)

수정

애니메이션 설정

이미지에
애니메이션 설정

진동

어플리케이션 구성
액티비티의 자바 클래스

어플리케이션 기본 정보

```
uses-permission
```

```
name android.permission.VIBRATE
```

```
application
```

```
icon @mipmap/ic_launcher
```

```
label @string/app_name
```

```
theme @style/AppTheme
```

```
activity
```

```
name MainActivity
```

AndroidManifest.xml (manifest)

수정

진동 허용 설정

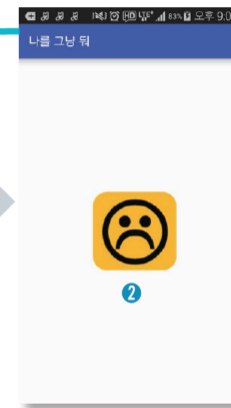


사용자

1 손을 가까이 댄

센서의 움직임이
감지될 때

컴파일/빌더



스마트폰



컴파일/빌더 정보

```
build gradle(Project)  
build gradle(Module app)  
gradle properties  
settings gradle  
local properties
```

(Gradle Scripts)

● 편집

1 이미지 파일의 복사

모듈	폴더	소스 파일	이미지
res	drawable	smile.png	 → A
		angry.png	 → B

2 텍스트 자원의 편집

소스 | strings.xml

```
01 <resources>
02     <string name="app_name">나를 그냥 뒤</string>
03 </resources>
```

app_name의 데이터를
'나를 그냥 뒤'로 수정

③ 화면 설계

소스 | activity_main.xml → C

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
03     xmlns:tools="http://schemas.android.com/tools"
04     android:layout_width="match_parent"
05     android:layout_height="match_parent"
06     android:paddingBottom="@dimen/activity_vertical_margin"
07     android:paddingLeft="@dimen/activity_horizontal_margin"
08     android:paddingRight="@dimen/activity_horizontal_margin"
09     android:paddingTop="@dimen/activity_vertical_margin"
10     tools:context="com.example.shaking.MainActivity">
11
12     <ImageView
13         android:id="@+id/img"
14         android:layout_width="match_parent"
15         android:layout_height="wrap_content"
16         android:layout_centerInParent="true"
17         android:src="@drawable/smile" />
18
19 </RelativeLayout>
```

D

A

근접 여부에 따른
이미지 출력을 위한 이미지뷰

4 애니메이션 편집

소스 | shaking.xml → E

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <set
03     xmlns:android="http://schemas.android.com/apk/res/android">
04     <translate
05         android:fromXDelta="-2%"
06         android:toXDelta="2%"
07         android:duration="100"
08         android:repeatCount="10" />
09 </set>
```

이동 위치 변화: x 방향(좌우)으로
이미지 너비의 2% 크기만큼을
100밀리초(0.1초) 동안 10회 반복

5 액티비티 제어

소스 | MainActivity.java

```
24  @Override
25  protected void onCreate(Bundle savedInstanceState) {
26      super.onCreate(savedInstanceState);
27      setContentView(R.layout.activity_main);
28
29      img = (ImageView) findViewById(R.id.img);
30
31      sm = (SensorManager) getSystemService(SENSOR_SERVICE);
32
33      sensor_proximity = sm.getDefaultSensor(Sensor.TYPE_PROXIMITY);
34
35      mVibe = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);
36  }
37
38  @Override
39  protected void onResume() {
40      super.onResume();
41
42      sm.registerListener(this, sensor_proximity, SensorManager.
43                          SENSOR_DELAY_NORMAL);
44  }
45  @Override
```

아이콘 이미지가
출력될 이미지뷰 인식

디바이스 센서
접근을 위한 객체 생성

근접 가속도 센서 측정을
위한 센서 객체 생성

시스템으로부터
진동을 위한 객체 생성

액티비티가 일시정지(pause) 상태에서 복귀될 때 호출(센서 리스너 등록)


```

46 protected void onPause() {
47     super.onPause();
48
49     sm.unregisterListener(this);
50 }

```

화면에 표시되는 상태에서
사용자와 상호작용하지
않을 때(센서 리스너 해제)

```

51
52 @Override

```

등록된 센서의 정확도가 변할 때 호출

```

53 public void onAccuracyChanged(Sensor sensor, int accuracy) {
54 }

```

```

55
56 @Override

```

센서 값이 변할 때 호출

```

57 public void onSensorChanged(SensorEvent event) {

```

센서 값 유형이 근접 센서일 때

```

58
59     if(event.sensor.getType() == Sensor.TYPE_PROXIMITY) {

```

```

60         if(event.values[0] <= 4) {

```

근접 센서 값이 4 이하일 때
(가까이 가면 0, 멀어지면 8의 값)

```

61         Animation ani = AnimationUtils.loadAnimation

```

아이콘 이미지를 angry.png로 변경

```

        (this, R.anim.shaking);

```

```

62         img.setImageResource(R.drawable.angry);

```

```

63         img.startAnimation(ani);

```

```

64         mVibe.vibrate(1000);

```

1초간 진동

아이콘 이미지에 대한
애니메이션 시작

anim 폴더에 있는
shaking.xml을
애니메이션으로 하는
애니메이션 객체 생성

```

65     } else

```

```

66         img.setImageResource(R.drawable.smile);

```

아이콘 이미지를 smile.png로 변경

```

67     }

```

```

68 }

```

```

69 }

```

3 환경 설정

소스 | AndroidManifest.xml

```
01 <?xml version="1.0" encoding="utf-8"?>
02 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
03     package="com.example.yschang.shaking">
04
05     <uses-permission android:name="android.permission.VIBRATE" />
06
07     <application
08         android:allowBackup="true"
09         android:icon="@mipmap/ic_launcher"
10         android:label="@string/app_name"
11         android:supportsRtl="true"
12         android:theme="@style/AppTheme">
13         <activity android:name=".MainActivity">
14             <intent-filter>
15                 <action android:name="android.intent.action.MAIN" />
16
17                 <category android:name="android.intent.category.LAUNCHER" />
18             </intent-filter>
19         </activity>
20     </application>
21
22 </manifest>
```

앱이 진동을 사용할 수 있도록 허가

STEP 3

프로젝트 실행

절차	내용
실행메뉴 선택	'Run' 메뉴에서 'Run app' 클릭(또는 'Run app' 아이콘 클릭)
디바이스 선택	스마트폰 디바이스를 선택하고 'OK' 버튼 클릭



Thank you