



출처: maxpixel.net

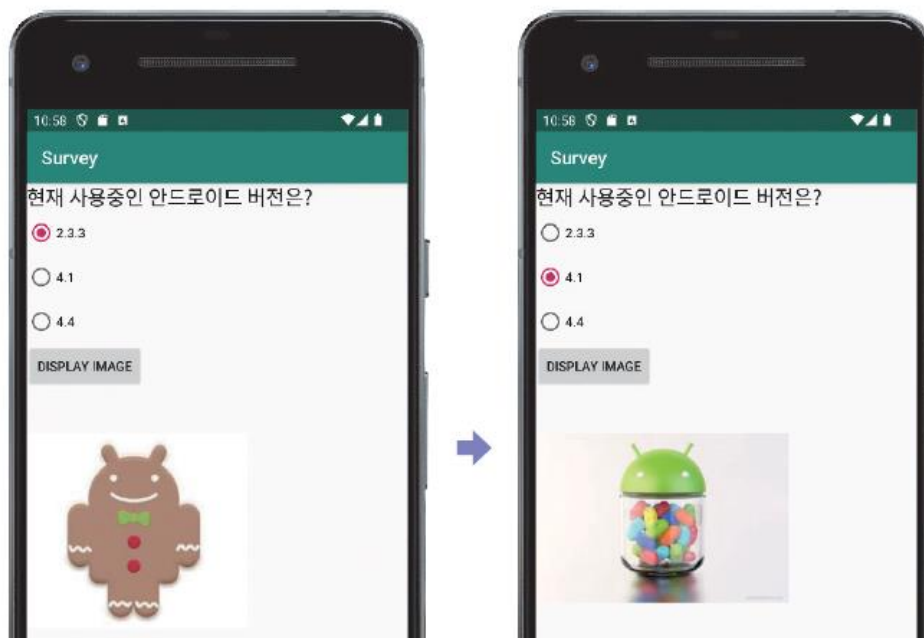
# CHAP 5. 고급 위젯과 이벤트 처리하기



# 고급 위젯과 이벤트 처리하기



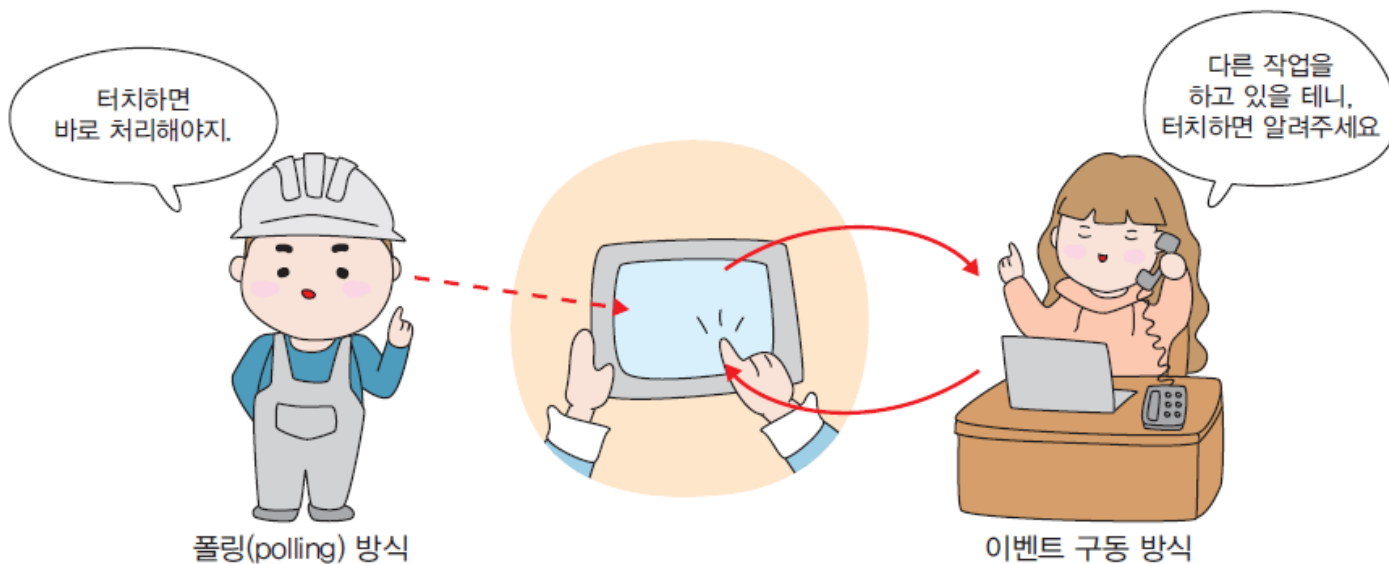
- 여론 조사 앱 만들기





# 안드로이드에서 이벤트 처리

- 이벤트 구동(event-driven)방식에서는 애플리케이션이 다른 작업을 하고 있다가 사용자의 입력으로 인하여 이벤트가 발생하면, 이 이벤트를 처리하면 된다.





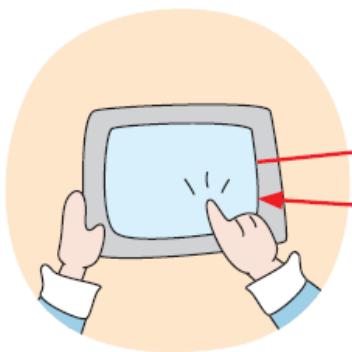
# 이벤트를 처리하는 3가지의 방법

- XML 파일에 이벤트 처리 메소드를 등록하는 방법
  - 위젯의 **onClick** 속성에 자바 메소드를 등록하는 방법이다.
- 이벤트를 처리하는 객체를 생성하여 이벤트를 처리하는 방법
  - 이벤트를 처리하는 객체를 별도로 생성하여 위젯에 등록한다. 이벤트를 처리하는 가장 일반적인 방법이다.
- 뷰 클래스의 이벤트 처리 메소드를 재정의하는 방법
  - 뷰 클래스의 이벤트 처리 메소드를 재정의한다. 커스텀 뷰(Custom View)를 작성하는 경우에만 사용할 수 있는 방법이다.



# 이벤트 처리하기(이벤트 처리 객체 사용)

- 이 방법에서는 이벤트를 처리하는 객체를 생성하여서 위젯에 등록한다. 이벤트를 처리하는 콜백 메소드가 정의된 인터페이스를 이벤트 리스너(event listener)라고 부른다.



```
public class MyClass {  
    ...  
    button.setOnClickListener(new View.OnClickListener() {  
        public void onClick(View v) {  
            ...  
        }  
    });  
}
```

● 그림 5-1  
이벤트 처리 객체  
사용



# 이벤트 리스너란?

- 이벤트 리스너는 특정 이벤트(예: 버튼 클릭, 터치 제스처, 키 입력 등)를 감지하고 해당 이벤트가 발생했을 때 실행되어야 하는 코드를 정의하는 인터페이스이다.





# 어떤 이벤트 리스너들이 있는가?

리스너	콜백 메소드	설명
<code>View.OnClickListener</code>	<code>onClick()</code>	사용자가 어떤 항목을 터치하거나 내비게이션 키나 트랙볼로 항목으로 이동한 후에 엔터 키를 눌러서 선택하면 호출된다.
<code>View. OnLongClickListener</code>	<code>onLongClick()</code>	사용자가 항목을 터치하여서 일정 시간 동안 그대로 누르고 있으면 발생한다.
<code>View. OnFocusChangeListener</code>	<code>onFocusChange()</code>	사용자가 하나의 항목에서 다른 항목으로 포커스를 이동할 때 호출된다.
<code>View.OnKeyListener</code>	<code>onKey()</code>	포커스를 가지고 있는 항목 위에서 키를 눌렀다가 놓았을 때 호출된다.
<code>View.OnTouchListener</code>	<code>onTouch()</code>	사용자의 터치 동작(예: 터치 다운, 이동, 터치 업)에 따라 호출된다.
<code>Adapter.OnItemSelectedListener</code>	<code>onItemSelected ()</code>	사용자가 항목을 선택할 때 호출된다.





# 리스너 객체를 생성하는 방법

- 리스너 클래스를 내부 클래스로 정의한다. -> 생략
- 리스너 인터페이스를 액티비티 클래스에 구현한다. -> 생략
- 리스너 클래스를 익명 클래스로 정의한다. -> 이것이 가장 많이 사용되는 방법이다.
- 람다식을 이용한다. -> 많이 사용되는 방법이다.



# 익명 클래스로 처리하는 방법

- 익명 클래스(**anonymous class**)는 클래스 몸체는 정의되지만 이름이 없는 클래스이다. 익명 클래스는 클래스를 정의하면서 동시에 객체를 생성하게 된다.
- 익명 클래스는 이름이 없으므로 한 번만 사용이 가능하다.

정상 클래스	익명 클래스
<pre>class MyClass implements     OnClickListener { ... } obj = new MyClass();</pre>	<pre>obj = new OnClickListener() { .... };</pre>



# 이름 클래스로 처리하는 방법

- 예를 들어서 버튼이 클릭되는 경우에 발생하는 클릭 이벤트를 처리하려면 **OnClickListener** 인터페이스를 구현하는 객체를 생성하고 **setOnClickListener()**를 호출하여서 이것을 버튼에 설정하면 된다.

```
final Button button = (Button) findViewById(R.id.button_send);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        // 버튼이 클릭되면 여기서 어떤 작업을 한다.
    }
});
```



# 이름 클래스로 처리하는 방법

버튼 참조  
변수 선언

```
Button b1;
```

버튼 위젯  
찾기

```
b1 = (Button) findViewById(R.id.button1)
```

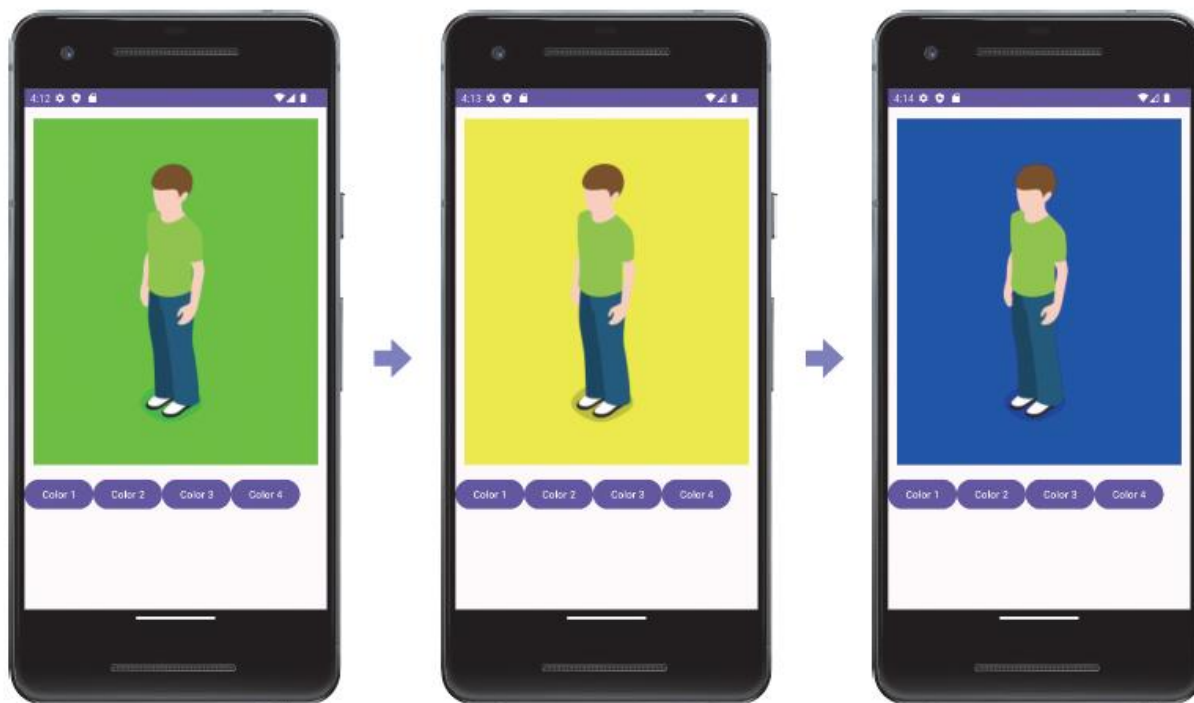
익명 클래스를  
버튼에 붙이기

```
b1.setOnClickListener(new View.OnClickListener() {  
    public void onClick(View v) {  
        // 여기에 작업을 기술한다.  
    }  
});
```



# 예제: 이름 클래스로 버튼 이벤트 처리

- 화면에 이미지가 있고 하단에 버튼이 4개 있다. 각 버튼을 누르면 이미지의 배경색이 변경되는 앱을 작성해보자.





# 예제: 이름 클래스로 버튼 이벤트 처리

activity\_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android">
    <ImageView
        android:id="@+id/clothingImageView"
        app:srcCompat="@drawable/img" />

    <!-- 4개의 색상 버튼 -->
    <Button
        android:id="@+id/colorButton1"
        android:text="Color 1" />
    <Button
        android:id="@+id/colorButton2"
        android:text="Color 2" />
    <Button
        android:id="@+id/colorButton3"
        android:text="Color 3" />
    <Button
        android:id="@+id/colorButton4"
        android:text="Color 4" />

</RelativeLayout>
```



# 예제: 익명 클래스로 버튼 이벤트 처리

MainActivity.java

```
public class MainActivity extends AppCompatActivity {  
    private ImageView clothingImageView;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        clothingImageView = findViewById(R.id.clothingImageView);  
  
        Button colorButton1 = findViewById(R.id.colorButton1);  
        Button colorButton2 = findViewById(R.id.colorButton2);  
        Button colorButton3 = findViewById(R.id.colorButton3);  
        Button colorButton4 = findViewById(R.id.colorButton4);  
  
        colorButton1.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                changeClothingColor(Color.RED);  
            }  
        });  
  
        colorButton2.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {
```

클릭 리스너를 구현하는 익명 클래스를 정의하고, 객체를 생성하여 버튼에 등록한다.  
한 곳에서 이벤트 처리와 관련된 모든 코드가 작성되는 장점이 있다.



# 예제: 기본 클래스로 버튼 이벤트 처리

```
        changeClothingColor(Color.BLUE);
    }
});

colorButton3.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        changeClothingColor(Color.GREEN);
    }
});

colorButton4.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        changeClothingColor(Color.YELLOW);
    }
});
}

private void changeClothingColor(int color) {
    clothingImageView.setBackgroundColor(color);
}
}
```





# 람다식을 이용하는 방법

- 단 이것은 이벤트 리스너가 하나의 콜백 메소드만을 가지고 있는 경우에만 가능하다.

```
button.setOnClickListener(new View.OnClickListener(){  
    public void onClick(View v) {v++;}  
});
```

익명 클래스

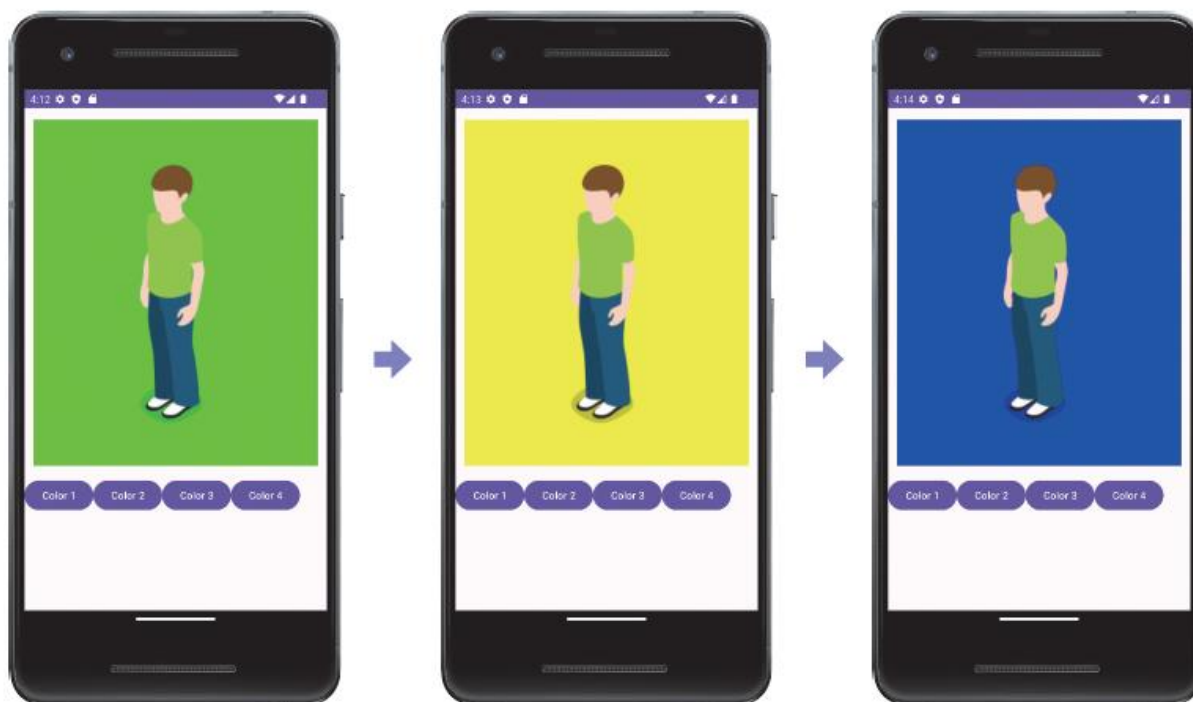
```
button.setOnClickListener(e->{v++;});
```

람다식



## 예제: 람다식으로 버튼 이벤트 처리

- 화면에 이미지가 있고 하단에 버튼이 4개 있다. 각 버튼을 누르면 이미지의 배경색이 변경되는 앱을 작성해보자.





# 예제: 람다식으로 버튼 이벤트 처리

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    private ImageView clothingImageView;

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

        clothingImageView = findViewById(R.id.clothingImageView);

        Button colorButton1 = findViewById(R.id.colorButton1);
        Button colorButton2 = findViewById(R.id.colorButton2);
        Button colorButton3 = findViewById(R.id.colorButton3);
        Button colorButton4 = findViewById(R.id.colorButton4);
```

람다식으로 이벤트를 처리한다.



## 예제: 람다식으로 버튼 이벤트 처리

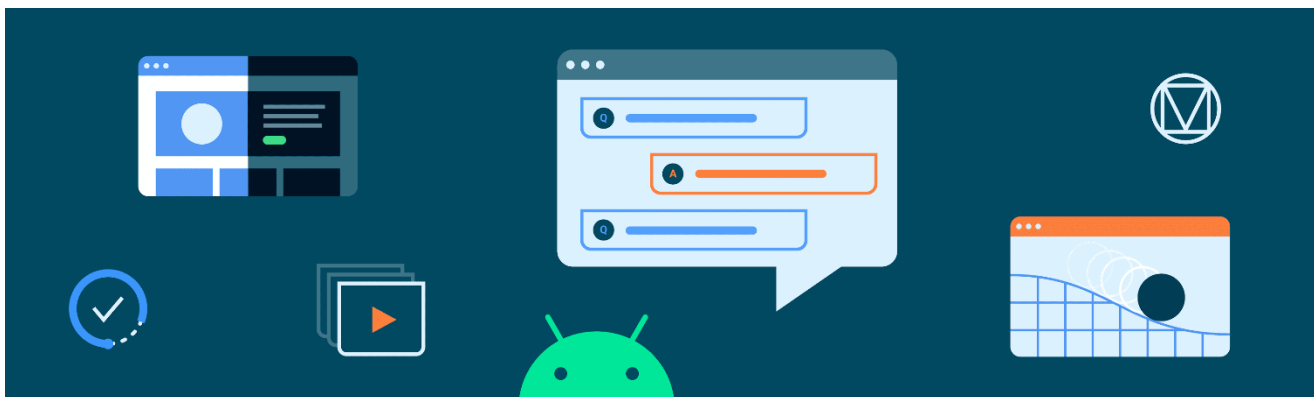
```
colorButton1.setOnClickListener(view -> changeClothingColor(Color.RED));  
colorButton2.setOnClickListener(view -> changeClothingColor(Color.BLUE));  
colorButton3.setOnClickListener(view -> changeClothingColor(Color.GREEN));  
colorButton4.setOnClickListener(view -> changeClothingColor(Color.YELLOW));  
}
```

```
private void changeClothingColor(int color) {  
    clothingImageView.setBackgroundColor(color);  
}  
}
```



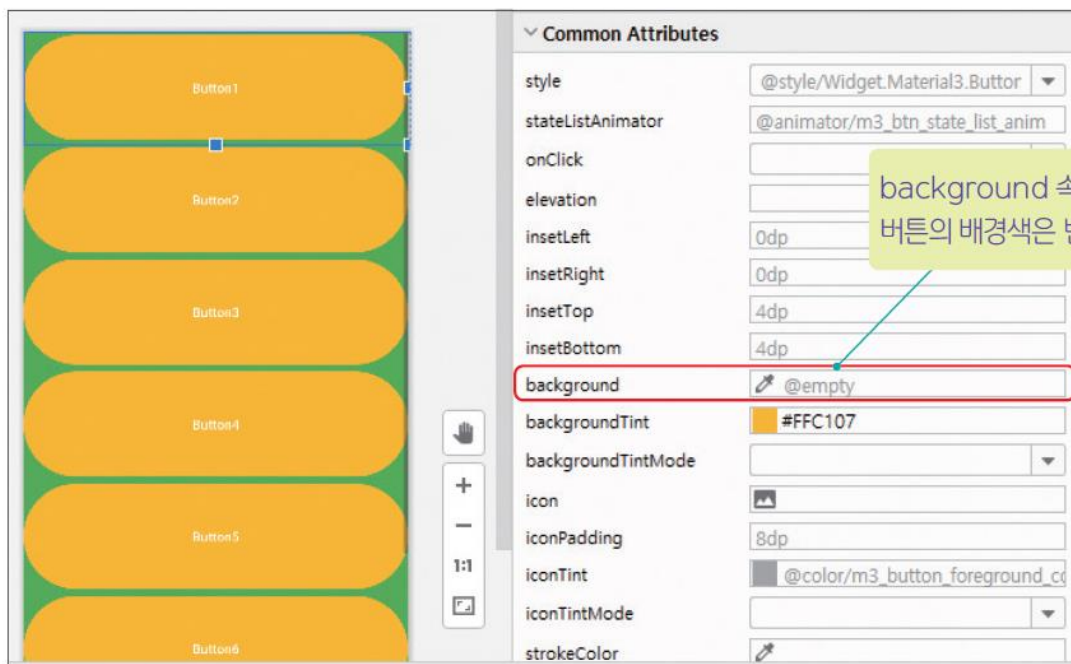
# 머티리얼 디자인

- MaterialComponents 테마는 구글의 디자인 가이드 라인인 “Material Design”을 기반으로 한 안드로이드 앱의 사용자 인터페이스(UI)를 구현하기 위한 테마 스타일이다.
- 머티리얼 디자인은 사용자 경험을 향상시키고 일관성 있는 디자인을 제공하기 위한 원칙과 가이드 라인을 제시하는 디자인 철학이다.



# MDC에서 개별적인 스타일 변경

- MaterialComponents 테마는 이처럼 장점이 많지만 단점도 있는데, 개별적인 스타일링이 무시되고 스타일에서 지정된 색상이나 모양을 따른다는 점이다.



# 예제: 머티리얼 버튼과 기본 버튼

- 화면에 머티리얼 디자인을 따르는 버튼과 기본 버튼을 동시에 만들어서 배경색을 변경해보자.



머티리얼 버튼으로 만들어진다.

`<android.widget.Button>` 버튼으로 만들어진다.



# 예제: 머티리얼 버튼과 기본 버튼

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#FF9800"
    android:text="Button" />
```

← 머티리얼 버튼, 배경색이 변경되지 않는다.

```
<android.widget.Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#FF9800"
    android:text="Button" />
```

← 기본 버튼, 배경색이 변경된다.

```
<TextView
    android:id="@+id/textView"
    android:text="머티리얼 버튼" />
```

```
<TextView
    android:id="@+id/textView2"
    android:text="기본 버튼" />
```

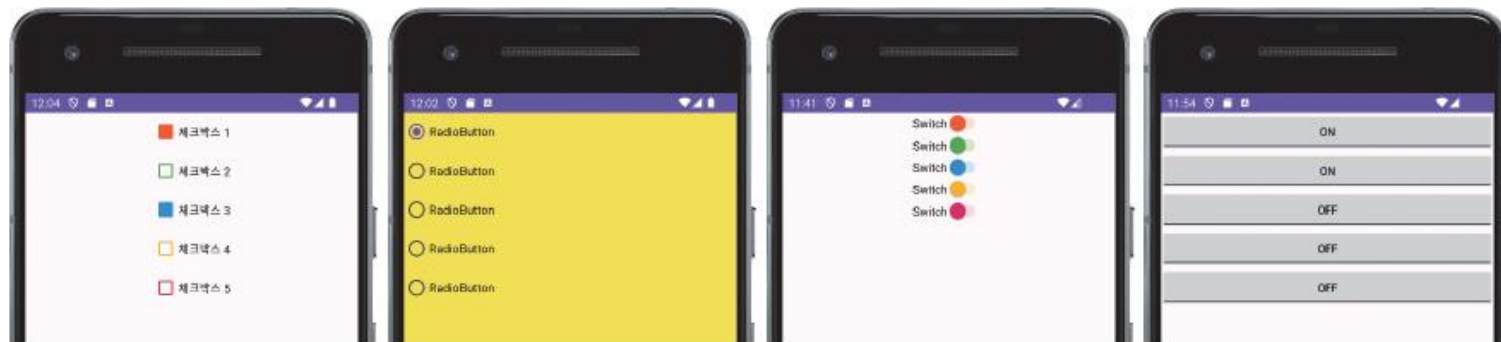
```
</androidx.constraintlayout.widget.ConstraintLayout>
```





# 컴파운드 버튼

- 안드로이드에서 컴파운드 버튼(Compound Button)은 단일 선택 또는 다중 선택을 허용하는 위젯이다.
- 대표적인 컴파운드 버튼으로는 체크 박스, 라디오 버튼, 스위치, 토글 버튼 등이 있다.



체크 박스

라디오 버튼

스위치

토글 버튼



# 체크 박스

- 체크 박스(**checkbox**)는 사용자가 하나의 그룹 안에서 여러 개의 버튼을 동시에 선택할 때 사용하는 위젯이다.
- 체크 박스를 생성하려면 레이아웃에서 **<CheckBox>** 요소를 추가하면 된다.

☒ Checkbox 1

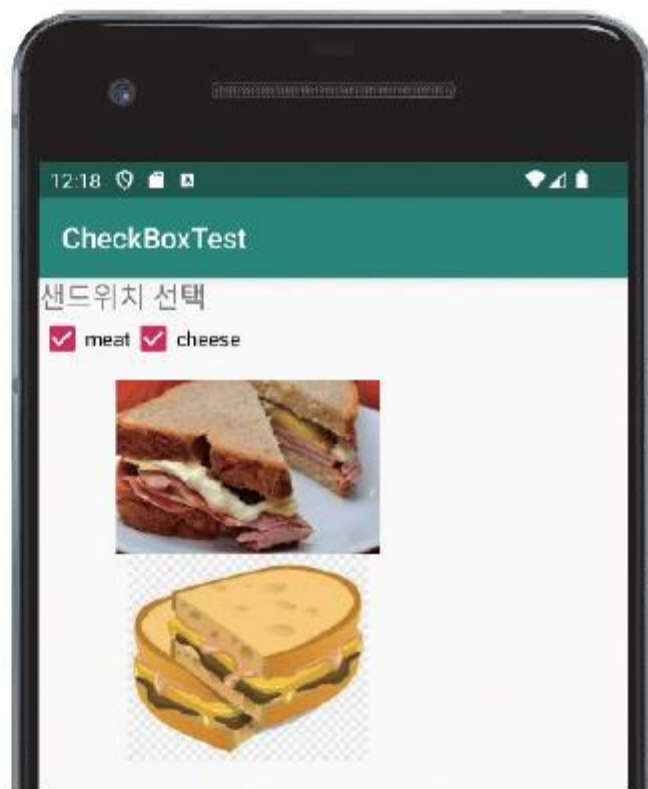
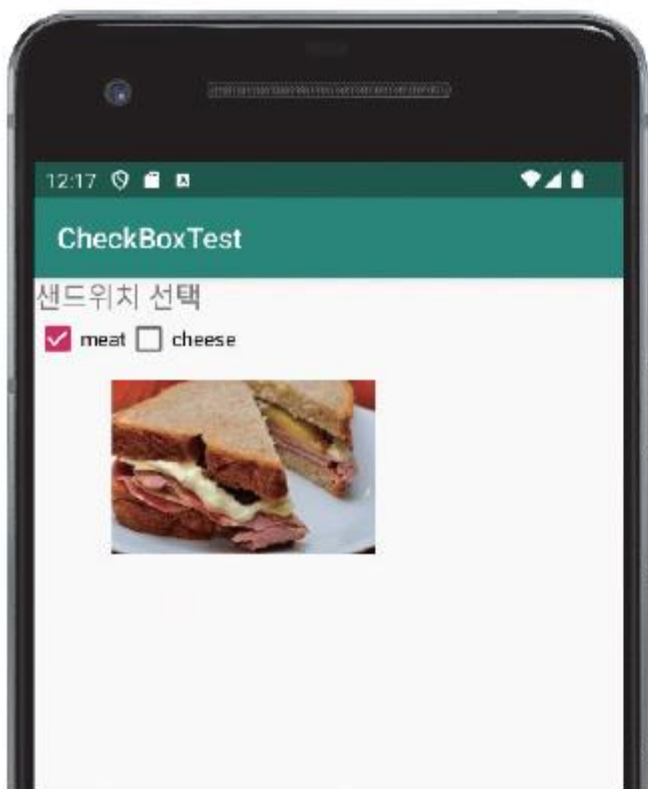
☐ Checkbox 2

☐ Checkbox 3

☐ Checkbox 4



# 예제: 체크 박스





# 예제: 체크 박스

activity\_main.xml

```
<LinearLayout    android:orientation="vertical">
    <TextView      android:text="샌드위치 선택">
    <LinearLayout  android:orientation="horizontal">

        <CheckBox
            android:id="@+id/checkBox"
            android:layout_width="wrap_content"
            android:layout_height="34dp"
            android:onClick="onCheckboxClicked"
            android:text="meat" />

        <CheckBox
            android:id="@+id/checkBox2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:onClick="onCheckboxClicked"
            android:text="cheese" />
    </LinearLayout>

    <ImageView
        android:id="@+id/imageView"
```



# 예제: 체크 박스

```
android:layout_width="262dp"  
android:layout_height="141dp" />
```

```
<ImageView  
    android:id="@+id/imageView2"  
    android:layout_width="259dp"  
    android:layout_height="166dp" />  
</LinearLayout>
```



// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.

```
public class MainActivity extends AppCompatActivity {
    ImageView imageView1, imageView2;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        imageView1 = (ImageView)findViewById(R.id.imageView);
        imageView2 = (ImageView)findViewById(R.id.imageView2);
    }
    public void onCheckboxClicked(View view) {
        boolean checked = ((CheckBox) view).isChecked();

        switch(view.getId()) {
            case R.id.checkBox:
                if (checked) imageView1.setImageResource(R.drawable.sand1);
                else imageView1.setImageResource(0);
                break;
            case R.id.checkBox2:
                if (checked) imageView2.setImageResource(R.drawable.sand2);
                else imageView2.setImageResource(0);
                break;
        }
    }
}
```



# 라디오 버튼

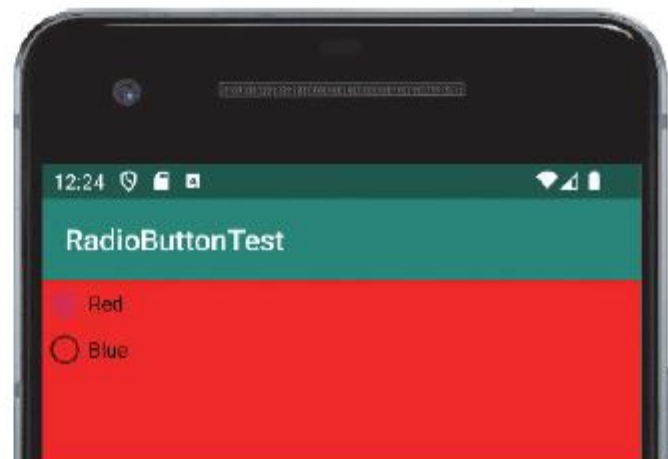
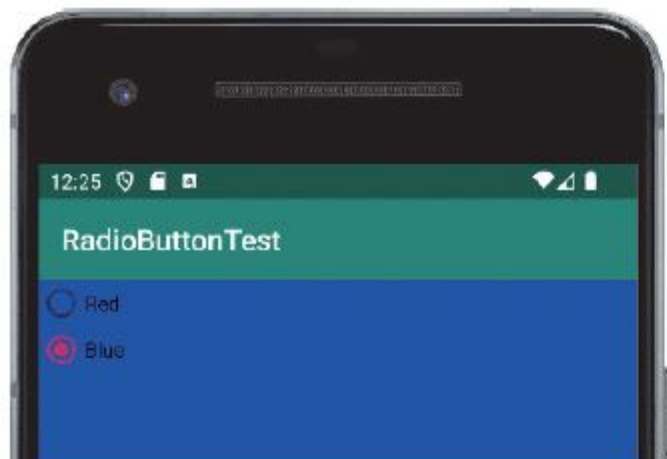
- 라디오 버튼(**radio button**)은 체크 박스와 비슷하지만 하나의 그룹 안에서는 한 개의 버튼만 선택할 수 있다는 점이 다르다.

A rectangular box containing five radio buttons, each followed by a label. The first radio button is selected, indicated by a teal dot in the center. The other four radio buttons are unselected, shown as empty circles.

- ☒ Radio button 1
- ☐ Radio button 2
- ☐ Radio button 3
- ☐ Radio button 4
- ☐ Radio button 5



# 예제: 라디오 버튼 만들기







예

activity\_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/layout"
    android:orientation="vertical" >
```

```
<RadioGroup    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >
```

```
<RadioButton
    android:id="@+id/radio_red"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="onRadioButtonClicked"
    android:text="Red" />
```

```
<RadioButton
    android:id="@+id/radio_blue"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="onRadioButtonClicked"
    android:text="Blue" />
```

```
</RadioGroup>
```

```
</LinearLayout>
```

라디오 그룹 안에  
라디오 버튼을 2개  
정의한다.



## 예제: 라디오 버튼 만들기

MainActivity.java

```
package kr.co.company.radiobutton;

// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.

public class MainActivity extends AppCompatActivity {
    LinearLayout layout;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        layout = (LinearLayout) findViewById(R.id.layout);
    }
}
```



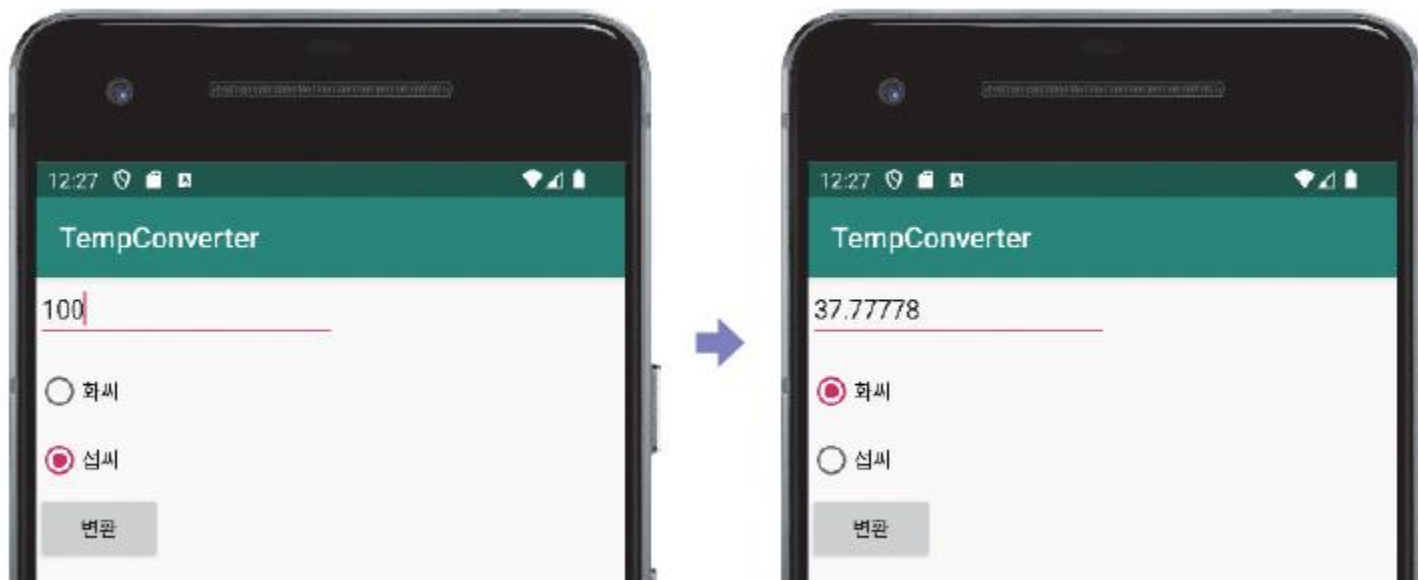
## 예제: 라디오 버튼 만들기

```
public void onRadioButtonClicked(View view) {  
    boolean checked = ((RadioButton) view).isChecked();  
  
    switch (view.getId()) {  
        case R.id.radio_red:  
            if (checked)  
                layout.setBackgroundColor(Color.RED);  
            break;  
        case R.id.radio_blue:  
            if (checked)  
                layout.setBackgroundColor(Color.BLUE);  
            break;  
        }  
    }  
}
```



## 예제: 온도 변환 앱 만들기

- 다음과 같이 섭씨 온도를 받아서 화씨 온도로 변환하는 앱을 작성해 보자.





## MainActivity.java

```
package kr.co.company.tempconverter;

// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.

public class MainActivity extends AppCompatActivity {

    private EditText text;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        text = (EditText) findViewById(R.id.edit_Input);

    }
    public void onClicked(View view) {
        switch (view.getId()) {
            case R.id.btn_change:
                RadioButton celsiusButton = (RadioButton) findViewById(R.id.celsius);
                RadioButton fahrenheitButton = (RadioButton) findViewById(R.id.fahrenheit);

                if (text.getText().length() == 0) {
                    Toast.makeText(this, "정확한 값을 입력하십시오.", Toast.LENGTH_LONG).show();
                    return;
                }
            }
        }
    }
}
```



```
float inputValue = Float.parseFloat(text.getText().toString());
if (celsiusButton.isChecked()) {
    text.setText(String.valueOf(convertFahrenheitToCelsius(inputValue)));
    celsiusButton.setChecked(false);
    fahrenheitButton.setChecked(true);
} else {
    text.setText(String.valueOf(convertCelsiusToFahrenheit(inputValue)));
    fahrenheitButton.setChecked(false);
    celsiusButton.setChecked(true);
}
break;
```

```
}
```

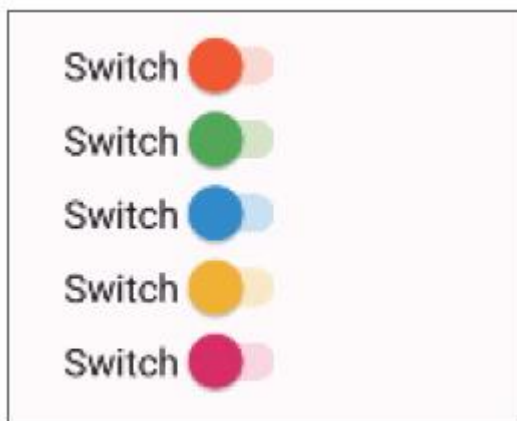
```
private float convertFahrenheitToCelsius(float fahrenheit) {
    return ((fahrenheit - 32) * 5 / 9.0);
}
```

```
private float convertCelsiusToFahrenheit(float celsius) {
    return ((celsius * 9) / 5) + 32.0;
}
}
```



# 스위치

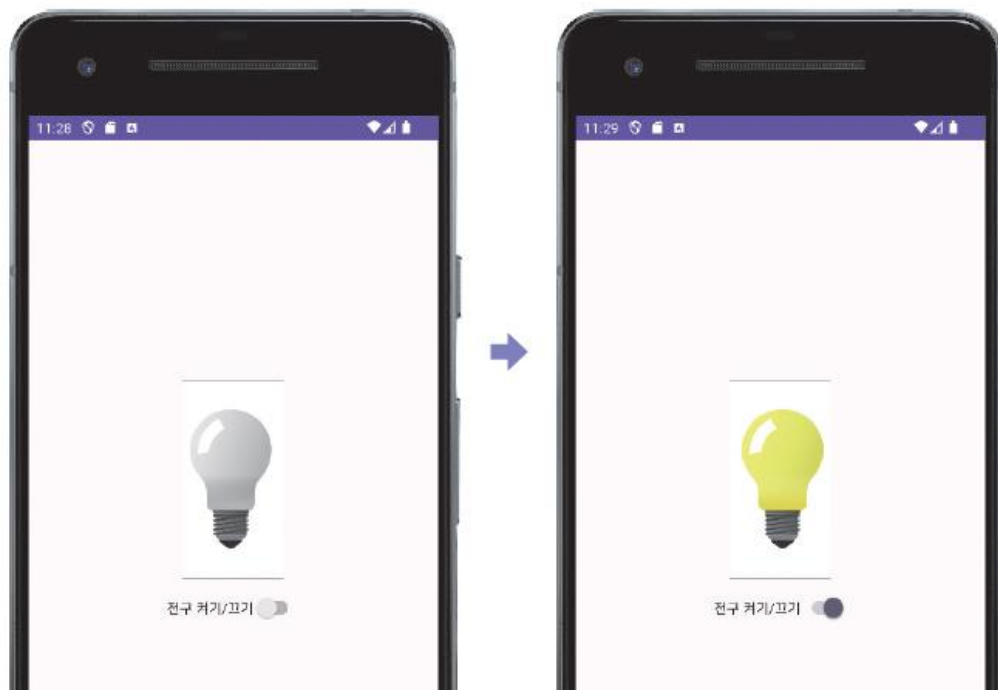
- 안드로이드 앱 개발에서 “스위치(Switch)”는 위젯 중 하나로, 주로 특정 설정이나 옵션을 켜고 끌 때 사용되는 위젯(Widget)이다.





# 예제: 조명 제어 앱 만들기

- 이번 실습에서는 스위치 위젯을 이용하여 전구를 켜거나 끄는 앱을 작성하여 보자.







activity\_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/bulbImageView"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerInParent="true"
        android:src="@drawable/off" />

    <Switch
        android:id="@+id/switchButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/bulbImageView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="16dp"
        android:text="전구 켜기/끄기" />

</RelativeLayout>
```



# 예제: 조명 제어 앱 만들기

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    private ImageView bulbImageView;
    private Switch switchButton;

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

        bulbImageView = findViewById(R.id.bulbImageView);
        switchButton = findViewById(R.id.switchButton);

        // 스위치 상태 변경 리스너 추가
```



# 예제: 조명 제어 앱 만들기

```
switchButton.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {  
    @Override  
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {  
        // 스위치 상태에 따라 전구 이미지 변경  
        if (isChecked) {  
            bulbImageView.setImageResource(R.drawable.on);  
        } else {  
            bulbImageView.setImageResource(R.drawable.off);  
        }  
    }  
});  
}
```



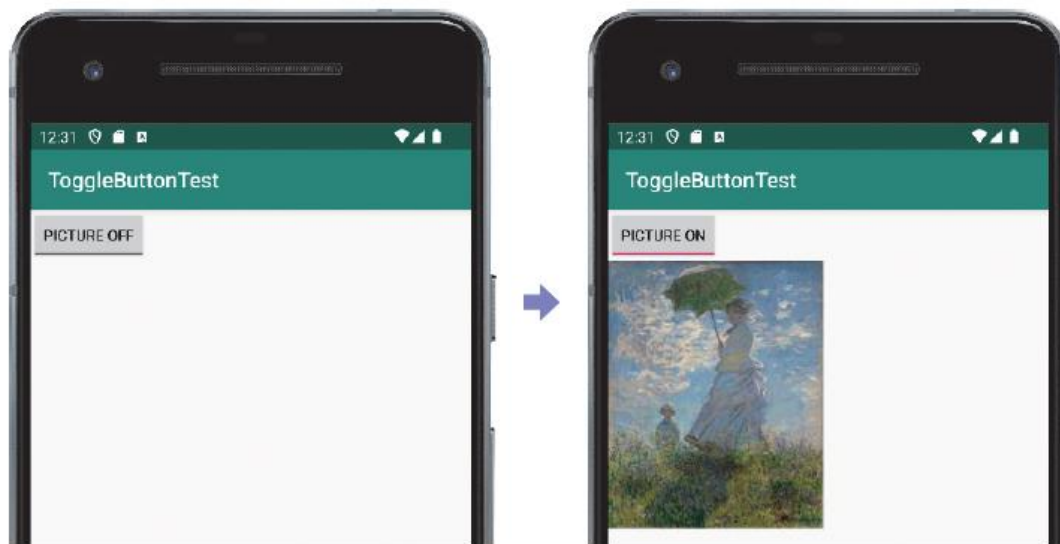
- 두 가지의 상태 중의 하나로 토글되도록 만들어진 버튼이 토글 버튼 (toggle button)이다.





## 예제: 토글 버튼

- 토글 버튼을 누르면 이미지가 나타나고, 다시 누르면 이미지가 사라지는 앱을 작성해보자.





## 예제: 토글 버튼

activity\_main.xml

```
<LinearLayout >
    <ToggleButton
        android:id="@+id/togglebutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="onToggleClicked"
        android:textOff="Picture Off"
        android:textOn="Picture On" />
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>
```



# 예제: 토글 버튼

MainActivity.java

```
package kr.co.company.togglebutton;  
// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.
```

```
public class MainActivity extends AppCompatActivity {  
    private ImageView imageView;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        imageView = (ImageView) findViewById(R.id.imageView);  
    }
```

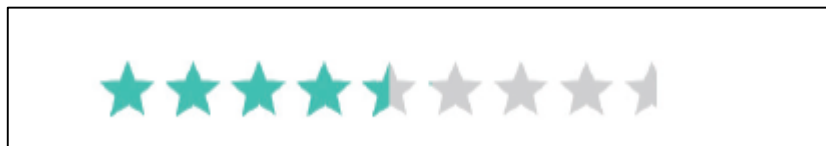
```
    public void onToggleClicked(View view) {  
        boolean on = ((ToggleButton) view).isChecked();  
        if (on) {  
            imageView.setImageResource(R.drawable.pic3);  
        } else {  
            imageView.setImageResource(0);  
        }  
    }  
}
```

이 메소드는 버튼이 클릭되었을 때 수행되어야 하는 동작을 정의한다. 이 예제에서는 버튼의 현재의 상태를 나타내는 토스트 메시지를 출력한다. 토글 버튼은 상태 변경을 스스로 처리하기 때문에 프로그래머는 단지 현재 상태만을 조사하면 된다.



# 레이팅바

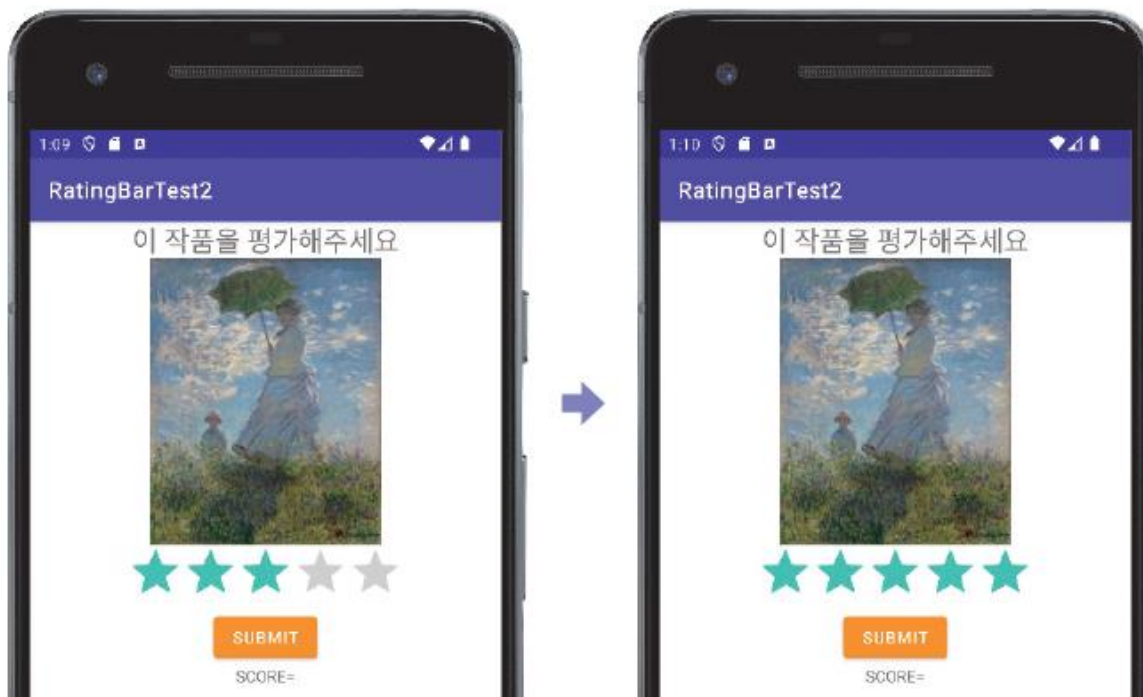
- 레이팅 바(RatingBar)는 시크 바와 프로그레스 바의 확장판이다. 레이팅 바는 별을 사용하여 점수를 표시한다.







# 예제: 그림 평가 앱





# 예제: 그림 평가 앱

activity\_main.xml

```
<LinearLayout >
    <TextView
        android:id="@+id/textView2"
        android:textAlignment="center"
        android:textSize="24sp" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:srcCompat="@drawable/pic3" />

    <RatingBar
        android:id="@+id/ratingBar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:numStars="5"
        android:rating="2.0"
        android:stepSize="1.0" />
```

레이팅 바를 선형 레이아웃 안에 배치한다.

← android:numStars 어트리뷰트는 레이팅 바에 등장하는 별의 최대 개수를 정의한다. android:stepSize 어트리뷰트는 각 별 간의 점수 차이를 정의한다. 즉 0.5라면 절반 크기의 별도 허용한다는 의미이다.



# 예제: 그림 평가 앱

```
<Button
    android:id="@+id/button"
    android:text="SUBMIT" />

<TextView
    android:id="@+id/textView"
    android:text="SCORE=" />
</LinearLayout>
```



# 예제

MainActivity.java

```
package kr.co.company.ratingbartest;  
// 소스만 입력하고 Alt+Enter를 눌러서 import 문장을 자동으로 생성한다.
```

```
public class MainActivity extends AppCompatActivity {
```

```
    private RatingBar ratingBar;
```

```
    private TextView value;
```

```
    private Button button;
```

```
    @Override
```

```
    public void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
```

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

```
        ratingBar = (RatingBar) findViewById(R.id.ratingBar);
```

```
        value = (TextView) findViewById(R.id.textView);
```

```
        button = (Button) findViewById(R.id.button);
```

```
        button.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
```

```
            public void onClick(View v) {
```

```
                float rating = ratingBar.getRating();
```

필요할 때마다 레이팅바에서  
읽어 오는 방법이 편리하다.

```
                value.setText(String.valueOf("SCORE=" + rating));
```

```
            }
```

```
        });
```

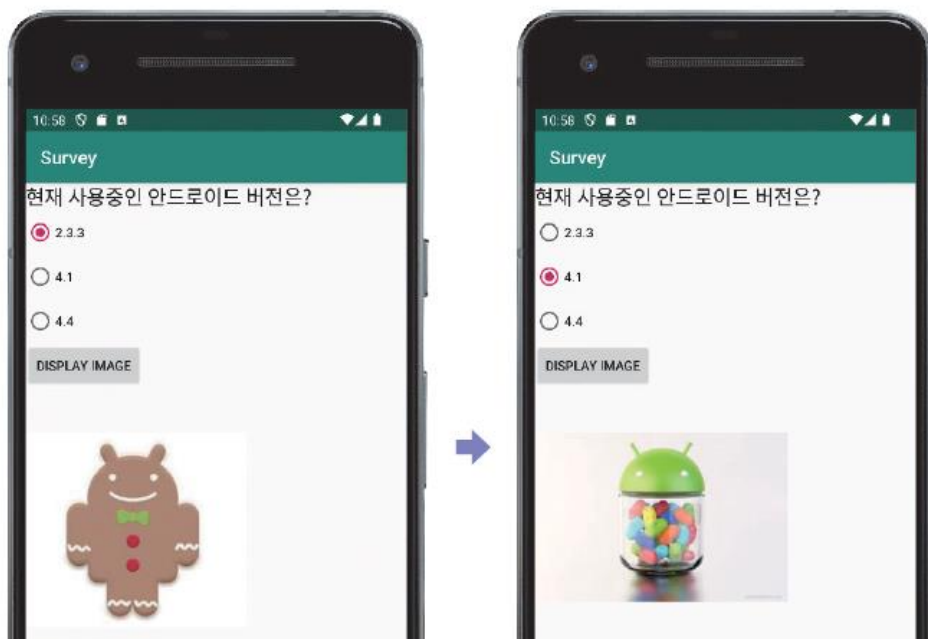
```
    }
```

```
}
```



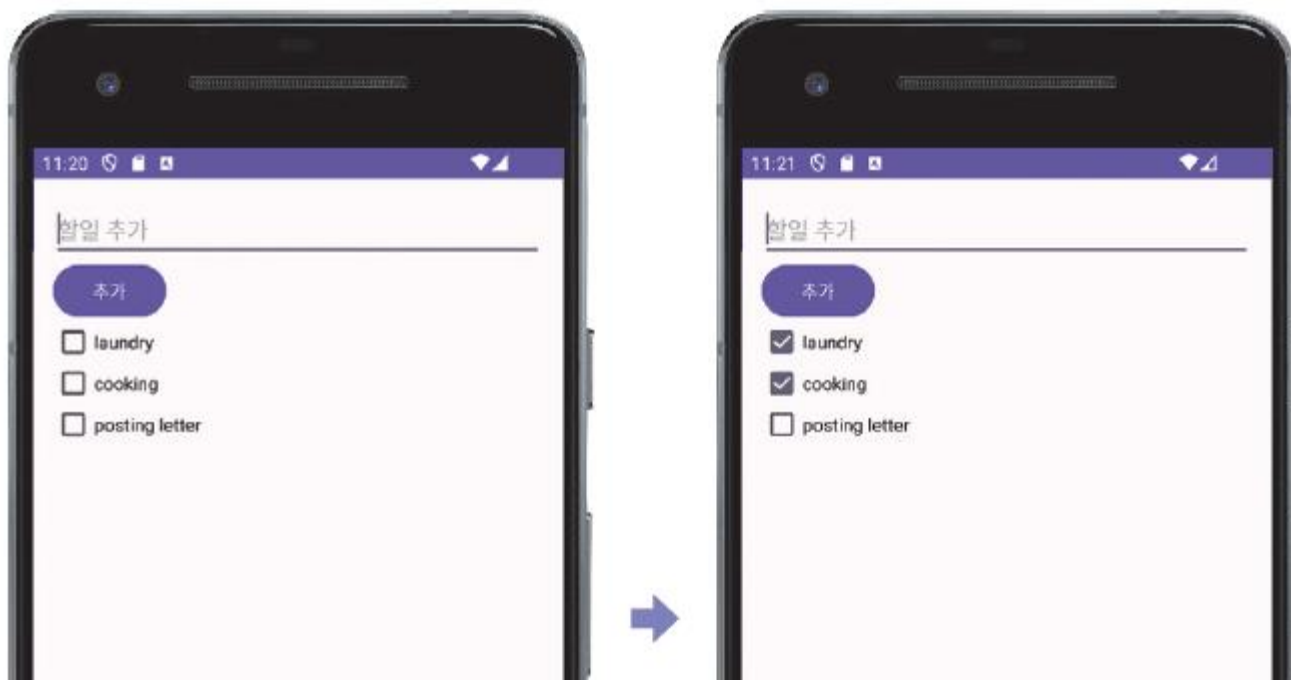
# Coding Challenge: 여론 조사 앱 작성

- 이번 실습에서는 간단한 여론 조사를 할 수 있는 애플리케이션을 작성하여 보자. 라디오 버튼을 클릭하면 해당되는 화면에 이미지가 표시된다.





# Coding Challenge: 할 일 목록 앱





# Q & A

