출처: maxpixel.net

CHAP 3. 기본 위젯

3장 목표

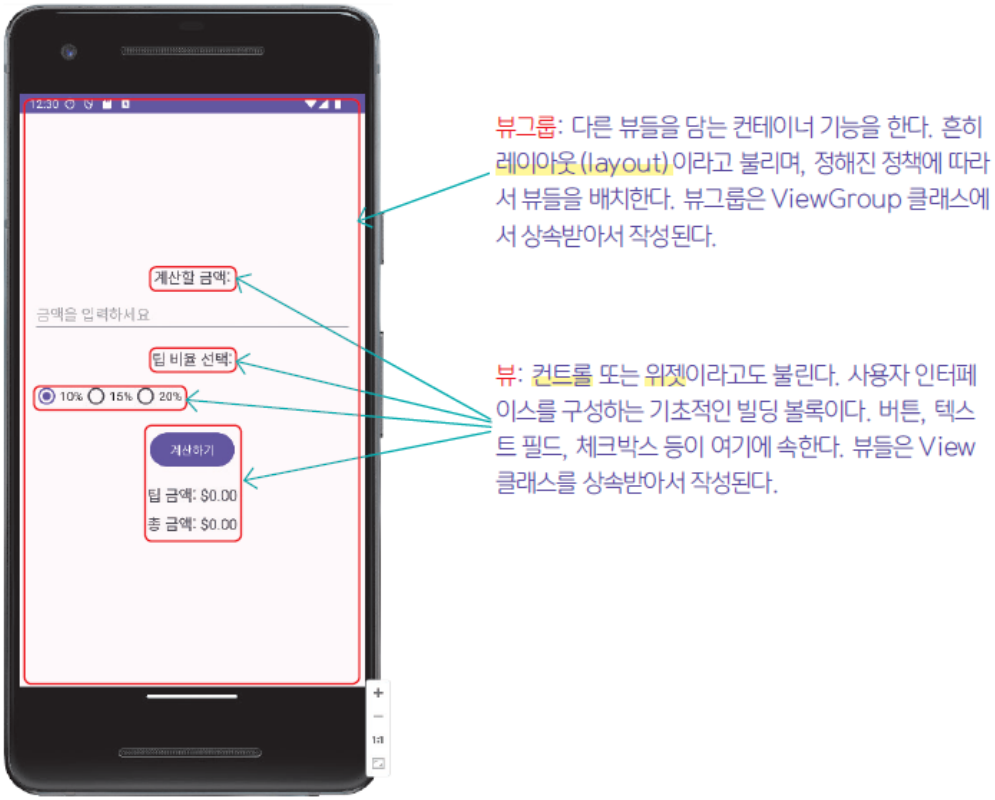
10번 연습 문제

사용자 인터페이스 기초

- 자바의 swing 은 사용하지 않음
  - 너무 리소스를 많이 잡아먹음!
- 독자적인 사용자 인터페이스 컨트롤 사용
  - 버튼, 리스트, 스크롤 바, 체크 박스, 메뉴, 대화 상자

사용자 인터페이스 개요

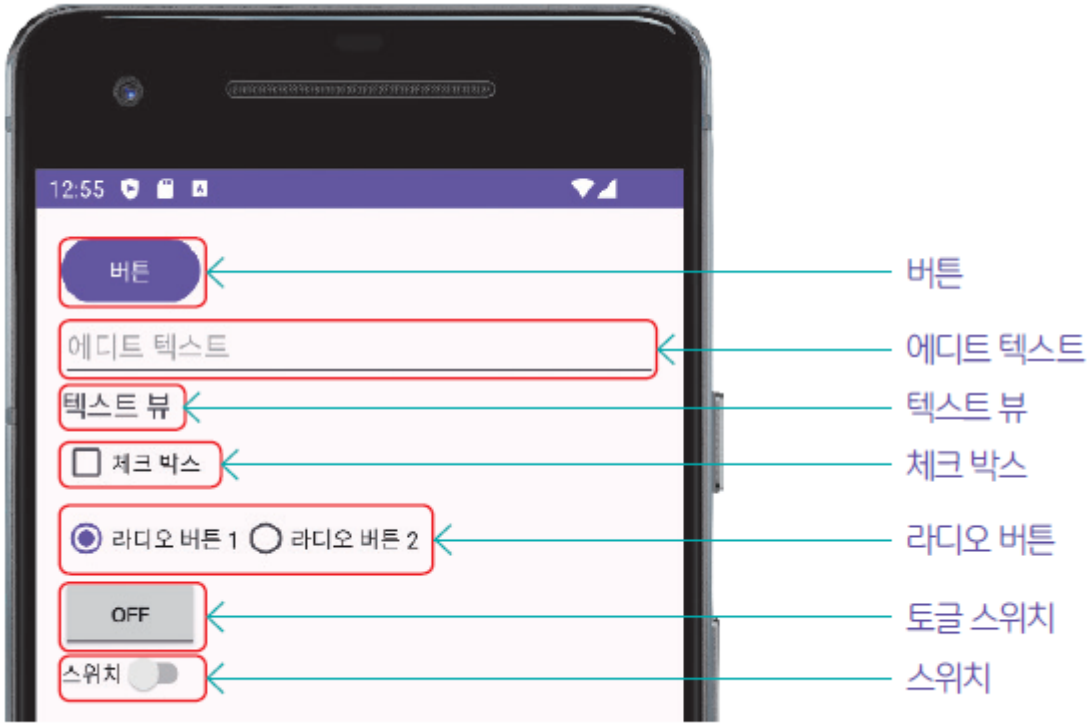
- 사용자 인터페이스 요소들을 크게 분류하면 뷰(View)와 뷰그룹(ViewGroup)으로 나눌 수 있다.



UI를 작성하는 절차

1. 뷰그룹을 생성한다.
2. 필요한 뷰를 추가한다.
3. 액티비티 화면으로 설정한다.

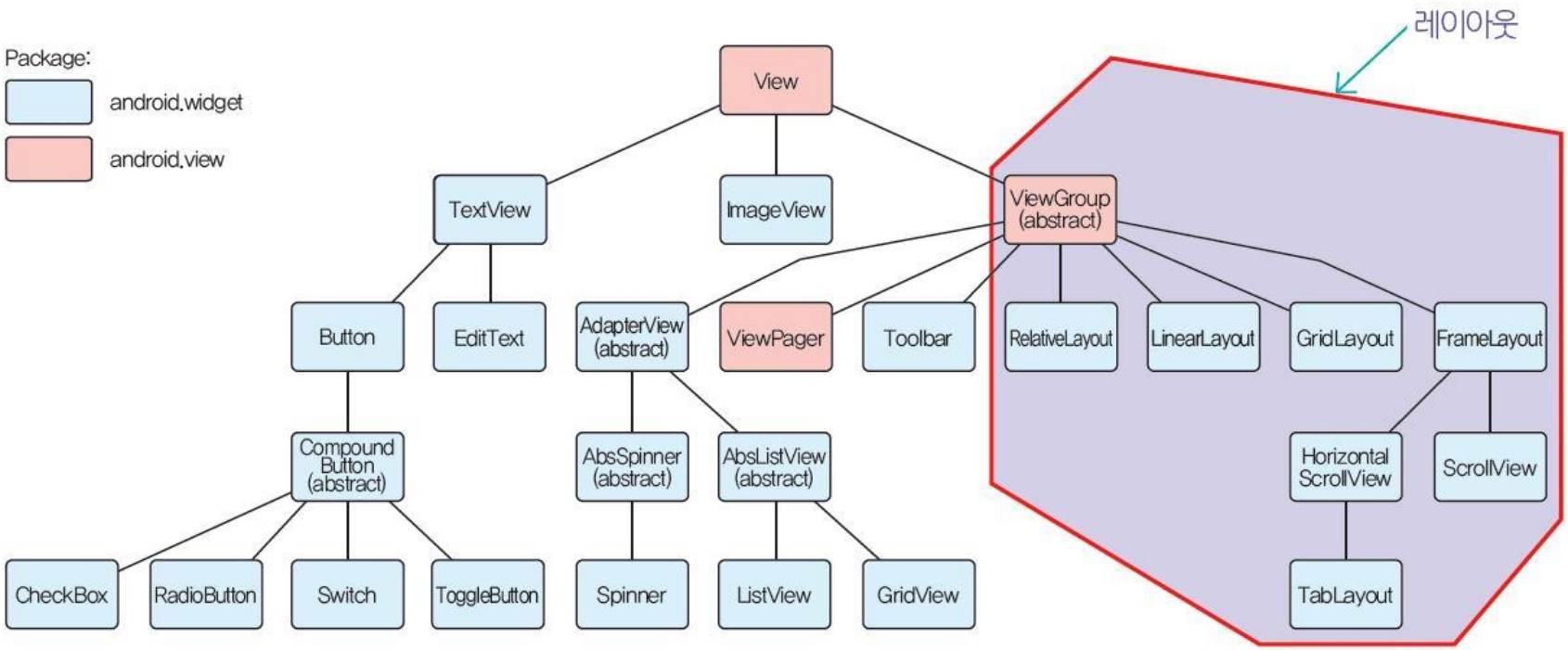
위젯의 유형


12:55
버튼
에디트 텍스트
텍스트 뷰
체크 박스
라디오 버튼 1
라디오 버튼 2
OFF
스위치
버튼
에디트 텍스트
텍스트 뷰
체크 박스
라디오 버튼
토글 스위치
스위치

위젯의 유형

위젯	설명	이미지
Button	클릭할 수 있는 푸시 버튼	BUTTON
EditText	편집이 가능한 텍스트 필드	Name
TextView	편집이 불가능한 텍스트	TextView
CheckBox	사용자가 체크할 수 있는 ON/OFF 스위치	CheckBox
RadioButton	그룹에서 하나의 옵션만 선택할 수 있다.	RadioButton RadioButton
ToggleButton	라이트 인디케이터가 있는 ON/OFF 버튼	ON
Switch	ON/OFF 스위치	Switch

위젯 클래스 계층도


Package:
android.widget
android.view
레이아웃
View
TextView
ImageView
ViewGroup (abstract)
Button
EditText
AdapterView (abstract)
ViewPager
Toolbar
RelativeLayout
LinearLayout
GridLayout
FrameLayout
Compound Button (abstract)
AbsSpinner (abstract)
AbsListView (abstract)
Horizontal ScrollView
ScrollView
CheckBox
RadioButton
Switch
ToggleButton
Spinner
ListView
GridView
TabLayout

UI를 작성하는 다양한 방법

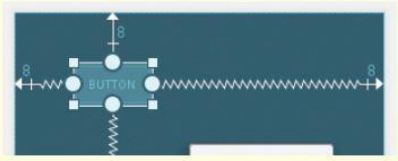
① XML로 사용자 인터페이스 기술

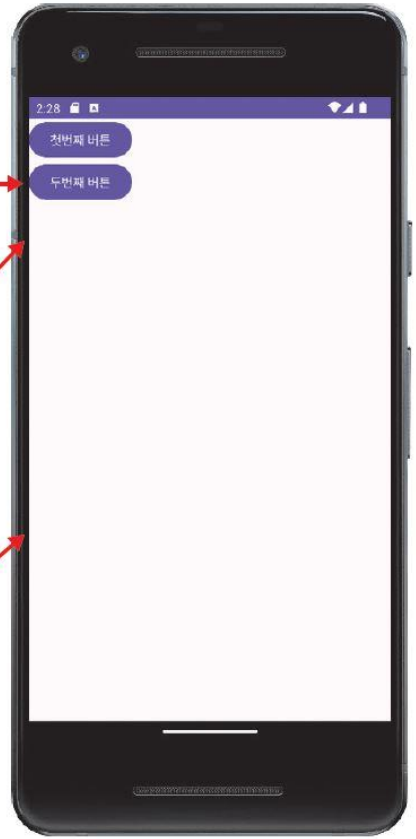
```
...
<Button
android:text="첫번째 버튼"
android:id="@+id/button1"
android:layout_width="match_parent"
android:layout_height="wrap_content">
</Button>
...
```

② 코드로 사용자 인터페이스 작성

```
...
Button b1 = new Button(this);
b1.setText("첫번째 버튼");
container.addView(b1);
...
```

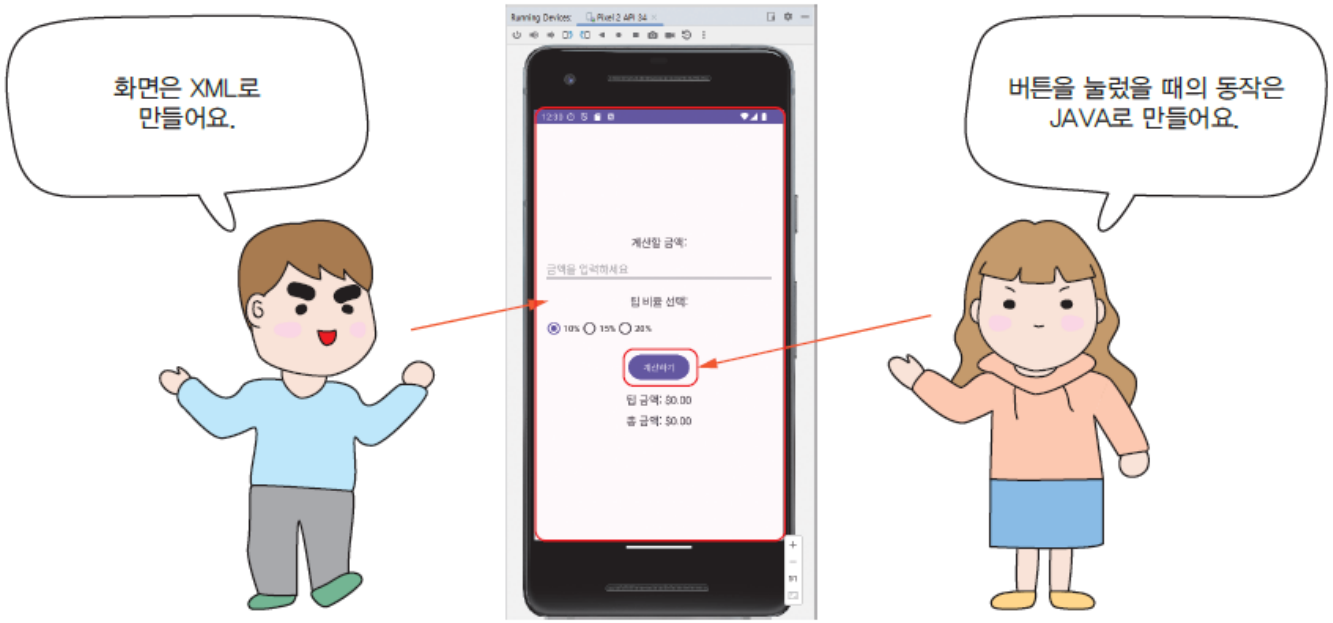
③ 비주얼 도구로 사용자 인터페이스 작성





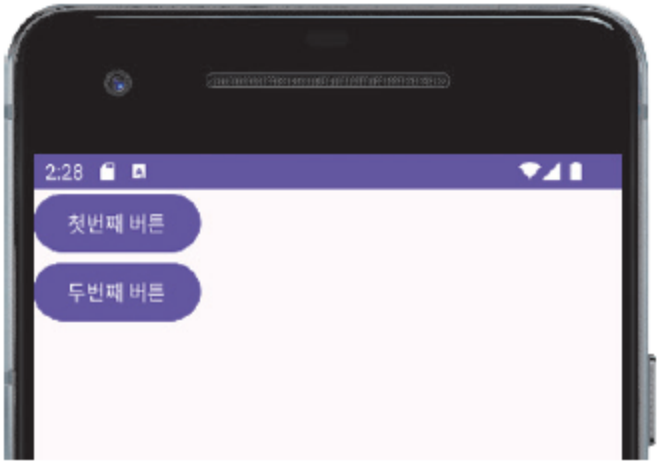
예제: XML로 UI 작성

- 이 방법에서는 코드와 화면 디자인이 완벽하게 분리된다. 즉 코드는 프로그래머가 담당하고 화면은 디자이너가 담당할 수 있다.



예제: XML 파일로 사용자 인터페이스를 작성해보자

- 이번 실습에서는 2개의 버튼으로 이루어진 앱의 화면을 XML 파일로 정의하여 본다.



예제: XML로 UI 작성

activity_main.xml

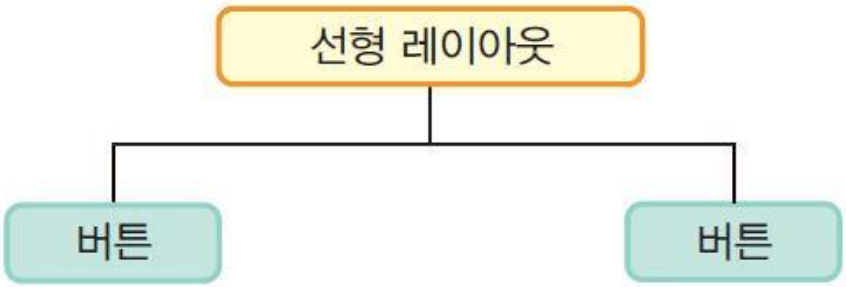
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="첫번째 버튼" >
    </Button>

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="두번째 버튼" >
    </Button>

</LinearLayout>
```

선형 레이아웃이라는 뷰그룹을 생성한다.

버튼이라는 뷰를 생성한다.

예제: XML로 UI 작성

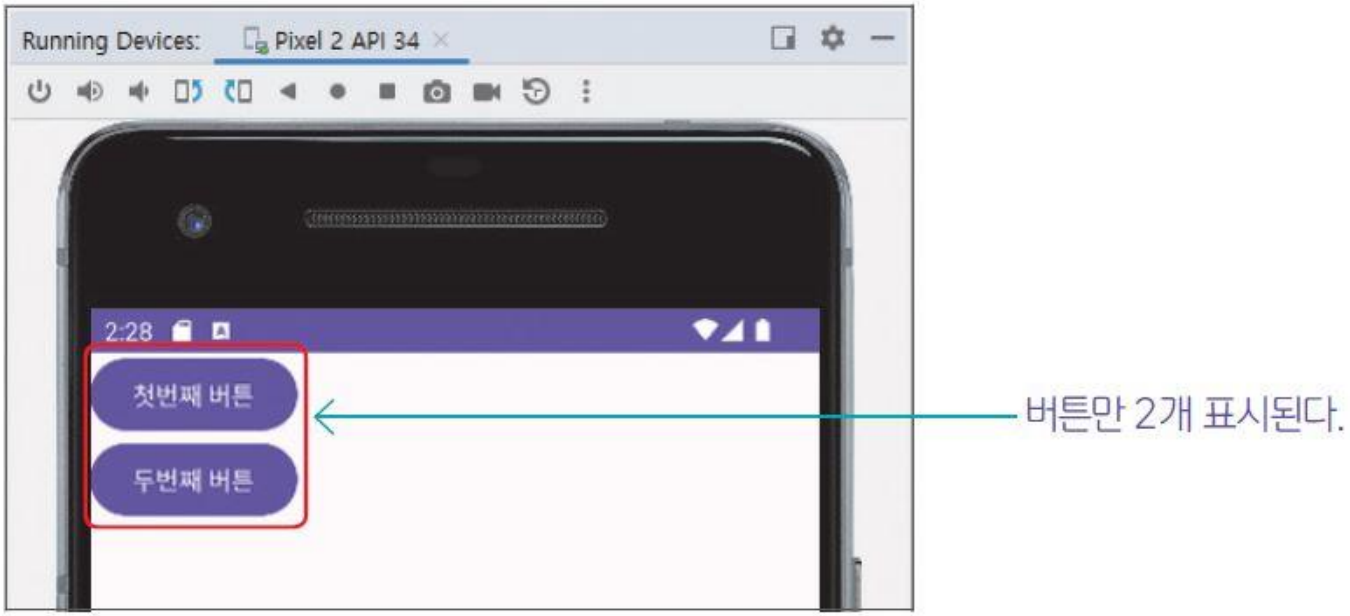
- LinearLayout 요소가 뷰그룹(컨테이너)의 역할을 한다. LinearLayout 요소 안에서 2개의 Button 요소가 정의된다. Button 요소는 버튼을 의미한다.



자바 코드는 변경하지 않는다!

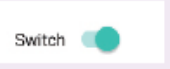
- 자동 생성된 코드 사용!

실행결과



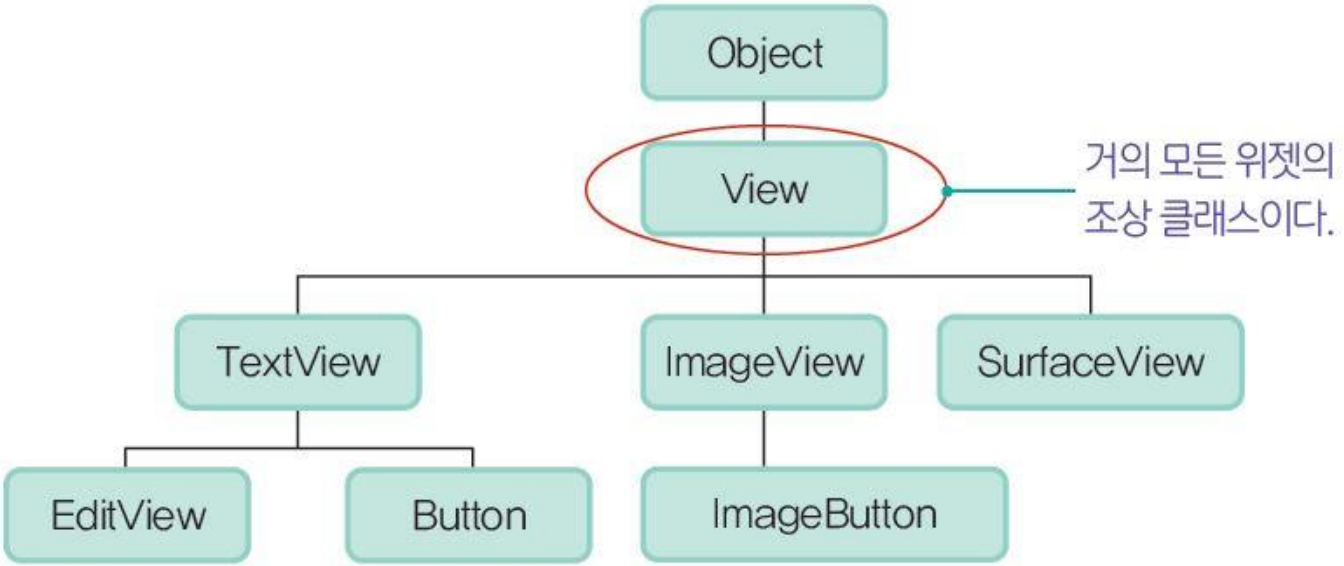
위젯의 속성

- 이번 절에서는 안드로이드 위젯들이 공통적으로 가지고 있는 속성들에 대하여 살펴보자.

위젯	설명	
Button	클릭할 수 있는 푸시 버튼	
EditText	편집이 가능한 텍스트 필드	
TextView	편집이 불가능한 텍스트	
CheckBox	사용자가 체크할 수 있는 ON/OFF 스위치	
RadioButton	그룹에서 하나의 옵션만 선택할 수 있다.	
ToggleButton	라이트 인디케이터가 있는 ON/OFF 버튼	
Switch	ON/OFF 스위치	

View 클래스

- **View** 클래스는 모든 뷰들의 부모 클래스이다.
- View 클래스가 가지고 있는 필드나 메소드는 모든 뷰에서 공통적으로 사용할 수 있다.



id 속성

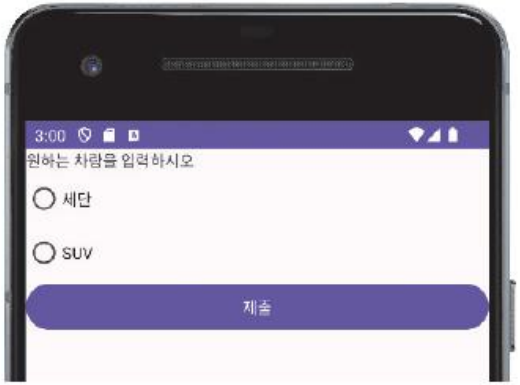
- 모든 위젯은 정수로 된 id(식별자)를 가질 수 있다.

activity_main.xml

```
<LinearLayout >
    <TextView
        android:text="원하는 차량을 입력하시오" />

    <RadioButton
        android:id="@+id/radioButton"
        android:text="세단" />

    <RadioButton
        android:id="@+id/radioButton2"
        android:text="SUV" />
    <Button
        android:id="@+id/button"
```

아이디가 필요 없다.

아이디가 필요하다.



위젯에 식별자를 부여하는 이유

- 자바 코드에서 findViewById( ) 메소드로 위젯을 찾아서 어떤 작업을 하기 위해서이다. 자바 코드에서 찾을 필요가 없는 위젯에는 식별자를 붙일 필요는 없다

```
Button button1;
button1 = (Button) findViewById(R.id.my_button);
```

위젯의 위치와 크기

- 위젯의 위치는 레이아웃 객체에 의하여 결정된다.
- 위젯의 크기는 다음과 같은 값으로 지정이 가능하다.

단위	설명
match_parent	부모의 크기를 꽉 채운다(fill_parent도 같은 의미).
wrap_content	뷰가 나타내는 내용물의 크기에 맞춘다.
px(pixels)	화면의 실제 픽셀을 나타낸다. 픽셀은 권장되는 단위는 아닌데 왜냐하면 장치마다 화면의 밀도가 다르기 때문이다.
dp (density-independent pixels)	dp는 화면의 밀도가 160dpi 화면에서 하나의 물리적인 픽셀을 말한다. 따라서 크기를 160dp로 지정하면 화면의 밀도와는 상관없이 항상 1인치가 된다. dp로 뷰의 크기를 지정하면 화면의 밀도가 다르더라도 항상 동일한 크기로 표시된다.
sp (scale-independent pixels)	화면 밀도와 사용자가 지정한 폰트 크기에 영향을 받아서 변환된다. 이 단위는 폰트 크기를 지정하는 경우에 추천된다.
pt(points)	1/72인치를 표시한다.
mm(millimeters)	밀리미터를 나타낸다.
in(inches)	인치를 나타낸다.

예제: 위젯의 크기 설정

- 레이아웃 파일에서 버튼을 하나 생성하고 버튼의 layout_width와 layout_height 속성을 wrap_content로 변경해보자.

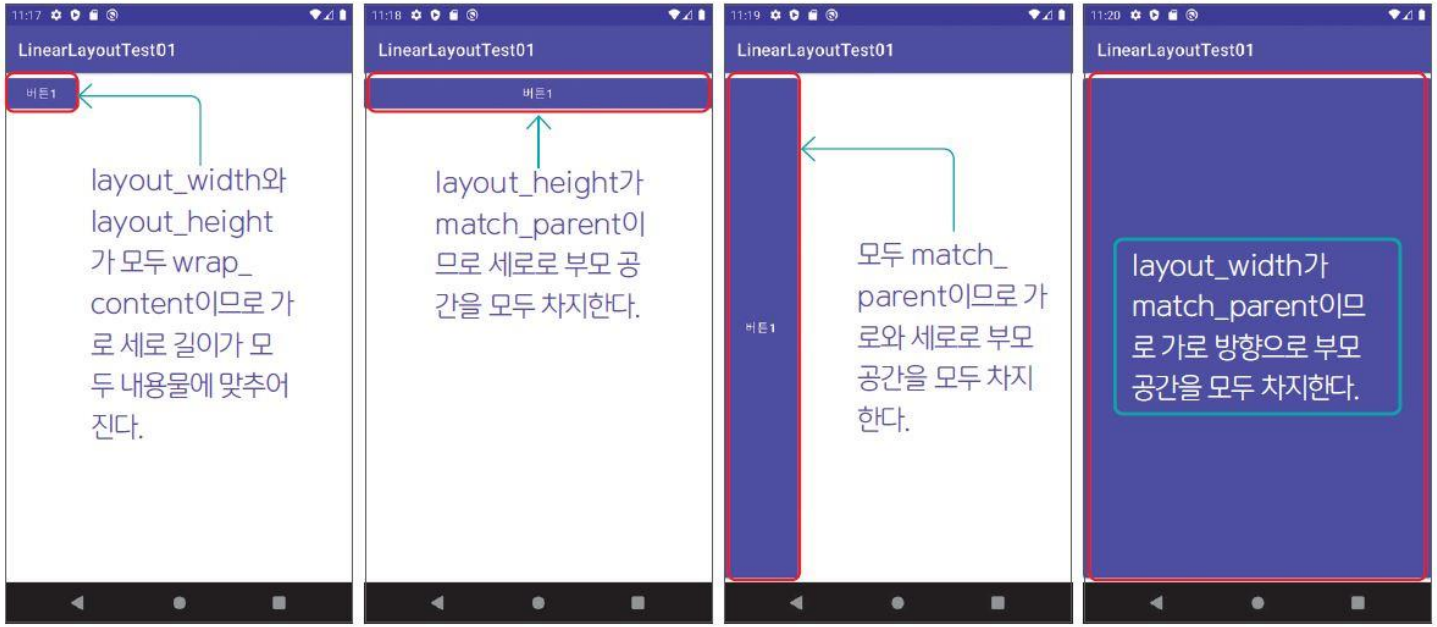
activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="버튼1" />

</LinearLayout>
```

버튼의 위치와 크기를 결정한다.

버튼의 layout_width와 layout_height 변경

실행 결과



(a) layout_width="wrap_content", layout_height="wrap_content"

(b) layout_width="match_parent", layout_height="wrap_content"

(c) layout_width="wrap_content", layout_height="match_parent"

(d) layout_width="match_parent", layout_height="match_parent"

● 그림 3-1 wrap_content와 match_parent

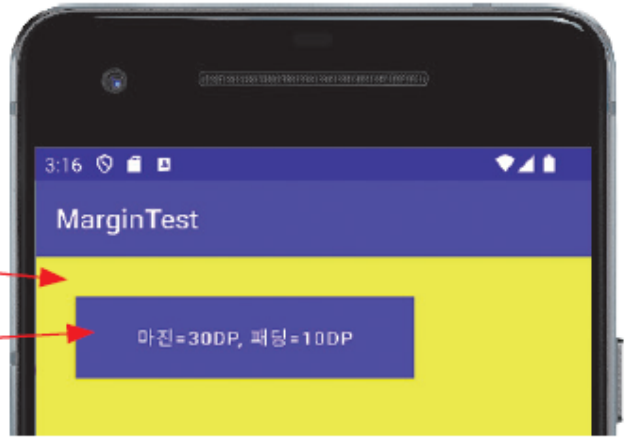
마진과 패딩

- 패딩이란 뷰의 경계와 뷰의 내용물 사이의 간격
- 마진이란 자식 뷰 주위의 여백

마진과 패딩

activity_main.xml

```
<LinearLayout >
    <Button
        android:id="@+id/button1"
        android:layout_width="250dp"
        android:layout_height="60dp"
        android:layout_margin="30dp"
        android:background="#00ffff"
        android:padding="10dp"
        android:text="마진=30dp, 패딩=10dp" />
</LinearLayout>
```



색상

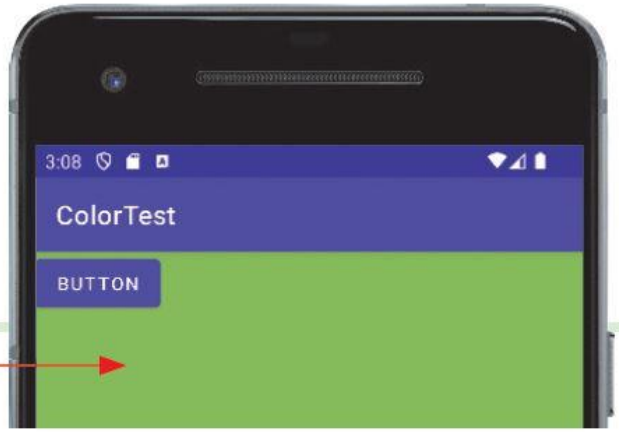
- 16진수로 투명도와 빛의 3원색인 RGB값을 표시

표시 방법	설명
#RRGGBB	RR은 빨간색 성분, GG는 녹색 성분, BB는 청색 성분을 나타낸다.
#AARRGGBB	AA는 투명도, RR은 빨간색 성분, GG는 녹색 성분, BB는 청색 성분을 나타낸다.

예제

activity_main.xml

```
<LinearLayout android:background="#8BC34A">
    <Button
        android:id="@+id/button"
        android:text="Button" />
</LinearLayout>
```



화면에 보이기 속성

- 만약 초기에 뷰의 표시 여부를 제어하려면 visibility 속성을 다음 중의 하나로 설정하면 된다.

상수	값	설명
visible	0	화면에 보이게 한다. 디폴트 값
invisible	1	표시되지 않는다. 그러나 배치에서 공간을 차지한다.
gone	2	완전히 숨겨진다.

화면에 보이기 속성

activity_main.xml

```
<LinearLayout >
    <Button
        android:id="@+id/button"
        android:text="Button1" />
    <Button
        android:id="@+id/button2"
        android:text="Button2"
        android:visibility="invisible" />

    <Button
        android:id="@+id/button3"
        android:text="Button3" />

    <Button
        android:id="@+id/button4"
        android:text="Button4"
        android:visibility="gone" />

    <Button
        android:id="@+id/button5"
        android:text="Button5" />
</LinearLayout>
```

enabled 속성과 rotation 속성

activity_main.xml

```
<LinearLayout >
    <Button
        android:text="Button1" />

    <Button
        android:enabled="false"
        android:text="Button2" />

    <Button
        android:rotation="45"
        android:text="Button3" />
</LinearLayout>
```

텍스트 뷰의 속성

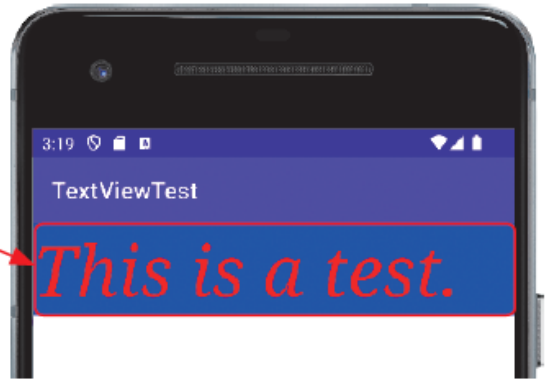
- 텍스트 뷰(TextView)는 화면에 간단한 텍스트를 출력하는 위젯이다. 다른 곳에서는 레이블(label)이라고도 한다.

XML 속성	설명	설정 메소드
text	표시할 텍스트	setText(CharSequence)
textColor	텍스트 색상	setTextColor(ColorStateList)
textSize	텍스트의 크기	setTextSize(float)
textStyle	텍스트 스타일(bold, italic, bolditalic)	setTextStyle(TextStyle)

텍스트뷰

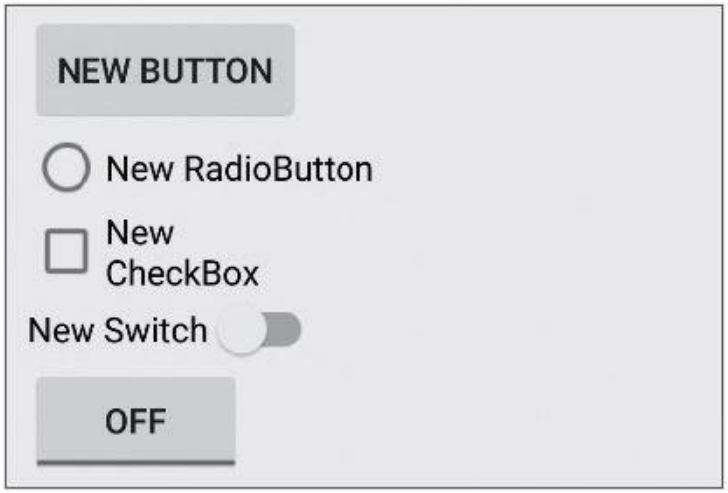
activity_main.xml

```
<LinearLayout>

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#0000ff"
        android:text="This is a test."
        android:textColor="#ff0000"
        android:textSize="60pt"
        android:textStyle="italic"
        android:typeface="serif" />

</LinearLayout>
```



버튼

- 버튼(Button)은 가장 기본적인 위젯으로 사용자 인터페이스에서 아주 많이 사용한다.

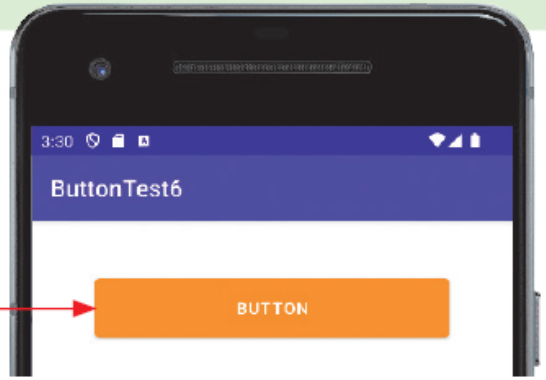


버튼의 색상

- 버튼의 색상은 backgroundTint 속성을 변경하면 된다.

activity_main.xml

```
<androidx.constraintlayout.widget.ConstraintLayout>
    <Button
        android:id="@+id/button"
        android:backgroundTint="#FF9800"
        android:text="Button" />
</androidx.constraintlayout.widget.ConstraintLayout>
```



버튼의 이벤트 처리

① XML 파일에 이벤트 처리 메소드를 등록하는 방법

- 클릭 이벤트만 처리할 수 있는 방법이다. 일반적인 방법은 아니지만 버튼과 같은 위젯의 경우, 가장 간단하게 이벤트를 처리할 수 있다.

② 이벤트를 처리하는 객체를 생성하여 이벤트를 처리하는 방법

- 이벤트를 처리하는 객체를 별도로 생성하여 위젯에 등록한다. 이벤트를 처리하는 가장 일반적인 방법이다. → 익명 클래스와 람다식으로 처리하는 방법을 살펴보자.

③ 뷰 클래스의 이벤트 처리 메소드를 재정의하는 방법

- 뷰 클래스의 이벤트 처리 메소드를 재정의한다. 커스텀 뷰(Custom View)를 작성하는경우에만 사용할 수 있는 방법이다. → 9장에서 살펴본다.

XML 파일을 이용한 이벤트 처리

```xml
<?xml version="1.0" encoding="utf-8"?>
<Button
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/button_send"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="sendMessage"
    android:text="@string/button_send" />
```

onClick 속성에 이벤트를 처리하는 메소드 이름을 적는다.

사용자가 클릭하면 호출된다.

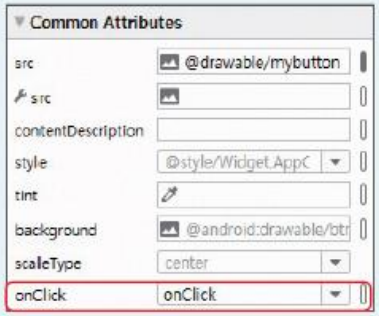
```java
public class MyActivity extends AppCompatActivity {
    @Override
    public void onCreate(...)  {
    //...
    }
    public void sendMessage(View view)
    {

    }
}
```

버튼 이벤트를 처리하는 메소드의 조건

- public이어야 한다.
- void 반환형을 가진다.
- View를 메소드의 인수로 가진다. 클릭된 View 객체가 전달된다.

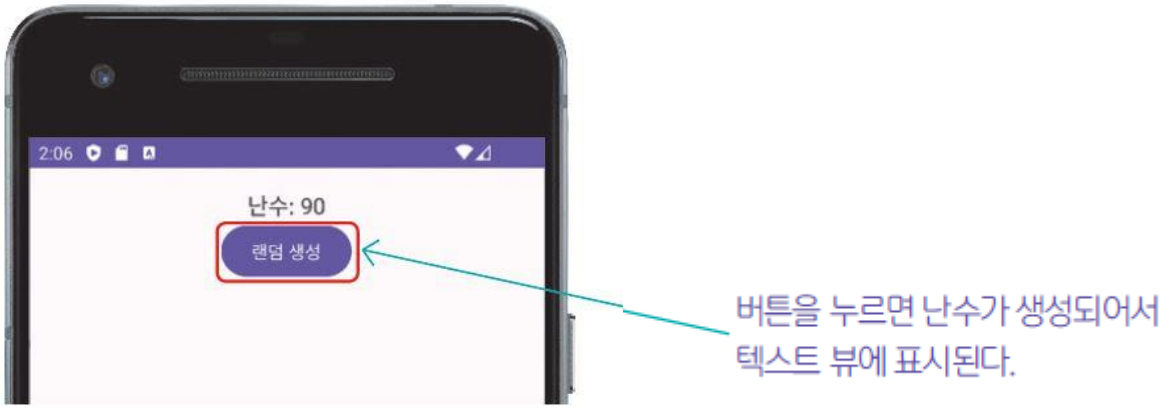
TIP

최근에는 상당히 많은 위젯이 onClick 속성을 지원한다. 위젯을 사용할 때 먼저 안드로이드 스튜디오의 비주얼 도구에서 onClick 속성을 찾아보자. 만약 onClick 속성이 있다면 onClick 속성을 사용하는 것이 제일 편리한 이벤트 처리 방법이다.



예제: 난수 표시 앱

- 텍스트 뷰와 버튼을 사용하여 버튼을 누를 때 텍스트 뷰에 난수로 표시되는 간단한 안드로이드 앱을 작성해보자



예제: 난수 표시 앱

activity_main.xml

```
<LinearLayout>

    <TextView
        android:id="@+id/textViewRandomNumber"
        android:text="난수:"
        android:textSize="18sp"/>

    <Button
        android:id="@+id/buttonGenerateRandom"
        android:text="랜덤 생성"
        android:onClick="generateRandomNumber"/>

</LinearLayout>
```

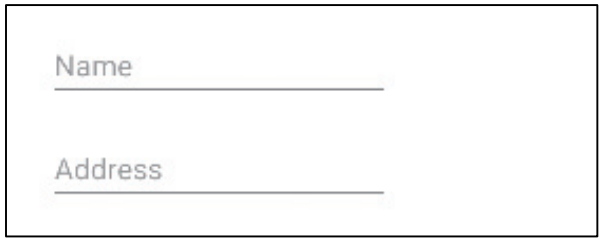
예제: 난수 표시 앱

MainActivity.java

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

  private TextView textViewRandomNumber;

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

      textViewRandomNumber = findViewById(R.id.textViewRandomNumber);
}

  public void generateRandomNumber(View view) {
      // 난수 생성
      Random random = new Random();
      int randomNumber = random.nextInt(100); // 0부터 99까지의 난수 생성

      // 텍스트 뷰에 난수 표시
      textViewRandomNumber.setText("난수: " + randomNumber);
  }
}
```

에디트 텍스트

- 에디트 텍스트(EditText)는 입력이 가능한 필드이다. 다른 곳에서는 텍스트 필드라고도 한다.



속성	설명
android:autoText	자동으로 타이핑 오류를 교정한다.
android:drawableBottom	텍스트의 아래에 표시되는 이미지 리소스이다.
android:drawableRight	텍스트의 오른쪽에 표시되는 이미지 리소스이다.
android:editable	편집 가능
android:text	표시되는 텍스트이다.
android:singleLine	true이면 한 줄만 받음
android:inputType	입력의 종류
android:hint	입력 필드에 표시되는 힌트 메시지

예제: 에디트 텍스트 사용하기 1

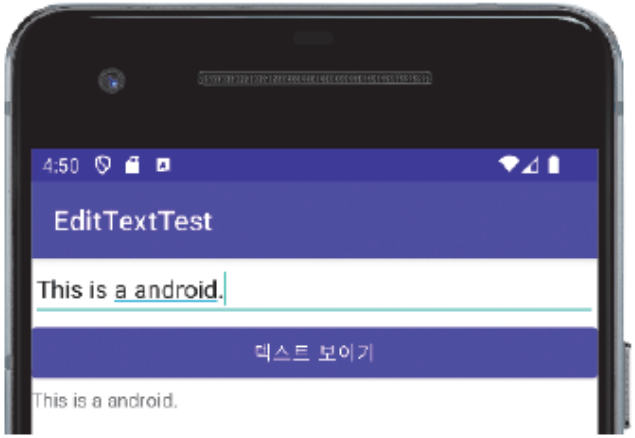
- 사용자가 텍스트를 입력하고 버튼을 누르면, 입력된 텍스트를 화면의 하단에 표시하는 예제를 작성해보자.

예제: 에디트 텍스트 사용하기 1

activity_main.xml

```
<LinearLayout>
    <EditText
        android:id="@+id/edittext"
        android:hint="여기에 텍스트를 입력하시오."
        android:inputType="text" />
    <Button
        android:id="@+id/button"
        android:onClick="onClicked"
        android:text="텍스트 보이기" />
    <TextView
        android:id="@+id/textView"
        android:text="TextView" />
</LinearLayout>
```

에디트 텍스트



예제: 에디트 텍스트 사용하기 1

MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    private TextView textView;
    EditText eText;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        eText = (EditText) findViewById(R.id.edittext);
        textView = (TextView) findViewById(R.id.textView);
    }
    public void onClicked(View v)      {
         String str = eText.getText().toString();
         textView.setText(str);
    }
}
```

버튼이 눌리면 아래의 텍스트 뷰에 입력받은 텍스트를 다시 출력한다.

CharSequence 형식의 문자열이므로 toString()을 불러서 String 객체로 변환

입력 형태 다르게 하기

- 에디트 텍스트에서 가장 중요한 속성은 inputType일 것이다. inputType 속성에 따라, 입력되는 내용을 제한할 수 있다

(textEmailAddress 입력 타입)

(phone 입력 타입)

입력 형태 다르게 하기

```
<EditText
    android:id="@+id/email_address"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="@string/email_hint"
    android:inputType="textEmailAddress" />
```

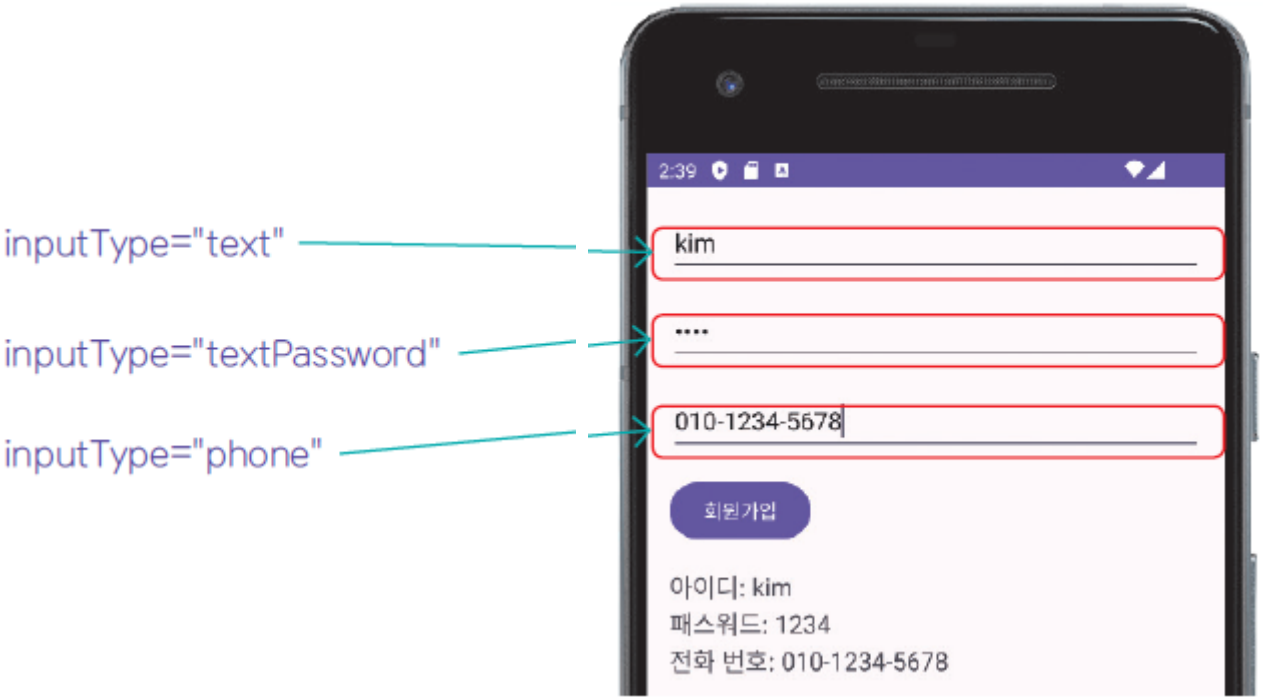
이메일 형태의 입력을 받는다.

inputType의 속성

inputType	설명
none	편집이 불가능한 문자열
text	일반적인 문자열
textMultiLine	여러 줄로 입력 가능
textPostalAddress	우편번호
textEmailAddress	이메일 주소
textPassword	패스워드
textVisiblePassword	패스워드 화면에 보인다.
number	숫자
numberSigned	부호가 붙은 숫자

예제: 에디트 텍스트 사용하기 2

- 사용자로부터 아이디, 패스워드, 전화번호를 입력받을 수 있는 안드로이드 앱을 만들어보자



예제: 에디트 텍스트 사용하기 2

activity_main.xml

```
<LinearLayout>
    <!-- 아이디 입력 -->
    <EditText
        android:id="@+id/editTextUsername"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="아이디를 입력하세요"
        android:inputType="text"
        android:layout_marginBottom="16dp"/>

    <!-- 패스워드 입력 -->
    <EditText
        android:id="@+id/editTextPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="패스워드를 입력하세요"
        android:inputType="textPassword"
        android:layout_marginBottom="16dp"/>
```

한 줄만 입력 가능

입력한 내용이 보이지 않는다.

```
    <!-- 전화번호 입력 -->
    <EditText
        android:id="@+id/editTextPhoneNumber"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="전화번호를 입력하세요"
        android:inputType="phone"
        android:layout_marginBottom="16dp"/>

    <!-- 회원가입 버튼 -->
    <Button
        android:id="@+id/buttonSignup"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="회원가입"
        android:onClick="onSignupButtonClick"/>

    <!-- 회원 정보 출력 텍스트 뷰 -->
    <TextView
        android:id="@+id/textViewUserInfo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text=""
        android:textSize="18sp"
        android:layout_marginTop="16dp"/>

</LinearLayout>
```

전화번호만 입력 가능

예제: 에디트 텍스트 사용하기 2

MainActivity.java

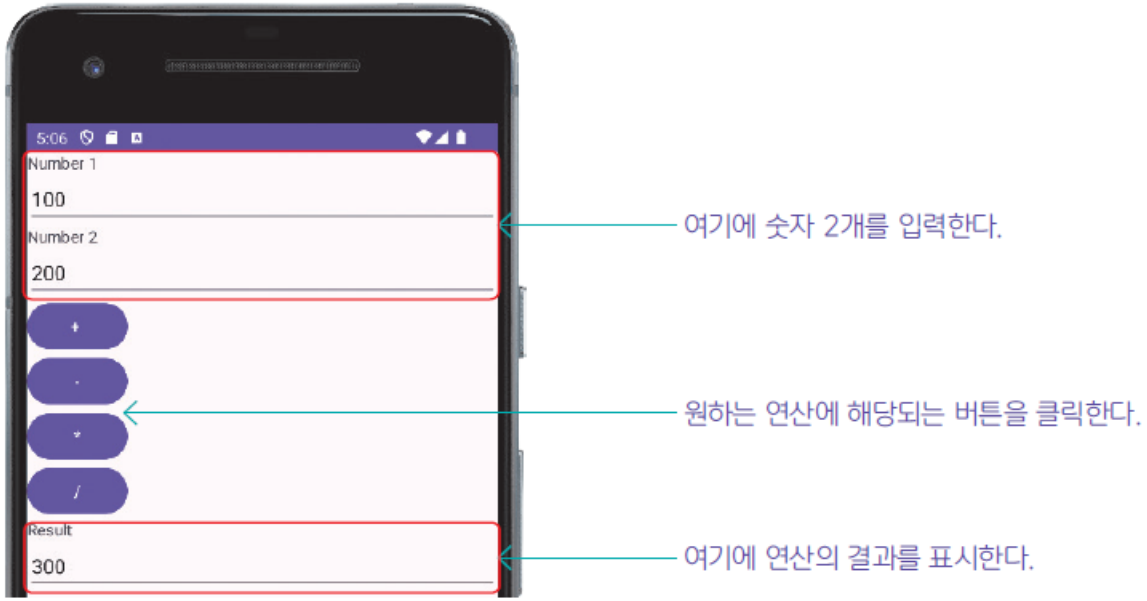
```
public class MainActivity extends AppCompatActivity {

    private EditText editTextUsername;
    private EditText editTextPassword;
    private EditText editTextPhoneNumber;
    private TextView textViewUserInfo;

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

        editTextUsername = findViewById(R.id.editTextUsername);
        editTextPassword = findViewById(R.id.editTextPassword);
```

예제: 에디트 텍스트 사용하기 2

```
        editTextPhoneNumber = findViewById(R.id.editTextPhoneNumber);
        textViewUserInfo = findViewById(R.id.textViewUserInfo);
    }

    public void onSignupButtonClick(View view) {
        String username = editTextUsername.getText().toString();
        String password = editTextPassword.getText().toString();
        String phoneNumber = editTextPhoneNumber.getText().toString();

        // 입력된 정보를 화면 하단에 출력
        String userInfo = "아이디: " + username + "\n패스워드: " + password + "\n전화번호: " +
        phoneNumber;
        textViewUserInfo.setText(userInfo);
    }
}
```

예제: 계산기 앱 #1

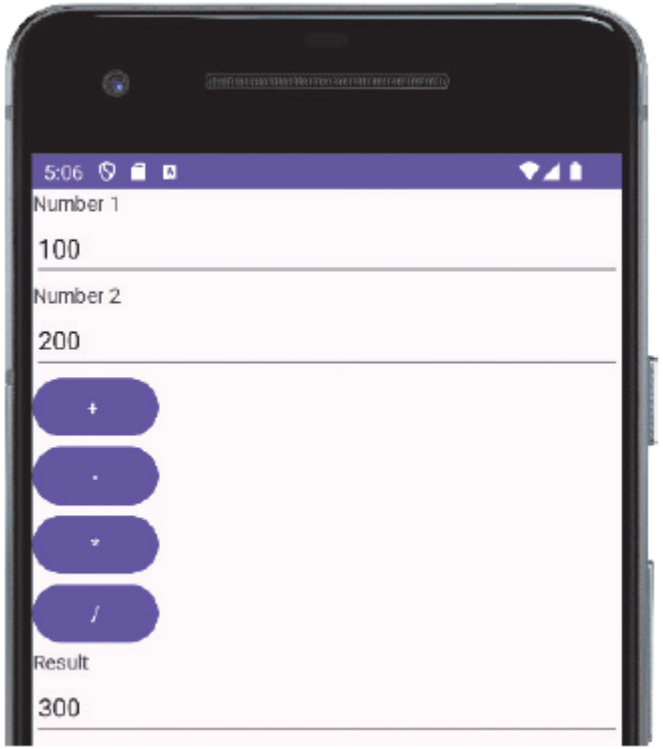
- 이번 실습에서는 다음과 같이 간단한 계산기 화면을 가지는 애플리케이션을 작성하여 보자.



예제: 계산기 앱 #1

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout android:orientation="vertical" >

    <TextView
        android:id="@+id/textView"
        android:text="Number 1"        />

    <EditText
        android:id="@+id/edit1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

    <TextView
        android:id="@+id/textView2"
        android:text="Number 2" />
```



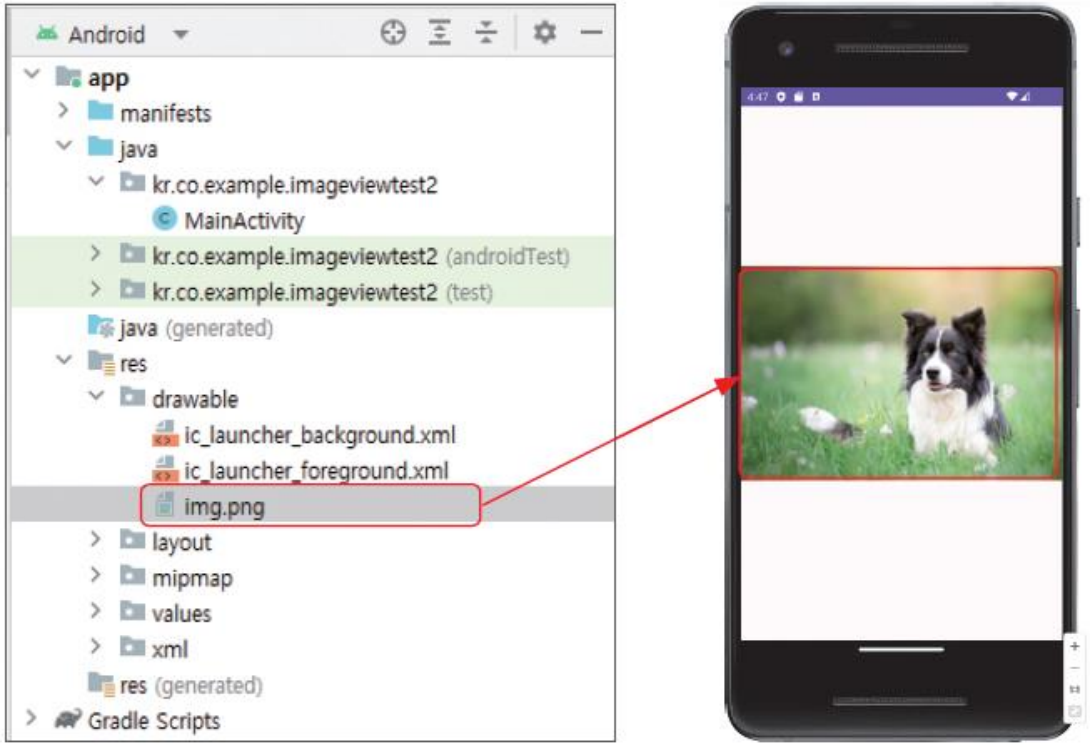
예제. 계산기 앱 #1

MainActivity.java

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

    EditText eText1;
    EditText eText2;
    EditText eText3;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button bPlus = (Button) findViewById(R.id.button1);
        eText1 = (EditText) findViewById(R.id.edit1);
        eText2 = (EditText) findViewById(R.id.edit2);
        eText3 = (EditText) findViewById(R.id.edit3);
    }

    public void cal_plus(View e) {
        String s1 = eText1.getText().toString();
        String s2 = eText2.getText().toString();
        int result = Integer.parseInt(s1) + Integer.parseInt(s2);
        eText3.setText("" + result);
    }
}
```

이미지 뷰와 이미지 버튼

- 이미지 뷰(ImageView)는 아이콘과 같은 이미지들을 간단히 표시하는 데 사용된다. 이미지 뷰는 TextView 클래스를 확장한 것으로 이미지를 표시할 수 있는 TextView라고 생각하면 된다.

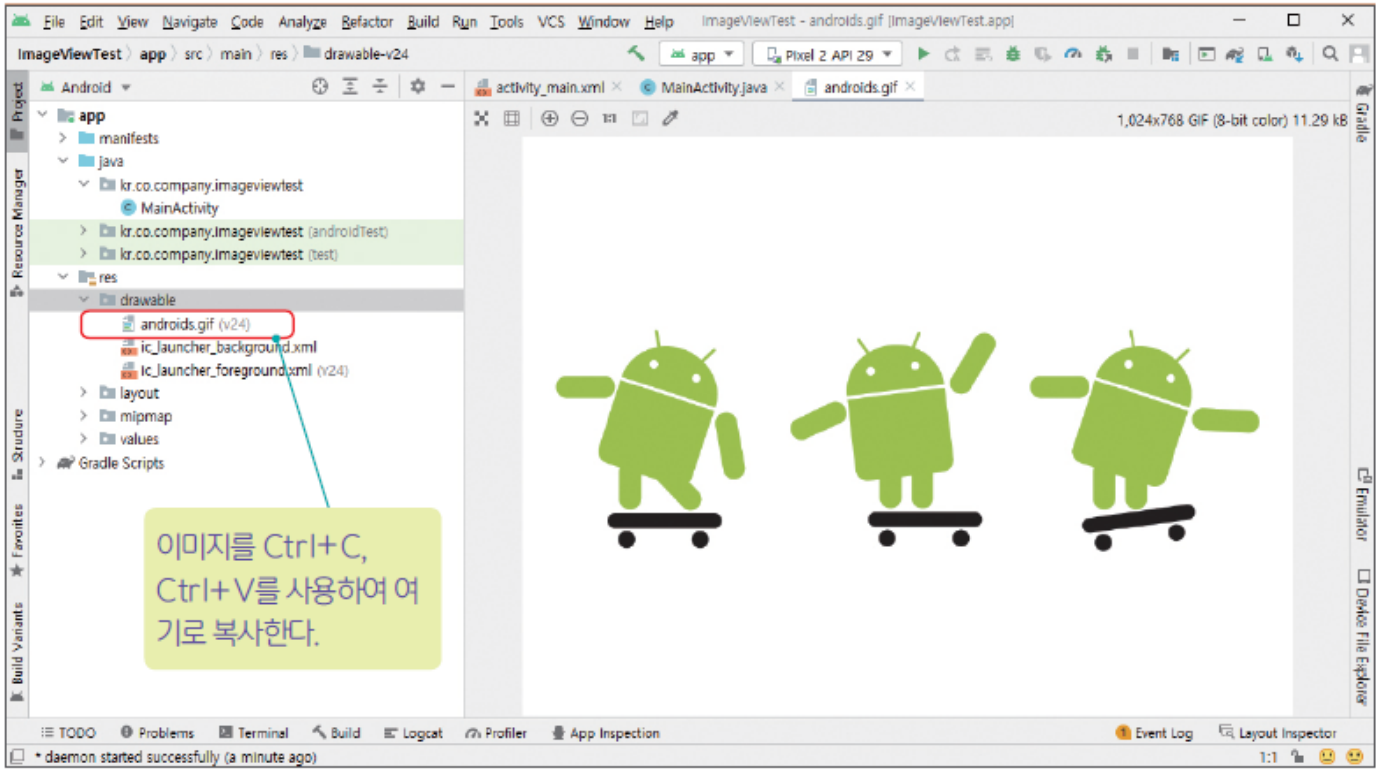


이미지 뷰의 속성

속성	설정 메소드	설명
android:adjustViewBounds	setAdjustViewBounds(boolean)	drawable의 종횡비를 유지하기 위하여 이미지 뷰의 가로, 세로를 조정
android:cropToPadding		true이면 패딩 안에 맞추어서 이미지를 자른다.
android:maxHeight	setMaxHeight(int)	이미지 뷰의 최대 높이
android:maxWidth	setMaxWidth(int)	이미지 뷰의 최대 너비
android:scaleType	setScaleType(ImageView.ScaleType)	이미지 뷰의 크기에 맞추어 어떻게 확대나 축소할 것인지 방법 선택
android:src	setImageResource(int)	이미지 소스
android:tint	setColorFilter(int, PorterDuff.Mode)	이미지 배경 색상

예제: 이미지

- 안드로이드가 지원하는 이미지 형식은 gif, png, jpg 등이다.
- androids.gif 파일을 Ctrl+C로 복사하여서 프로젝트의 app/src/main/res/Drawable 폴더에 Ctrl+V로 붙여넣기한다.



예제: 이미지

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ImageView
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/icon"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:adjustViewBounds="true"
  android:src="@drawable/androids"
/>
```

src 속성이 이미지 파일 이름을 가지고 있다.

예제: 이미지

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<ImageView
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:id="@+id/icon"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:maxHeight="100dp"
  android:maxWidth="200dp"
  android:adjustViewBounds="true"
  android:tint="#80ff0000"
  android:src="@drawable/androids"
  />
```

이미지 속성 변경

- 이미지의 크기, 배치, 스케일링, 이미지 리소스 등을 조절하여 원하는 디자인을 만들 수 있다.
- android:layout_width와 android:layout_height: 이미지 뷰의 크기를 조절한다. wrap_content, match_parent, 고정 크기(dp, px 등)로 설정할 수 있다.
- android:scaleType: 이미지의 스케일링 및 크롭 방식을 지정한다. 일반적으로 centerCrop, fitCenter, centerInside 등을 사용하여 이미지를 화면에 맞게 조절한다.
- android:background: ImageView의 배경색을 설정한다.

코드로 이미지를 동적으로 변경하는 방법

- setImageResource( ) 메서드를 사용하여 자바 코드에서도 이미지를 변경할 수 있다.

```
// ImageView를 찾는다.
ImageView imageView = findViewById(R.id.imageView);

// 이미지 리소스를 설정한다. 여기서 "your_image_name"은 이미지 리소스의 이름이다.
imageView.setImageResource(R.drawable.your_image_name);
```

예제: 이미지 속성 변경

- 각 버튼을 누르면 이미지를 회전시킨다거나 투명도를 변경하고 ScaleType을 변경하여서 이미지의 크기나 비율을 변경하는 앱을 작성해보자.

예제: 이미지 속성 변경

activity_main.xml

```
<LinearLayout>
    <ImageView
        android:id="@+id/imageView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:src="@drawable/img"
        android:layout_marginBottom="16dp"/>
    <LinearLayout  android:orientation="horizontal">
        <Button
            android:text="Scale Type 변경"
            android:onClick="changeScaleType"/>
        <Button
            android:text="회전 변경"
            android:onClick="changeRotation"/>
        <Button
            android:text="Alpha 변경"
            android:onClick="changeAlpha"/>
    </LinearLayout>
</LinearLayout>
```

예제: 이미지 속성 변경

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    private ImageView imageView;
    private int scaleTypeIndex = 0;

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

        imageView = findViewById(R.id.imageView);
    }
```

예제

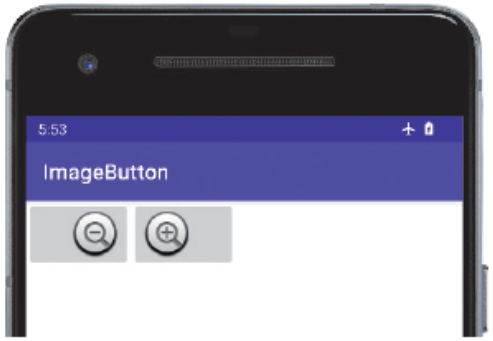
```
    // 버튼 클릭 이벤트 처리: Scale Type 변경
    public void changeScaleType(View view) {
        ImageView.ScaleType[] scaleTypes = {
                ImageView.ScaleType.CENTER,
                ImageView.ScaleType.CENTER_CROP,
                ImageView.ScaleType.CENTER_INSIDE,
                ImageView.ScaleType.FIT_CENTER,
                ImageView.ScaleType.FIT_XY
        };

        imageView.setScaleType(scaleTypes[scaleTypeIndex]);
        scaleTypeIndex = (scaleTypeIndex + 1) % scaleTypes.length;
    }

    // 버튼 클릭 이벤트 처리: 회전 변경
    public void changeRotation(View view) {
        imageView.setRotation(imageView.getRotation() + 45);
    }

    // 버튼 클릭 이벤트 처리: Alpha 변경
    public void changeAlpha(View view) {
        float alpha = imageView.getAlpha();
        alpha = (alpha == 1.0f) ? 0.5f : 1.0f;
        imageView.setAlpha(alpha);
    }
}
```

예제: 이미지 버튼 이벤트 처리

- 안드로이드에서 이미지로 버튼을 만드는 경우는 상당히 많다. 가장 쉬운 방법은 레이아웃 파일에서 <ImageButton> 태그를 사용하는 방법이다.

activity_main.xml

```
<LinearLayout>
    <ImageButton
        android:id="@+id/imageButton"
        app:srcCompat="@android:drawable/btn_minus" />

    <ImageButton
        android:id="@+id/imageButton2"
        app:srcCompat="@android:drawable/btn_plus" />
</LinearLayout>
```

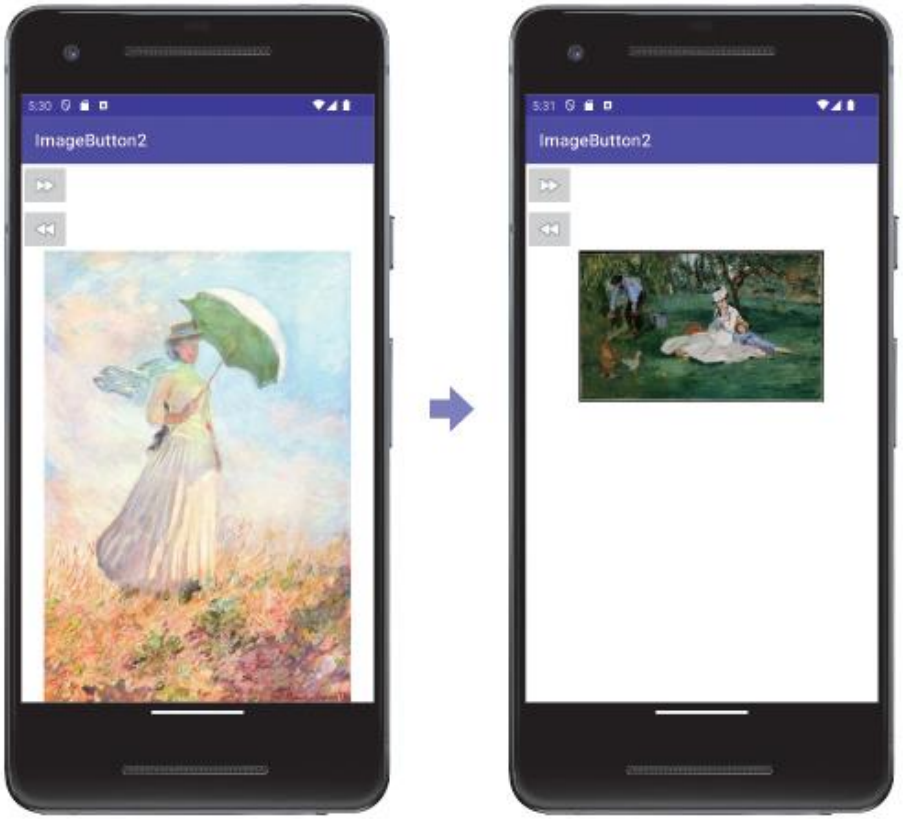


src와 srcCompat

- src와 srcCompat는 안드로이드 앱에서 이미지를 표시하는 데 사용되는 두 가지 다른 속성이다.
- src 속성은 Android 2.2(API 레벨 8) 이상에서 사용 가능한 오래된 속성이다. src 속성을 사용하면 이미지를 설정할 때 기본적으로 앱의 테마에 따라 이미지가 렌더링된다. 이 속성은 벡터 이미지를 지원하지 않는다. 따라서 높은 해상도의 화면에서 이미지가 흐릿하게 보일 수 있다.
- srcCompat 속성은 AppCompat 라이브러리와 호환되는 AndroidX 라이브러리를 사용하는 경우, 사용할 수 있는 상대적으로 최신의 속성이다.

예제: 이미지 뷰어 만들기

- 버튼을 누르면 앱 안의 이미지들을 차례대로 보여주는 앱을 작성해 보자. 이미지 뷰의 setImageResource( ) 메소드를 호출한다.



예제: 이미지 뷰어 만들기

activity_main.xml

```
<LinearLayout>

    <ImageButton
        android:id="@+id/imageButton"
        android:onClick="setImage1"
        app:srcCompat="@android:drawable/ic_media_ff" />

    <ImageButton
        android:id="@+id/imageButton2"
        android:onClick="setImage2"
        app:srcCompat="@android:drawable/ic_media_rew" />

   <ImageView
       android:id="@+id/imageView"
       android:src="@drawable/pic" />
</LinearLayout>
```

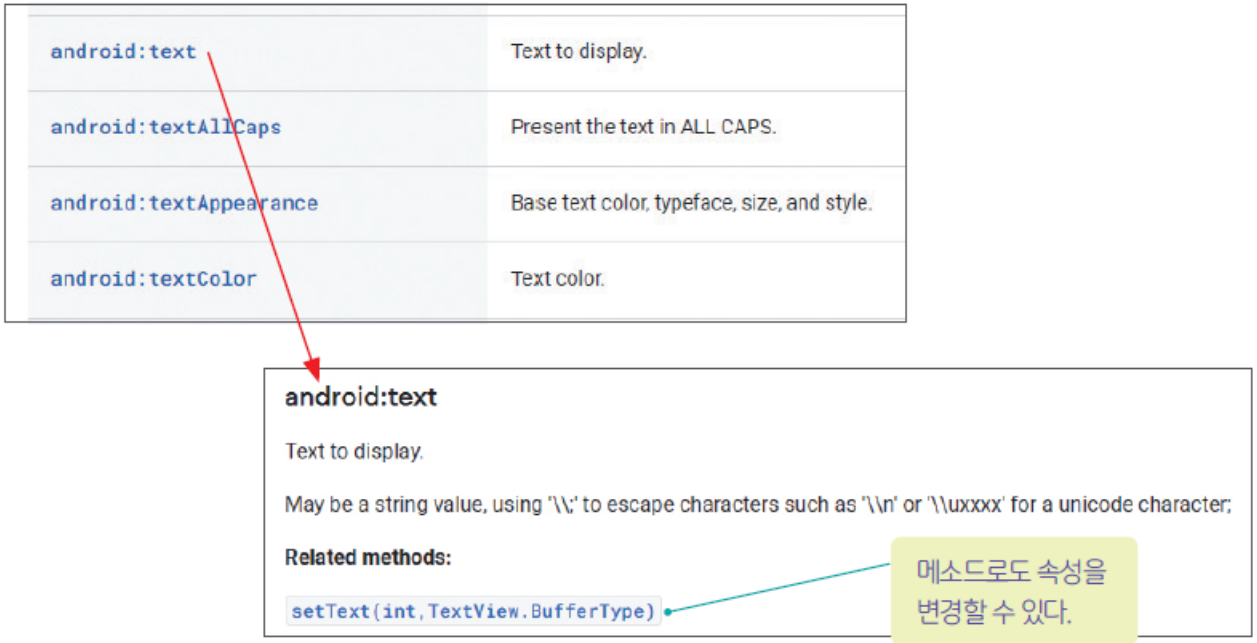
예제: 이미지 뷰어 만들기

MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    ImageView imageview;
    ImageButton button1, button2;

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

        imageview = (ImageView) findViewById(R.id.imageView);
        button1 = (ImageButton) findViewById(R.id.imageButton);
        button2 = (ImageButton) findViewById(R.id.imageButton2);
    }
    public void setImage1(View v) {
        imageview.setImageResource(R.drawable.pic);
    }
    public void setImage2(View v) {
        imageview.setImageResource(R.drawable.pic2);
    }
}
```

코드로 위젯의 속성 변경하기

- 앞에서는 레이아웃 파일을 통하여 위젯의 속성을 지정하였다. 하지만 레이아웃 속성은 코드를 통해서도 얼마든지 변경이 가능하다.



코드로 위젯의 속성 변경하기

- 예를 들어 뷰가 가지고 있는 background라는 속성은 setBackgroundResource(int)로 변경 가능하다.

▼From class android.view.View

Attribute Name	*Related Method*	*Description*
android:background	setBackgroundResource(int)	A drawable to use as the background.
android:clickable	setClickable(boolean)	Defines whether this view reacts to click events.
android:contentDescription	setContentDescription (CharSequence)	Defines text that briefly describes content of the view.
android:drawingCacheQuality	setDrawingCacheQuality(int)	Defines the quality of translucent drawing caches.
android:duplicateParentState		When this attribute is set to true, the view gets its drawable state (focused, pressed, etc.) from its direct parent rather than from itself.

메소드로도 속성을 변경할 수 있다.

예제: 코드로 텍스트 뷰 속성 변경하기 #1

예제: 코드로 텍스트 뷰 속성 변경하기 #1

activity_main.xml

```
<LinearLayout>
    <TextView
        android:id="@+id/textView"
        android:text="TextView" />
    <TextView
        android:id="@+id/textView2"
        android:text="TextView" />
    <TextView
        android:id="@+id/textView3"
        android:text="TextView" />
</LinearLayout>
```

예제: 코드로 텍스트 뷰 속성 변경하기 #1

MainActivity.java

```
package kr.co.company.textviewtest2;
public class MainActivity extends AppCompatActivity {
    TextView tv1, tv2, tv3;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tv1 = (TextView) findViewById(R.id.textView);
        tv2 = (TextView) findViewById(R.id.textView2);
        tv3 = (TextView) findViewById(R.id.textView3);

        tv1.setText("자바 코드로 변경하였습니다.");
        tv2.setTextColor(Color.BLUE);
        tv2.setTextSize(60);
        tv3.setTextSize(60);
        tv3.setTypeface(Typeface.SERIF, Typeface.ITALIC);
    }
}
```

CC: 카운터 만들어보기

- 다음과 같은 간단한 카운터 앱을 작성해보자. “카운터 증가” 버튼을 누르면 카운터가 증가한다. “카운터 감소” 버튼을 누르면 카운터가 감소한다.

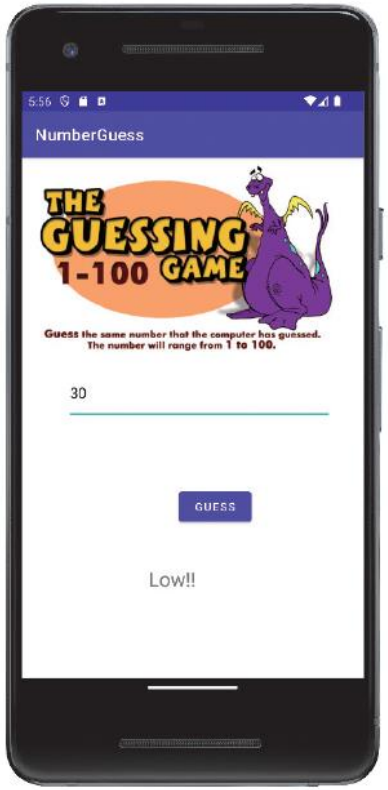
CC: 카운터 만들어보기

- 다음과 같이 버튼을 누르면 주사위가 굴려지고 화면에 주사위를 이미지로 표시하는 앱을 작성해보자.

연습 문제 10번

10 컴퓨터가 난수로 생성한 정수를 사용자가 알아 맞추는 게임을 안드로이드 앱으로 작성해보자. 컴퓨터는 1에서 100 사이의 정수를 하나 선택한다. 사용자가 정수를 예측하면 정답과 비교하여 높은지 낮은지를 사용자에게 힌트로 알려준다.

(주제: 에디트 텍스트 사용하기, 버튼 이벤트 처리, 난이도: 상)



Q & A