

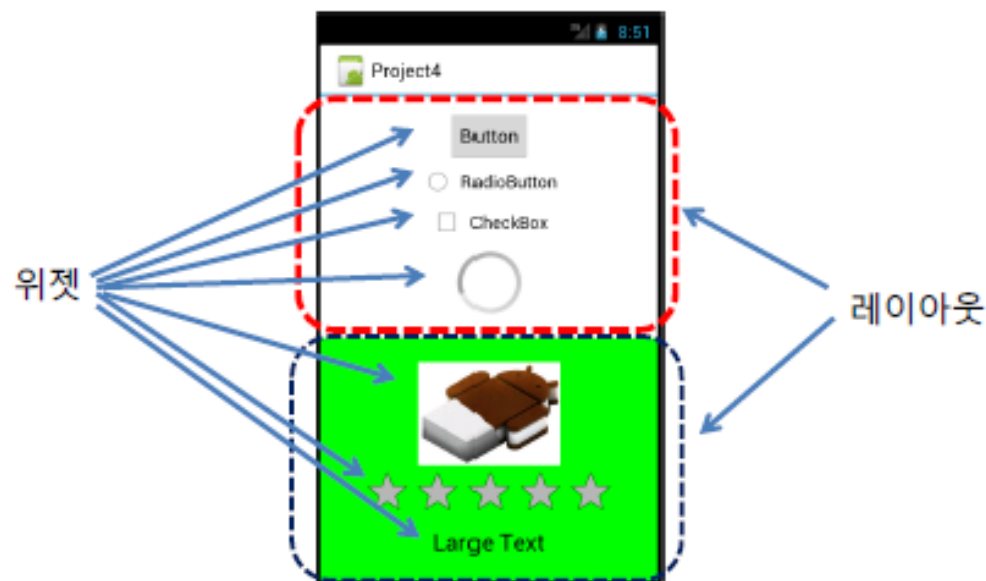


화면 구성



1. 뷰(View)의 개요

- ❖ 안드로이드 화면에서 실제로 사용되는 것은 모두 **View** 클래스에서 상속을 받으며, 이러한 클래스를 모두 '**위젯**' (또는 뷰)이라고도 부름.
- ❖ 위젯 중에서 다른 위젯을 담을 수 있는 것들은 주로 **ViewGroup** 클래스의 하위에 존재한다. 이렇게 다른 위젯을 담을 수 있는 클래스를 레이아웃이라고 함.



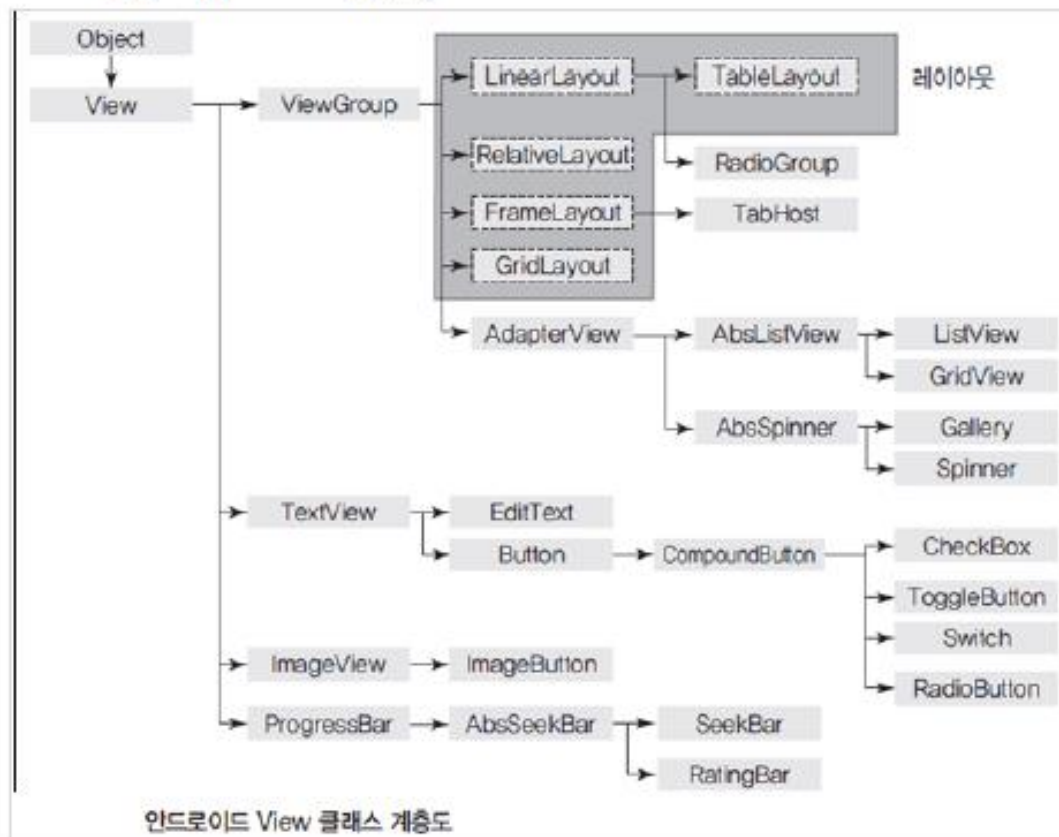
위젯 종류

- 버튼, 텍스트뷰, 에디트 텍스트, 라디오 버튼, 이미지뷰 등



1. 뷰(View)의 개요

❖ View 클래스 계층도



A circular logo featuring the Android robot icon. The robot is black with a pink heart on its chest and pink hearts floating around its head. The text "android" is written in a stylized font above the robot.

1. 뷰(View)의 개요

- 위젯과 레이아웃 등은 모두 **View** 클래스의 속성과 메소드를 상속받는다.
- 버튼과 텍스트 예)



```
<Button  
    android:id="@+id/button01"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Hello World!" />
```

1. 뷰(View)의 개요



<Button

```
android:id="@+id/button01"  
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:text="Hello World!" />
```

<TextView

```
android:id="@+id/button01"  
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:text="Hello World!" />
```



○ id 속성

- id속성은 모든 위젯의 아이디를 나타내는데, 주로 Java 코드에서 버튼 등의 위젯에 접근할 때 id속성에 지정한 아이디를 사용
- **android:id="@+id/btn1"**
(의미- 버튼 위젯의 아이디를 btn1으로 부여한 것)
- 위젯에 접근하기 위해서 Java 코드와 버튼의 예

위젯 변수 = (위젯 형) findViewById(R.id.위젯id);

```
Button button1;  
button1 = (Button) findViewById(R.id.btn1);
```



❖ id 속성 예제

id 속성 XML 코드

```
1 <LinearLayout ~~~ 중간 생략 ~~~
2   android:orientation="vertical" >
3   <TextView
4     android:id="@+id/textView1"
5     android:layout_width="wrap_content"
6     android:layout_height="wrap_content"
7     android:text="성별 선택" />
8   <RadioButton
9     android:id="@+id/female"
10    android:layout_width="wrap_content"
11    android:layout_height="wrap_content"
12    android:text="여성" />
13  <RadioButton
14    android:id="@+id/male"
15    android:layout_width="wrap_content"
16    android:layout_height="wrap_content"
17    android:text="남성" />
18 </LinearLayout>
```

id 무방

id 필요

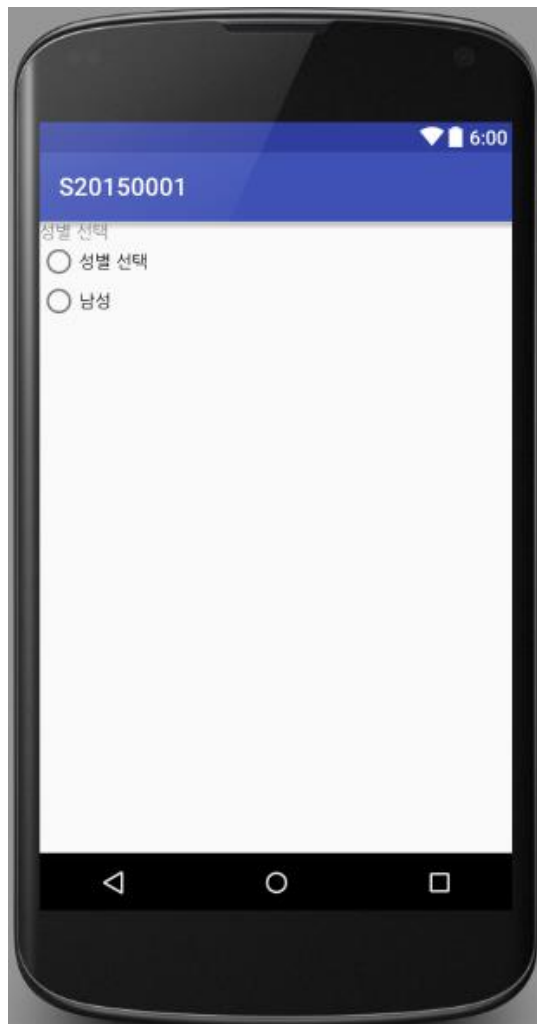
id 필요

성별 선택

☐ 여성

☒ 남성

id는 어떠한 동작을 위한 것으로 단순 텍스트나 배경이미지를 지정하는 경우에는 필요 없음(무방)



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

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="성별 선택" />

    <RadioButton
        android:id="@+id/female"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="성별 선택" />

    <RadioButton
        android:id="@+id/male"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="남성" />

</LinearLayout>
```

○ layout_width, layout_height 속성

- 속성값 match_parent(=fill_parent)는 이름 그대로 자신의 부모에 폭이나 높이를 맞춘다는 의미
- 속성값 wrap_content는 자신의 폭이나 높이를 자신 안의 글자가 꼭 들어갈 정도로 한다는 의미

The screenshot displays the Android Studio environment. On the left, the 'activity_main.xml' file is open, showing the following XML code:

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

    <Button
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼입니다." />

</LinearLayout>
```

On the right, the preview window shows a visual representation of the layout. It features a blue header bar with the text 'S20150001' and a grey button with the text '버튼입니다.' (Button).

○ 뷰의 위치와 크기

상수	설명
<code>fill_parent</code>	부모의 크기를 꽉 채운다. (<code>match_parent</code> 도 같은 의미)
<code>wrap_content</code>	뷰가 나타내는 내용물의 크기에 맞춘다.
숫자	크기를 정확히 지정한다.

```
<Button
```

```
    android:id="@+id/button1"
```

```
    android:layout_width="wrap_content"
```

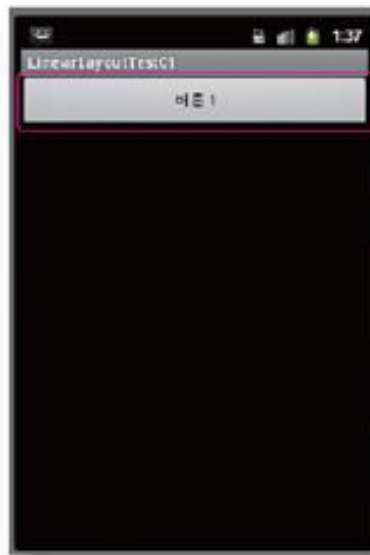
```
    android:layout_height="wrap_content"
```

```
    android:text="@string/button_text" />
```

버튼의 위치와 크기를
결정한다.



`layout_width = "wrap_content"`
`layout_height = "wrap_content"`



`layout_width = "fill_parent"`
`layout_height = "wrap_content"`



`layout_width = "wrap_content"`
`layout_height = "fill_parent"`



`layout_width = "fill_parent"`
`layout_height = "fill_parent"`

- 뷰의 실제 위치
 - `getLeft()`, `getTop()`
- 뷰의 실제 너비-`getWidth()`, 뷰의 실제 높이-`getHeight()`
- 안드로이드에서 사용할 수 있는 단위(px, dp, sp, in, mm, pt)
 - 예) 200픽셀의 경우, "200px"

단위	설명
px(pixels)	화면의 실제 픽셀을 나타낸다. 픽셀은 권장되는 단위는 아니데, 그 이유는 장치마다 화면의 밀도가 다르기 때문이다.
dp(density-independent pixels)	dp는 화면의 밀도가 160dpi 화면에서 하나의 물리적인 픽셀을 말한다. 따라서 크기를 160dp로 지정하면 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 추천
pt(points)	1/72 인치를 표시한다.
mm(millimeters)	밀리미터를 나타낸다.
In(inches)	인치를 나타낸다.

❖ background 속성

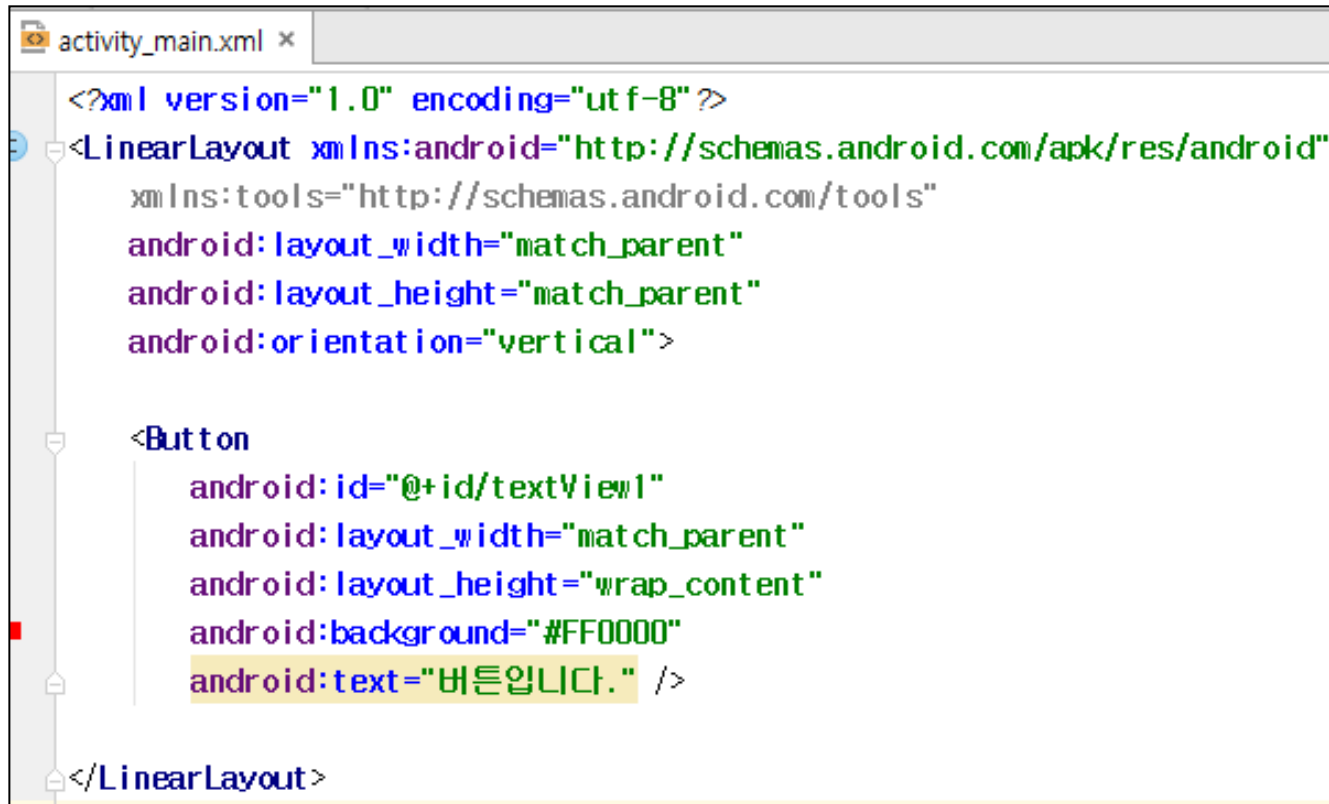
- 위젯의 색상을 주로 RGB 값으로 설정("#RRGGBB" 값으로 지정)
- "#AARRGGBB"로 표현이 가능
(AA는 투명도로서, 00-FF 까지 설정가능 00(투명), FF(불투명))

❖ 예제

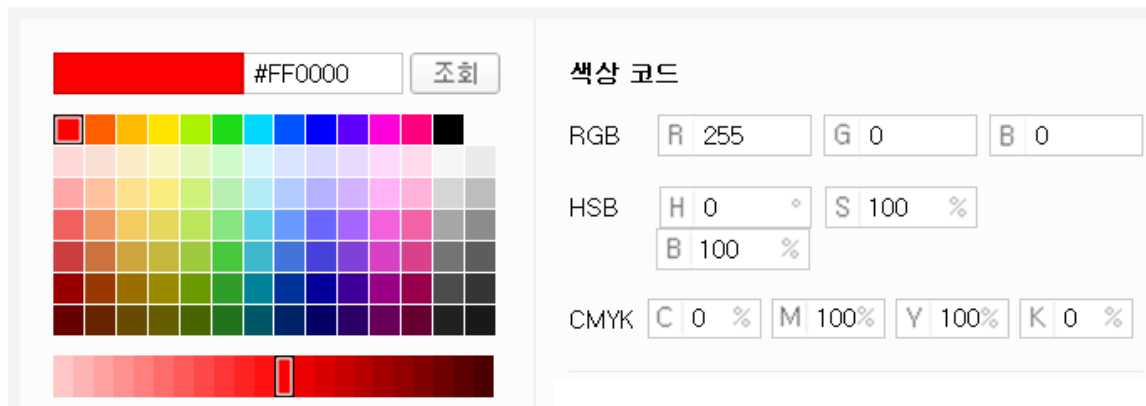
background 속성 XML 코드

```
1 <LinearLayout
2   android:background="#ff0000"
3   ~~~~ 중간 생략 ~~~~ >
4   <Button
5     android:layout_width="wrap_content"
6     android:layout_height="wrap_content"
7     android:background="#00ff00"
8     android:text="버튼입니다" />
9 </LinearLayout>
```





색상 팔레트



padding 속성

- padding은 위젯의 경계선으로부터 위젯 안의 내용물이 좀 떨어지도록 설정

dp는 해상도에 따라 크기 유지

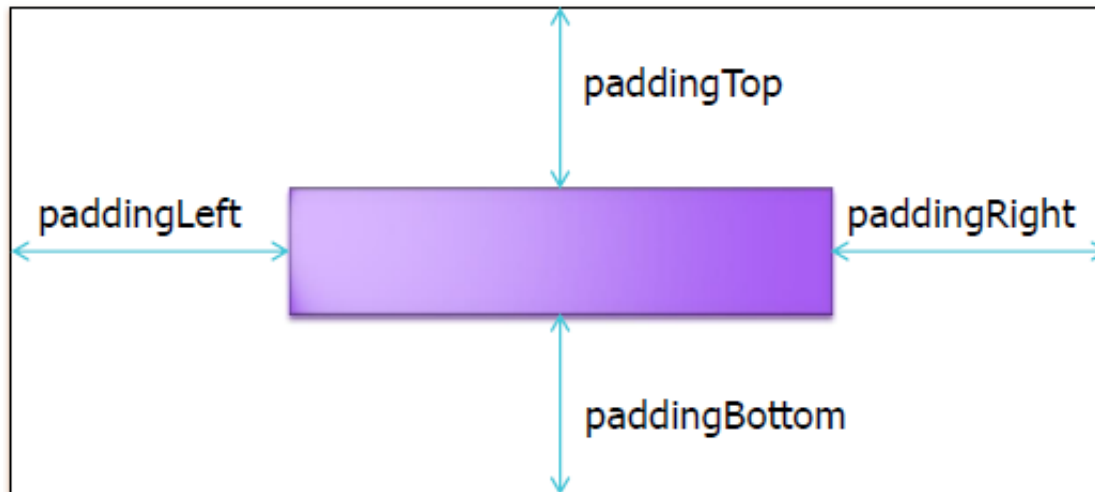
padding 속성 XML 코드

```
1 <LinearLayout
2   android:padding="30dp" >
3   <TextView
4     android:text="아래에 이름을 입력 :" />
5   <EditText
6     android:hint="여기에 채우세요" />
7   <Button
8     android:text="확인" />
9 </LinearLayout>
```

아래에 이름을 입력 :
여기에 채우세요
확인



- Padding은 상하좌우 모두 지정하는 속성으로 개별 설정
 - ⊙ `paddingTop`, `paddingBottom`, `paddingLeft`, `paddingRight`
- 패딩을 메소드에서 설정(java code)
 - ⊙ `setPadding(int, int, int, int)`
- 설정된 패딩 값 확인(java code)
 - ⊙ `getPaddingLeft()`, `getPaddingTop()`, `getPaddingRight()`, `getPaddingBottom()`



○ layout_margin 속성

- 위젯과 위젯 사이를 여유 있게 하고 싶다면 layout_margin 속성을 사용

layout_margin 속성 XML 코드

```
1 <LinearLayout
2     android:padding="30dp" >
3     <TextView
4         android:layout_margin="20dp"
5         android:text="아래에 이름을 입력 :" />
6     <EditText
7         android:layout_margin="20dp"
8         android:hint="여기에 채우세요" />
9     <Button
10        android:layout_margin="20dp"
11        android:text="확인" />
12 </LinearLayout>
```

아래에 이름을 입력 :

여기에 채우세요

확인



padding과 layout_margin

- **padding**은 자신(레이아웃)의 내부에 들어 있는 위젯과 자신의 경계선과의 간격을 지정
- **layout_magin**은 자신과 부모 레이아웃이나 위젯과의 간격이나 주위의 다른 위젯과의 간격을 지정

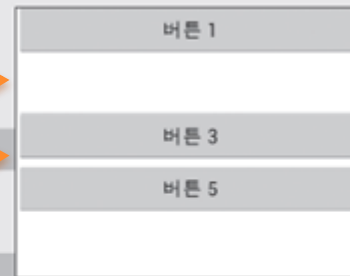
○ visibility 속성

- 위젯을 보일 것인지 여부를 결정하는 속성
- visible은 보이는 상태, invisible과 gone은 안 보임.

- **invisible** 은 자리는 확보하나 보이지 않음.
- **gone** 는 자리도 차지하지 않고 보이지도 않음

visibility 속성 XML 코드

```
1 <Button  
2     android:text="버튼 1" />  
3 <Button  
4     android:visibility="invisible"  
5     android:text="버튼 2" />  
6 <Button  
7     android:visibility="visible"  
8     android:text="버튼 3" />  
9 <Button  
10    android:visibility="gone"  
11    android:text="버튼 4" />  
12 <Button  
13    android:text="버튼 5" />
```



activity_main.xml

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

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼1" />

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼2"
        android:visibility="invisible"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼3"
        android:visibility="visible"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼4"
        android:visibility="gone"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼5" />

</LinearLayout>
```



○ enabled, clickable 속성

- 위젯의 작동 여부 는 enabled 속성
- 클릭이나 터치가 가능하도록 하는 것은 clickable 속성
- True나 false로 설정(디폴트는 true)

○ 예제

enabled 및 clickable 속성 XML 코드

```
1 <Button  
2   android:text="버튼 1" />  
3 <Button  
4   android:enabled="false"  
5   android:text="버튼 2" />  
6 <Button  
7   android:clickable="false"  
8   android:text="버튼 3" />
```

동작(x)

클릭이나 터치(x)





○ rotation 속성

- 위젯을 회전시켜서 출력
- Android 3.0 부터 지원됨.

○ 예제

rotation 속성 XML 코드

```
1 <Button  
2     android:text="버튼 1" />  
3 <Button  
4     android:text="버튼 2" />  
5 <Button  
6     android:rotation="45"  
7     android:text="버튼 3" />
```



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

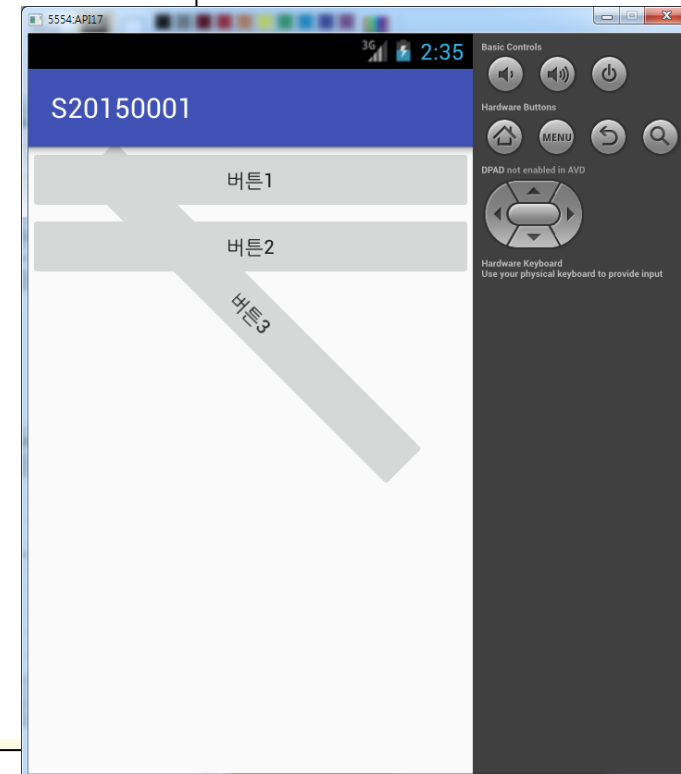
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼1" />

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="버튼2" />

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:rotation="45"
        android:text="버튼3" />

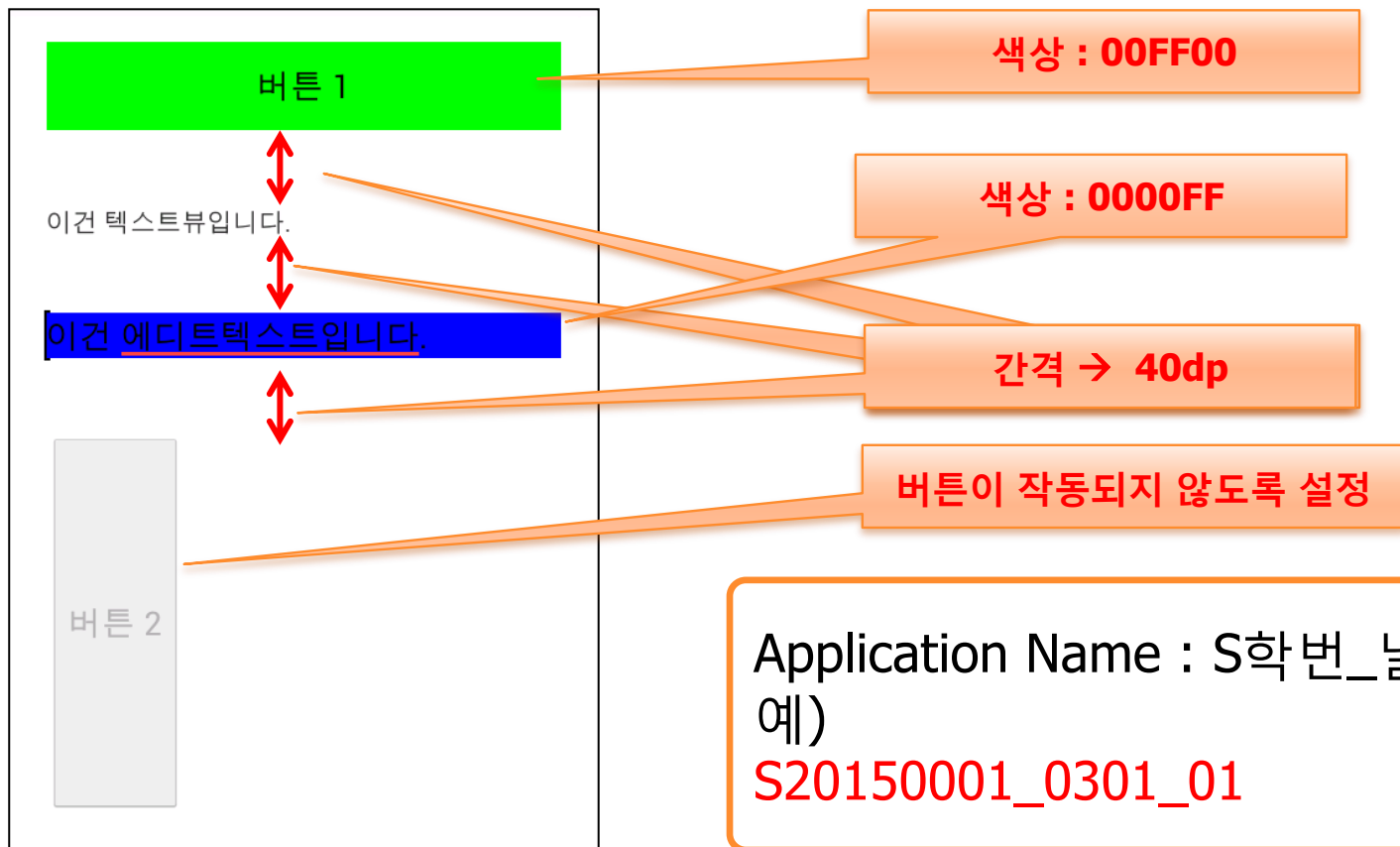
</LinearLayout>
```

프로젝트를 실행해야
회전 기능 적용됨.



연습 문제

- 다음 그림과 같은 XML 코드를 코딩하자. 버튼, 텍스트뷰, 에디트텍스트, 버튼 차례로 지정한다. 또 앞에서 배운 다양한 속성을 지정해 본다.



main.xml

```
1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2   xmlns:tools="http://schemas.android.com/tools"
3   android:layout_width="fill_parent"
4   android:layout_height="fill_parent"
5   android:orientation="vertical" >
6   <Button
7       android:layout_width="match_parent"
8       android:layout_height="wrap_content"
9       android:layout_margin="20dp"
10      android:background="#00ff00"
11      android:text="버튼1" />
12   <TextView
13       android:layout_width="match_parent"
14       android:layout_height="wrap_content"
15       android:layout_margin="20dp"
16       android:text="이건 텍스트뷰입니다." />
17   <EditText
18       android:layout_width="match_parent"
19       android:layout_height="wrap_content"
20       android:layout_margin="20dp"
21       android:background="#0000ff"
22       android:text="이건 에디트 텍스트입니다." />
23   <Button
24       android:layout_width="wrap_content"
25       android:layout_height="match_parent"
26       android:layout_margin="20dp"
27       android:enabled="false"
28       android:text="버튼2" />
29 </LinearLayout>
```


2. 기본 위젯 다루기

○ 계층도

```
java.lang.Object
```

```
└ android.view.View
```

```
└ android.widget.TextView
```

TextView 계층도

- 텍스트뷰(TextView): 화면에 간단한 텍스트를 출력하는 뷰
- 텍스트뷰 하위 에디트텍스트, 버튼, 체크박스 등

2. 기본 위젯 다루기



❖ 관련 속성

속성	설정 메소드	설명
Text	setText(CharSequence)	표시할 텍스트
textColor	setTextColor(ColorStateList)	텍스트 색상(#RRGGBB, #AARRGGBB)
textSize	setTextSize(float)	텍스트의 크기(단위-dp/px/in/mm/sp)
textStyle	setTextStyle(TextStyle)	텍스트 스타일(bold , <i>italic</i> , bolditalic)
Typeface	setTypeface(Typeface)	텍스트 폰트(normal(디폴트)/, sans, serif, monospace)
Width	setWidth(int)	픽셀 단위로 텍스트뷰의 길이를 설정
Height	setHeight(int)	픽셀 단위로 텍스트뷰의 높이를 설정
lines	setLines(int)	텍스트뷰의 높이를 설정
singleLine	setSingleLine()	한 줄에 출력(디폴트 false)



글자 관련 속성 XML 코드

```

1  <TextView
2      android:textSize="30dp"
3      android:text="textSize 속성" />
4  <TextView
5      android:textSize="30dp"
6      android:textColor="#00FF00"
7      android:text="textColor 속성" />
8  <TextView
9      android:textSize="30dp"
10     android:textStyle="bold|italic"
11     android:text="textStyle 속성" />
12 <TextView
13     android:textSize="30dp"
14     android:typeface="serif"
15     android:text="typeface 속성" />
16 <TextView
17     android:textSize="30dp"
18     android:singleLine="true"
19     android:text="singleLine 속성 singleLine 속성" />

```

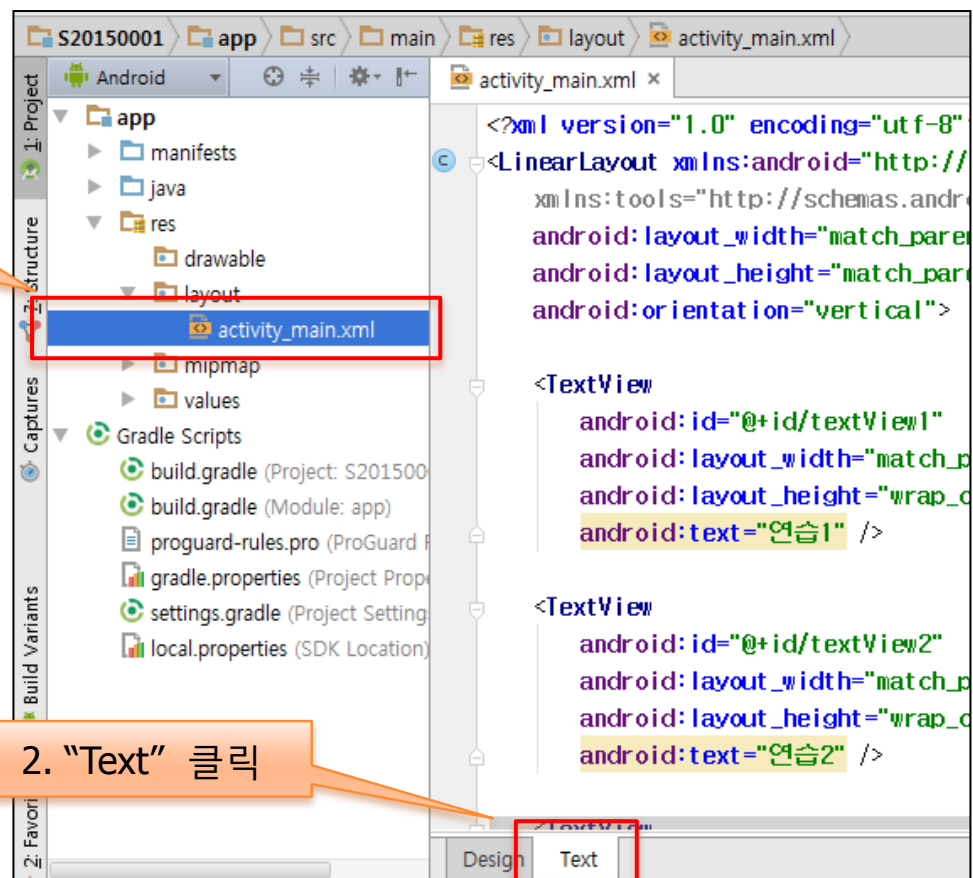
textSize 속성
 textColor 속성
 textStyle 속성
 typeface 속성
 singleLine 속성 single...

```
1 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
2   xmlns:tools="http://schemas.android.com/tools"
3   android:layout_width="fill_parent"
4   android:layout_height="fill_parent"
5   android:orientation="vertical" >
6   <TextView
7       android:layout_width="wrap_content"
8       android:layout_height="wrap_content"
9       android:text="textSize 속성"
10      android:textSize="30dp" />
11   <TextView
12       android:layout_width="wrap_content"
13       android:layout_height="wrap_content"
14       android:text="textColor 속성"
15       android:textColor="#00FF00"
16       android:textSize="30dp" />
17   <TextView
18       android:layout_width="wrap_content"
19       android:layout_height="wrap_content"
20       android:text="textStyle 속성"
21       android:textSize="30dp"
22       android:textStyle="bold|italic" />
23   <TextView
24       android:layout_width="wrap_content"
25       android:layout_height="wrap_content"
26       android:text="typeface 속성"
27       android:textSize="30dp"
28       android:typeface="serif" />
29   <TextView
30       android:layout_width="wrap_content"
31       android:layout_height="wrap_content"
32       android:singleLine="true"
33       android:text="singleLine 속성 singleLine 속성"
34       android:textSize="30dp" />
35 </LinearLayout>
```

코딩

Application Name : S학번_날짜_순번
예)
S20150001_0301_01

1. [layout]→
[activity_main.xml] 클릭



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

    <TextView
        android:id="@+id/textView1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="연습1" />

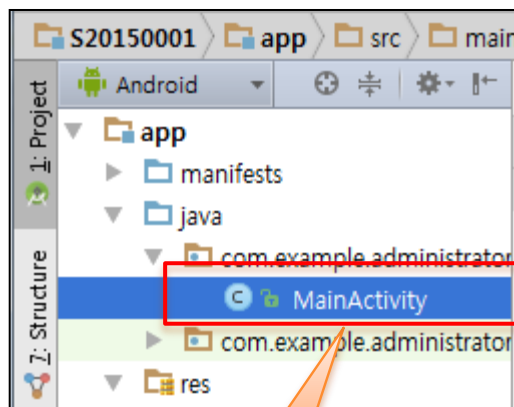
    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="연습2" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="연습3" />

</LinearLayout>
```

3. 코딩





```
1 package com.example.administrator.s20150001;
2
3
4 import android.os.Bundle;
5 import android.support.v7.app.AppCompatActivity;
6 import android.widget.TextView;
7
8 public class MainActivity extends AppCompatActivity {
9
10     @Override
11     protected void onCreate(Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         setContentView(R.layout.activity_main);
14         TextView tv1, tv2, tv3;
15     }
16 }
```

자동 import 설정의
경우 import가 자동
으로 추가됨

5. 코딩

```
1 package com.example.administrator.s20150001;
2
3 import android.graphics.Color;
4 import android.graphics.Typeface;
5 import android.os.Bundle;
6 import android.support.v7.app.AppCompatActivity;
7 import android.widget.TextView;
8
9 public class MainActivity extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_main);
15         TextView tv1, tv2, tv3;
16         tv1 = (TextView)findViewById(R.id.textView1);
17         tv2 = (TextView)findViewById(R.id.textView2);
18         tv3 = (TextView)findViewById(R.id.textView3);
19
20         tv1.setText("안녕하세요?");
21         tv2.setTextColor(Color.RED);
22         tv3.setTypeface(Typeface.SERIF, Typeface.BOLD_ITALIC);
23         tv3.setText("가나다라마바라아자차카타마파하" +
24             "ABCDEFGHIJKLMNOPQRSTUVWXYZ");
25         tv3.setSingleLine();
26     }
27 }
```

TextView 선언

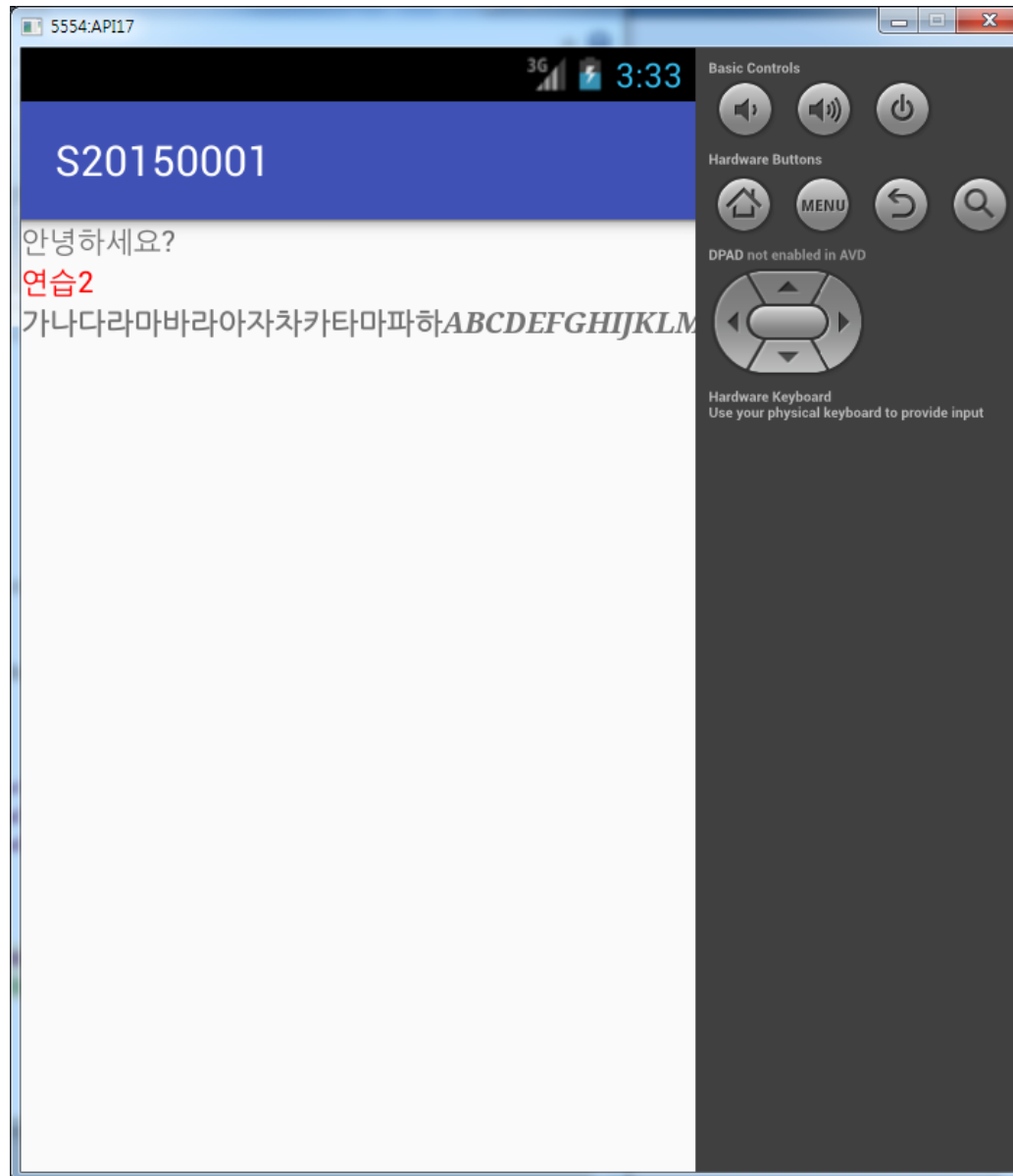
Acvitivity_main.xml에서 생성한
TextView

텍스트 색상 지정

텍스트 글꼴 지정

텍스트 지정

텍스트를 한 줄로 출력(디폴트
false)



Java 코드로 XML 속성 지정

○ XML 속성과 관련 메소드

XML 속성	관련 메소드	비고
background	setBackgroundColor()	View 클래스
clickable	setClickable()	View 클래스
focusable	setFocusable()	View 클래스
id	setId()	View 클래스
longClickable	setLongClickable()	View 클래스
padding	setPadding()	View 클래스
rotation	setRotation()	View 클래스
scaleX, scaleY	setScaleX(), setScaleY()	View 클래스
visibility	setVisibility()	View 클래스
gravity	setGravity()	TextView 클래스
inputType	setRawInputType()	TextView 클래스
password	setTransformationMethod()	TextView 클래스
text	setText()	TextView 클래스
textColor	setTextColor()	TextView 클래스
textSize	setTextSize()	TextView 클래스

버튼과 에디트텍스트

❖ 텍스트뷰와 버튼의 호환

<TextView

```
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:text="나는 어떤 위젯일까요?" />
```

텍스트뷰를 버튼으로 변환

<Button

```
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:text="나는 어떤 위젯일까요?" />
```



❖ 버튼 계층도

```
java.lang.Object
    ↳ android.view.View
        ↳ android.widget.TextView
            ↳ android.widget.Button
```

Button 계층도

❖ 버튼 XML과 Java 코드 작성 3단계

Button을 사용하기 위한 3단계

```
<Button
    android:id="@+id/button1"
    android:text="확인" />
```

XML 파일



```
① 버튼 변수 선언
    Button mybutton;

② 변수에 버튼 위젯 대입
    mybutton = (Button) findViewById(R.id.button1);

③ 버튼 클릭할 때 동작하는 클래스 정의
    mybutton.setOnClickListener( new View.OnClickListener() {
        public void onClick(View v) {
            // 동작할 내용을 코딩
        }
    });
```

Java 파일

❖ 에디트텍스트 계층도

```
java.lang.Object
    ↳ android.view.View
        ↳ android.widget.TextView
            ↳ android.widget.EditText
```

EditText 계층도

❖ 에디트텍스트 XML과 Java 코드 작성 3단계

```
<EditText
    android:id="@id/edittext1" />
```

① 에디트텍스트 변수 선언

```
EditText myEdit;
```

② 변수에 에디트텍스트 위젯 대입

```
myEdit = (EditText) findViewById(R.id.edittext1);
```

③ 에디트텍스트에 입력된 값을 가져오기

```
String myStr = myEdit.getText( ).toString( );
```

EditText에 입력된 값을 문자열로 변환하여 받아온다.

○ 에디트 텍스트

- EditText는 입력이 가능한 텍스트

속성	설명
android:autoText	자동 철자 교정 기능 사용 여부
android:capitalize	true이면 문장의 첫 글자를 대문자로 변경
android:digits	숫자만 받도록 설정
android:password	암호를 입력 받는다. (●●●●●)
android:phoneNumber	전화번호를 입력 받는다.
android:singleLine	True이면 한 줄만 받음

○ 에디트 텍스트

main.xml

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

```
<EditText
    android:id="@+id/edit1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:text="여기에 입력할 수 있습니다. " />
```

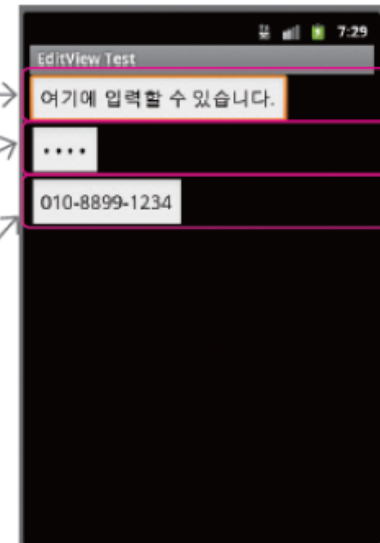
한줄만 입력 가능

```
<EditText
    android:id="@+id/edit2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:password="true"
    android:text="패스워드 " />
```

입력한 내용이 보이지 않는다.

```
<EditText
```

전화번호만 입력 가능



```
<EditText
```

```
    android:id="@+id/edit3"
```

```
    android:layout_width="wrap_content"
```

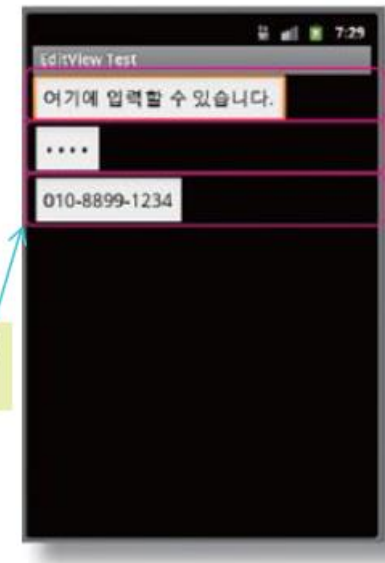
```
    android:layout_height="wrap_content"
```

```
    android:phoneNumber="true"
```

```
    android:text="010-8899-1234"/>
```

```
</LinearLayout>
```

전화번호만 입력
가능




```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <EditText
        android:id="@+id/edit1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:singleLine="true"
        android:text="여기는 입력할 수 있습니다." />

    <EditText
        android:id="@+id/edit2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:password="true"
        android:text="패스워드" />

    <EditText
        android:id="@+id/edit3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:phoneNumber="true"
        android:text="010-8899-1234" />

</LinearLayout>
```

