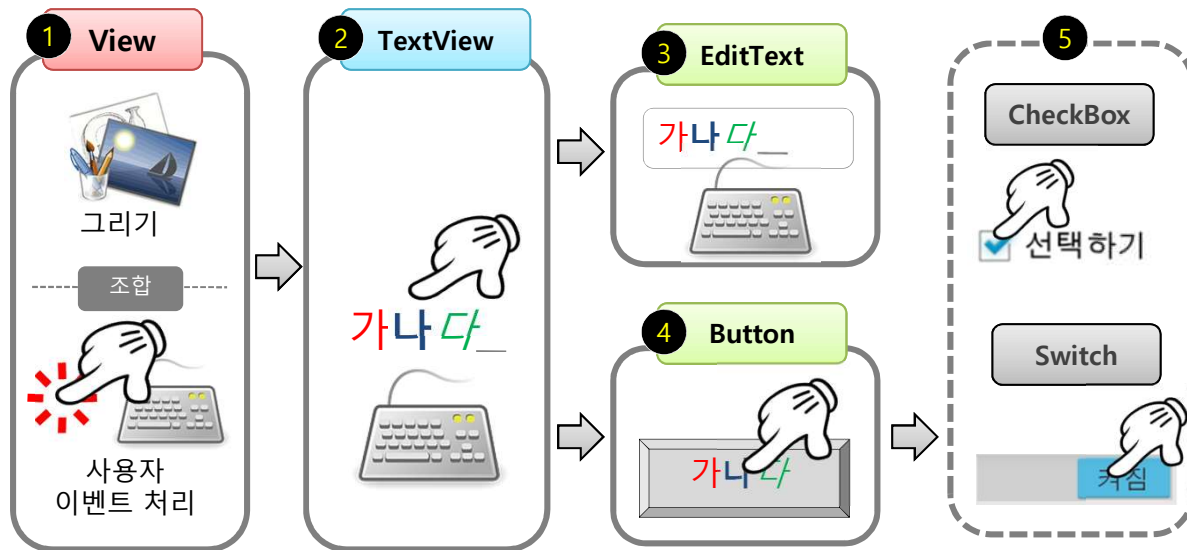
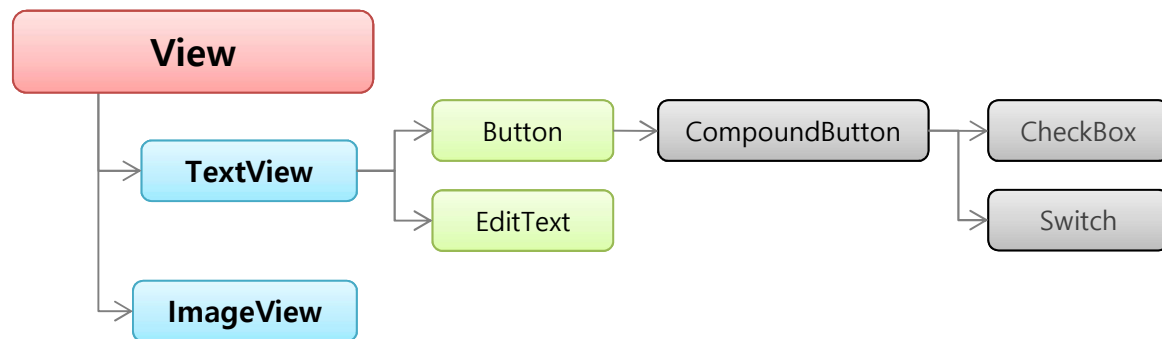


대표적인 뷰

# 최상위 뷰의 파생 클래스

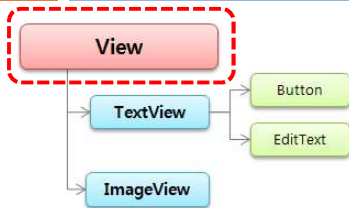
2

뷰의 상속 관계를 확인하면 특정 뷰의 특징을 빠르게 파악할 수 있다.



# 최상위 뷰 – android:id 속성

3



res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">
```

```
<TextView android:id="@+id/id_test_view"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="superdroid" />
```

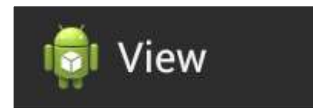
```
</LinearLayout>
```

필수 속성 X

src/MainActivity.java

```
public class MainActivity extends AppCompatActivity
{
    @Override
    protected void onCreate( Bundle savedInstanceState )
    {
        super.onCreate( savedInstanceState );
        setContentView( R.layout. activity_main );

        TextView textView = (TextView) findViewById( R.id.id_test_view );
        textView.setText( "Hello World!" );
    }
}
```



Hello World!

# 최상위 뷰 – android:background 속성

4

res/layout/activity\_main.xml

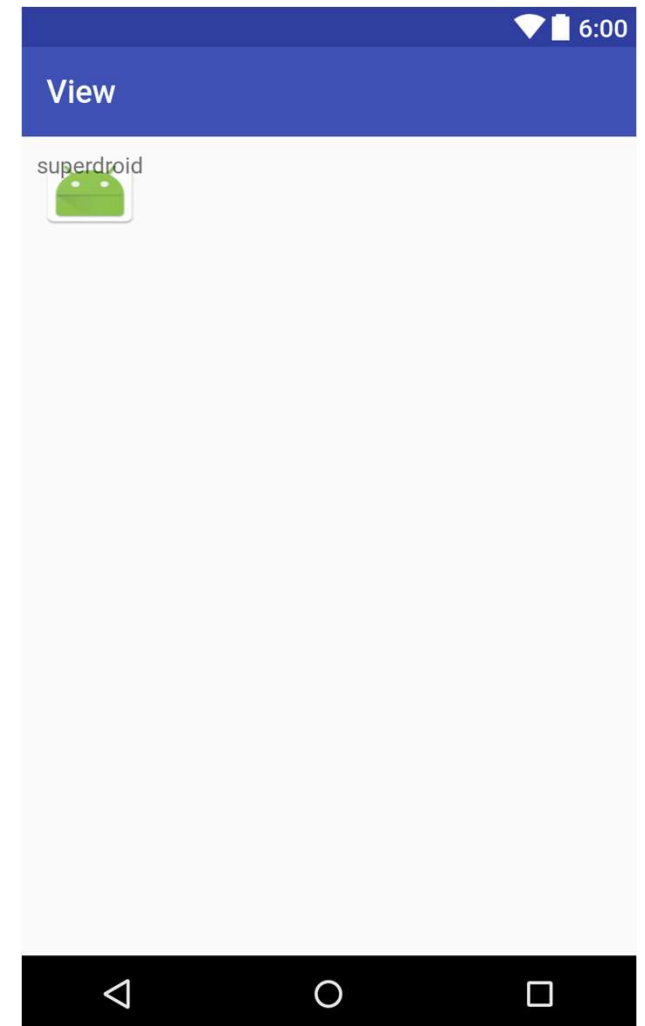
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="superdroid"
        android:background="@mipmap/ic_launcher"/>

</LinearLayout>
```



원본 이미지



# 최상위 뷰 - 패딩 속성

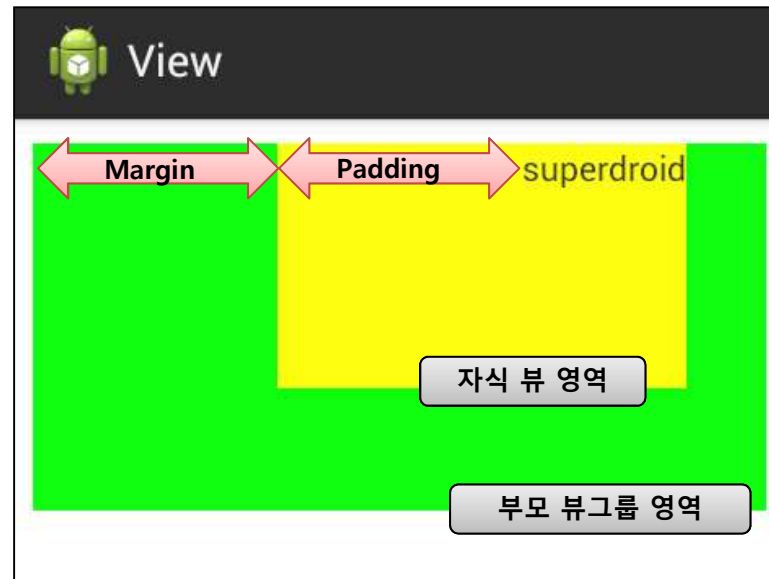
5

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."  
    android:layout_width="match_parent"  
    android:layout_height="150dp"  
    android:background="#0F0"  
    android:layout_margin="10dp"  
    android:orientation="vertical">
```

```
    <TextView android:layout_width="wrap_content"  
        android:layout_height="100dp"  
        android:text="superdroid"  
        android:layout_marginLeft="100dp"  
        android:paddingLeft="100dp"  
        android:background="#FF0"/>
```

```
</LinearLayout>
```



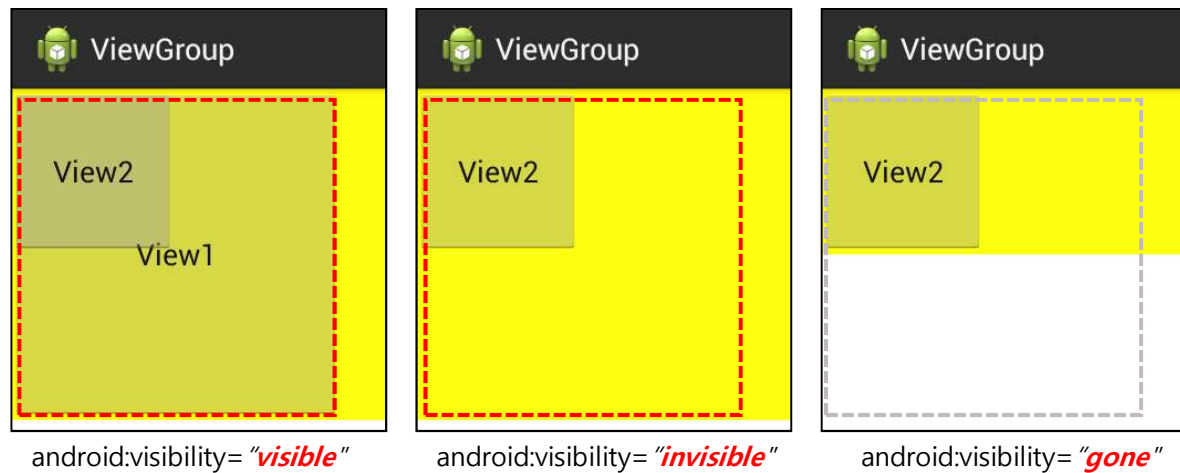
## ■ 패딩과 관련된 속성

- android:paddingTop : 뷰의 상단 여백
- android:paddingBottom : 뷰의 하단 여백
- android:paddingLeft : 뷰의 좌측 여백
- android:paddingRight : 뷰의 우측 여백
- android:padding : 뷰의 상하좌우 여백

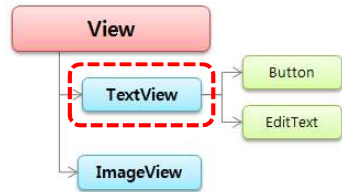
마진 속성은 LayoutParams의 속성

# 최상위 뷰 – android:visibility 속성

6



# 텍스트뷰 - 텍스트 내용과 글꼴에 관련된 속성



## ■ android:text 속성

문자열을 입력하면 화면에 출력된다. 텍스트뷰의 가장 핵심적인 기능이라 볼 수 있다.

## ■ android:textColor 속성

출력되는 텍스트의 색상을 지정할 수 있다. textColor 속성값은 빨강(R), 초록(G), 파랑(B)의 조합으로 모든 색상을 표현한다. 그뿐만 아니라 투명도(A)를 뜻하는 알파값도 지정할 수 있다. 16진수의 수로 표현이 가능하며 다음 형식도 사용할 수 있다.

1 Red (빨강) 0~F    Green (초록) 0~F    Blue (파랑) 0~F    #RGB

## ■ Alpha 값

값이 커질수록 불투명함.

2 Alpha (투명도) 0~F    Red (빨강) 0~F    Green (초록) 0~F    Blue (파랑) 0~F    #ARGB

3 Red (빨강) 00~FF    Green (초록) 00~FF    Blue (파랑) 00~FF    #RRGGBB

4 Alpha (투명도) 00~FF    Red (빨강) 00~FF    Green (초록) 00~FF    Blue (파랑) 00~FF    #AARRGGBB

# 텍스트뷰 - 텍스트 내용과 글꼴에 관련된 속성

## ■ android:textSize 속성

텍스트 크기를 설정하는 속성이다.

## ■ android:textStyle 속성

텍스트의 두 가지 스타일을 지정하는 속성이다.

- bold : 글자를 두껍게 표시한다.
- italic : 글자를 기울여서 표시한다

두 가지 속성값은 '|' 기호를 이용하여 조합하여 사용할 수도 있다.

## ■ android:typeface 속성

텍스트의 글자체를 설정하는 속성이다. 설정 값은 아래와 같이 세 가지가 존재한다.

- sans : 글자의 굵기가 같고 장식이 없는 글자체
- serif : 글자의 굵기가 다르고 장식이 있는 글자체
- monospace : 고정 폭의 글자체





# 텍스트뷰 - 텍스트 내용과 글꼴에 관련된 속성

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello TextView"
        android:textColor="#F00"
        android:textSize="20dp"
        android:textStyle="normal"
        android:typeface="sans" />

    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello EditText"
        android:textColor="#0F0"
        android:textSize="30dp"
        android:textStyle="italic"
        android:typeface="serif" />

    <Button android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello TextView"
        android:textColor="#00F"
        android:textSize="40dp"
        android:textStyle="bold"
        android:typeface="monospace" />

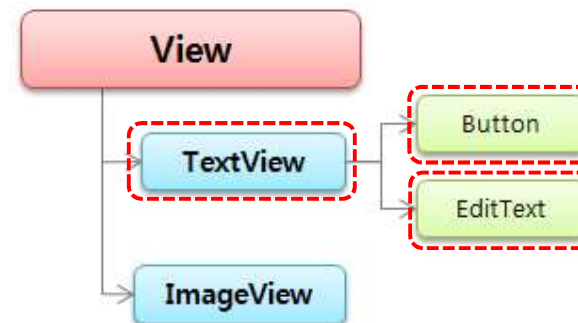
</LinearLayout>
```

 View

Hello TextView

Hello EditText

Hello  
TextView



# 텍스트뷰 – android:singleLine 속성

10

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="20dp"
        android:background="#FF0"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"/>

    <TextView android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:singleLine="true"
        android:background="#FF0"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"/>
</LinearLayout>
```



View

동해물과 백두산이 마르고  
닳도록 하느님이

동해물과 백두산이 마르고 닳...

# 텍스트뷰 – android:ellipsize 속성

```
res/layout/activity_main.xml
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="20dp"
        android:background="#FF0"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"
        android:singleLine="true"
        android:ellipsize="start"/>

    <TextView android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="20dp"
        android:background="#FF0"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"
        android:singleLine="true"
        android:ellipsize="middle"/>

    <TextView android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="20dp"
        android:background="#FF0"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"
        android:singleLine="true"
        android:ellipsize="end"/>

</LinearLayout>
```

## View

...도록 하느님이

동해물과...느님이

동해물과 백두...

### ■ ellipsize 속성값

- start : 문장의 앞 부분을 생략하고 생략기호 ...을 문장 앞에 배치
- middle : 문장의 중간 부분을 생략하고 생략기호 ...을 문장 중간에 배치
- end : 문장의 끝 부분을 생략하고 생략기호 ...을 문장 끝에 배치
- marquee : 다음 페이지에서 설명

# 텍스트뷰 -

android:ellipsize 속성값 "marquee"와 android:marqueeRepeatLimit 속성

12

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:id="@+id/textview"
        android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="20dp"
        android:background="#FF0"
        android:text="동해물과 백두산이 ..."
        android:singleLine="true"
        android:ellipsize="marquee"
        android:marqueeRepeatLimit="1"/>

</LinearLayout>
```

src/MainActivity.java

```
public class MainActivity extends Activity
{
    @Override
    protected void onCreate( Bundle savedInstanceState )
    {
        super.onCreate( savedInstanceState );

        setContentView( R.layout.activity_main );

        TextView textView = (TextView) findViewById( R.id.textview );
        textView.setSelected( true );
    }
}
```

## ■ marqueeRepeatLimit 속성값

- 횟수 상수: 글이 지정한 수만큼 반복해서 흐름
- marquee\_forever: 글이 무한정 반복해서 흐름
- 만일 marqueeRepeatLimit 속성을 설정하지 않으면 기본적으로 3번 반복해서 흐른 뒤 멈춤



View

백두산이 마르고 닳도록 하느님



좌측으로 글이 흐른다.

# 텍스트뷰 -

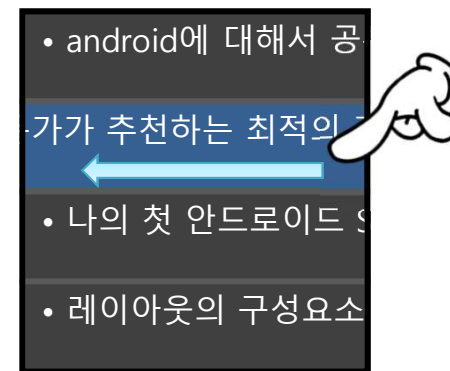
android:ellipsize 속성값 "marquee"와 android:marqueeRepeatLimit 속성

13

## ■ 왜 텍스트뷰는 선택된 상태에서만 글이 흐를까?



리스트 환경에서 전체  
아이템들이 모두 글이 흐른다.



리스트 환경에서 선택된  
아이템만 글이 흐른다.

# 텍스트뷰 – android:gravity 속성

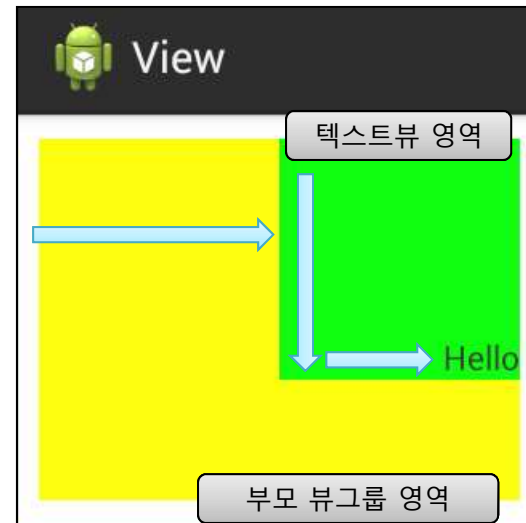
14

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."  
    android:layout_width="200dp"  
    android:layout_height="150dp"  
    android:layout_margin="10dp"  
    android:background="#FF0"  
    android:orientation="vertical">
```

```
    <TextView android:layout_width="100dp"  
        android:layout_height="100dp"  
        android:background="#0F0"  
        android:text="Hello"  
        android:layout_gravity="right"  
        android:gravity="bottom/right"/>
```

```
</LinearLayout>
```



# 텍스트뷰 -

android:lines, android:minLines, android:maxLines 속성

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="80dp"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#FF0"
        android:text="동해물과 백두산이"
        android:minLines="3"/>

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#0F0"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"
        android:maxLines="3"/>

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#00F"
        android:text="동해물과 백두산이"
        android:lines="3"/>

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#00F"
        android:text="동해물과 백두산이 마르고 닳도록 하느님이"
        android:lines="3"/>

</LinearLayout>
```



## ■ 그런데 왜 생략 기호가 표시되지 않을까?

android:singleLine 속성과는 달리 생략기호 ...이 표시되지 않았다. 텍스트뷰는 android:singleLine 속성이 없는 경우 명시적으로 android:ellipsize="end"를 설정해야 하기 때문이다. 또한 android:ellipsize 속성값은 여러 줄인 경우 "end"만 지원한다.

# 텍스트뷰 -

## android:lineSpacingExtra, android:lineSpacingMultiplier 속성

16

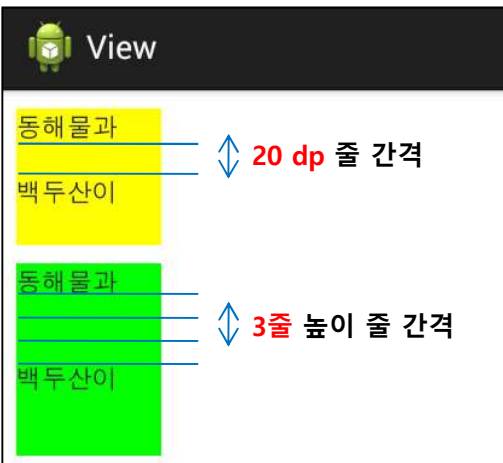
res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="80dp"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#FF0"
        android:text="동해물과 백두산이"
        android:lineSpacingExtra="20dp"/>

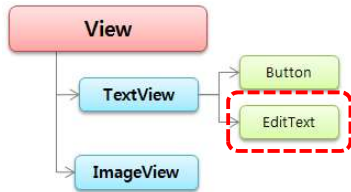
    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#0F0"
        android:text="동해물과 백두산이"
        android:lineSpacingMultiplier="3"/>

</LinearLayout>
```





# 에디트 텍스트 – android:editable, android:enabled 속성



res/layout/activity\_main.xml

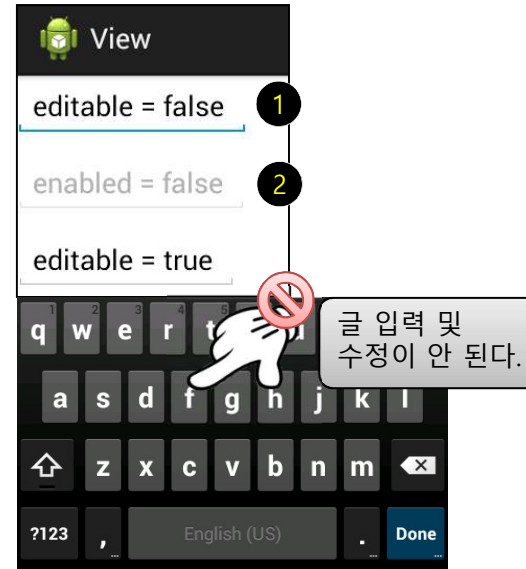
```
<LinearLayout xmlns:android="..."
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:text="editable = false "
        1 android:editable="false"/>

    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:text="enabled = false "
        2 android:enabled="false"/>

    <EditText android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="editable = true "
        3 android:editable="true"/>

</LinearLayout>
```



■ android:enabled는 View의 속성이다.

## 에디트 텍스트 – android:editable, android:enabled 속성

18

- 에뮬레이터에 소프트웨어 입력기가 나타나지 않을 때
  1. AVD Manager 실행
  2. 사용하는 Virtual Device의 “Edit this AVD” 버튼 클릭
  3. Enable keyboard input 체크 박스 해제
  4. Finish 버튼 클릭
  5. 에뮬레이터 재구동

# 에디트 텍스트 – android:digits 속성

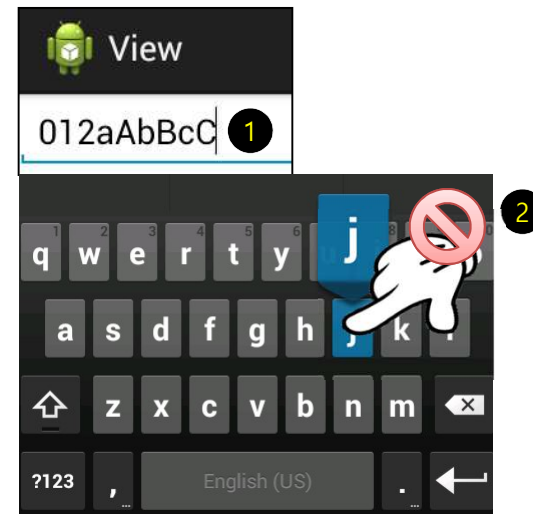
19

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <EditText android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:digits="012aAbBcC"/>

</LinearLayout>
```



# 에디트 텍스트 – android:hint, android:textColorHint

20

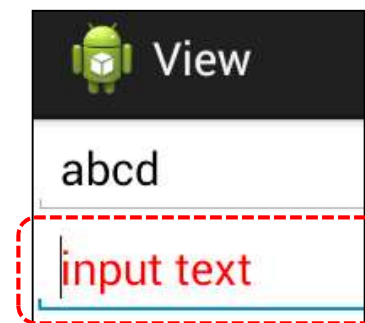
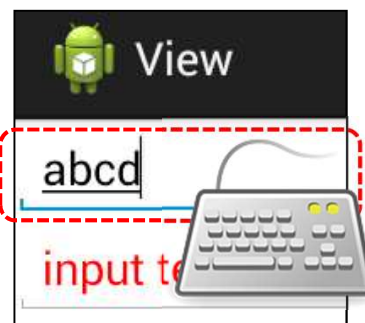
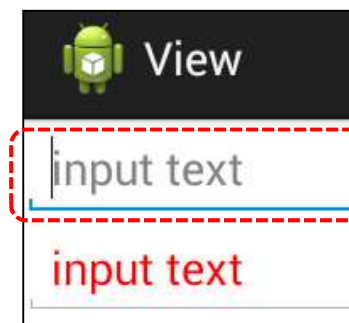
res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <EditText android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="input text"/>

    <EditText android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="input text"
        android:textColorHint="#F00"/>

</LinearLayout>
```



# 에디트 텍스트 –

android:selectAllOnFocus, android:textColorHighlight

21

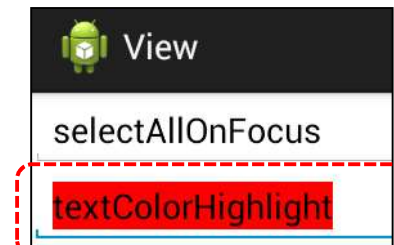
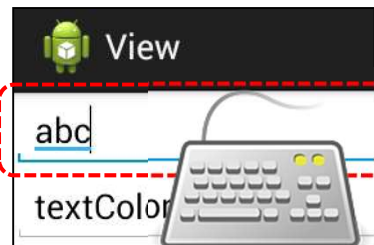
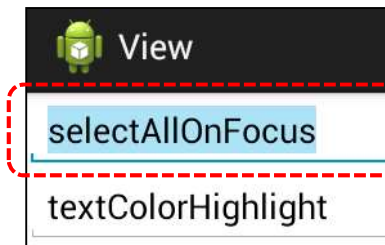
res/layout/activity\_main.xml

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

    <EditText android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="selectAllOnFocus"
        android:selectAllOnFocus="true"/>




    <EditText android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="textColorHighlight"
        android:selectAllOnFocus="true"
        android:textColorHighlight="#F00"/>

</LinearLayout>
```





# 에디트텍스트 – android:inputType 속성

22

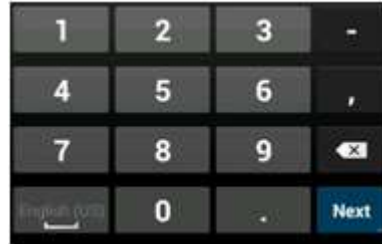
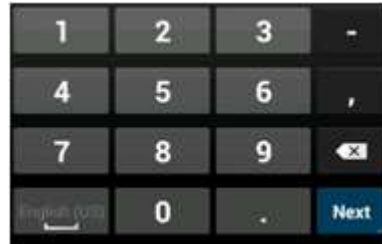

속성값	의미	적용 예
none	일반적인 입력기 모양	
text	일반적인 입력기 모양에서 Next라는 기능이 추가된다. Next 키를 누르면 현재 에디트텍스트에 있는 커서가 다음 에디트텍스트로 이동한다. (줄 바꿈 불가능)	
phone <b>phoneNumber 속성 대체</b>	전화번호를 입력하기 편리한 입력기다. (줄 바꿈 불가능)	

# 에디트 텍스트 – android:inputType 속성

23

<p>textNoSuggestions</p>	<p>특정 단어를 한 자씩 입력할 때마다 추천 단어가 상단에 표시된다. 하지만 textNoSuggestions 속성값으로 설정하면 추천어를 표시하지 않는다.</p>	
<p>number numeric 속성 대체</p>	<p>대부분 숫자 키로만 형성된 입력기가 나타난다. 해당 입력기는 양수만 입력 가능하다(소수점, 음수 입력 불가). 숫자 입력 시에는 추천어가 필요치 않으므로 생략된다. (줄 바꿈 불가능)</p>	

# 에디트 텍스트 – android:inputType 속성

numbersigned <b>numeric 속성 대체</b>	대부분 숫자 키로만 형성된 입력기가 나타난다. 양수와 음수 입력이 가능하다(소수점 입력 불가). 숫자 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않는다.	
numberDecimal <b>numeric 속성 대체</b>	대부분 숫자 키로만 형성된 입력기가 나타난다. 해당 입력기는 소수점 입력이 가능하다(음수 입력 불가). 숫자 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않는다.	
time	시간 정보를 입력하기에 적합한 입력기가 나타난다. 특히 시분초 사이에 입력하는 기호 : 키가 존재한다. 시간 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않는다.	






# 에디트 텍스트 – android:inputType 속성

date	날짜 정보를 입력하기에 적합한 입력기가 나타난다. 특히 연월일 사이에 입력하는 기호 / 키가 존재한다. 날짜 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않는다.	
datetime	날짜와 시간 정보를 입력하기에 적합한 입력기가 나타난다. 특히 연월일과 시분초 사이에 입력하는 기호 /: 키가 존재한다. 날짜와 시간 입력 시에는 추천어가 필요 없으므로 추천어는 제공되지 않는다.	
textCapCharacters capitalize 속성 대체	모든 영문이 대문자로 입력된다. 입력 전에 Caps Lock 키를 해제하면 소문자도 입력 가능하나 한 글자를 입력하고 나면 다시 Caps Lock 키가 활성화되어 대문자 입력을 유도한다.	

# 에디트 텍스트 – android:inputType 속성

<p>textCapWords  <b>capitalize 속성 대체</b></p>	<p>영문 한 단어 첫 자를 대문자로 입력한다. 입력 전에 Caps Lock 키를 해제하면 소문자도 입력 가능하나 한 단어를 입력하고 나면 다시 Caps Lock 키가 활성화되어 단어의 첫 글자 입력을 대문자로 입력하도록 유도한다.</p>	
<p>textCapSentences  <b>capitalize 속성 대체</b></p>	<p>영문 한 문장의 첫 자를 대문자로 입력한다. 입력 전에 Caps Lock 키를 해제하면 소문자도 입력 가능하나, 한 문장을 입력하고 나면 다시 Caps Lock 키가 활성화되어 문장의 첫 글자 입력을 대문자로 입력하도록 유도한다.</p>	
<p>textPassword  <b>password 속성을 대체</b></p>	<p>문자 기반의 입력기 형태이며, 입력되는 글자가 모두 • 기호로 표시된다.</p>	

# 에디트 텍스트 – android:inputType 속성

<p>numberPassword  <b>!!! 주의 API 11 부터 지원</b></p>	<p>숫자 기반의 입력기 형태이며, 입력되는 글자가 모두 • 기호로 표시된다.</p>	
<p>textEmailAddress</p>	<p>이메일 주소를 입력하기 위한 입력기가 나타난다. 특히 메일 주소에 꼭 들어가는 @와 . 기호 키가 추가된다.          이메일 주소 입력 시에는 추천어가 필요치 않으므로 생략된다.          (줄 바꿈 불가능)</p>	
<p>textShortMessage</p>	<p>SMS 메시지를 입력할 때 많이 사용되는 이모티콘 모음 키가 추가된다.</p>	

# 에디트 텍스트 – android:inputType 속성

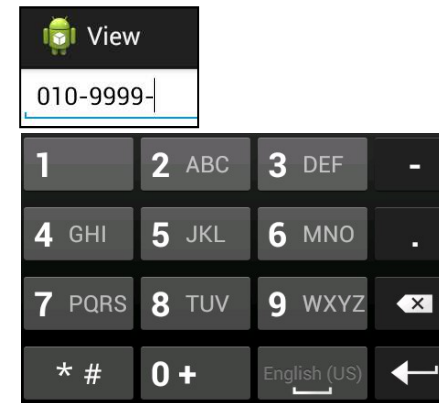
28

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <EditText android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="phone"/>

</LinearLayout>
```



# 버튼

29

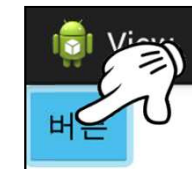
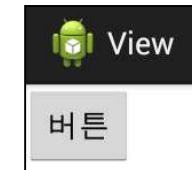
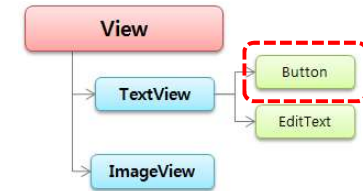
## res/layout/activity\_main.xml

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

    <Button android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="버튼"
        android:onClick="onButtonClick" />

</LinearLayout>
```

android.view.View의 속성  
※ android:clickable



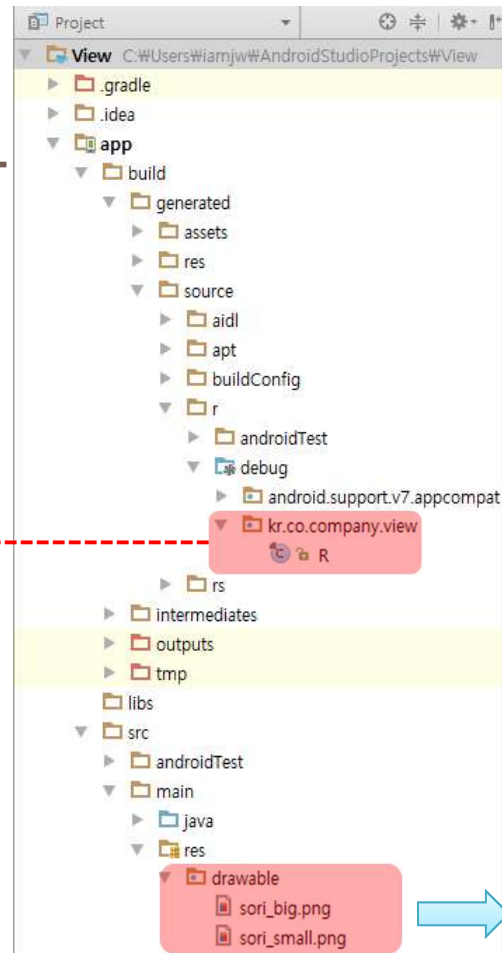
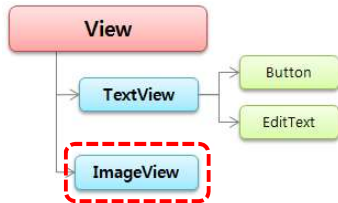
## src/MainActivity.java

```
public class MainActivity extends Activity
{
    @Override
    protected void onCreate( Bundle savedInstanceState )
    {
        super.onCreate( savedInstanceState );

        setContentView( R.layout. activity_main );
    }

    public void onButtonClick( View v )
    {
        Toast.makeText( this, "버튼을 눌렀습니다.", Toast.LENGTH_LONG ).show();
    }
}
```

# 이미지뷰



## R.java

```

package kr.co.company.View;
public final class R {
    ...
}
public static final class drawable {
    public static final int ic_launcher=0x7f020000;
    public static final int sori_big=0x7f020001;
    public static final int sori_small=0x7f020002;
}
...
    
```

# 이미지뷰 – android:src 속성

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:background="#7F7F7F"
    android:orientation="vertical">

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#3F48CC"
        android:src="@drawable/sori_small"/>

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#3F48CC"
        android:src="@drawable/sori_big"/>

</LinearLayout>
```



원본파일

파일명 : sori\_big.png  
해상도 : 960X1280



원본파일

파일명 : sori\_small.png  
해상도 : 100X133

이미지 크기에 따라 화면에 출력된 이미지뷰의 모양이 다르다. 왜 그럴까?

이미지뷰는 원본 이미지의 비율을 깨지 않는 것을 기본으로 한다.

View



# 이미지뷰 – android:adjustViewBounds 속성

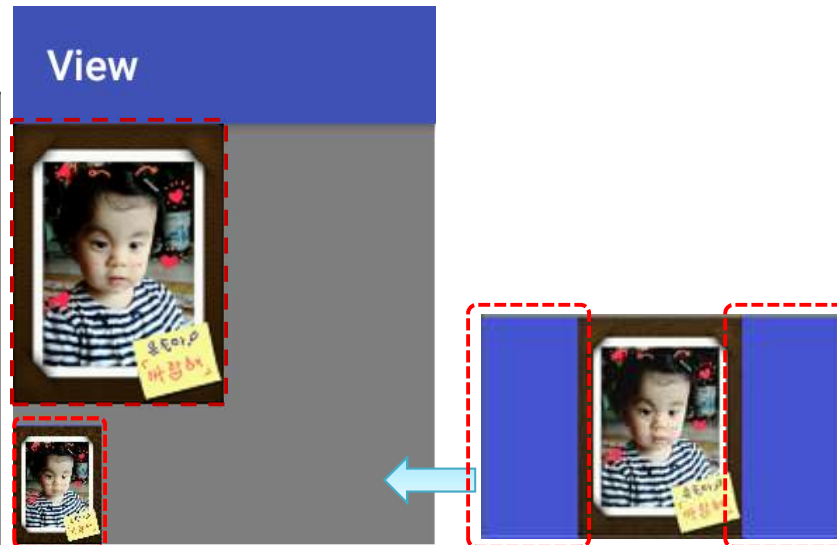
res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:background="#7F7F7F"
    android:orientation="vertical">

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="10dp"
        android:background="#3F48CC"
        android:src="@drawable/sori_small"
        android:adjustViewBounds="true"/>

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#3F48CC"
        android:src="@drawable/sori_big"
        android:adjustViewBounds="true"/>

</LinearLayout>
```



특정 뷰그룹에 이미지뷰를 추가하면, 부모 뷰그룹은 이미지뷰의 영역을 결정해준다.

그리고 이미지뷰는 뷰그룹이 할당한 영역에 자신의 콘텐츠인 원본이미지를 그린다.

여기서 원본이미지가 뷰영역 보다 크다면 종횡비를 유지하여 축소하는데, 대부분 이미지뷰와 원본이미지의 종횡비 차이가 있다. 따라서 이미지뷰에 좌우 빈 공간이 생긴다.



# 이미지뷰 – android:maxWidth, android:maxHeight 속성

이 속성은 이미지뷰 영역에 그려질 **원본 이미지의 최대 크기**를 지정한다.

android:maxWidth 속성은 최대 너비를 설정하고, android:maxHeight 속성은 최대 높이를 설정한다.

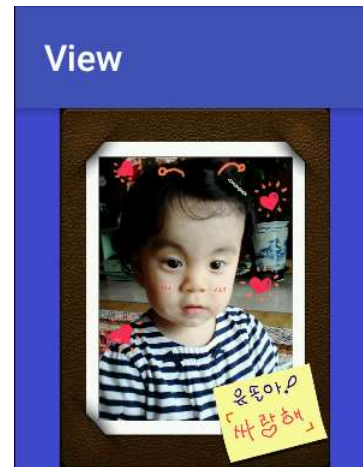
res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:background="#7F7F7F"
    android:orientation="vertical">

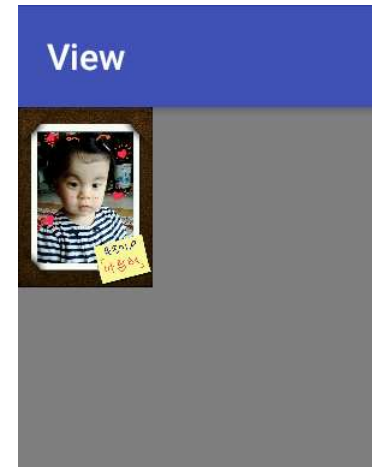
    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#3F48CC"
        android:src="@drawable/sori_big"
        android:adjustViewBounds="false"
        android:maxWidth="100dp"
        android:maxHeight="100dp"/>

</LinearLayout>
```

함께 사용!!



android:adjustViewBounds="false"



android:adjustViewBounds="true"

android:maxWidth, android:maxHeight 속성을 사용할 때는  
android:adjustViewBounds 속성을 true로 설정해야 한다.

# 이미지뷰 – android:tint 속성

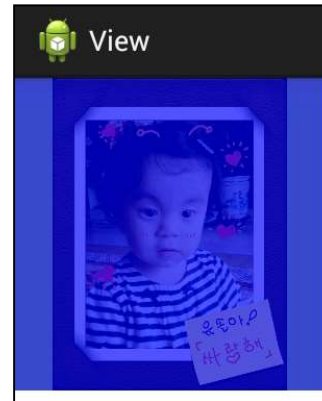
이미지뷰의 상단에 색조를 입힌다.

res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:background="#7F7F7F"
    android:orientation="vertical">

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#3F48CC"
        android:src="@drawable/sori_big"
        android:tint="#AA0000FF"/>

</LinearLayout>
```



tint = "#AA0000FF"

색조값에 투명도를 지정하지 않으면  
원본 이미지가 보이지 않게 되므로 대부분 투명도를 설정한다.



# 이미지뷰 – android:baselineAlignBottom 속성

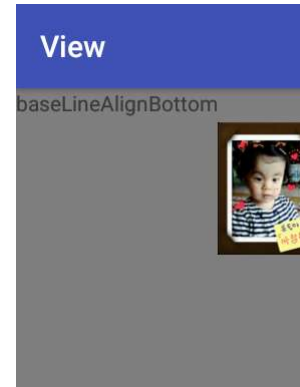
res/layout/activity\_main.xml

```
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:background="#7F7F7F"
    android:orientation="horizontal">

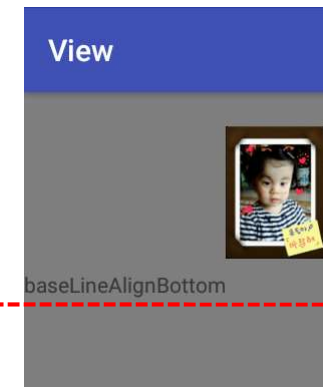
    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="baselineAlignBottom"/>

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/sori_small"
        android:baselineAlignBottom="false"/>

</LinearLayout>
```



android:baselineAlignBottom="false"



android:baselineAlignBottom="true"

이 속성은 이미지뷰 아래에 정렬 기준선을 설정하는 기능이다.

텍스트뷰의 경우 문자열의 하단이 기준선이며, 텍스트뷰를 제외한 대부분의 뷰들은 기준선 자체가 없다.

사실 정렬 기준선 자체는 텍스트뷰를 위해 존재하는 기능이나 다름없다.

이유는 다음과 같이 다양한 글자크기를 가진 텍스트뷰를 수평으로 배치할 때,  
글자하단을 기준으로 배치해야 보기 좋기 때문이다.

# 이미지뷰 – android:baseLine 속성

36

res/layout/activity\_main.xml

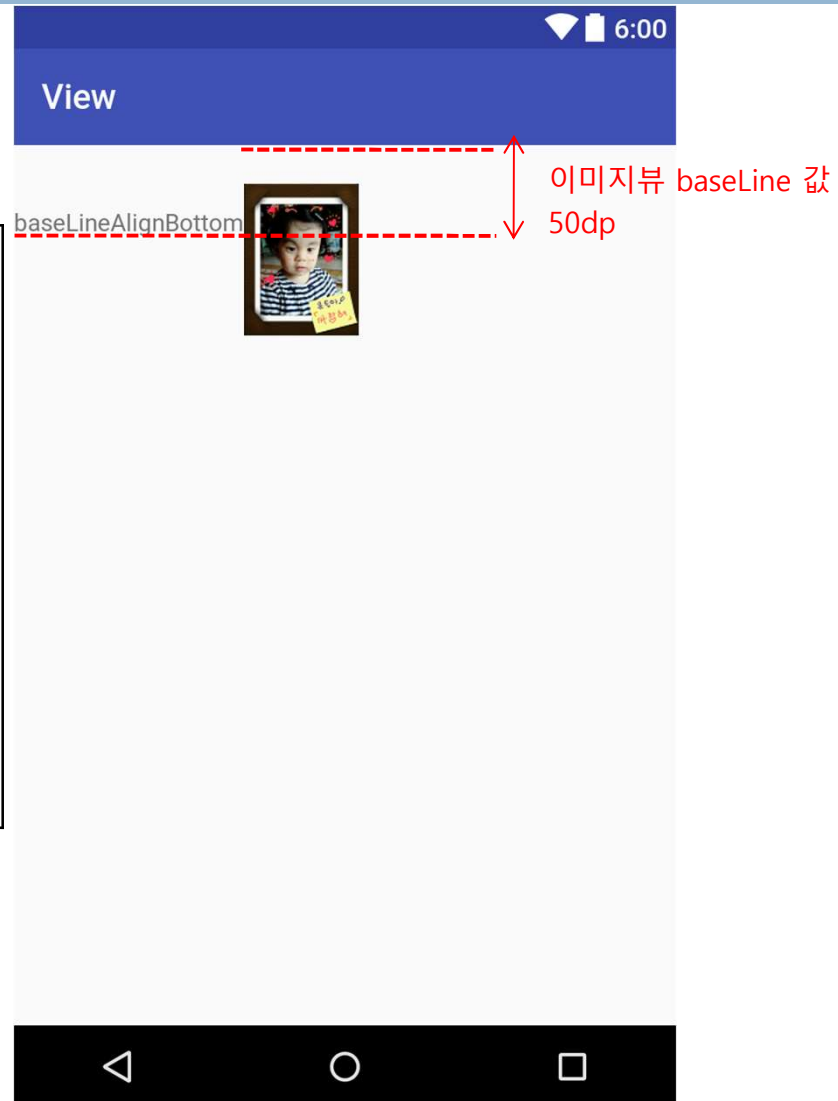
```
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:orientation="horizontal">

    <TextView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="baseLineAlignBottom"/>

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/sori_small"
        android:baseline="50dp"/>

</LinearLayout>
```

이 속성은 API 11부터 지원된다.



# 이미지뷰 – android:scaleType 속성

이미지뷰에 표시되는 원본이미지 크기를 다양한 형태로 조정할 수 있다.

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>matrix</b> 원본이미지를 이미지뷰 좌측 상단부터 표시한다. <ul style="list-style-type: none"> <li>• 이미지 비율 유지</li> <li>• 이미지 크기 유지</li> </ul>	 <p>빨강색 영역은 이미지 뷰의 영역이다.</p>	 <p>이미지뷰 밖의 영역으로 화면에 보이지 않는 영역</p>
		

# 이미지뷰 – android:scaleType 속성

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>center</b> 원본이미지를 이미지뷰 정 중앙에 표시한다. <ul style="list-style-type: none"> <li>• 이미지 비율 유지</li> <li>• 이미지 크기 유지</li> </ul>		
		



# 이미지뷰 – android:scaleType 속성

39

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>centerInside</b>  원본 이미지를 이미지뷰 영역 내에 중앙 정렬하여 맞춘다. 만일 이미지 크기가 뷰 영역보 다 작으면 이미지 크기를 변경 하지 않는다.  • 이미지 비율 유지 • 뷰 영역보다 이미지가 크면 원본 이미지를 뷰 크기에 맞게 변경, 크지 않으면 유지		
		

# 이미지뷰 – android:scaleType 속성

40

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>centerCrop</b>  원본이미지를 이미지뷰 정 중앙에 표시한다. 원본이미지는 이미지뷰 영역 안에 비율을 유지하면서 여백을 남기지 않고 맞춘다.  • 이미지 비율 유지 • 이미지 크기 변경		
		



# 이미지뷰 – android:scaleType 속성

41

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>fitXY</b>  원본 이미지를 이미지뷰 영역 안에 여백을 남기지 않고 맞춘 다.  • 이미지 비율 변경 • 이미지 크기 변경		
		

# 이미지뷰 – android:scaleType 속성

42

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>fitStart</b>  원본이미지를 이미지뷰 영역 안에 좌측 혹은 상단 정렬하여 맞춘다.  • 이미지 비율 유지 • 이미지 크기 변경		
		

# 이미지뷰 – android:scaleType 속성

43

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>fitCenter</b>  원본이미지를 이미지뷰 영역 안에 중앙 정렬하여 맞춘다. <ul style="list-style-type: none"><li>• 이미지 비율 유지</li><li>• 이미지 크기 변경</li></ul>		
		

# 이미지뷰 – android:scaleType 속성

44

속성값 및 의미	뷰 영역보다 이미지가 작을 때	뷰 영역보다 이미지가 클 때
<b>fitEnd</b>  원본이미지를 이미지뷰 영역 내에 우측 혹은 하단 정렬하여 맞춘다. <ul style="list-style-type: none"><li>• 이미지 비율 유지</li><li>• 이미지 크기 변경</li></ul>		
		

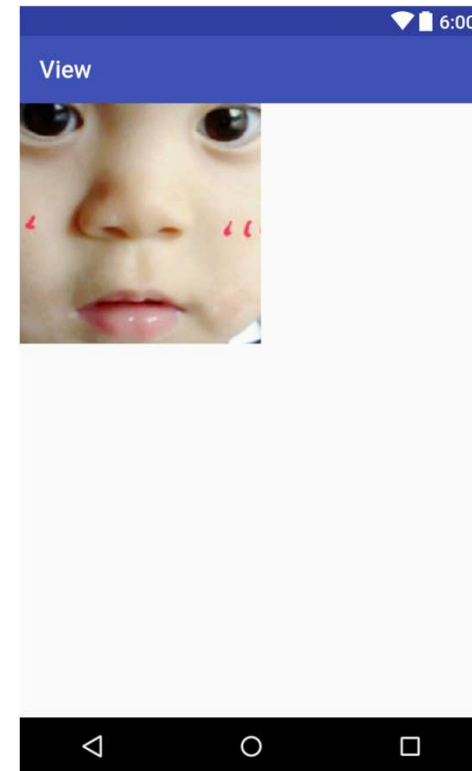
# 이미지뷰 – android:scaleType 속성

45

```
res/layout/activity_main.xml
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:orientation="horizontal">

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/sori_big"
        android:scaleType="center"/>

</LinearLayout>
```



# 이미지뷰 – android:cropToPadding 속성

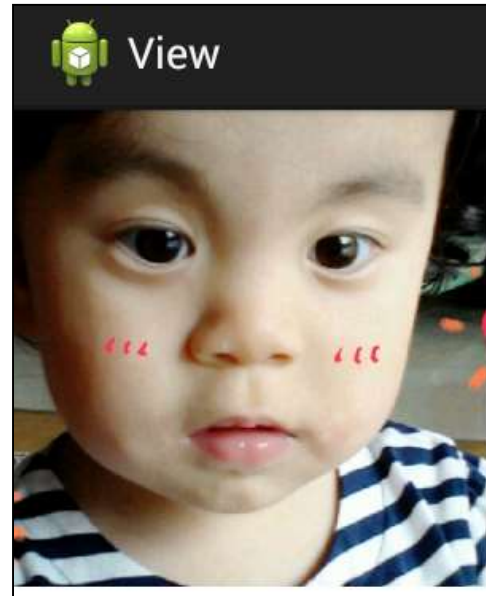
46

res/layout/activity\_main.xml

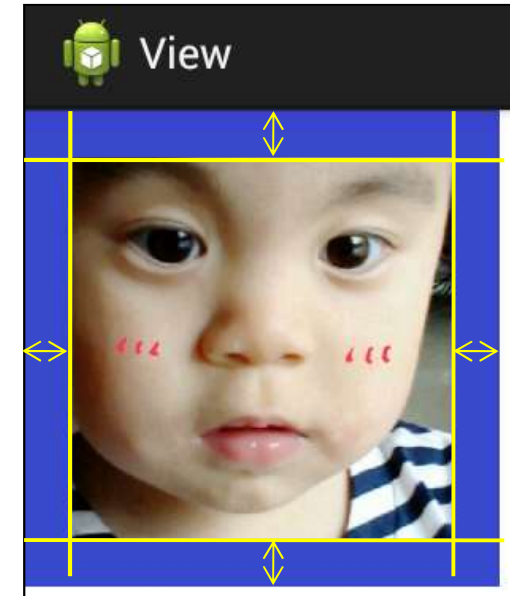
```
<LinearLayout xmlns:android="..."
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:orientation="horizontal">

    <ImageView android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/sori_big"
        android:background="#3F48CC"
        android:padding="20dp"
        android:scaleType="center"
        android:cropToPadding="false"/>

</LinearLayout>
```



android:cropToPadding="false"



android:cropToPadding="true"

기본적으로 이미지뷰는 아무리 패딩을 설정하더라도 원본 이미지가 이미지뷰의 크기를 벗어나는 경우에는 적용되지 않는다.

이 속성은 API 16부터 지원된다.