

자바로 레이아웃 구성하기

UI를 작성하는 3가지 방법

2

- XML 파일로 사용자 인터페이스를 작성하는 방법
- 코드로 사용자 인터페이스를 작성하는 방법
- XML과 코드를 동시에 사용하는 방법

초기화면
Activity 내에 동적으로 화면 제어

액티비티

3

src/MainActivity.java

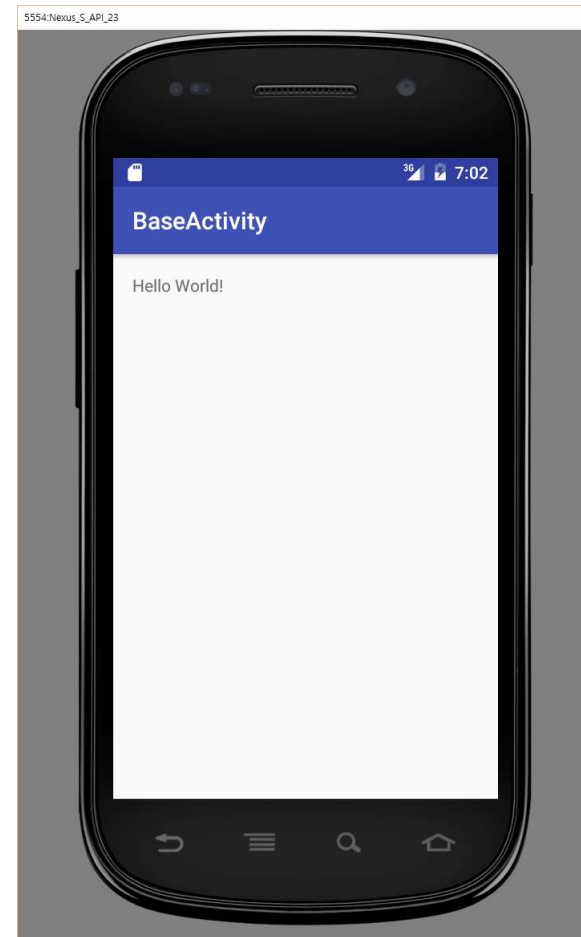
```
public class MainActivity extends AppCompatActivity
{
    @Override
    protected void onCreate( Bundle savedInstanceState )
    {
        super.onCreate( savedInstanceState );
        setContentView( R.layout.activity_main );
    }
}
```

res/layout/activity_main.xml

```
<RelativeLayout xmlns:android="http://..."
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="hello world!" />

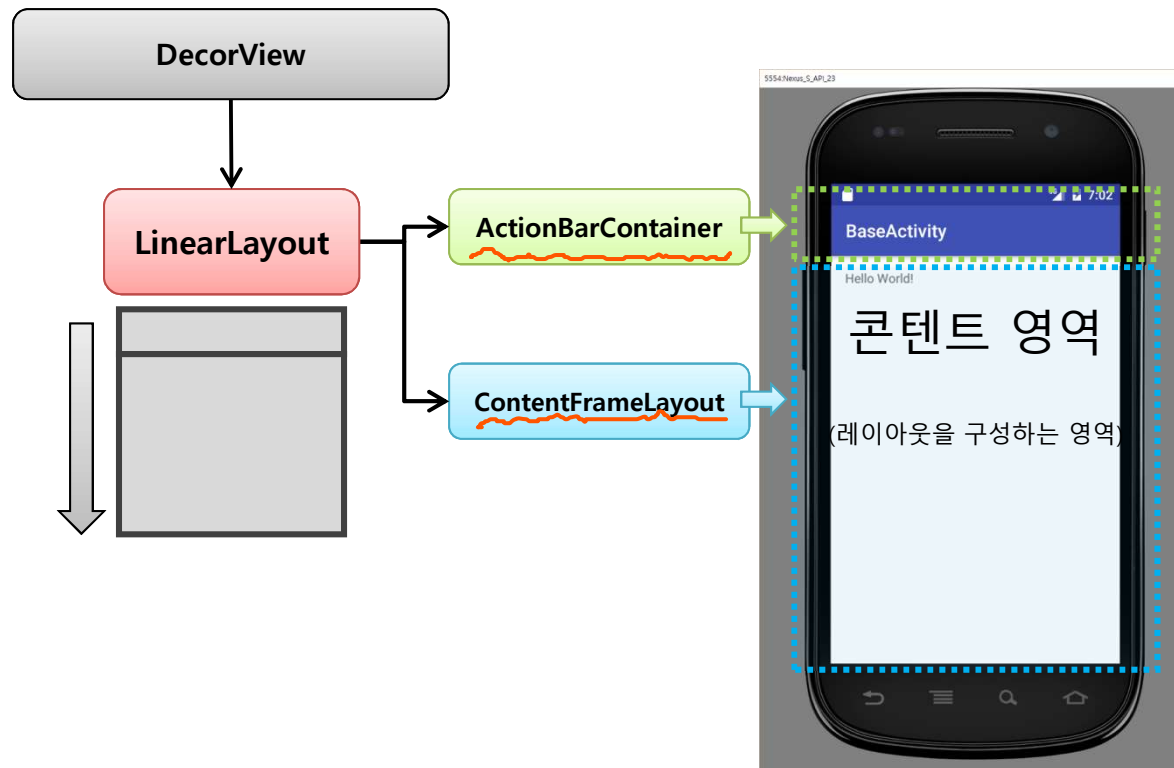
</RelativeLayout>
```





액티비티 기본 화면 구성 분석

5



콘텐츠 영역을 채워보자

6

res/layout/activity_main.xml

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20px">

    <LinearLayout android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView android:text="이름을 입력하세요 "
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>

        <EditText android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>
    </LinearLayout>

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

</LinearLayout>
```

URI

C:\Users\Wiamjw\AppData\Local\Android\Sdk\platforms\android-24\data\res\values\attrs.xml

↓
namespace 불러

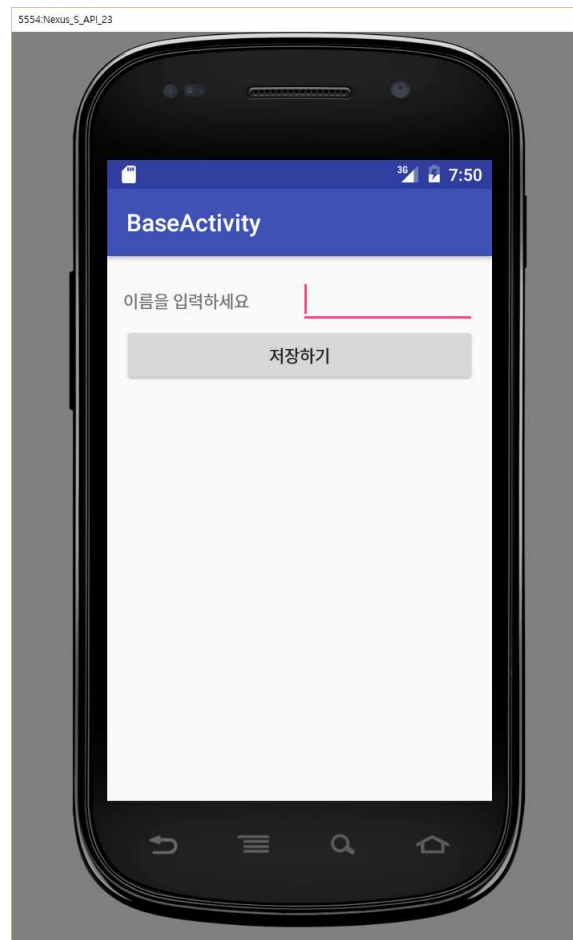
src/MainActivity.java

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

activity_main.xml의 id

콘텐츠 영역을 채워보자

7



setContentView 함수의 역할

8

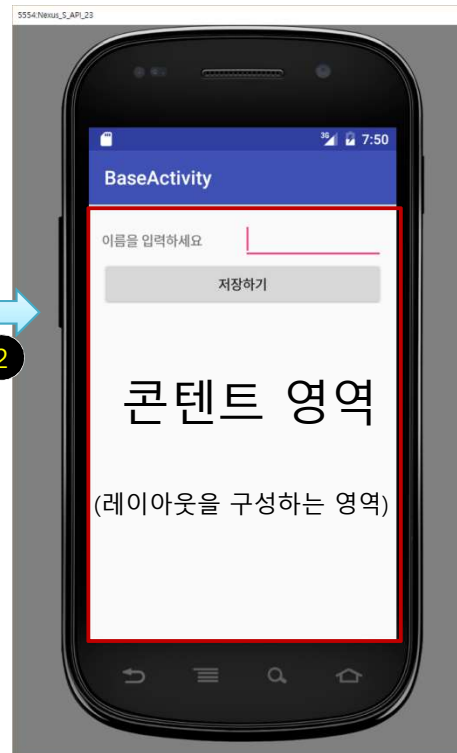
```
<LinearLayout android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="이름을 입력하세요" />
    <Button android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="저장하기" />
</LinearLayout>
```

1

이름을 입력하세요
저장하기

2



인플레이션(Inflation)

- XML 레이아웃에 정의된 정보를 메모리 상에서 객체로 만드는 객체화 과정

1 setContentView 함수는 레이아웃 XML의 내용을 파싱하여 뷰들을 생성하고, 뷰에 정의된 속성들을 설정한다. 또한 생성된 뷰들을 상하관계에 맞춰 배치한다. 참고로 setContentView 함수 내에 이러한 처리는 모두 LayoutInflater라는 클래스에 의존한다.

2 레이아웃 XML의 처리 결과로 생성된 뷰들을 콘텐츠 영역에 추가한다.

직접 인플레이션 제어

자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들

9

res/layout/activity_main.xml

```
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20px">
```

→ 대응하는 함수들

```
<LinearLayout android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
```

```
<TextView android:text="이름을 입력하세요 "
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"/>
```

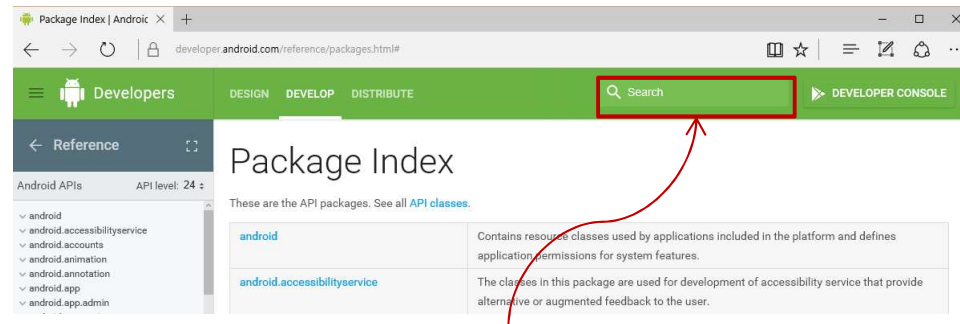
```
<EditText android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="1"/>
```

```
</LinearLayout>
```

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

```
</LinearLayout>
```

<https://developer.android.com/reference> 방문



1. LinearLayout 입력
2. 추천어들 중에서 android.widget.LinearLayout 선택

자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들

10

LinearLayout | Android | X +

developer.android.com/reference/android/widget/LinearLayout.html

Developers

DESIGN DEVELOP DISTRIBUTE

Search

DEVELOPER CONSOLE

Reference

Android APIs API level: 24

- android.os
- android.os.health
- android.os.storage
- android.preference
- android.print
- android.print.pdf
- android.printservice
- android.provider
- android.renderscript
- android.sax
- android.security
- android.security.keystore
- android.service.carrier
- android.service.chooser
- android.service.dreams
- android.service.media
- android.service.notification
- android.service.quicksettings
- android.service.restrictions
- android.service.textservice
- android.service.voice
- android.service.vr
- android.service.wallpaper
- android.speech

Summary

Nested classes

class	LinearLayout.LayoutParams
Per-child layout information associated with ViewLinearLayout.	

XML attributes

android:baselineAligned	When set to false, prevents the layout from aligning its children's baselines.
android:baselineAlignedChildIndex	When a linear layout is part of another layout that is baseline aligned, it can specify which of its children to baseline align to (that is, which child TextView).
android:divider	Drawable to use as a vertical divider between buttons.
android:gravity	Specifies how an object should position its content, on both the X and Y axes, within its own bounds.
android:measureWithLargestChild	When set to true, all children with a weight will be considered having the minimum size of the largest child.
android:orientation	Should the layout be a column or a row? Use "horizontal" for a row, "vertical" for a column.
android:weightSum	Defines the maximum weight sum.

클릭

자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들

11

The screenshot shows the Android Developer website. The left sidebar lists various Android APIs, with 'android.os' expanded. The main content area is titled 'android:orientation' and explains that it determines if a layout is a column or a row. It provides a table of constant values: 'horizontal' (0) and 'vertical' (1). Below the table, it mentions that this corresponds to the global attribute resource symbol 'orientation'. Under the 'Related methods' section, 'setOrientation(int)' is listed. A red arrow points from the Korean word '클릭' (click) to the 'setOrientation(int)' method.

LinearLayout | Android | X

developer.android.com/reference/android/widget/LinearLayout.html#attr_android:orientation

Developers

DESIGN DEVELOP DISTRIBUTE

Search

DEVELOPER CONSOLE

Reference

Android APIs API level: 24

- android.os
- android.os.health
- android.os.storage
- android.preference
- android.print
- android.print.pdf
- android.printservice
- android.provider
- android.renderscript
- android.sax
- android.security
- android.security.keystore
- android.service.carrier
- android.service chooser
- android.service.dreams
- android.service.media
- android.service.notification
- android.service.quicksettings
- android.service.restrictions
- android.service.textservice
- android.service.voice
- android.service.vr
- android.service.wallpaper
- android.speech

android:orientation

Should the layout be a column or a row? Use "horizontal" for a row, "vertical" for a column. The default is horizontal.

Must be one of the following constant values.

Constant	Value	Description
horizontal	0	Defines an horizontal widget.
vertical	1	Defines a vertical widget.

This corresponds to the global attribute resource symbol `orientation`.

Related methods:

`setOrientation(int)`

android:weightSum

Defines the maximum weight sum. If unspecified, the sum is computed by adding the layout_weight of all of the children. This can be used for

클릭

자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들

12

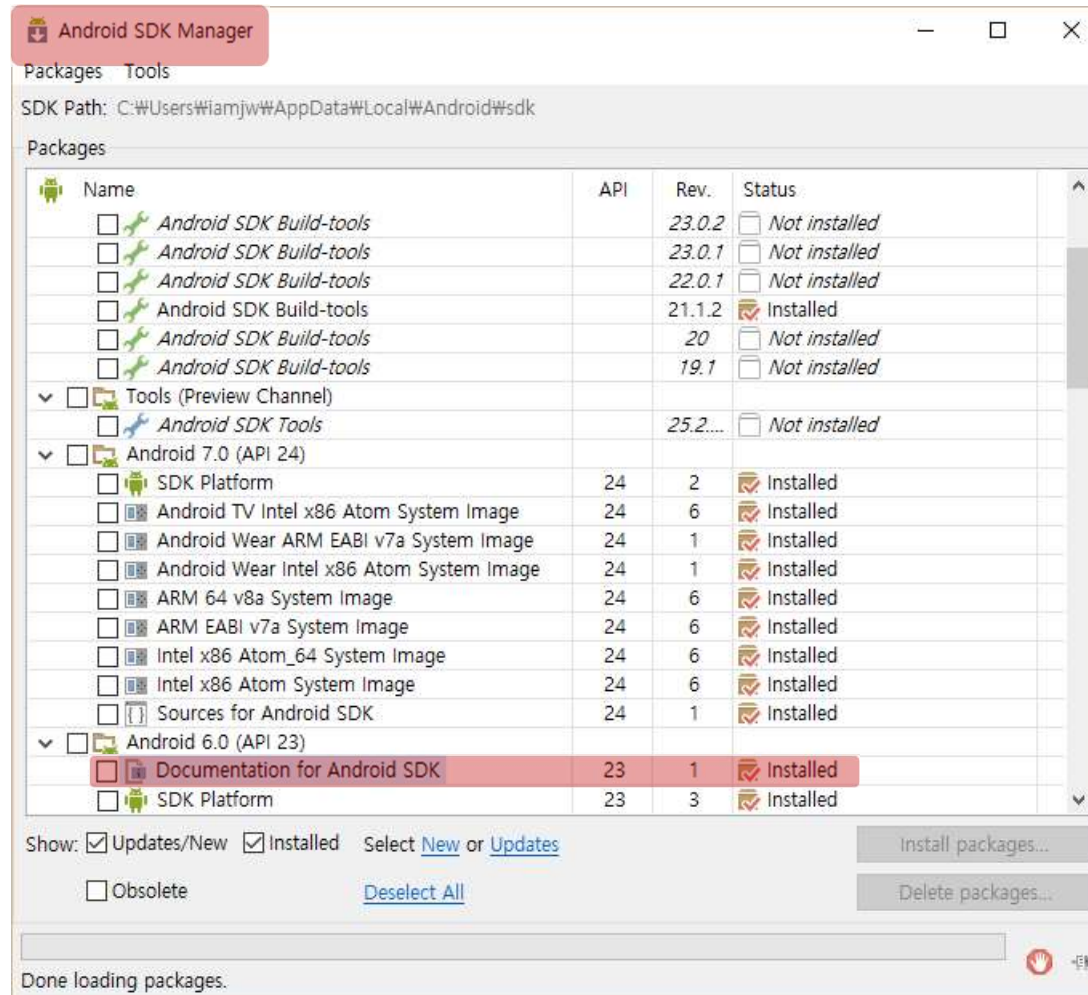
The screenshot shows the Android Developer website's reference page for the `setOrientation` method of `LinearLayout`. The page is titled "setOrientation" and includes a description: "Should the layout be a column or a row." It also lists the related XML attribute as `android:orientation`. The parameters section shows `orientation` as an `int` with the description "Pass HORIZONTAL or VERTICAL. Default value is HORIZONTAL." A red box highlights the word `VERTICAL` in the parameter description. A red arrow points from the text "1. 클릭" to this box. Another red arrow points from the text "2. LinearLayout.VERTICAL 이므로 setOrientation(LinearLayout.VERTICAL) 이라고 사용" to the `setOrientation` method name.

1. 클릭
2. `LinearLayout.VERTICAL` 이므로 `setOrientation(LinearLayout.VERTICAL)` 이라고 사용

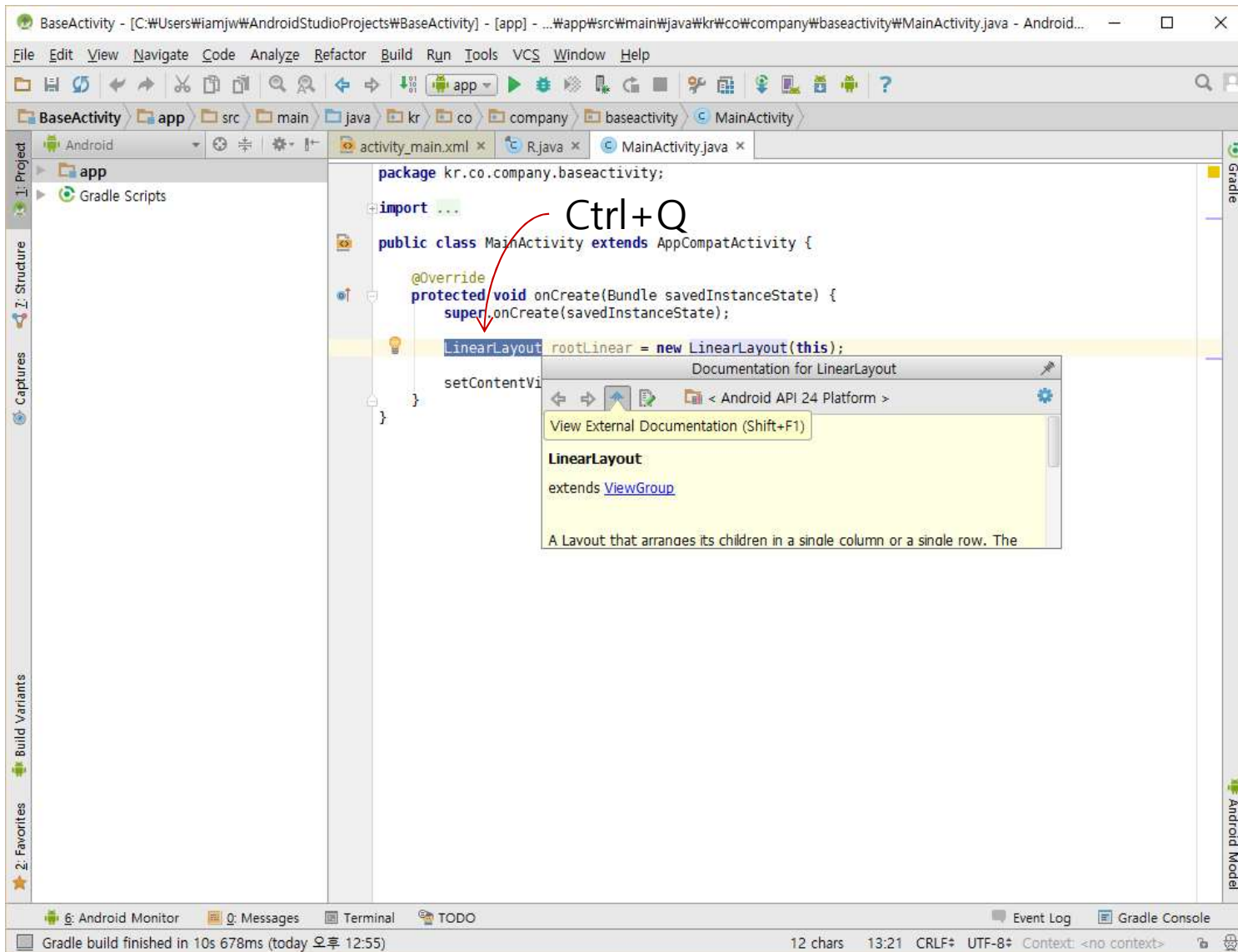
자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들

13

□ 오프라인 안드로이드 개발자 문서



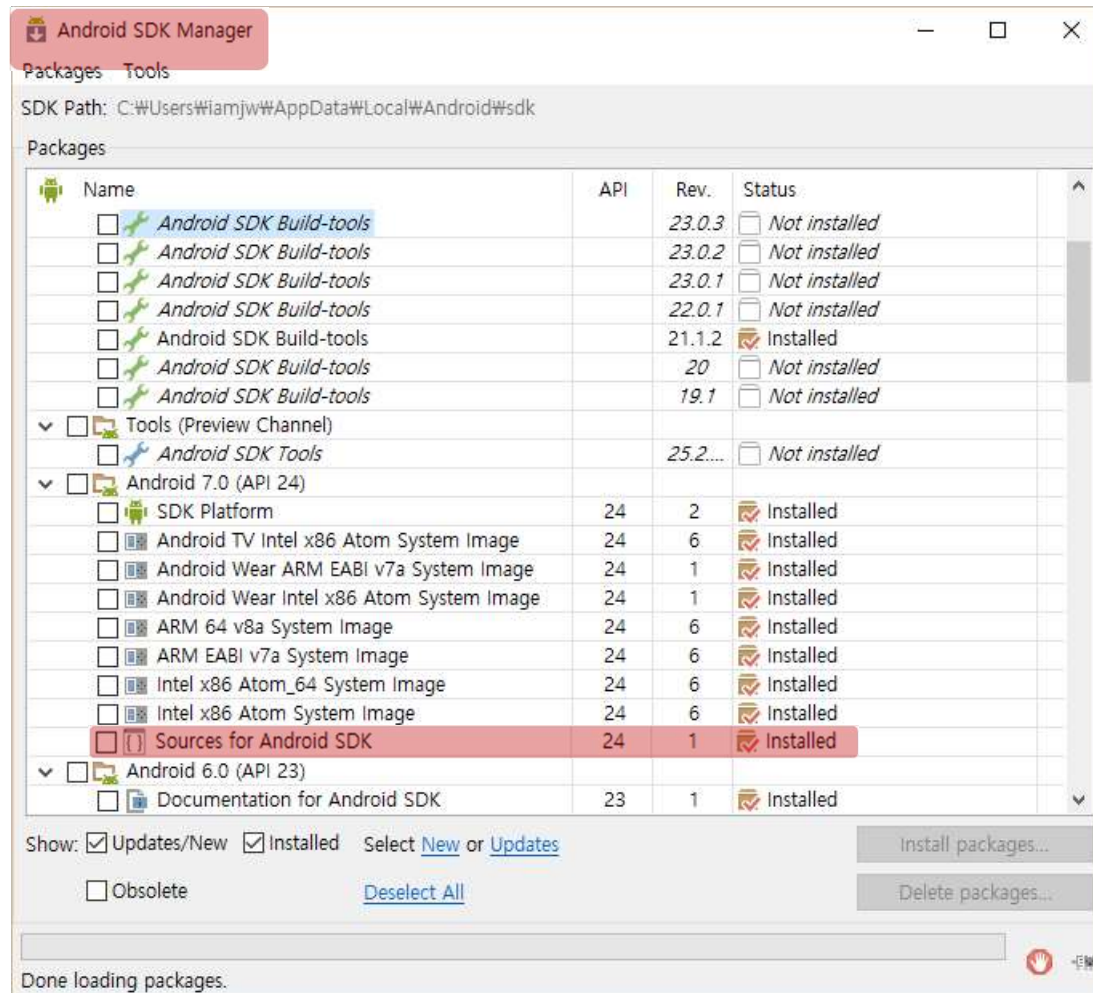
자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들



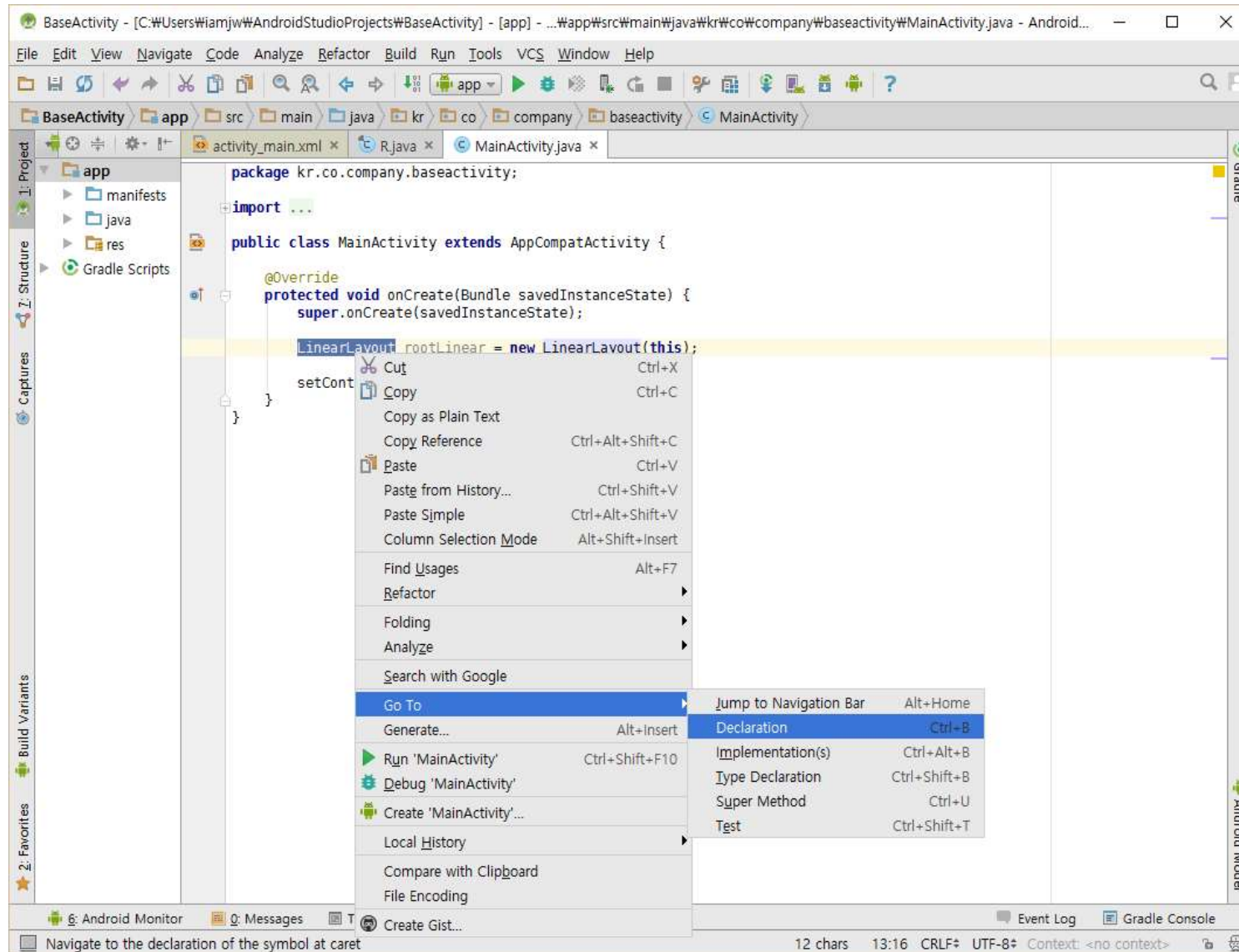
자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들

15

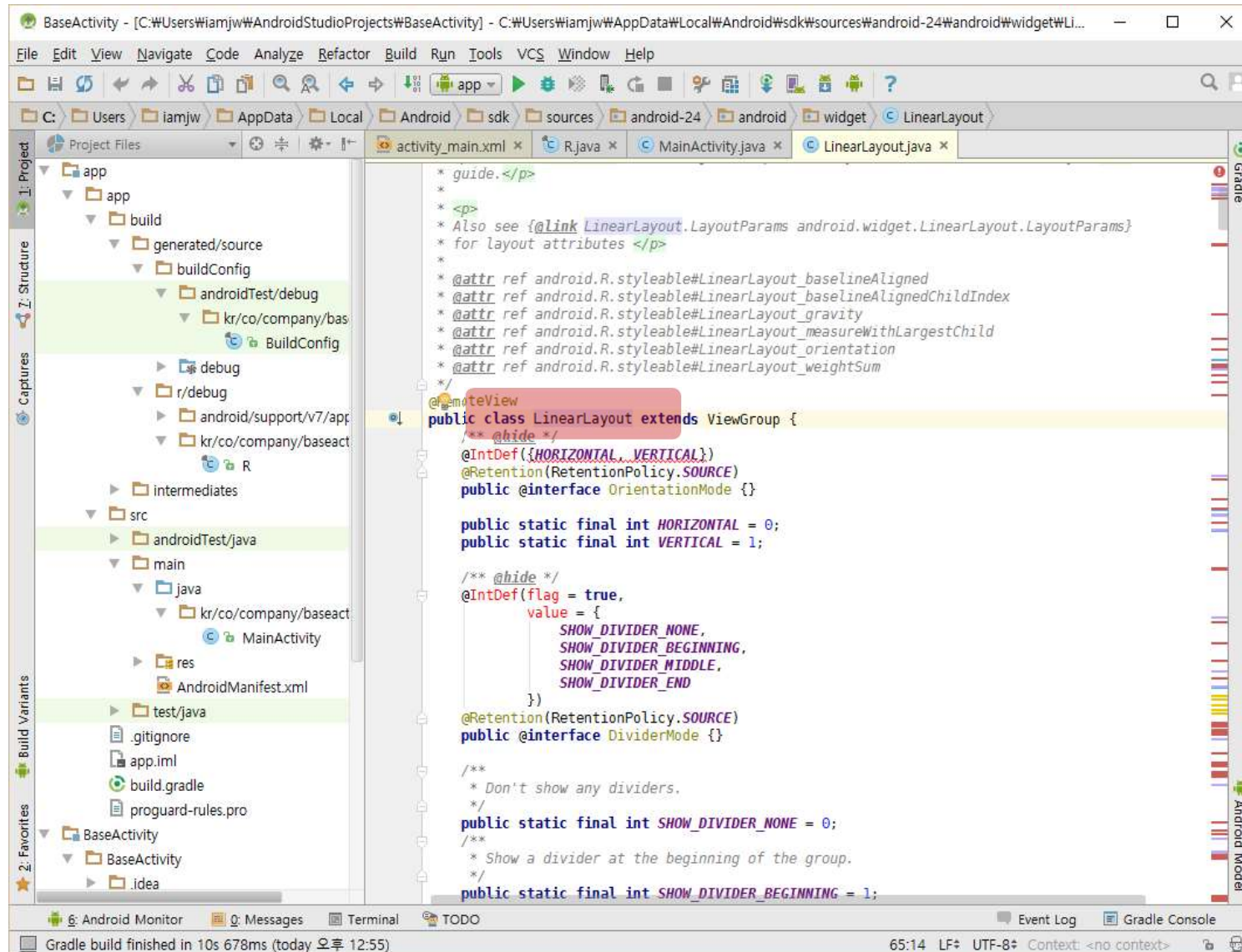
□ 프레임워크 소스 활용하기



자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들



자바 소스 작성 전에 꼭 알고 넘어가야 할 기능들



자바 소스로 구현하기

첫 번째 과정: 전체 뷰를 감싸고 있는 수직 배치 LinearLayout을 자바 소스로 옮긴다.

18

res/layout/activity_main.xml

```
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20px">

    <LinearLayout android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

        <TextView android:text="이름을 입력하세요"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>

        <EditText android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>
    </LinearLayout>

    <Button android:text="저장하기"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

LinearLayout 수직 배치

이름을 입력하세요

저장하기

자바 소스로 구현하기

첫 번째 과정: 전체 뷰를 감싸고 있는 수직 배치 LinearLayout을 자바 소스로 옮긴다.

19

src/MainActivity.java

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

```
    @Override
```

```
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);
```

```
        LinearLayout rootLinear = new LinearLayout(this);  
        rootLinear.setOrientation(LinearLayout.VERTICAL);
```

```
        ContentFrameLayout.LayoutParams rootLp =  
            new ContentFrameLayout.LayoutParams(  
                ContentFrameLayout.LayoutParams.MATCH_PARENT,  
                ContentFrameLayout.LayoutParams.MATCH_PARENT,
```

```
        rootLp.setMargins(20, 20, 20, 20);
```

view

Layout Params 객체

context

생성자

자바 소스로 구현하기

두 번째 과정: 이름 입력 부분의 레이아웃을 자바 소스로 옮긴다.

20

res/layout/activity_main.xml

```
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20px">

    <LinearLayout android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

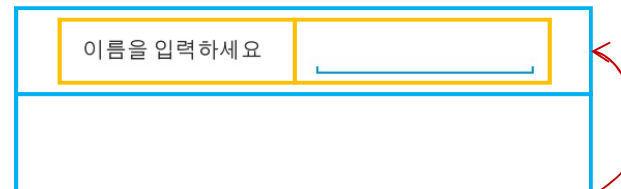
        <TextView android:text="이름을 입력하세요"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>

        <EditText android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>

    </LinearLayout>

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

</LinearLayout>
```



LinearLayout 수평 배치



자바 소스로 구현하기

두 번째 과정: 이름 입력 부분의 레이아웃을 자바 소스로 옮긴다.

21

```
src/MainActivity.java
...
LinearLayout nameInputLinear = new LinearLayout(this);
nameInputLinear.setOrientation(LinearLayout.HORIZONTAL);
LinearLayout.LayoutParams nameInputLp =
    new LinearLayout.LayoutParams(
        LinearLayout.LayoutParams.MATCH_PARENT,
        LinearLayout.LayoutParams.WRAP_CONTENT);

TextView nameTv = new TextView(this);
nameTv.setText("이름을 입력하세요");
LinearLayout.LayoutParams nameTextLp = new LinearLayout.LayoutParams(
    0,
    LinearLayout.LayoutParams.WRAP_CONTENT);
nameTextLp.weight = 1;

EditText nameEt = new EditText(this);
LinearLayout.LayoutParams nameEditLp = new LinearLayout.LayoutParams(
    0,
    LinearLayout.LayoutParams.WRAP_CONTENT);
nameEditLp.weight = 1;

nameInputLinear.addView(nameTv, nameTextLp);
nameInputLinear.addView(nameEt, nameEditLp);

rootLinear.addView(nameInputLinear, nameInputLp);
```

자바 소스로 구현하기

세 번째 과정: 저장하기 버튼을 자바 소스로 옮긴다.

22

res/layout/activity_main.xml

```
<LinearLayout xmlns:android="http://..."
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20px">

    <LinearLayout android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">

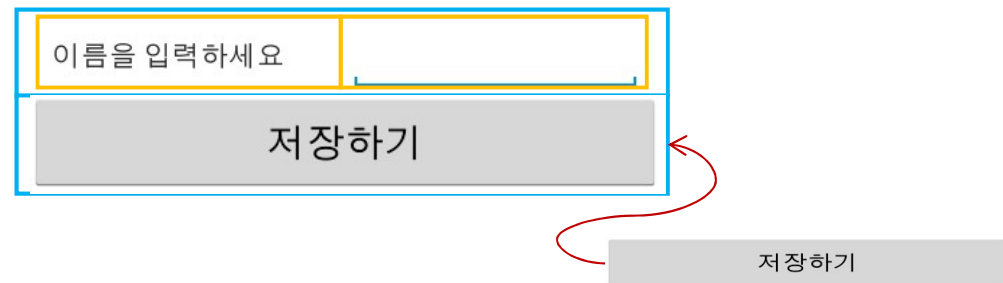
        <TextView android:text="이름을 입력하세요 "
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>

        <EditText android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"/>

    </LinearLayout>

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

</LinearLayout>
```



src/MainActivity.java

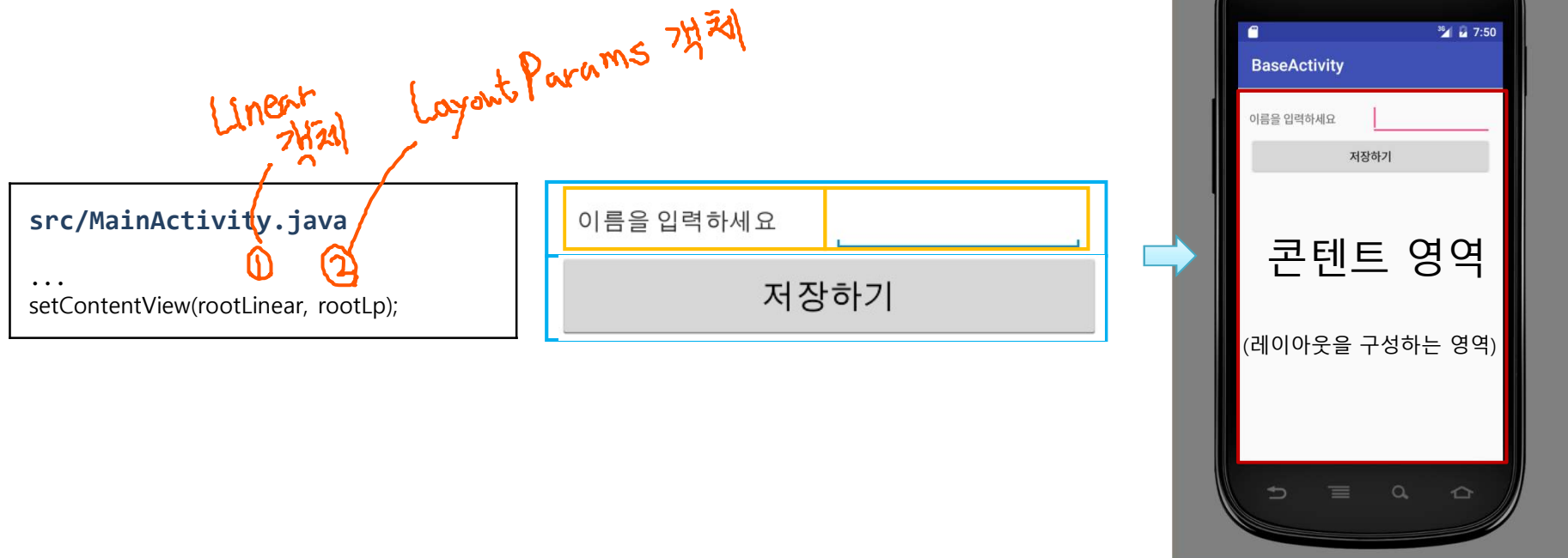
```
...
Button saveBtn = new Button(this);
saveBtn.setText("저장하기");
LinearLayout.LayoutParams saveBtnLp = new LinearLayout.LayoutParams(
    LinearLayout.LayoutParams.MATCH_PARENT,
    LinearLayout.LayoutParams.WRAP_CONTENT);

rootLinear.addView(saveBtn, saveBtnLp);
```

자바 소스로 구현하기

네 번째 과정: 전체 레이아웃을 콘텐츠 영역에 추가한다.

23



XML과 코드를 동시에 사용하는 방법

24

1. 프로젝트 UserInterface3를 생성한다.
2. activity_main.xml을 다음과 같이 수정한다.

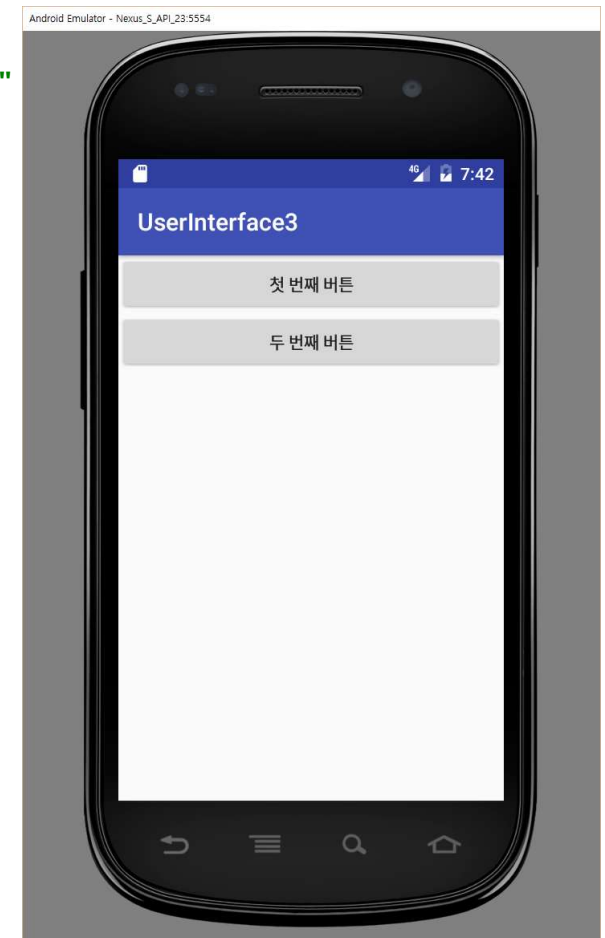
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

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

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

</LinearLayout>
```

버튼에 식별자를 부여한다.



XML과 코드를 동시에 사용하는 방법

25

3. MainActivity.java 파일을 다음과 같이 수정한다.

```
package kr.co.company.userinterface3;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

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

        Button b1 = (Button) findViewById(R.id.button1);
        b1.setText("코드에서도 변경 가능");

        Button b2 = (Button) findViewById(R.id.button2);
        b2.setEnabled(false);
    }
}
```

(인플레이션)
캐치가
안들거짐

id가 button1인 버튼을 찾는다.

