

Week07.  
**비디오 재생**



안드로이드 앱

# 개발환경 구축 절차

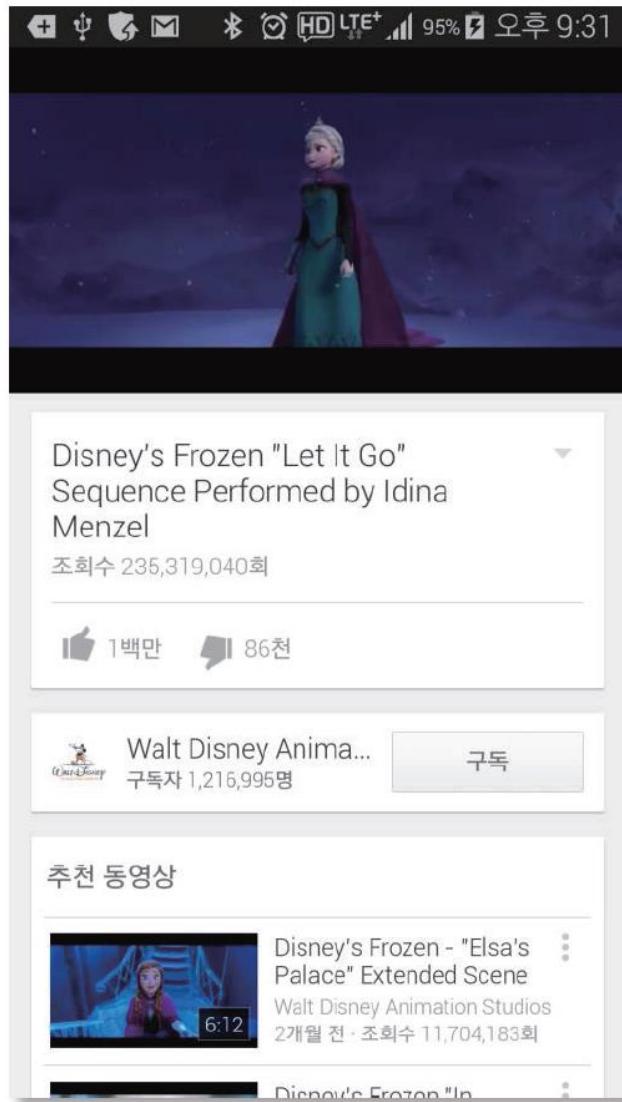
2

주 차	수업 내용
1	수업 소개
2	개발 환경 구축과 맛보기 프로젝트
3	텍스트 출력과 레이아웃
4	이미지의 출력
5	이벤트 처리와 액티비티 간 이동
6	오디오 재생
7	<b>비디오 재생</b>
8	중간고사
9	애니메이션
10	사물인터넷과 센서 – 터치 센서, 모션 센서
11	사물인터넷과 센서 – 위치 센서, 환경 센서
12	NFC 활용
13	공공 DB 오픈 API 활용
14	구글 맵과 위치 추적
15	기말 고사

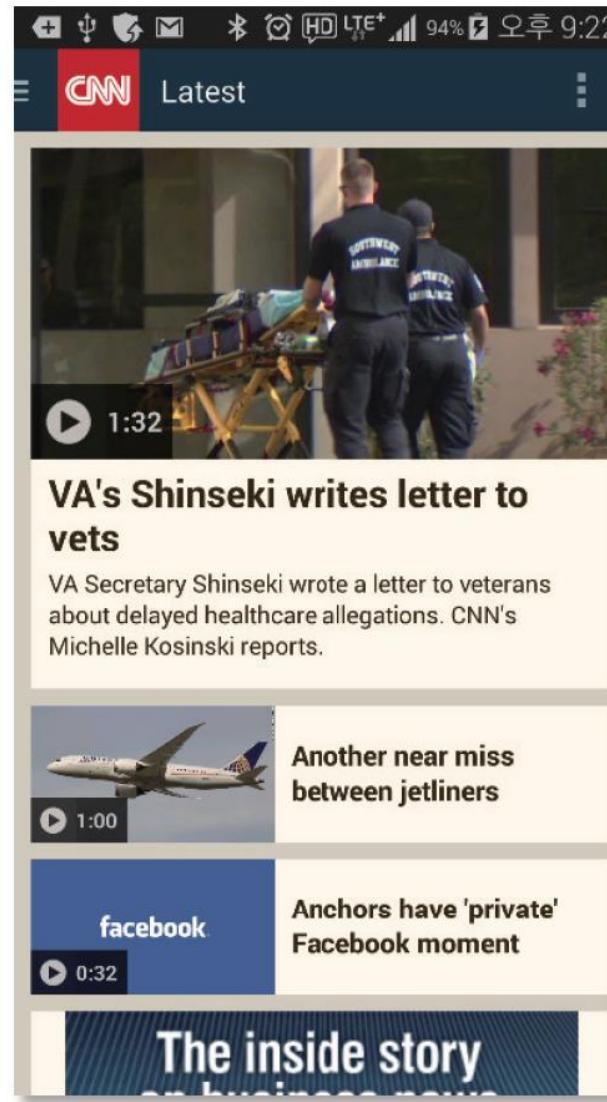


강의 자료-<https://github.com/hopypark>

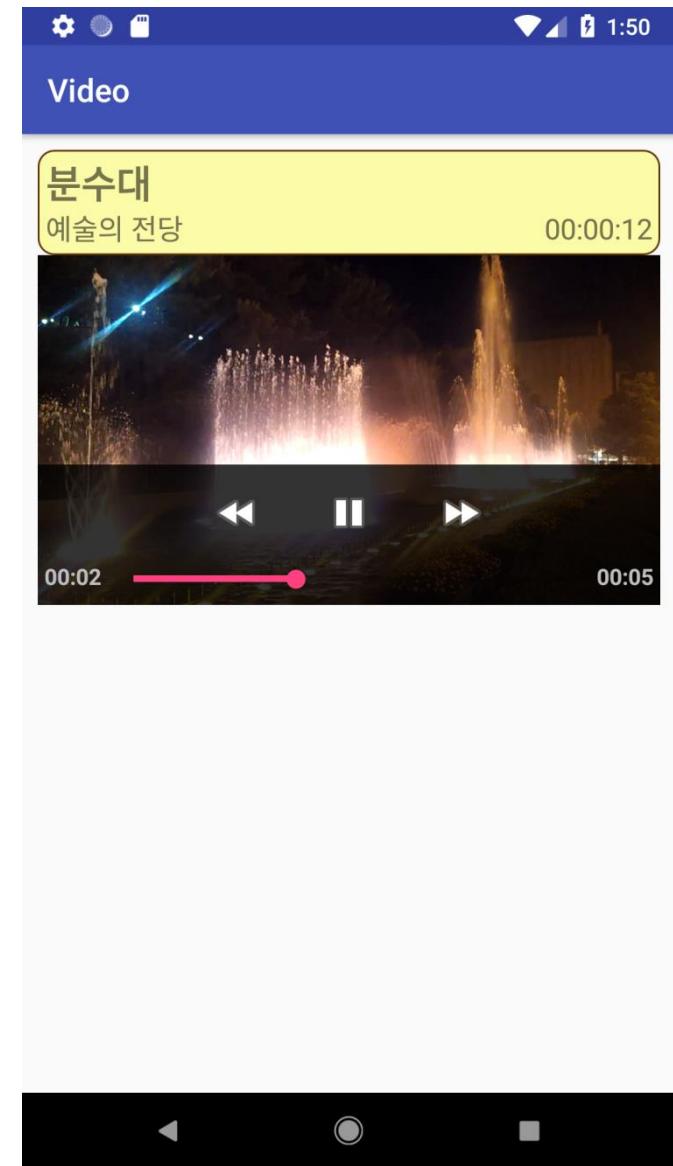
# 비디오 재생 앱의 예



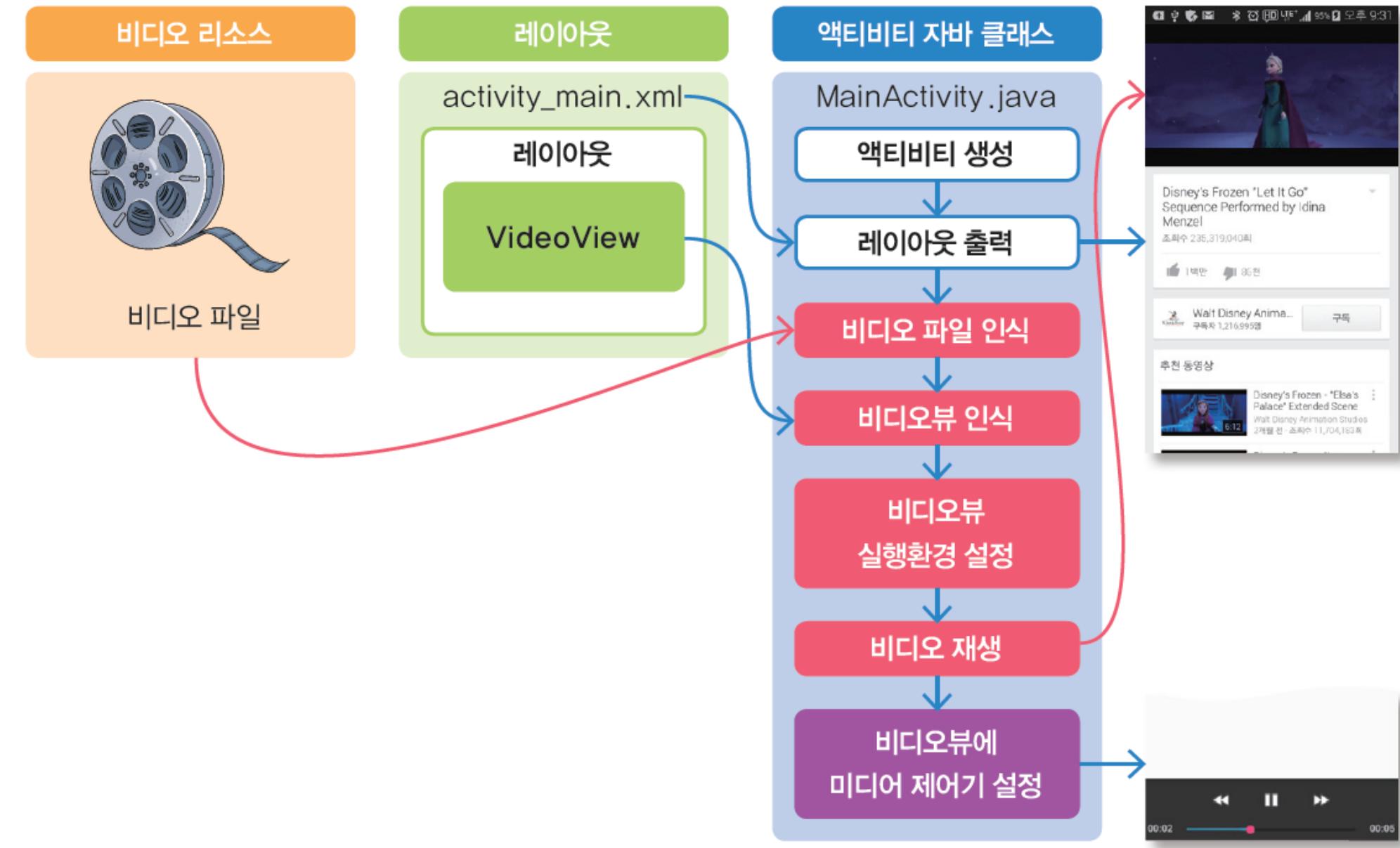
(a) YouTube 동영상



(b) CNN Video

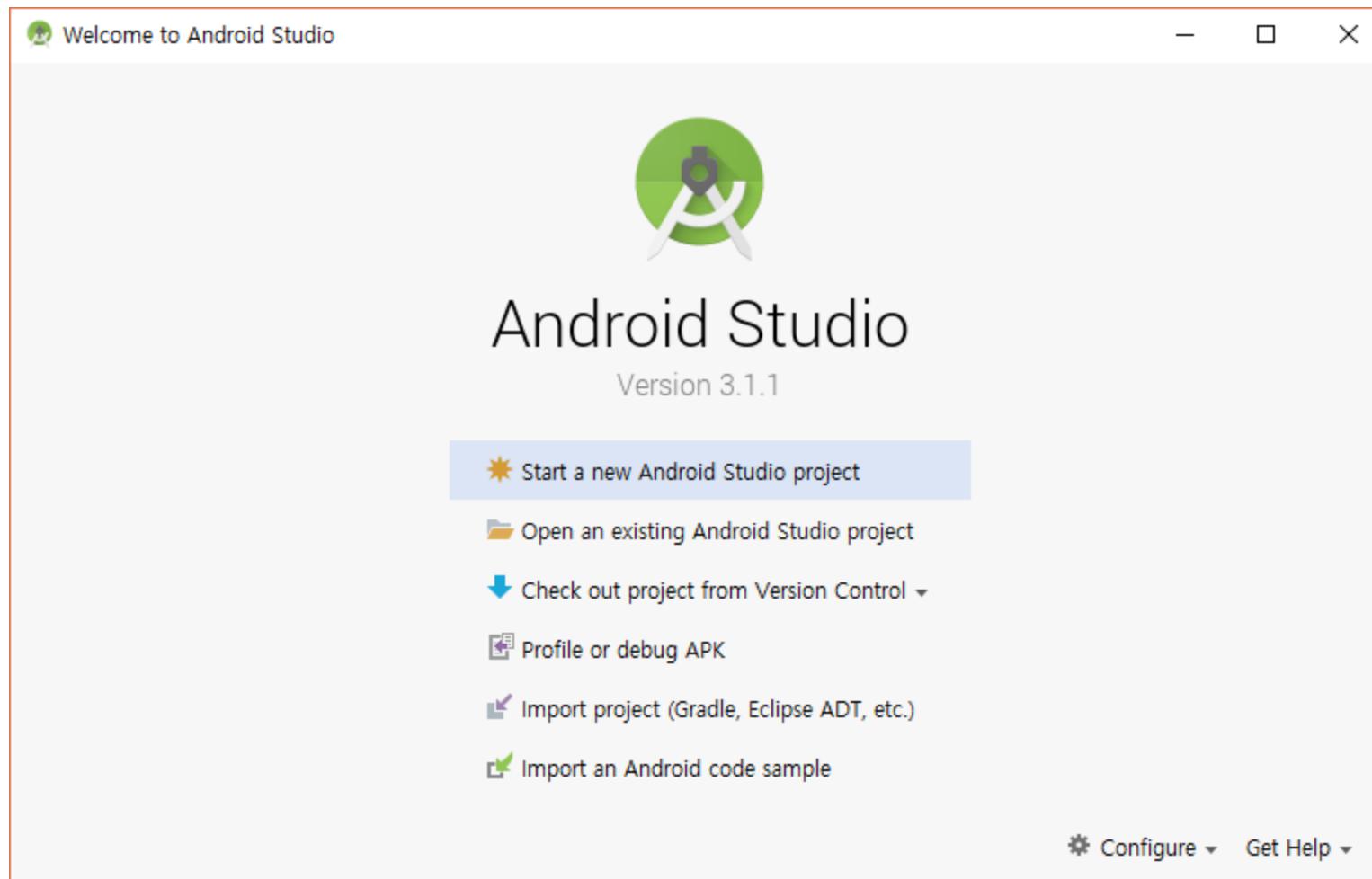


# 오디오 재생 원리



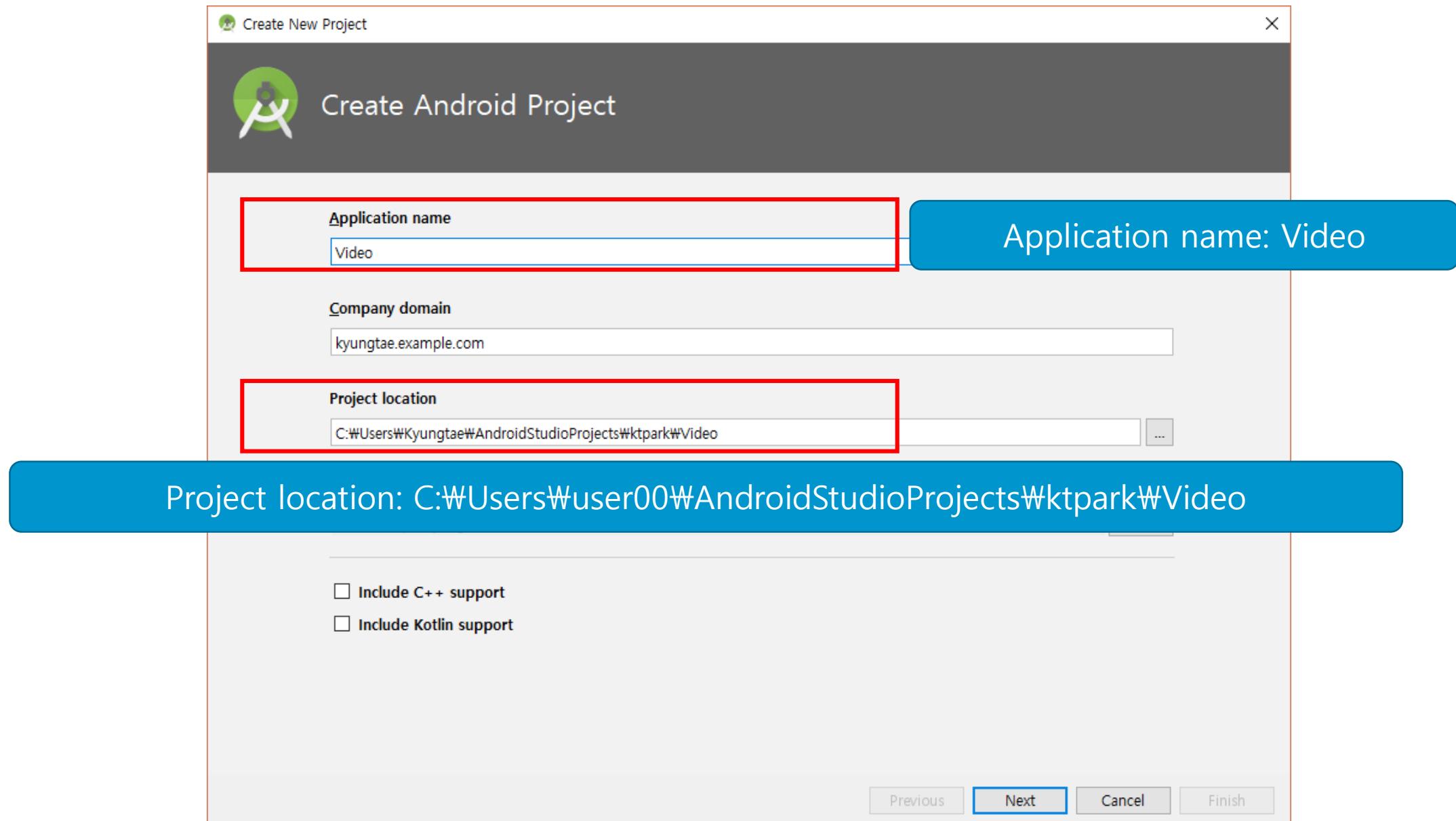
# Start a new Android Studio project-type1

6



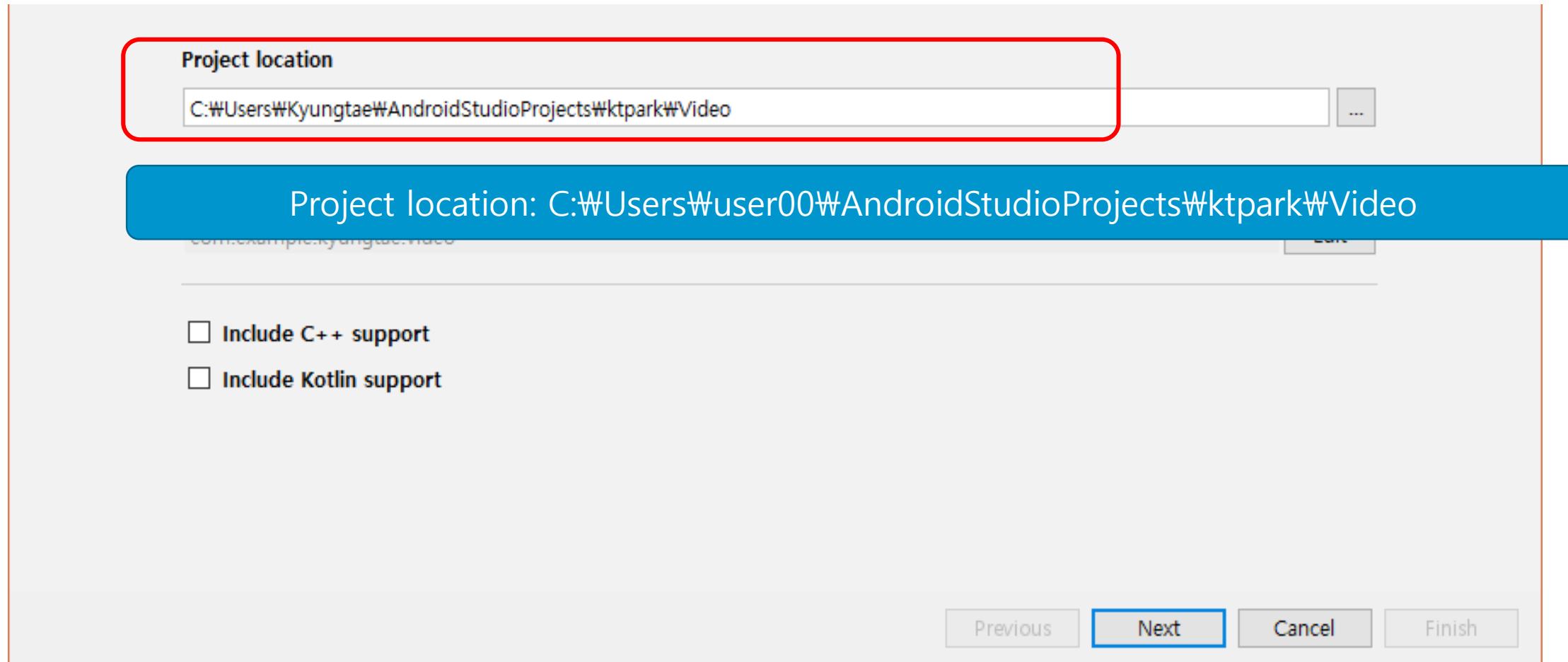
# Create Android Project

8



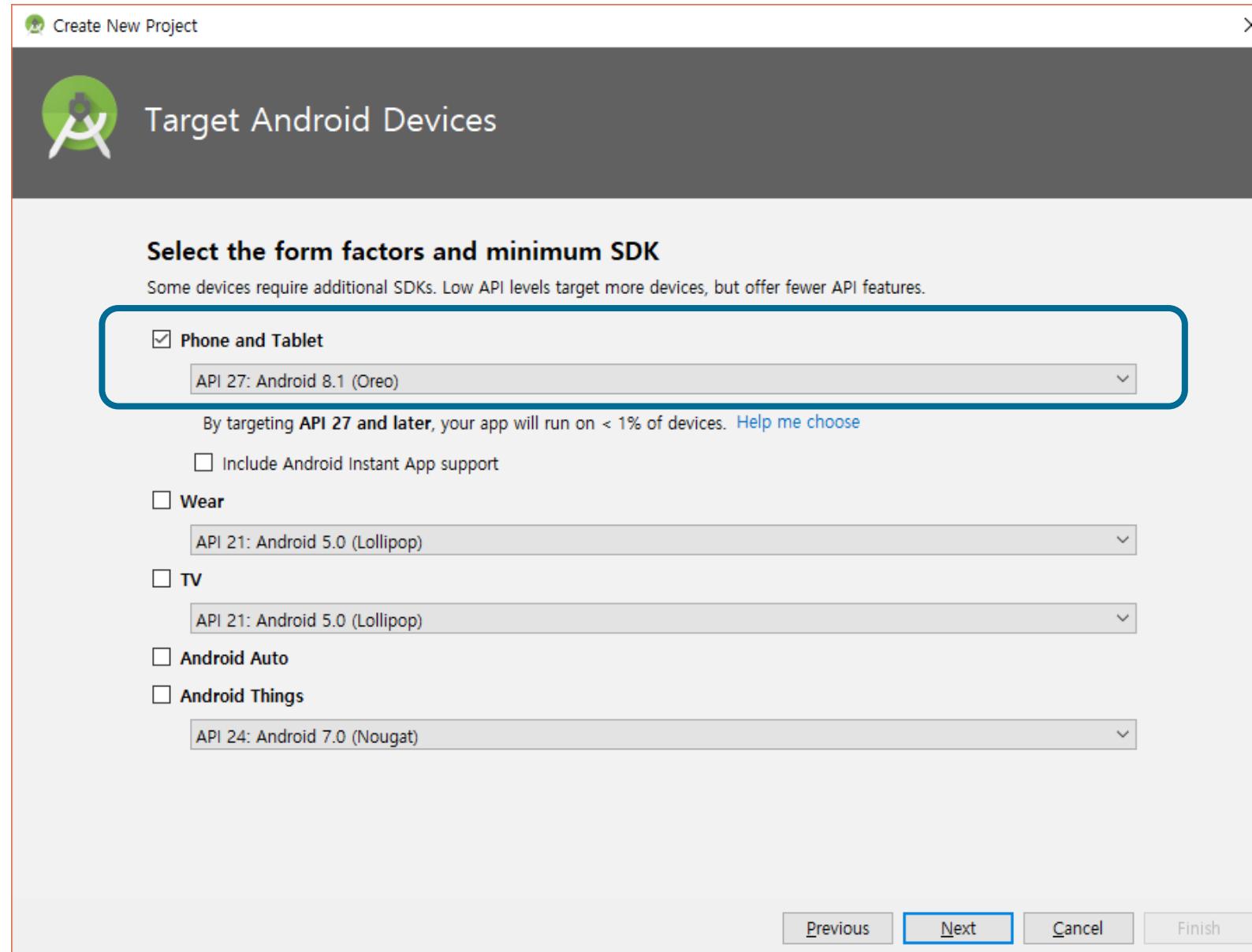
# Create Android Project

9



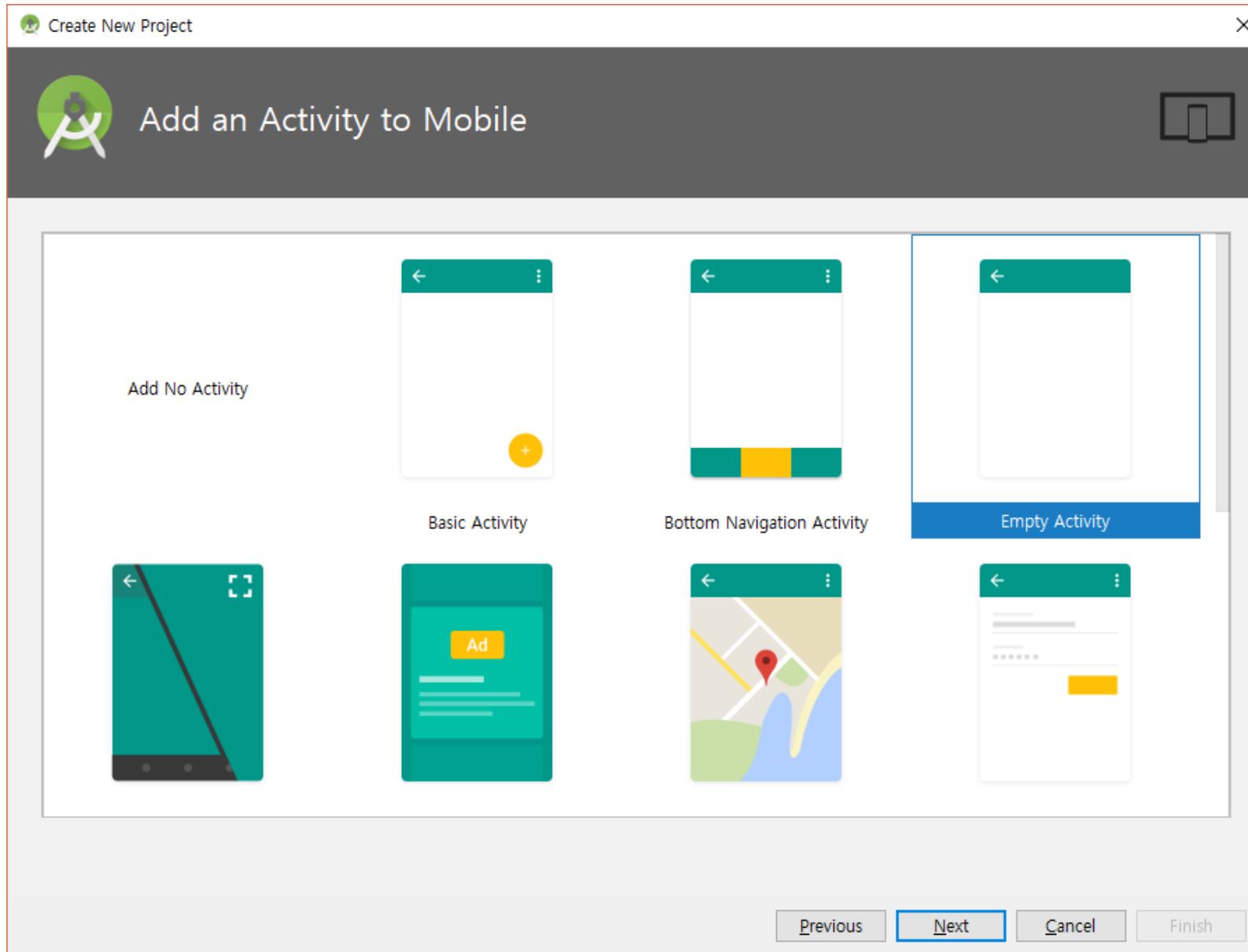
# Target Android Devices

11



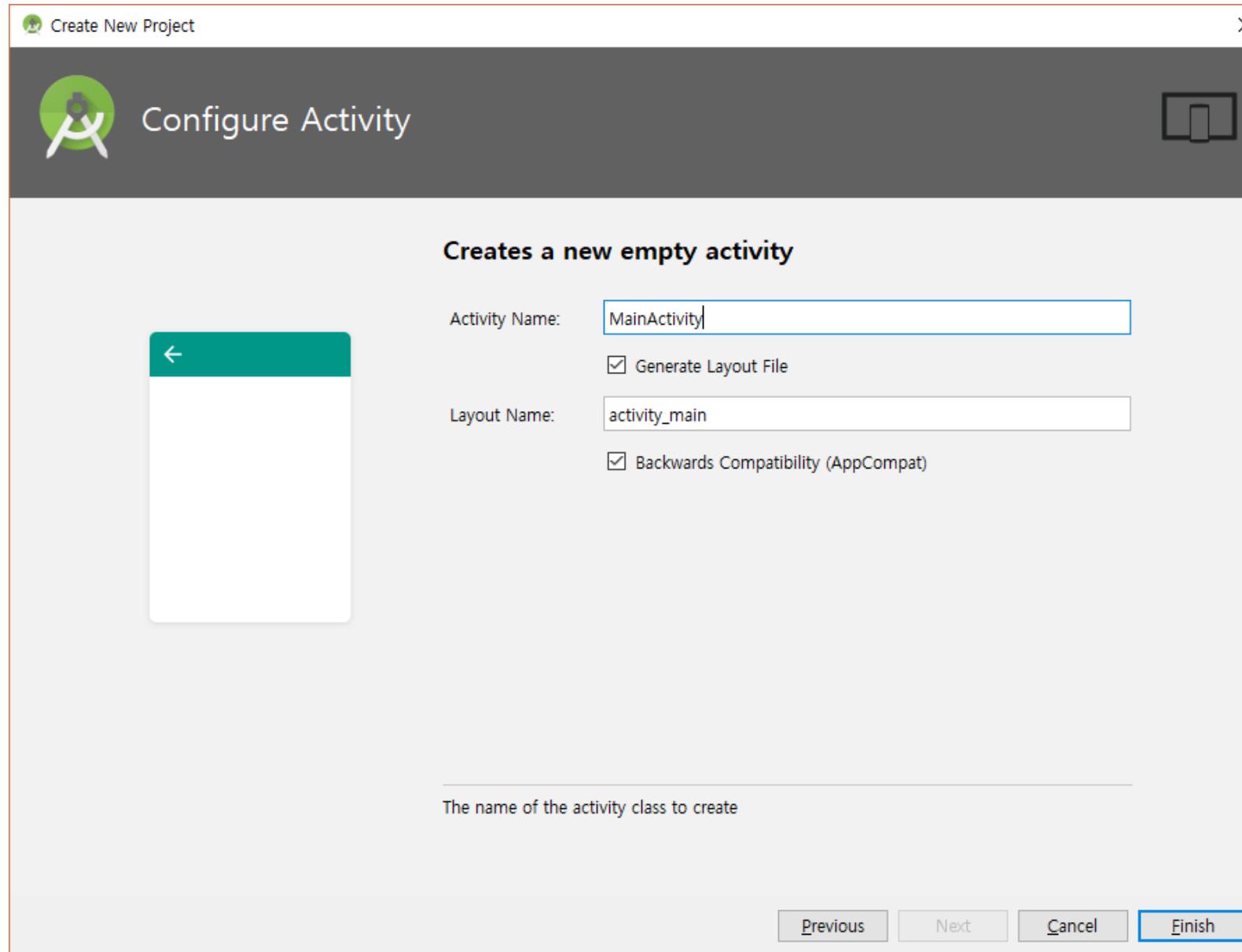
# Add an Activity to Mobile

12



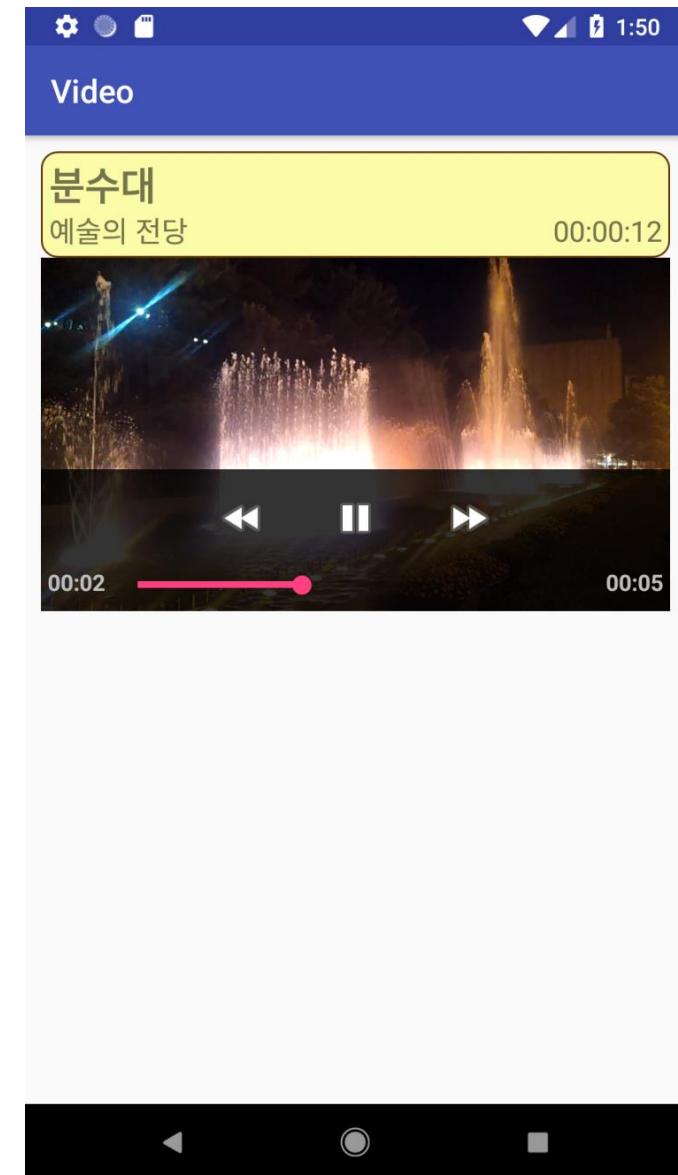
# Configure Activity

13



# Step 0. 프로젝트 개요

14



# Step 1. 프로젝트 생성

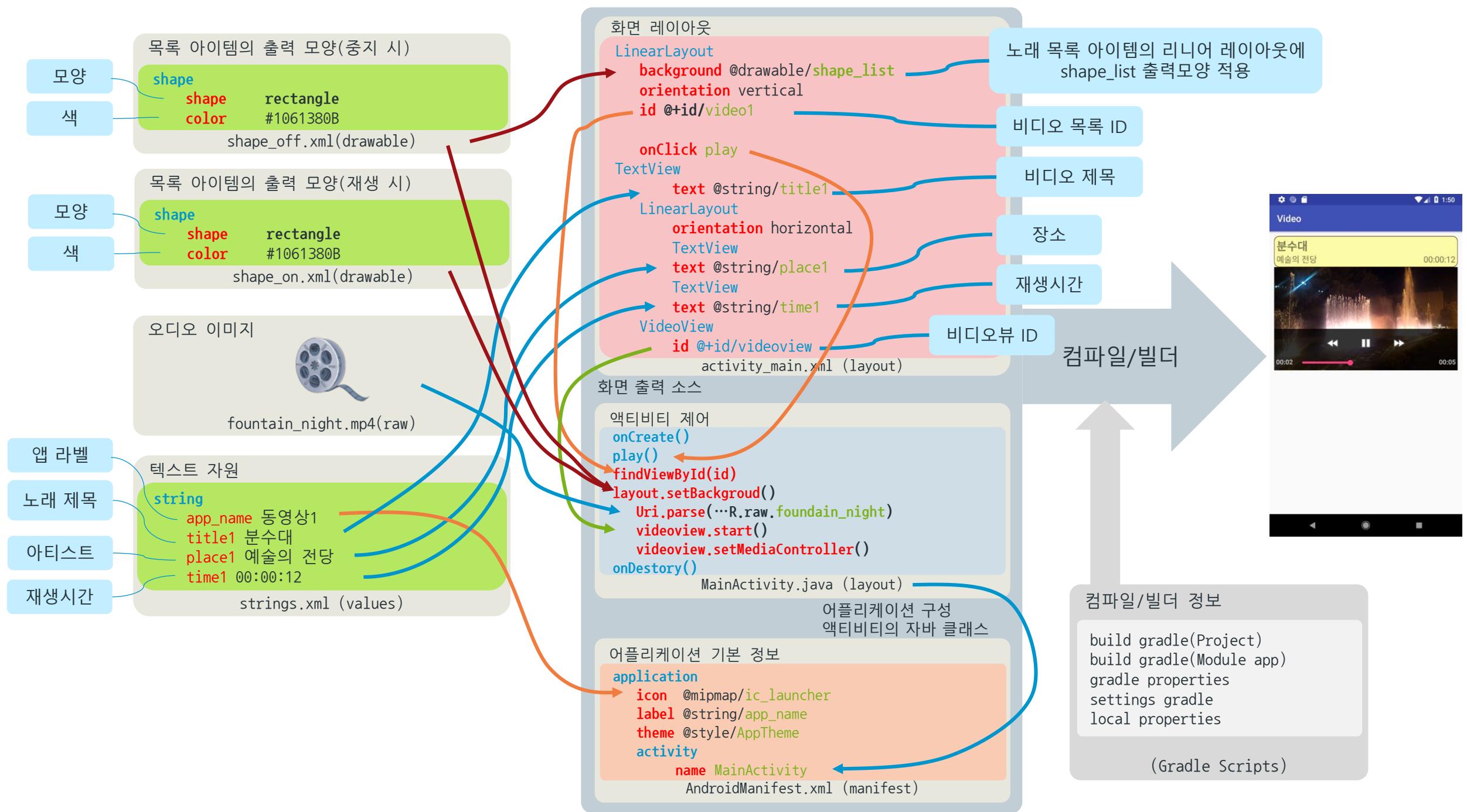
15

절차	내용
①프로젝트 시작	메뉴에서 ‘File → New Project’ 클릭
②프로젝트 구성	Application Name: Video
	Company Domain: 사용자계정.example.com(디폴트 사용)
	Project location: ~\user00\AndroidStudioProject\ktpark\Video
③제품형태	Phone and Tablet(사용할 안드로이드 버전 지정: Android 8.1 Oreo)
④액티비티 유형	Empty Activity
⑤파일 옵션	Activity Name: MainActivity(디폴트 사용)
	Layout Name: activity_main(디폴트 사용)

# Step 2. 파일 편집

16

모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	
java	com.example.사용자계정.video	MainActivity.java	<ul style="list-style-type: none"><li>비디오 목록 출력과 비디오 자동 재생</li></ul>
	drawable	shape_off.xml	<ul style="list-style-type: none"><li>목록 아이템에 대한 출력 스타일 설계 (테두리, 패딩, 모서리) – 중지 시</li></ul>
		shape_on.xml	<ul style="list-style-type: none"><li>목록 아이템에 대한 출력 스타일 설계 (테두리, 패딩, 모서리) - 재생시</li></ul>
	layout	activity_main.xml	<ul style="list-style-type: none"><li>비디오 목록의 화면 배치</li><li>목록 아이템에 출력 모양 적용 (shape_list.xml)</li></ul>
res	mipmap	ic_launcher.png	
	raw	fountain_night.mp4	<ul style="list-style-type: none"><li>비디오 파일</li></ul>
	values	colors.xml	
		dimens.xml	<ul style="list-style-type: none"><li>화면의 구성자원 크기(여백, 글자크기 등)</li></ul>
		strings.xml	<ul style="list-style-type: none"><li>어플리케이션 라벨</li><li>비디오에 대한 제목, 제작자, 재생시간에 대한 텍스트 리소스 정의</li></ul>
		styles.xml	



# Step 2.1 이미지 파일 복사

18

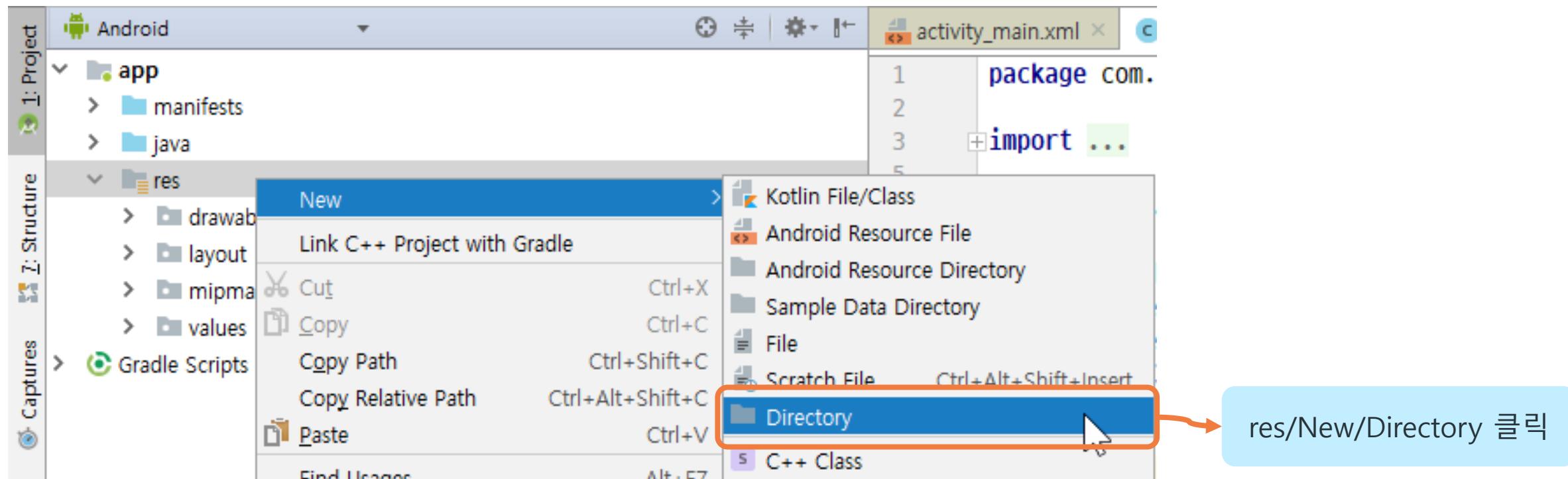
- res 폴더에 있는 **raw** 폴더에 fountain\_night.mp4 파일 저장

모듈	폴더	소스 파일	내용
res	raw	fountain_night.mp4	동영상 파일

res/raw 폴더에  
비디오 파일 올리기

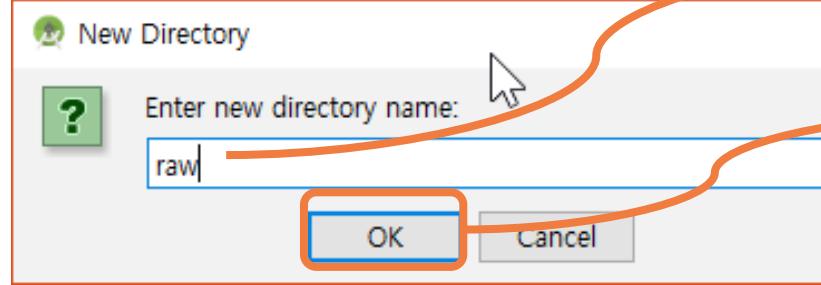
# drawable/raw 폴더에 오디오 파일 추가하기

- app → res → New → Directory 클릭



## • 폴더 이름 작성

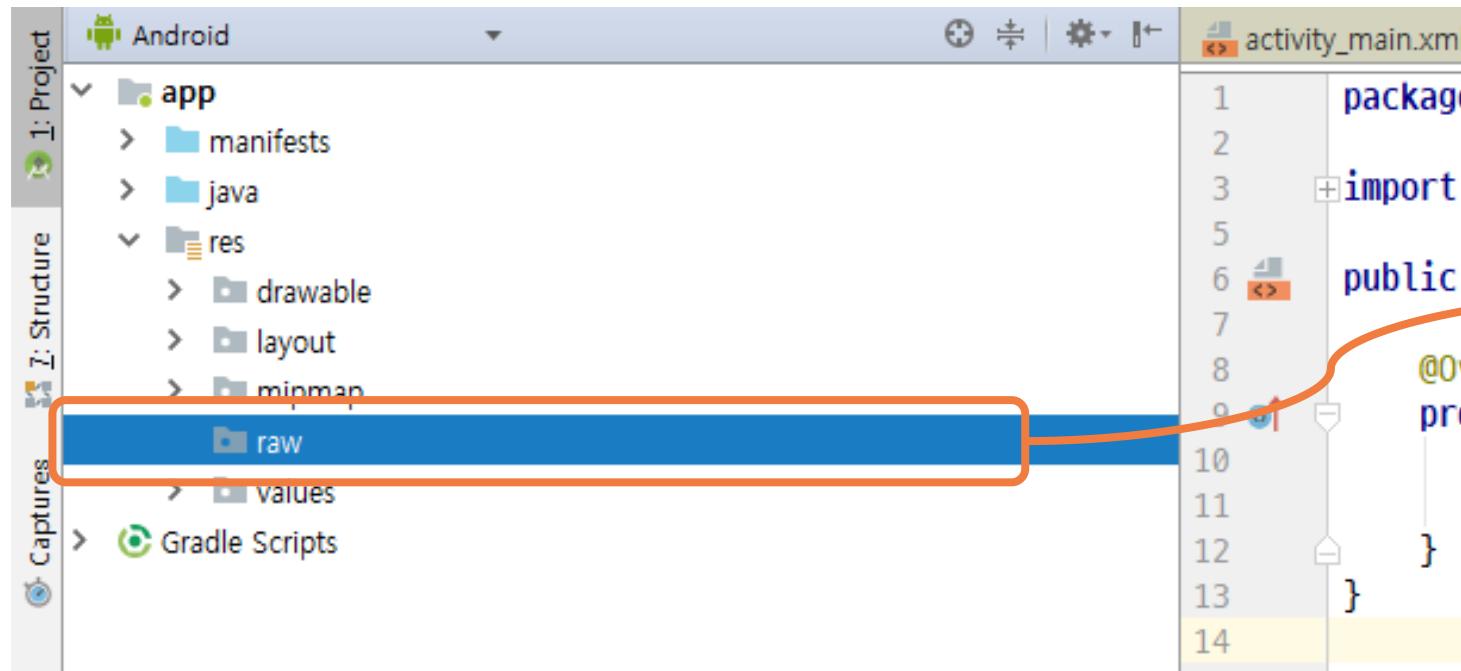
21



폴더 이름: raw

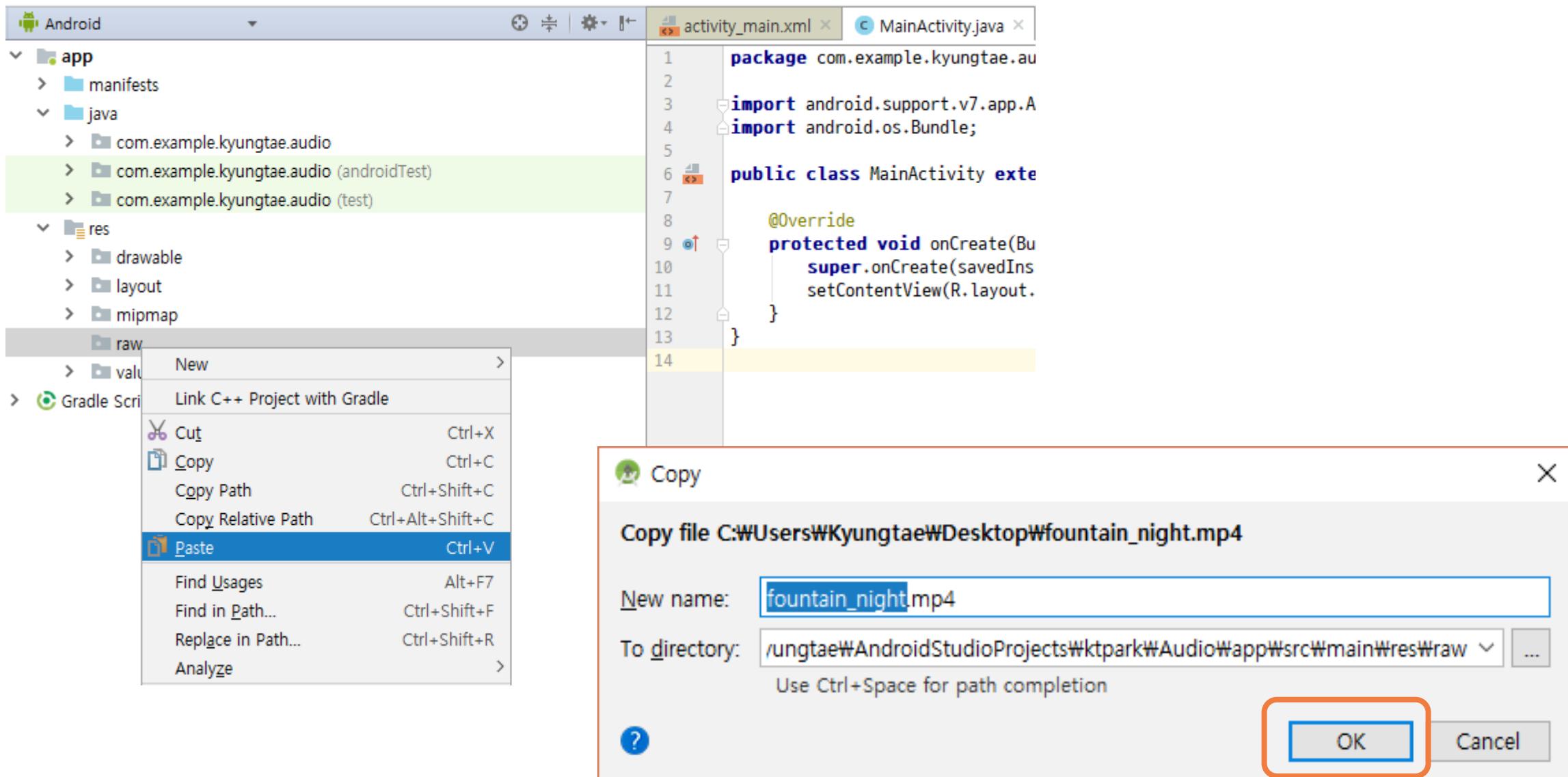
OK 클릭

## • 실행 결과



raw 폴더 생성됨

# fountain\_night.mp4 추가하기



# fountain\_night.mp4

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left pane displays the project structure under the `app` module. It includes `manifests`, `java` (containing `com.example.kyungtae.audio`), `res` (containing `drawable`, `layout`, `miomap`, and `raw`), and `values`. The `raw` folder contains a file named `fountain_night.mp4`, which is highlighted with a red rectangle.
- MainActivity.java:** The right pane shows the Java code for `MainActivity`. The code imports `AppCompatActivity` and `Bundle`, extends `AppCompatActivity`, and overrides the `onCreate` method to set the content view to `R.layout.activity_main`.
- Toolbars:** The top bar includes standard Android Studio icons for creating new files, opening recent projects, and navigating between tabs.

```
1 package com.example.kyungtae.audio;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 public class MainActivity extends AppCompatActivity {
7
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12     }
13 }
14
```

# Step 2.2 텍스트 자원의 편집

- strings.xml

The screenshot shows the Android Studio interface with the following details:

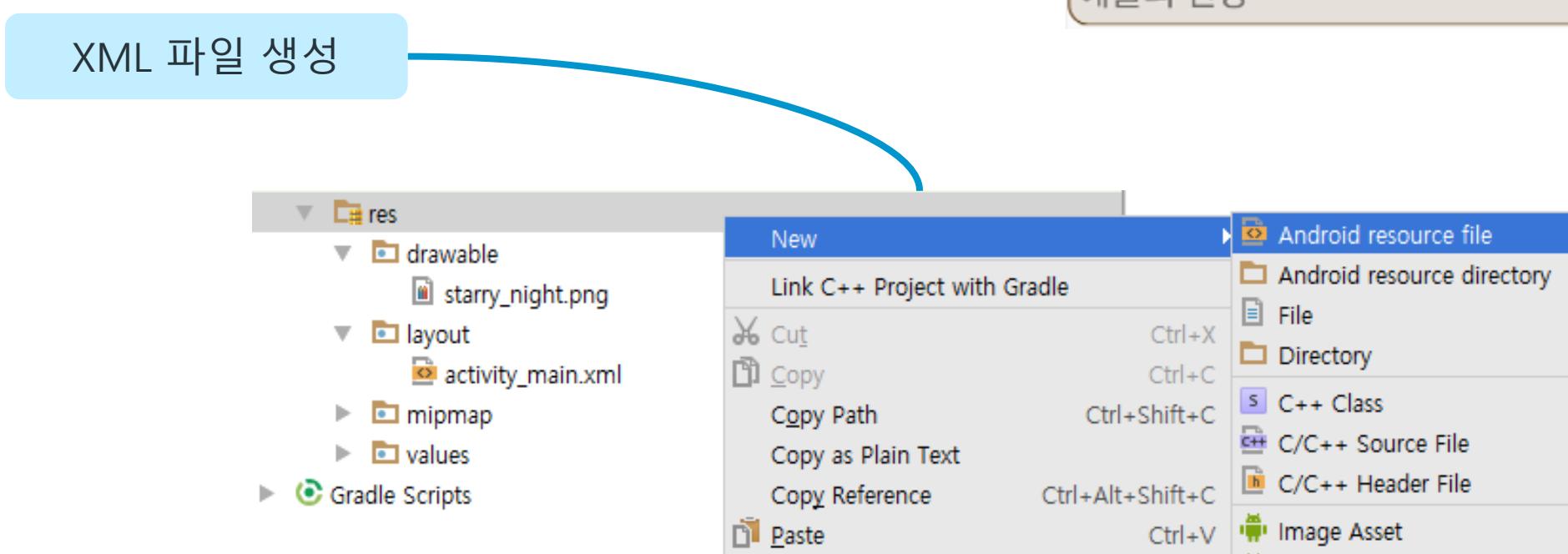
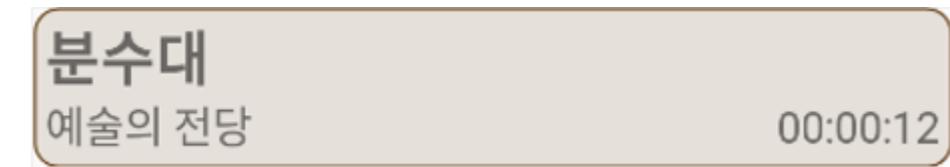
- Project Structure:** The left sidebar shows the project structure under "app". Key folders include "manifests", "java", "res" (with "drawable", "layout", "mipmap", and "raw" subfolders), and "values" (with "colors.xml", "strings.xml", and "styles.xml"). The "strings.xml" file is highlighted with an orange rectangle.
- Editor Tab:** The top bar shows tabs for "activity\_main.xml", "MainActivity.java", and "strings.xml".
- Editor Content:** The main editor area displays the XML code for "strings.xml".

```
<resources>
    <string name="app_name">Video</string>
    <string name="title">분수대</string>
    <string name="place">예술의 전당</string>
    <string name="time">00:00:12</string>
</resources>
```

# Step 2.3 Drawable Resource 추가 및 편집

27

- **shape\_list.xml** 생성(res/drawable 폴더)
  - drawable resource를 이용한 그림 출력



- Set New Resource File

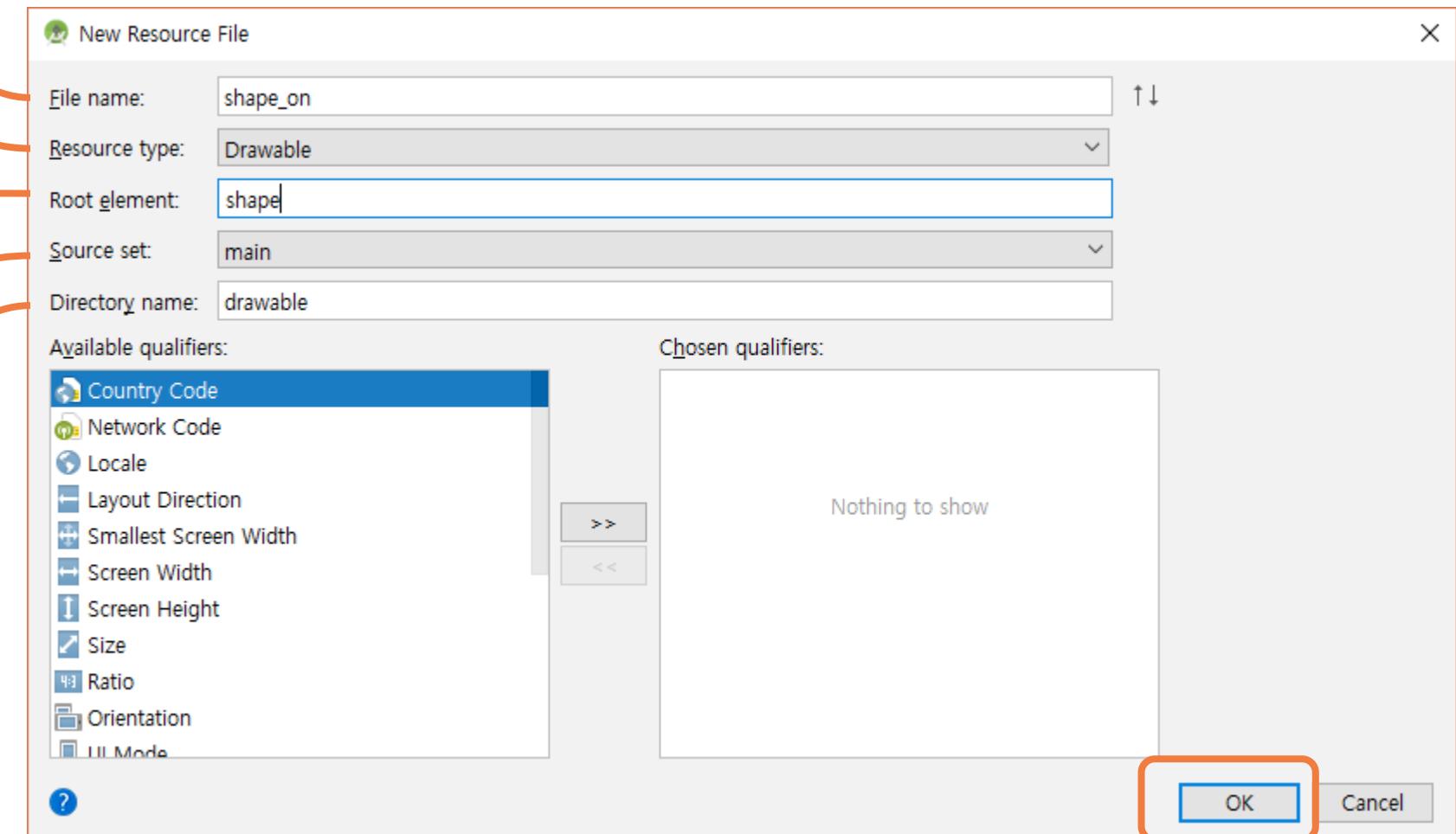
File name: shape\_on

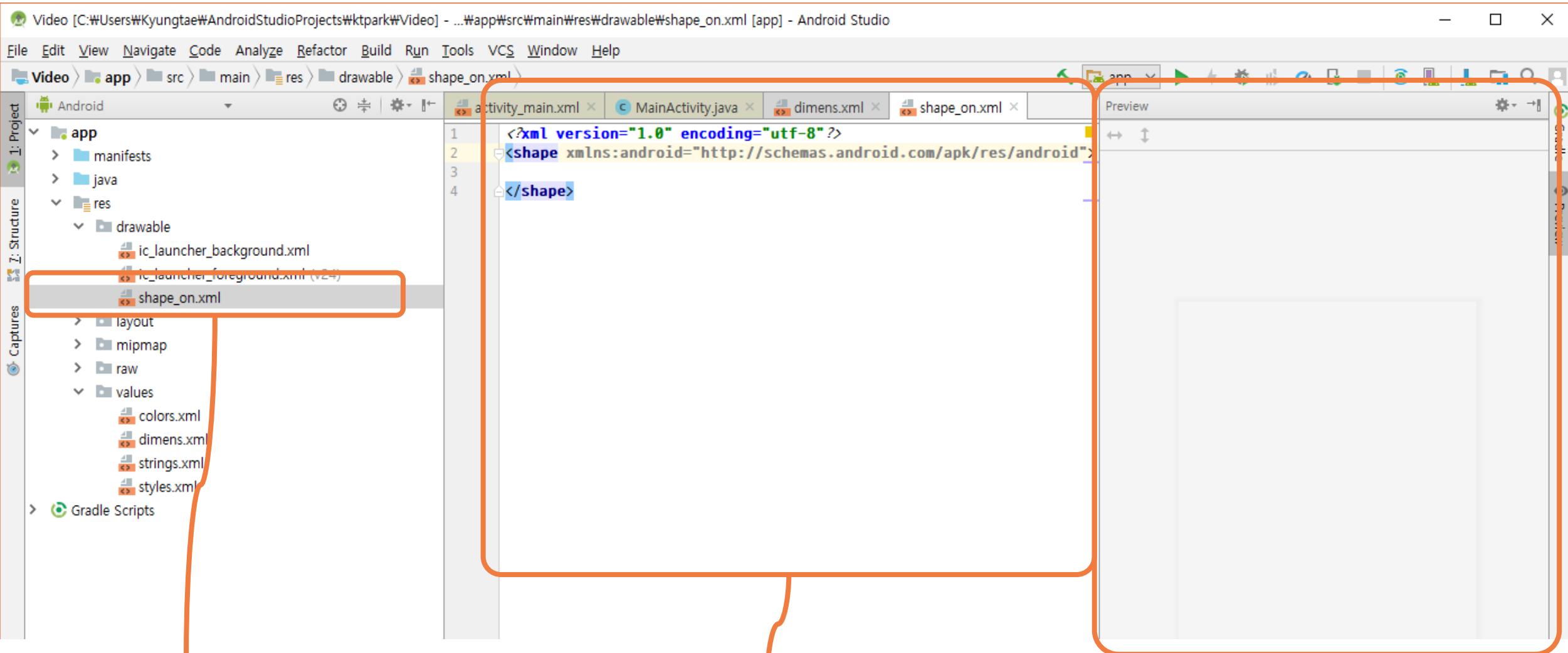
Resource type: Drawable

Root element: shape

Source set: main

Directory name: drawable





shape\_on.xml 파일

shape\_on.xml 파일의  
텍스트 코딩 영역

shape\_on 파일에 의한  
shape 미리보기 영역

## • shape\_on.xml 소스(동영상 재생 시)

출력모양을 사각형으로 지정

출력모양을 내부의 색

출력모양을 테두리의 색

내부 패딩 정보

출력모양 모서리를 둥근 모양  
으로 지정(반지름은 5dp)

```
<?xml version="1.0" encoding="utf-8" ?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

    <solid android:color="#ffff00"/>
    <stroke android:width="1dp" android:color="#61380B"/>
    <padding android:bottom="5dp"
        android:left="5dp"
        android:right="5dp"
        android:top="5dp"/>
    <corners android:radius="10dp"/>
</shape>
```

- shape\_off.xml 소스(동영상 중지 시)

The screenshot shows the Android Studio interface with the XML Editor open. The tab bar at the top includes activity\_main.xml, MainActivity.java, dimens.xml, shape\_on.xml, shape\_off.xml (which is the active tab), and Preview. The code editor displays the following XML:

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">

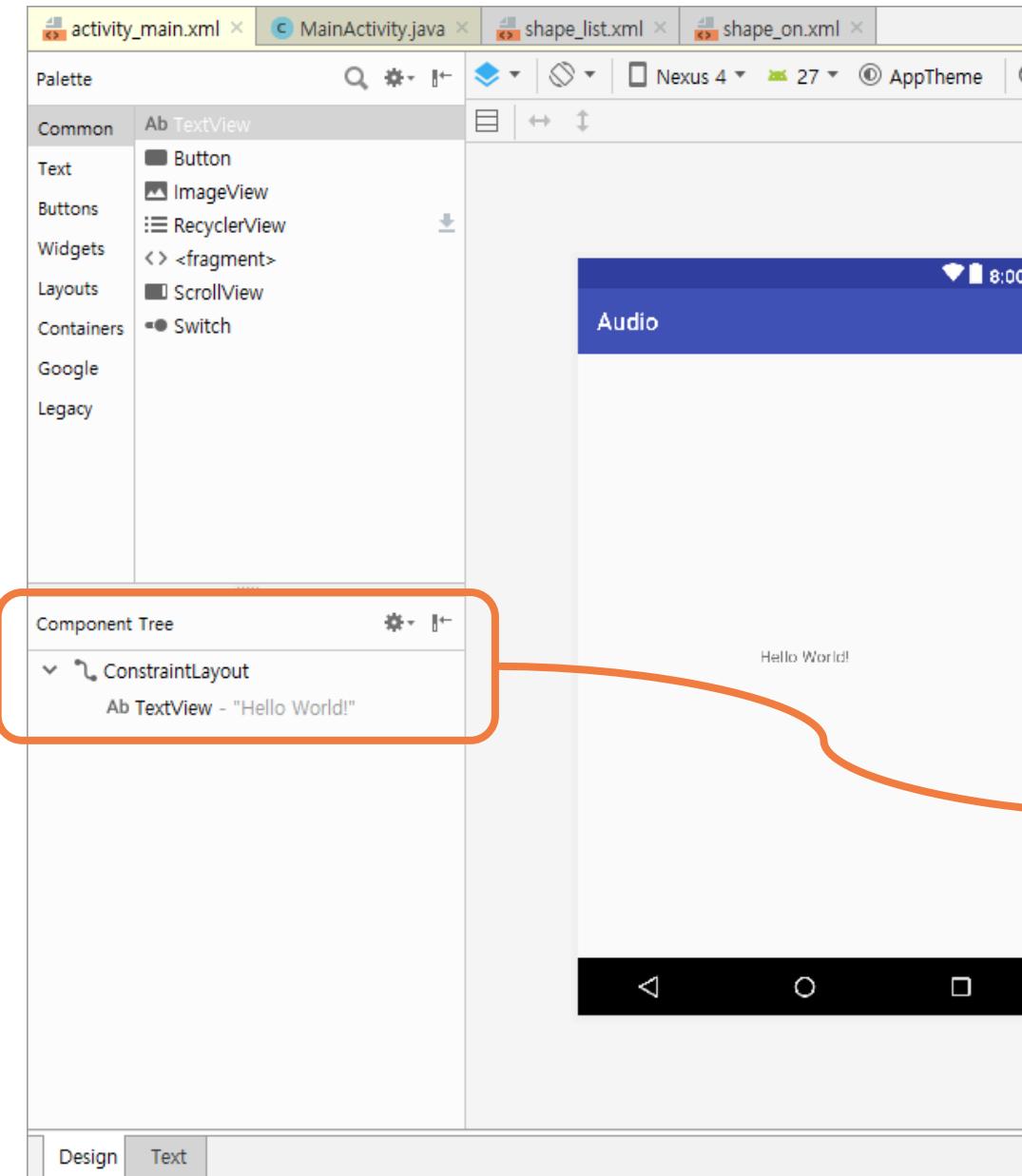
    <solid android:color="#2261380B"/>
    <stroke android:width="1dp" android:color="#61380B"/>

    <padding android:bottom="5dp"
        android:left="5dp"
        android:right="5dp"
        android:top="5dp"/>

    <corners android:radius="10dp"/>
</shape>
```

The code editor highlights several lines of code in yellow, specifically lines 5 through 11, which define the padding for the shape. To the right of the code editor is a preview window showing a light brown rectangle with rounded corners and a thin brown border, representing the current state of the shape definition.

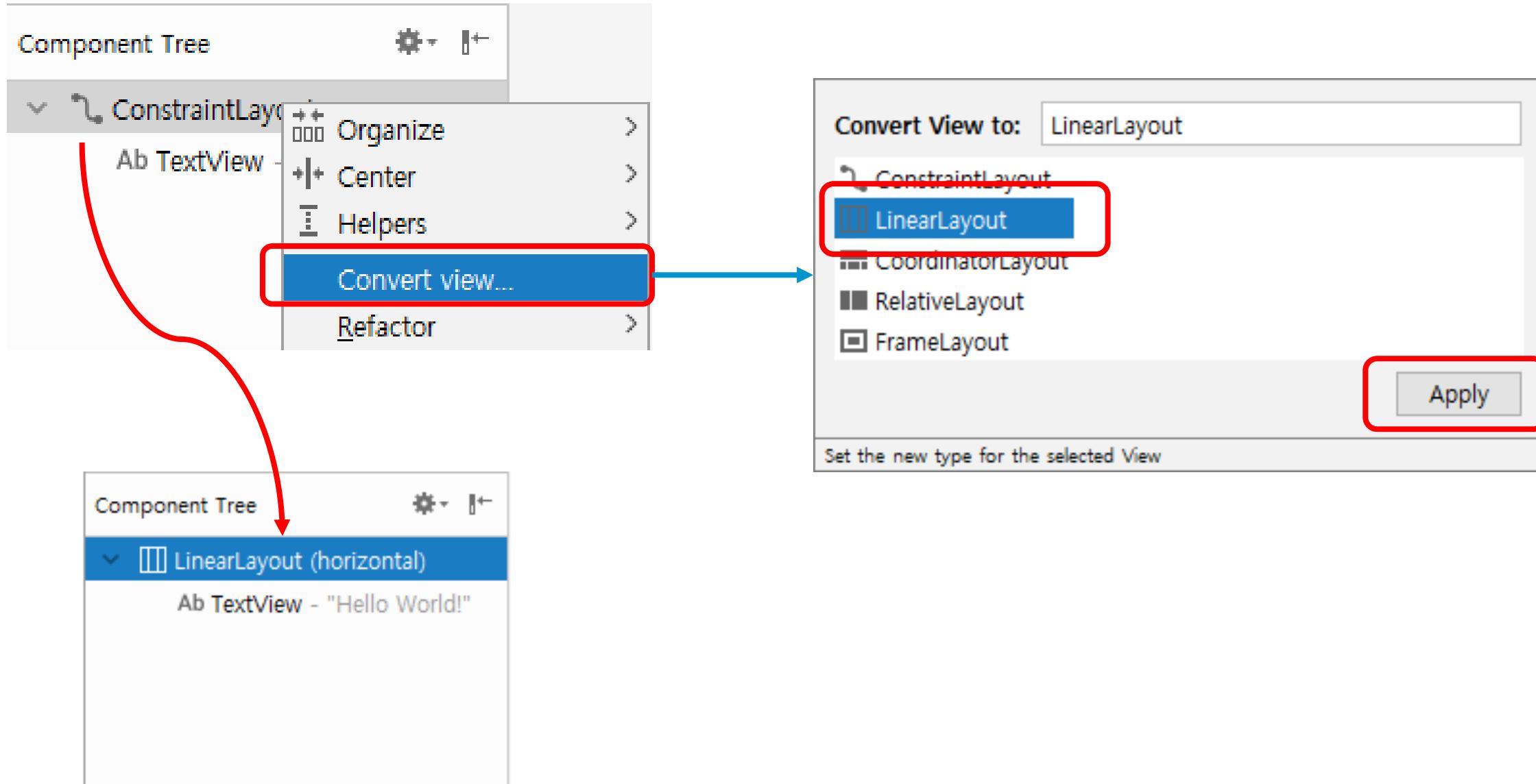
## 2.4 화면 설계



ConstraintLayout →  
LinearLayout으로 변경

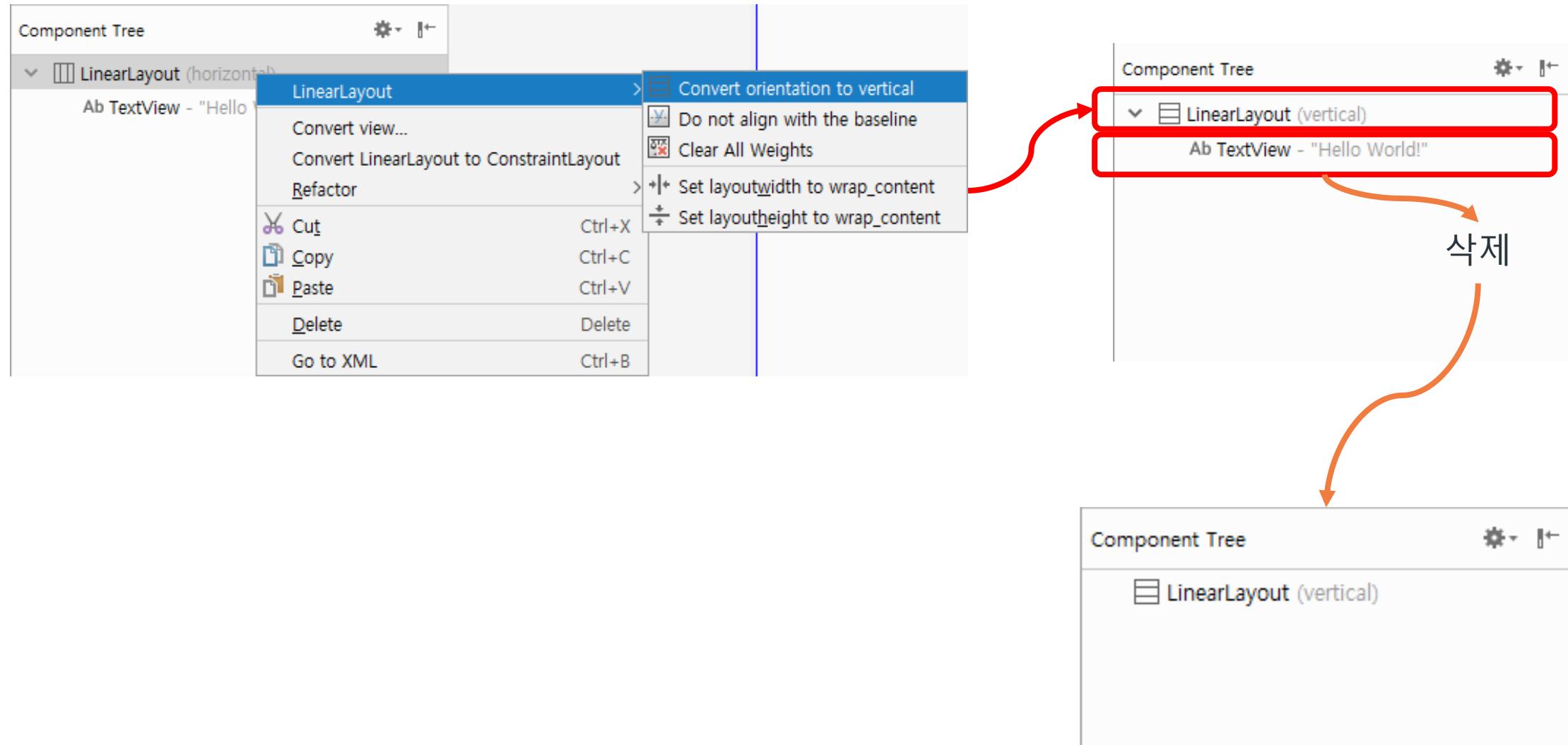
# ConstraintLayout을 LinearLayout로 바꾸기

33

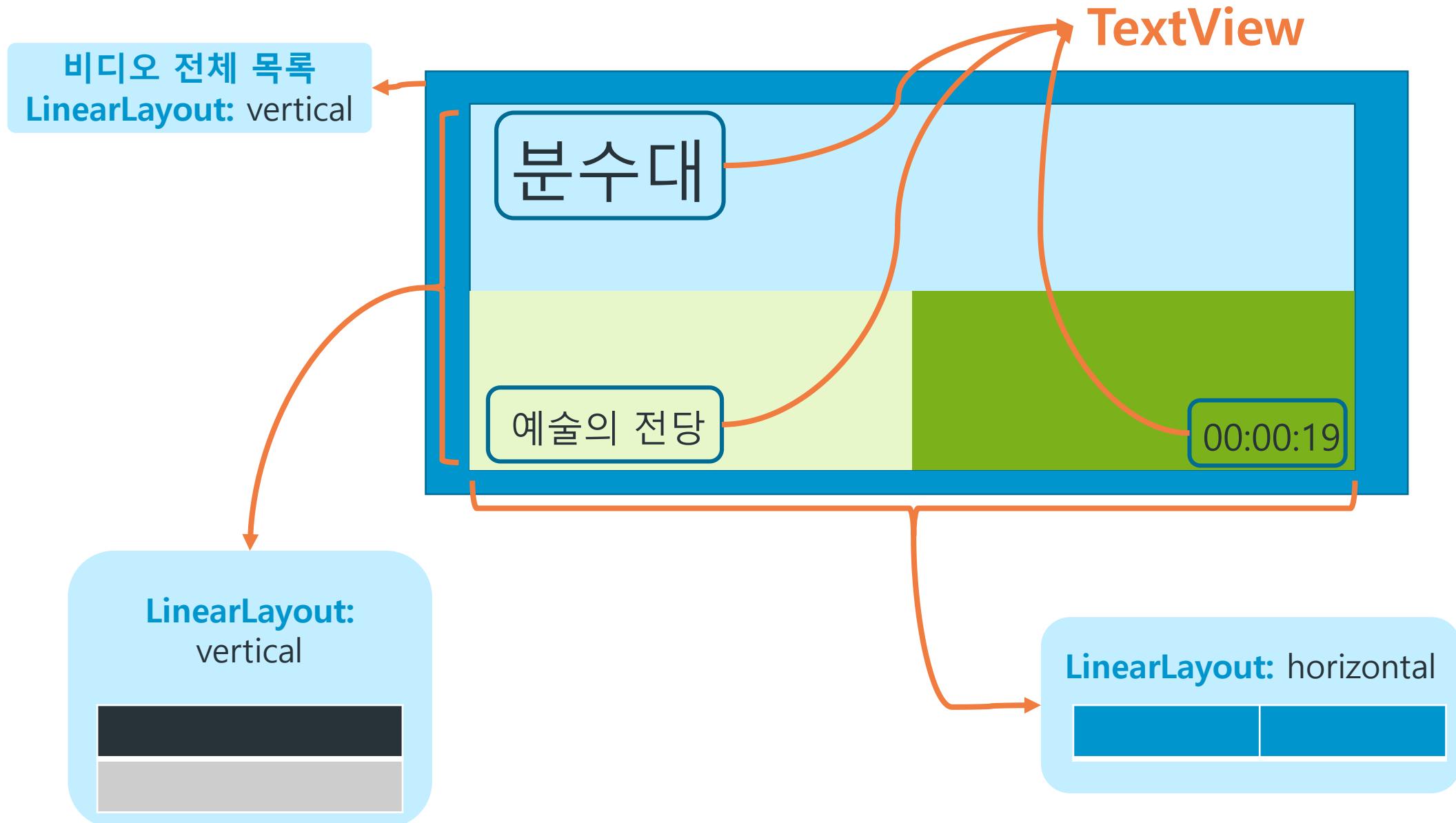


# LinearLayout의 방향을 Horizontal → Vertical로 변경하기

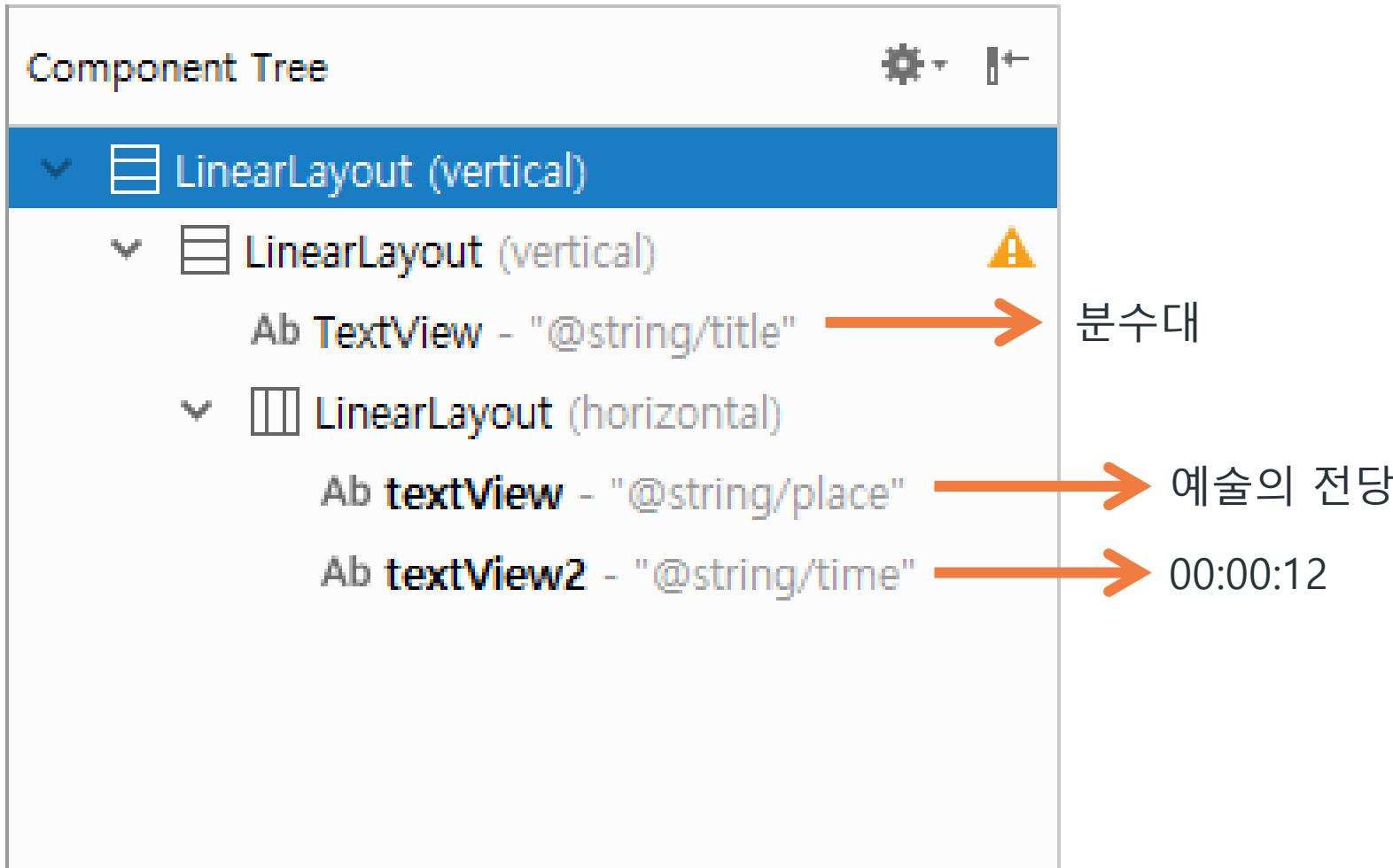
34



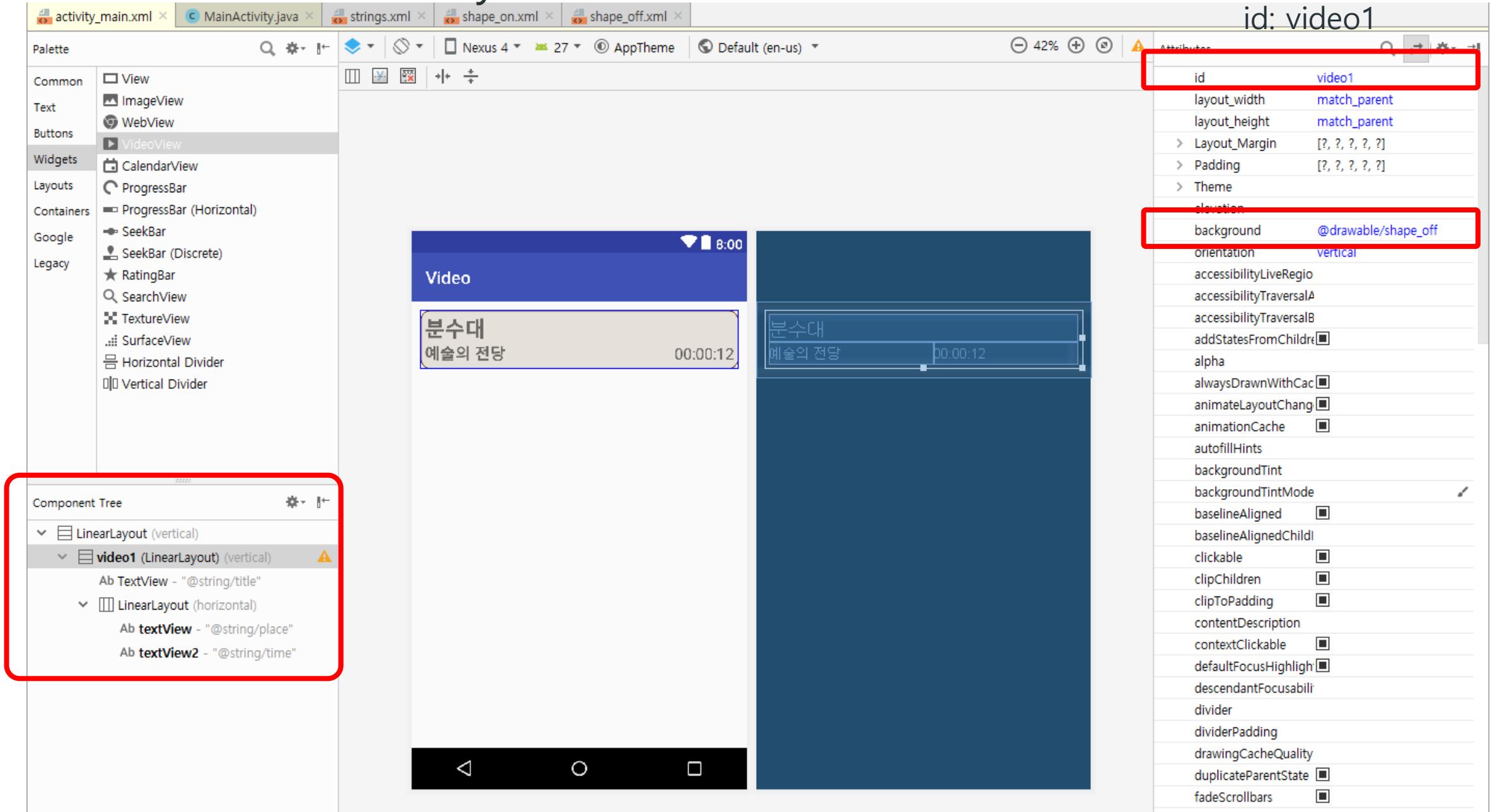
- 비디오 목록 표시를 위한 Layout 구조

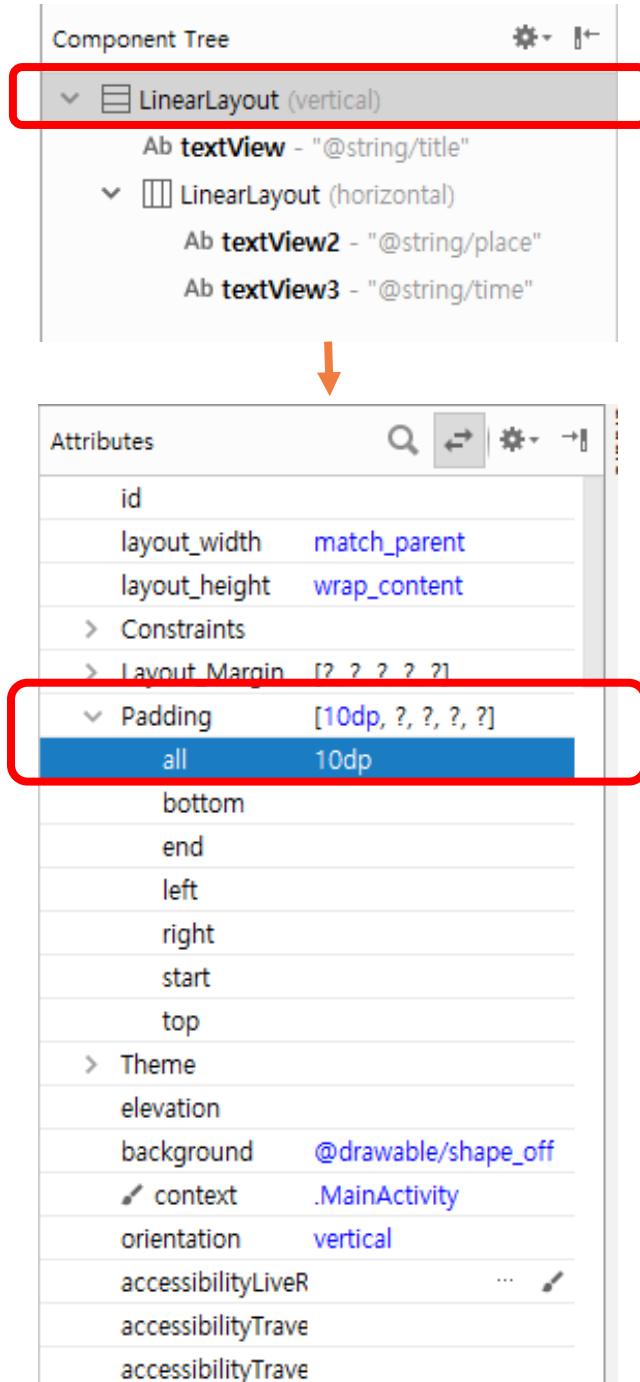
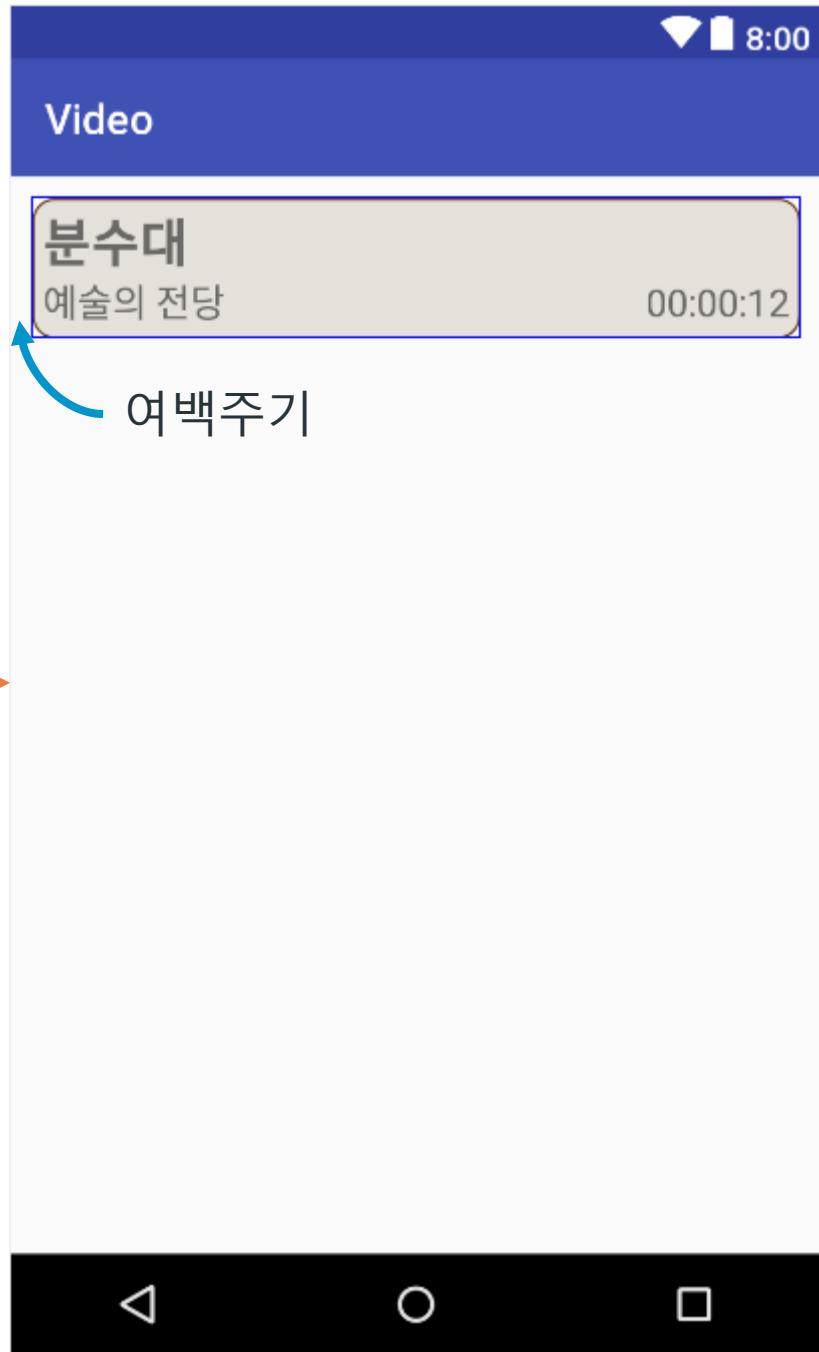
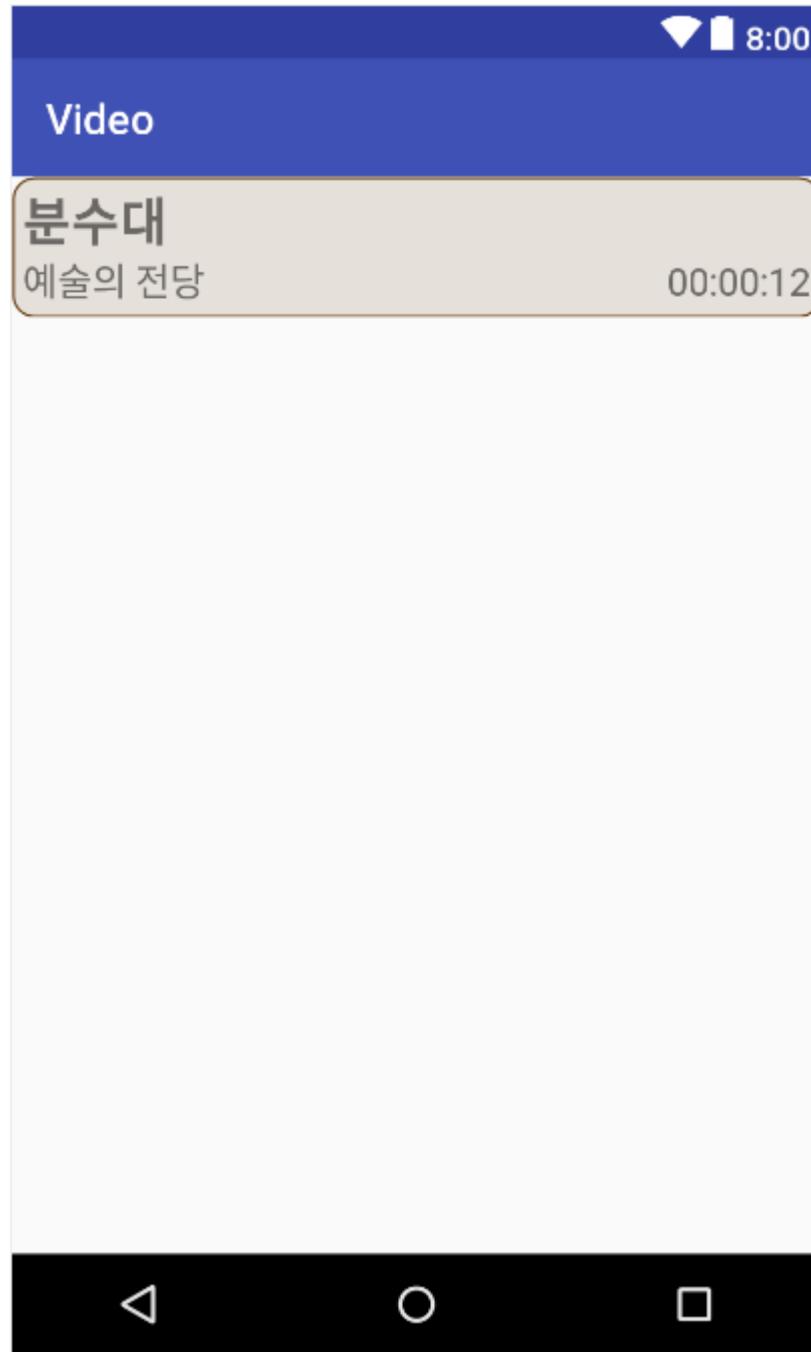


- 동영상 내용 표시를 위한 Layout 구조-Component Tree

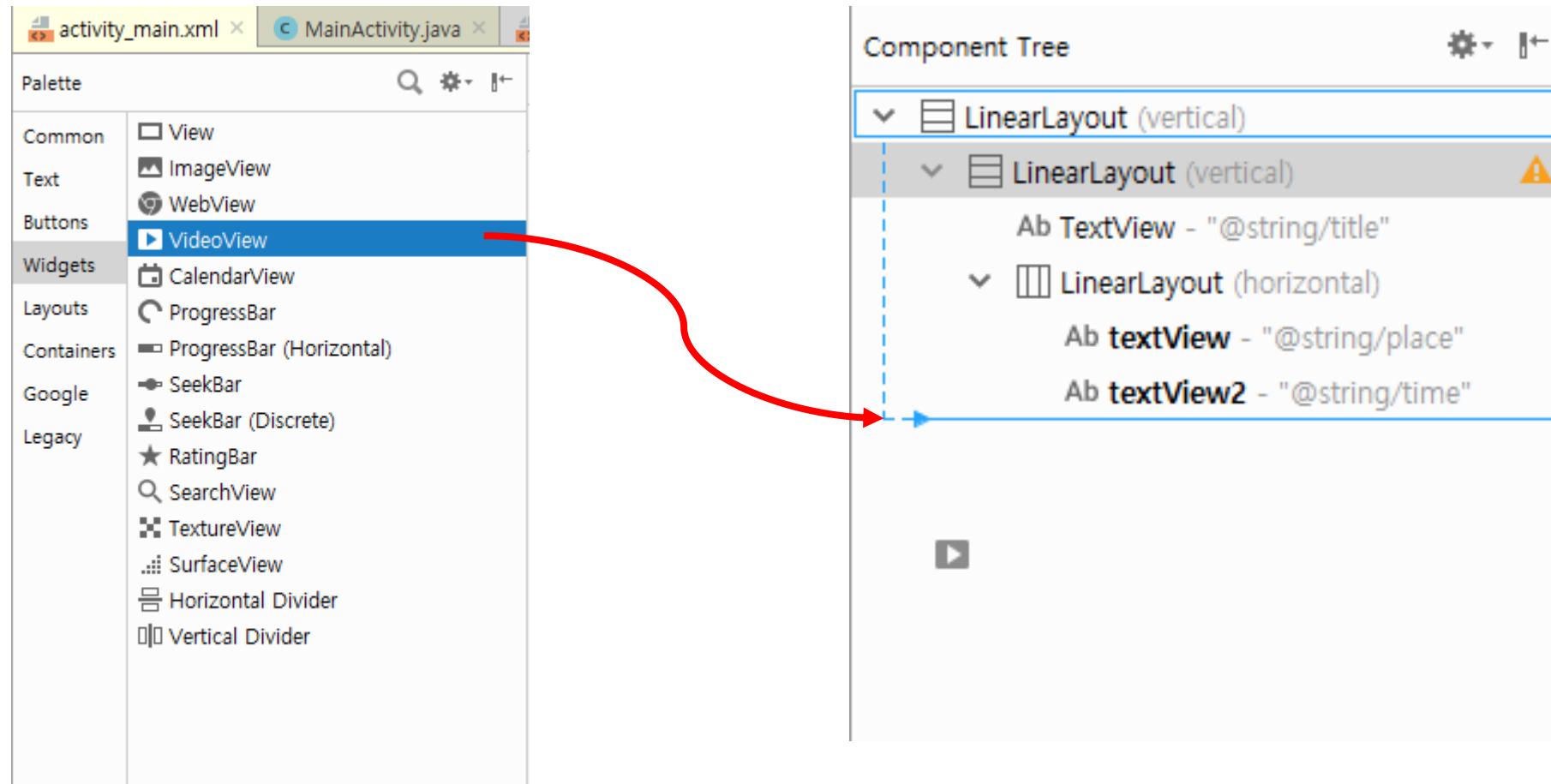


# • 동영상 제목 표시 Layout

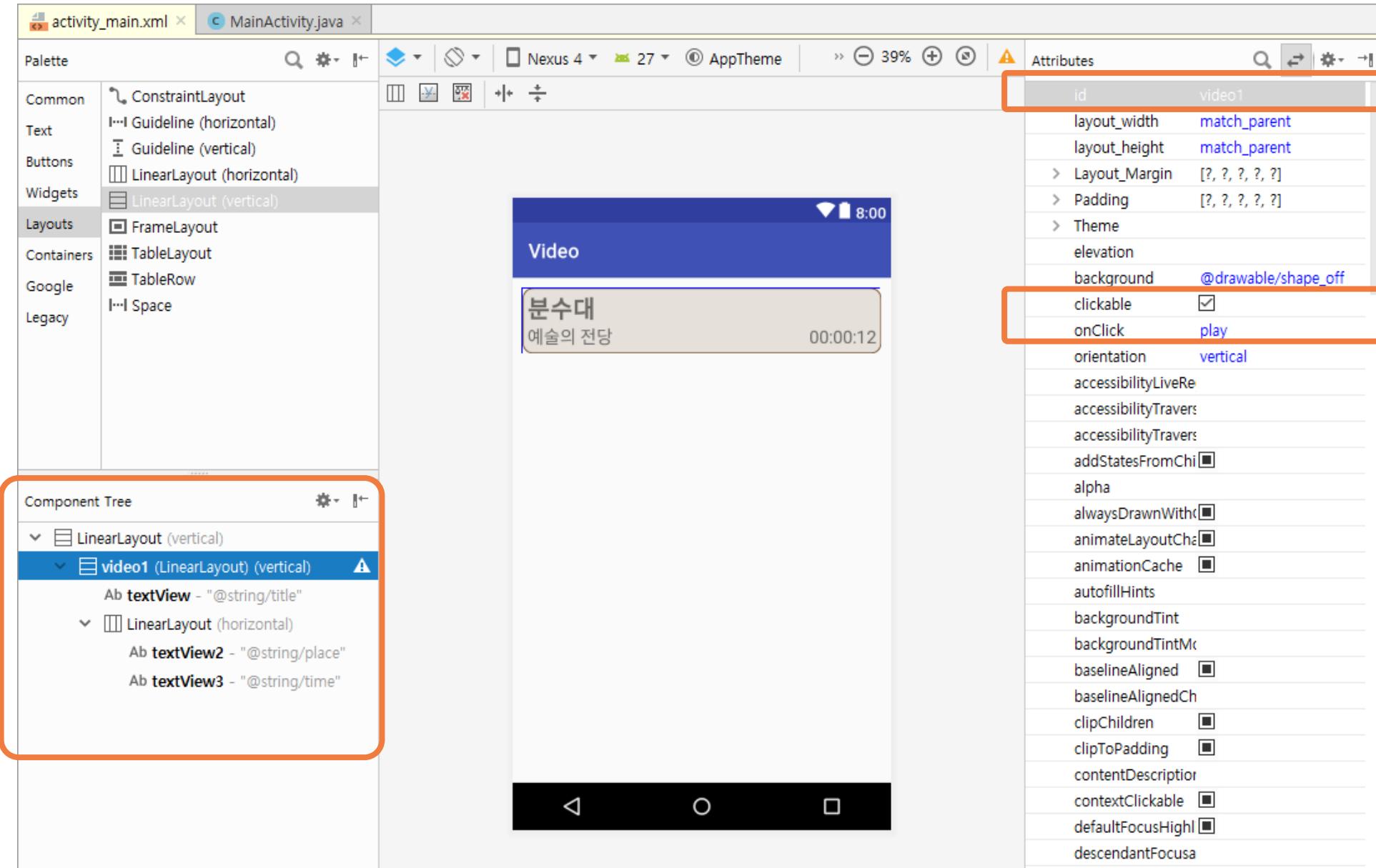




# 동영상 VideoView 추가



# 동영상 제목 LinearLayout 설정



activity\_main.xml   MainActivity.java   strings.xml   shape\_on.xml   shape\_off.xml

Nexus 4   27   AppTheme   Default (en-us)

42%   Attributes

Palette   Common   Text   Buttons   Widgets   Layouts   Containers   Google   Legacy

View   ImageView   WebView   VideoView   CalendarView   ProgressBar   ProgressBar (Horizontal)   SeekBar   SeekBar (Discrete)   RatingBar   SearchView   TextureView   SurfaceView   Horizontal Divider   Vertical Divider

id   videoView  
layout\_width   match\_parent  
layout\_height   wrap\_content  
Layout\_Margin   [?, ?, ?, ?]  
Padding   [?, ?, ?, ?]  
Theme  
elevation  
accessibilityLiveRegion  
accessibilityTraversalA  
accessibilityTraversalB  
alpha  
autofillHints  
background  
backgroundTint  
backgroundTintMode  
clickable     
contentDescription  
contextClickable     
defaultFocusHighlight     
drawingCacheQuality  
duplicateParentState     
fadeScrollbars     
fadingEdge     
fadingEdgeLength  
filterTouchesWhenObserved     
fitsSystemWindows     
focusable  
focusableInTouchMode     
focusedByDefault     
forceHasOverlappingRegions  
foreground  
foregroundGravity     
foregroundTint  
foregroundTintMode  
hapticFeedbackEnabled     
importantForAccessibility  
importantForAutofill  

Video

분수대  
예술의 전당 00:00:12

Component Tree

- LinearLayout (vertical)
  - video1 (LinearLayout) (vertical)
    - TextView - "@string/title"
    - LinearLayout (horizontal)
      - textView - "@string/place"
      - textView2 - "@string/time"
  - videoView

The screenshot shows the Android Studio interface with the XML layout editor open. The top navigation bar includes tabs for activity\_main.xml, MainActivity.java, strings.xml, shape\_on.xml, and shape\_off.xml. Below the tabs are toolbars for search, settings, and file operations, followed by device and theme selection. The main workspace displays a preview of an Android application screen titled "Video". The screen contains a title "분수대", a subtitle "예술의 전당", a timestamp "00:00:12", and a large video view area. The bottom of the screen features standard Android navigation icons. To the left of the preview is the "Component Tree" panel, which lists the hierarchical structure of the layout: a vertical LinearLayout containing a video1 sub-view (another vertical LinearLayout with a title TextView and a horizontal LinearLayout containing two textView elements), and a final videoView element. The right side of the editor is the "Attributes" panel, which lists numerous properties for the selected videoView component, many of which have their values set to match\_parent or wrap\_content. The entire screenshot is framed by a red border.

# 2.5 Activity 제어(MainActivity.java)

- 비디오 리소스에 대한 VideoView를 생성

```

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

```

```

1 package com.example.kyungtae.video;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.widget.VideoView;
6
7 public class MainActivity extends AppCompatActivity {
8
9     VideoView
10
11    @Override
12    protected void onCreate(Bundle savedInstanceState) {
13        super.onCreate(savedInstanceState);
14        setContentView(R.layout.activity_main);
15    }
16
17

```

## • VideoView 객체생성과 VideoView 컴포넌트 연결

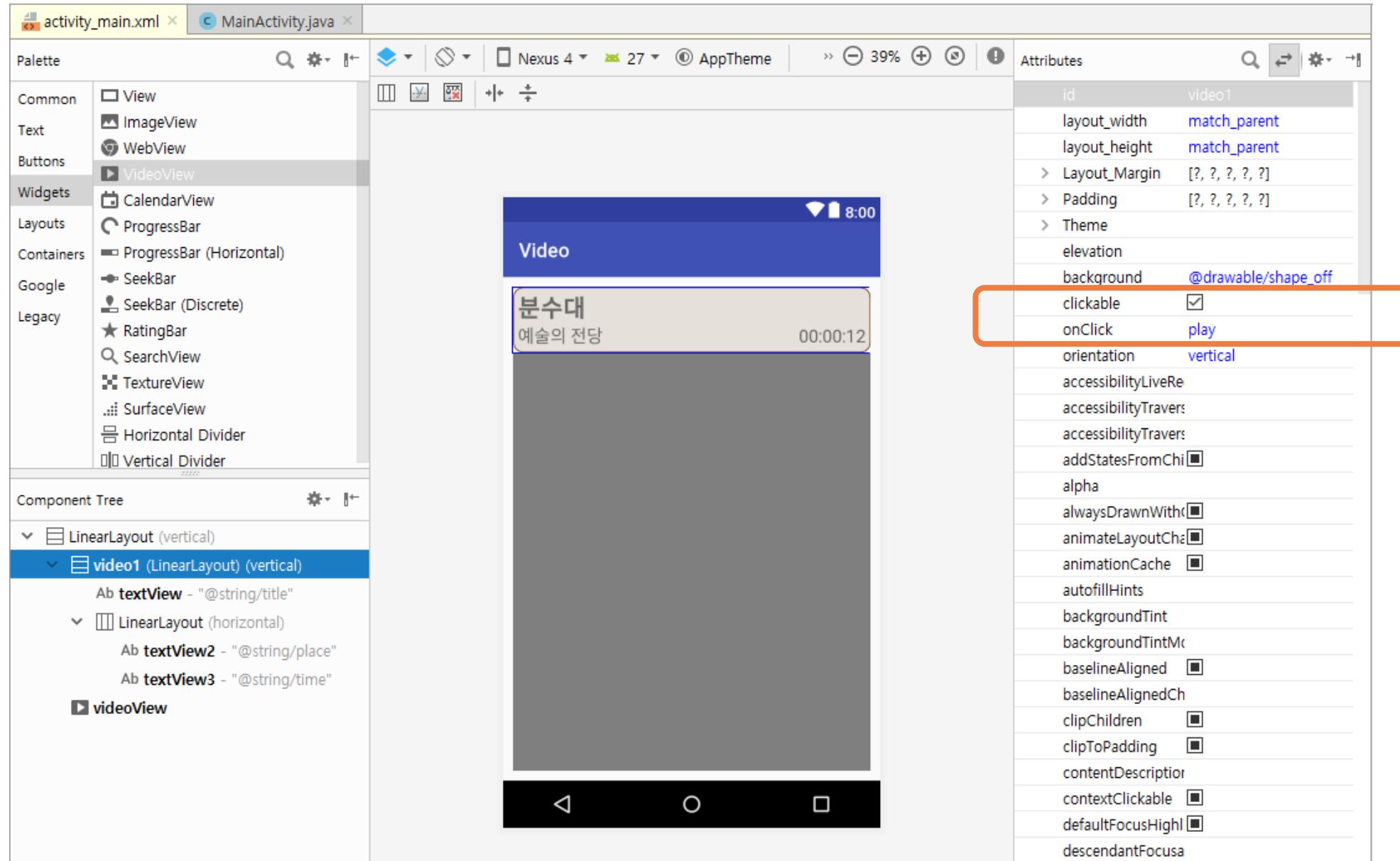
54

The diagram illustrates the connection between the Java code and the XML layout. In the Java code, a VideoView object is created and assigned to the variable `vView`. This variable is highlighted with an orange box and connected by an orange arrow to a callout bubble containing the text "비디오뷰 객체 생성(null)". In the XML layout file, there is a `<VideoView` element with the `id` attribute set to `videoView`. This element is also highlighted with an orange box and connected by an orange arrow to another callout bubble containing the text "비디오뷰 컴포넌트 연결".

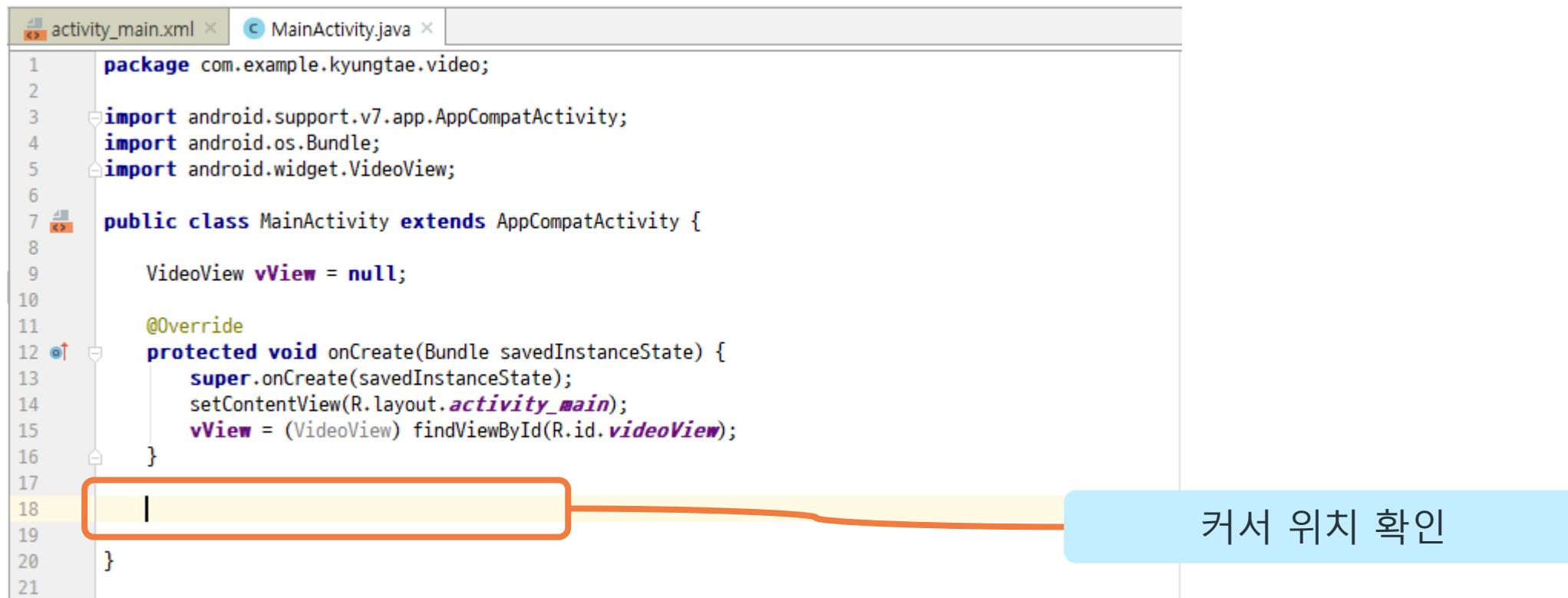
```
activity_main.xml MainActivity.java
```

```
1 package com.example.kyungtae.video;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.widget.VideoView;
6
7 public class MainActivity extends AppCompatActivity {
8
9     VideoView vView = null;
10
11    @Override
12    protected void onCreate(Bundle savedInstanceState) {
13        super.onCreate(savedInstanceState);
14        setContentView(R.layout.activity_main);
15        vView = (VideoView) findViewById(R.id.videoView);
16    }
17
18 }
```

# 동영상 제목 onClick 이벤트 추가하기-play()



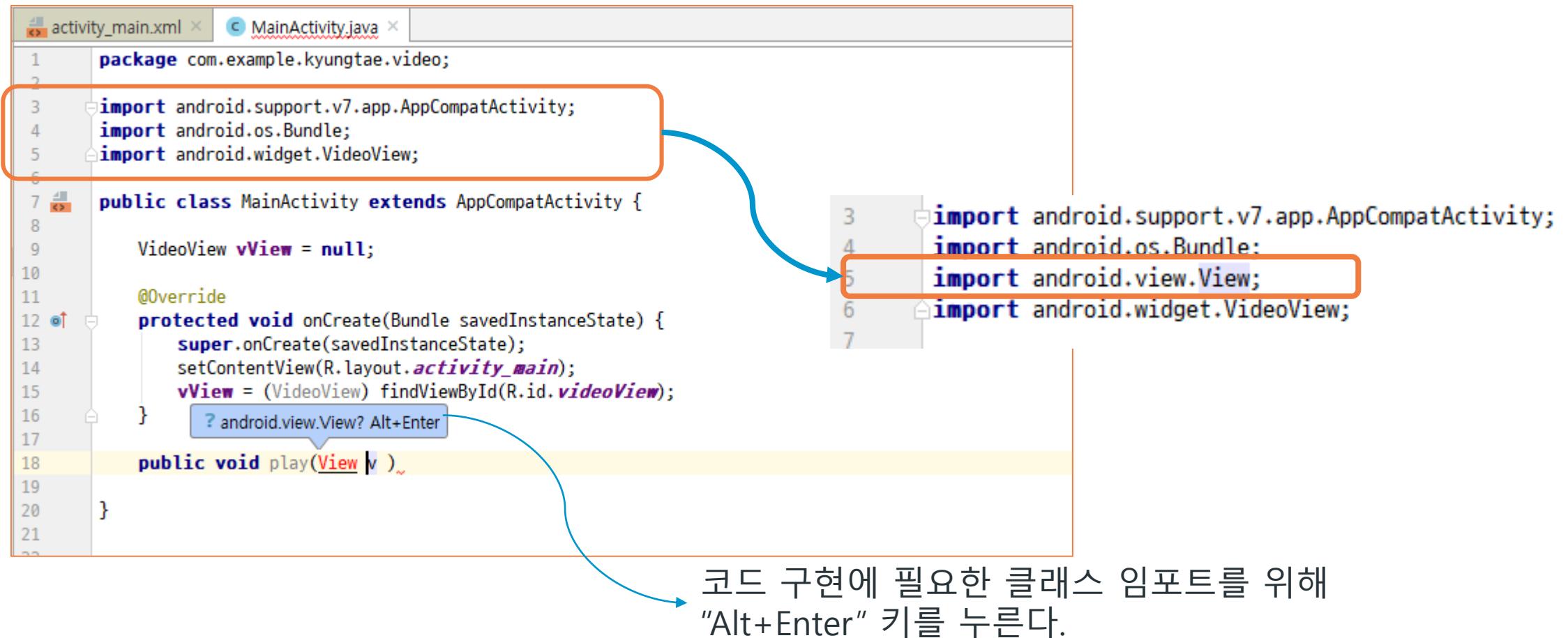
- 동영상 제목을 클릭했을 때 Video 재생을 위한 함수(**play()**) 생성



```
activity_main.xml × MainActivity.java ×
1 package com.example.kyungtae.video;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.widget.VideoView;
6
7 public class MainActivity extends AppCompatActivity {
8
9     VideoView vView = null;
10
11    @Override
12    protected void onCreate(Bundle savedInstanceState) {
13        super.onCreate(savedInstanceState);
14        setContentView(R.layout.activity_main);
15        vView = (VideoView) findViewById(R.id.videoView);
16    }
17
18    |
19
20 }
21
```

커서 위치 확인

## • 코드 입력 중 블루 팝업



- 비디오 제목을 클릭했을 때 호출되는 메소드(play()) 추가

```
30
31     public void play(View v){
32         int id = v.getId();
33         LinearLayout layout = (LinearLayout) findViewById(id);
34
35         Resources res = getResources();
36
37         if (vView.isPlaying()){
38             vView.pause();
39             Drawable drawable = res.getDrawable(R.drawable.shape_off, theme: null);
40             layout.setBackground(drawable);
41
42         }else{
43             Uri uri = Uri.parse("android.resource://com.example.kyungtae.video/" + R.raw.fountain_night);
44             vView.setVideoURI(uri);
45             vView.start();
46             vView.setVisibility(View.VISIBLE);
47
48             Drawable drawable = res.getDrawable(R.drawable.shape_on, theme: null);
49             layout.setBackground(drawable);
50
51             MediaController mc = new MediaController(context: this);
52             vView.setMediaController(mc);
53         }
54     }
55 }
```

생 중 일 때

정지 일 때

미디어 플레이어 중지

새로운 Drawable 객체 인식해서 동영상 제목 Layout 배경을 재 설정

동영상 파일의 uri 인식

uri위치의 동영상재생

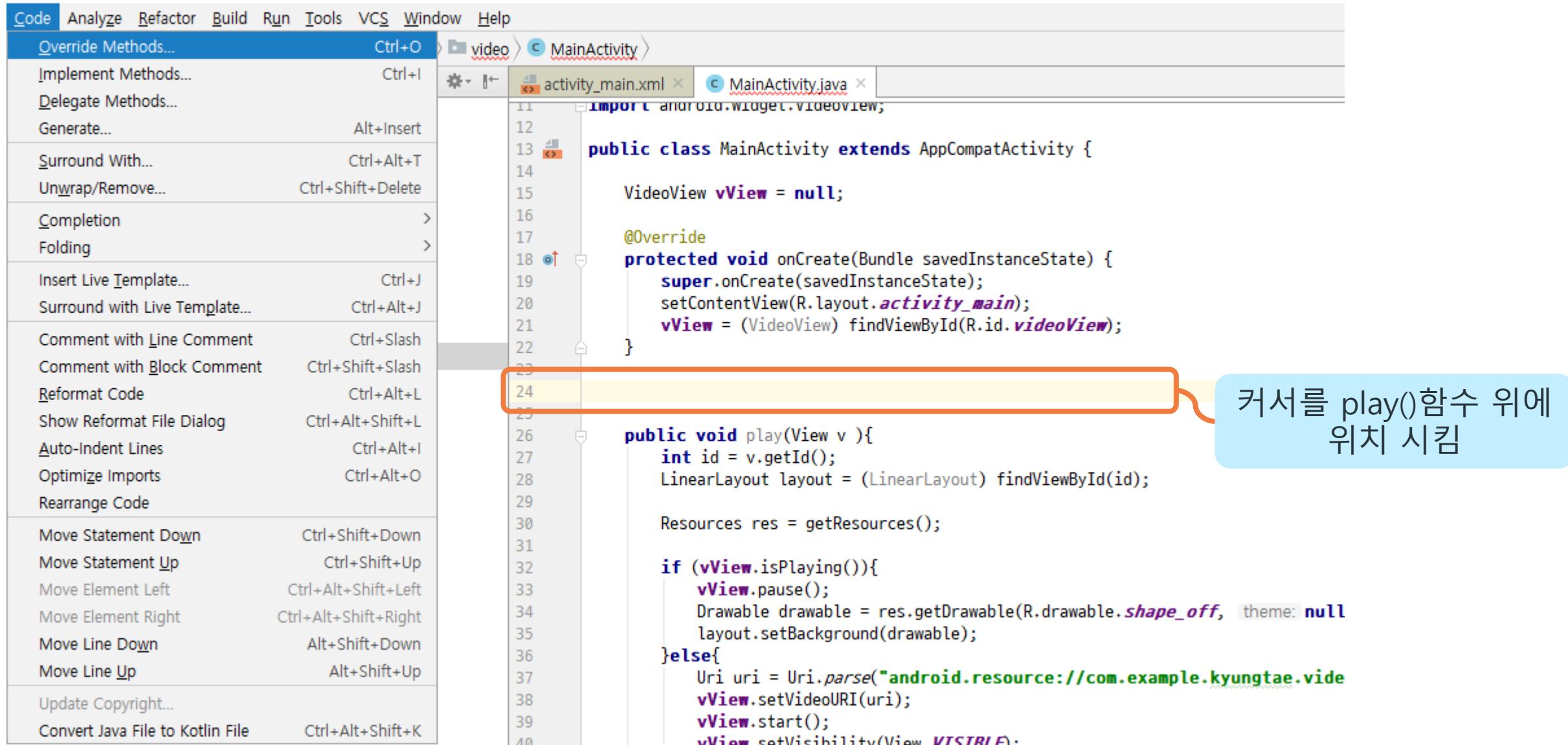
새로운 Drawable 객체 인식해서 동영상 제목 Layout 배경을 재 설정

미디어 제어기 생성

# 재생 중 일 때

정지 일 때

# • Video 재생을 종료했을 때 - onDestroy() 함수 추가



Code Analyze Refactor Build Run Tools VCS Window Help

Override Methods... Ctrl+O video MainActivity

Implement Methods... Ctrl+I

Delegate Methods...

Generate... Alt+Insert

Surround With... Ctrl+Alt+T

Unwrap/Remove... Ctrl+Shift+Delete

Completion >

Folding >

Insert Live Template... Ctrl+J

Surround with Live Template... Ctrl+Alt+J

Comment with Line Comment Ctrl+Slash

Comment with Block Comment Ctrl+Shift+Slash

Reformat Code Ctrl+Alt+L

Show Reformat File Dialog Ctrl+Alt+Shift+L

Auto-Indent Lines Ctrl+Alt+I

Optimize Imports Ctrl+Alt+O

Rearrange Code

Move Statement Down Ctrl+Shift+Down

Move Statement Up Ctrl+Shift+Up

Move Element Left Ctrl+Alt+Shift+Left

Move Element Right Ctrl+Alt+Shift+Right

Move Line Down Alt+Shift+Down

Move Line Up Alt+Shift+Up

Update Copyright...

Convert Java File to Kotlin File Ctrl+Alt+Shift+K

```
activity_main.xml MainActivity.java
```

```
11 import android.widget.VideoView;
12
13 public class MainActivity extends AppCompatActivity {
14
15     VideoView vView = null;
16
17     @Override
18     protected void onCreate(Bundle savedInstanceState) {
19         super.onCreate(savedInstanceState);
20         setContentView(R.layout.activity_main);
21         vView = (VideoView) findViewById(R.id.videoView);
22     }
23
24     public void play(View v){
25         int id = v.getId();
26         LinearLayout layout = (LinearLayout) findViewById(id);
27
28         Resources res = getResources();
29
30         if (vView.isPlaying()){
31             vView.pause();
32             Drawable drawable = res.getDrawable(R.drawable.shape_off, theme: null);
33             layout.setBackground(drawable);
34         }else{
35             Uri uri = Uri.parse("android.resource://com.example.kyungtae.vide
36             vView.setVideoURI(uri);
37             vView.start();
38             vView.setVisibility(View.VISIBLE);
39         }
40     }
41 }
```

커서를 play()함수 위에 위치 시킴

# • onDestroy()는 수퍼 클래스에 정의 되어 있으므로 Override 함

The screenshot shows an IDE interface with two main windows. On the left is a dialog titled "Select Methods to Override/Implement". It lists methods from the class "android.support.v7.app.AppCompatActivity". One method, "onDestroy()", is highlighted with a blue selection bar and has an orange arrow pointing to its implementation in the code editor on the right. The code editor window is titled "activity\_main.xml" and "MainActivity.java". The Java code defines a class "MainActivity" that extends "AppCompatActivity". It overrides the "onCreate()" method and implements the "onDestroy()" method, which calls the super's "onDestroy()".

activity\_main.xml x MainActivity.java x

```
import android.widget.LinearLayout;
import android.widget.MediaController;
import android.widget.VideoView;

public class MainActivity extends AppCompatActivity {
    VideoView vView = null;

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

    @Override
    protected void onDestroy() {
        super.onDestroy();
    }

    public void play(View v) {
        int id = v.getId();
        LinearLayout layout = (LinearLayout) findViewById(id);
    }
}
```

Select Methods to Override/Implement

android.support.v7.app.AppCompatActivity

- m AppCompatActivity()
- m setTheme(resid:int):void
- m onPostCreate(savedInstanceState:Bundle):void
- m getSupportActionBar():ActionBar
- m setSupportActionBar(toolbar:Toolbar):void
- m getMenuInflater():MenuInflater
- m setContentView(layoutResID:int):void
- m setContentView(view:View):void
- m setContentView(view:View, params:LayoutParams):void
- m addContentView(view:View, params:LayoutParams):void
- m onConfigurationChanged(newConfig:Configuration):void
- m onPostResume():void
- m onStart():void
- m onStop():void
- m findViewById(id:int):T
- m onDestroy():void**
- m onTitleChanged(title:CharSequence, color:int):void
- m supportRequestWindowFeature(featureId:int):boolean
- m supportInvalidateOptionsMenu():void
- m invalidateOptionsMenu():void

Copy JavaDoc

Insert @Override

OK Cancel

- 프로젝트 리소스를 얻기 위한 함수 추가

```
23  
24     @Override  
25     protected void onDestroy() {  
26         vView.pause();  
27         super.onDestroy();  
28     }  
29 }
```

액티비티 종료

비디오 뷰 중지

# 클래스와 속성/메소드

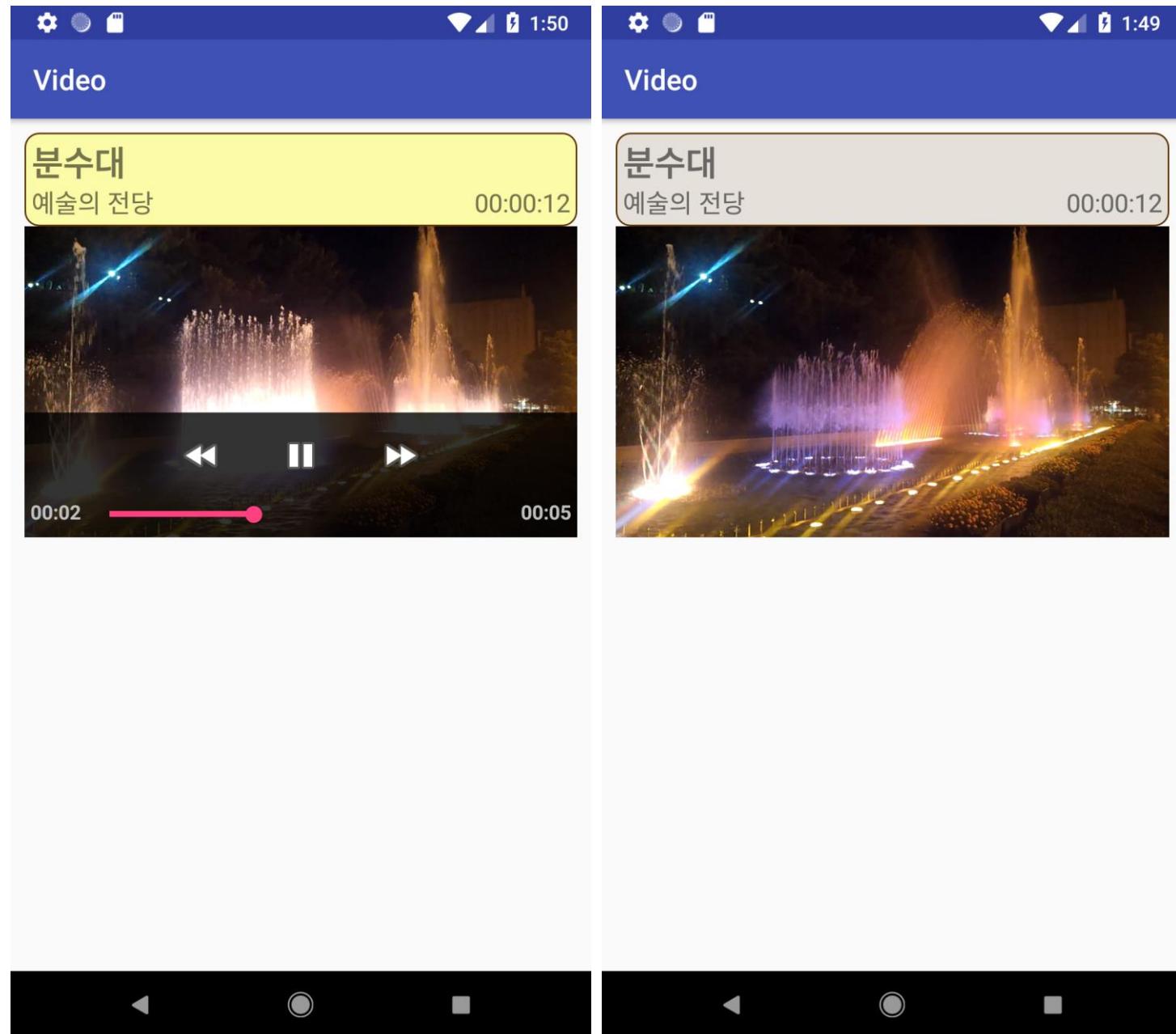
- 클래스

클래스	설명
MediaController	미디어 실행 제어를 포함하는 뷰

- 메소드

클래스	메소드	설명
VideoView	Boolean <code>isPlaying()</code>	비디오뷰의 실행 여부
	void <code>pause()</code>	비디오뷰의 중지
	void <code>setMediaController(MediaController controller)</code>	미디어 제어기를 설정함

# O outputs





question

&



answer

