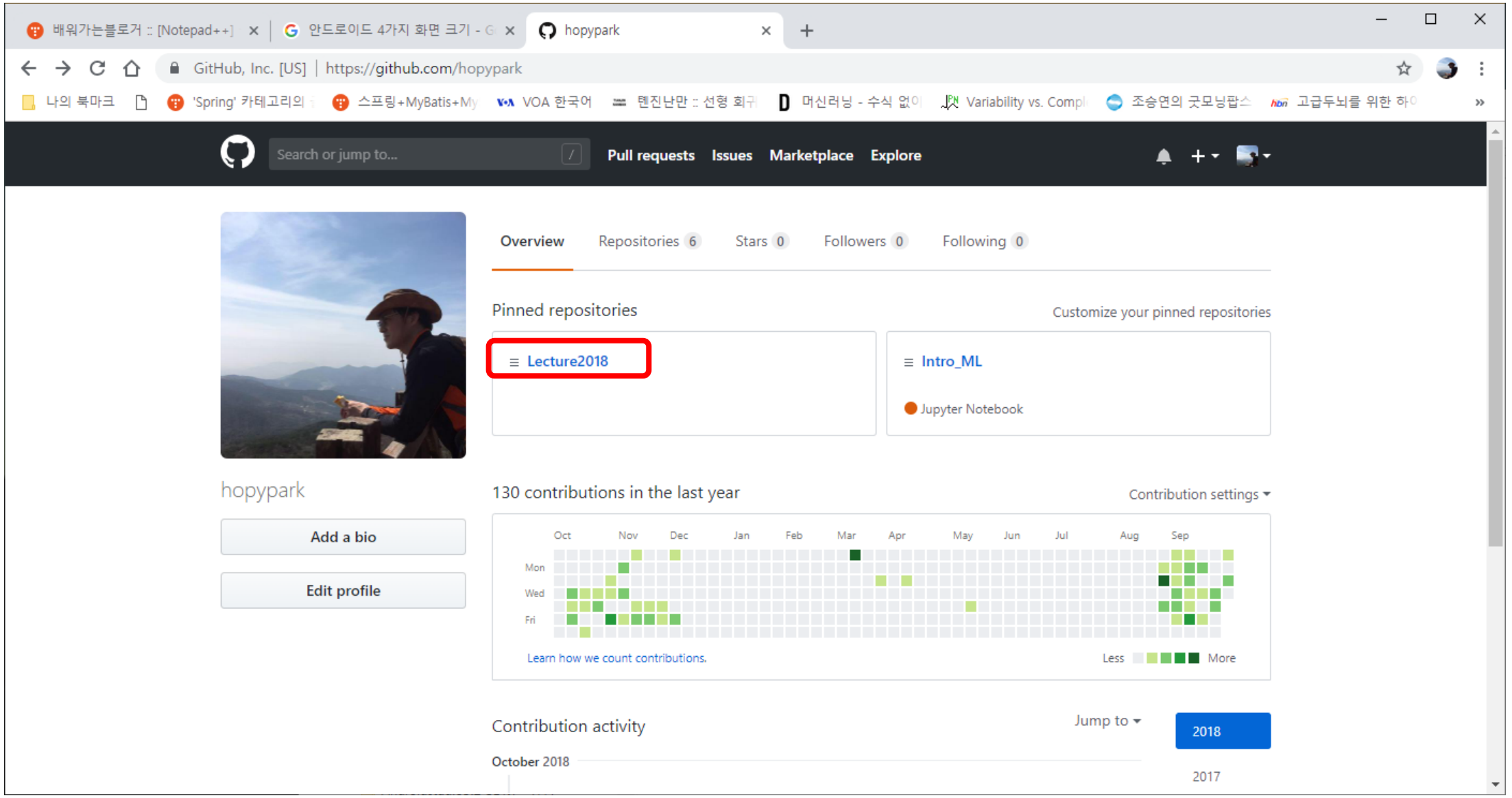
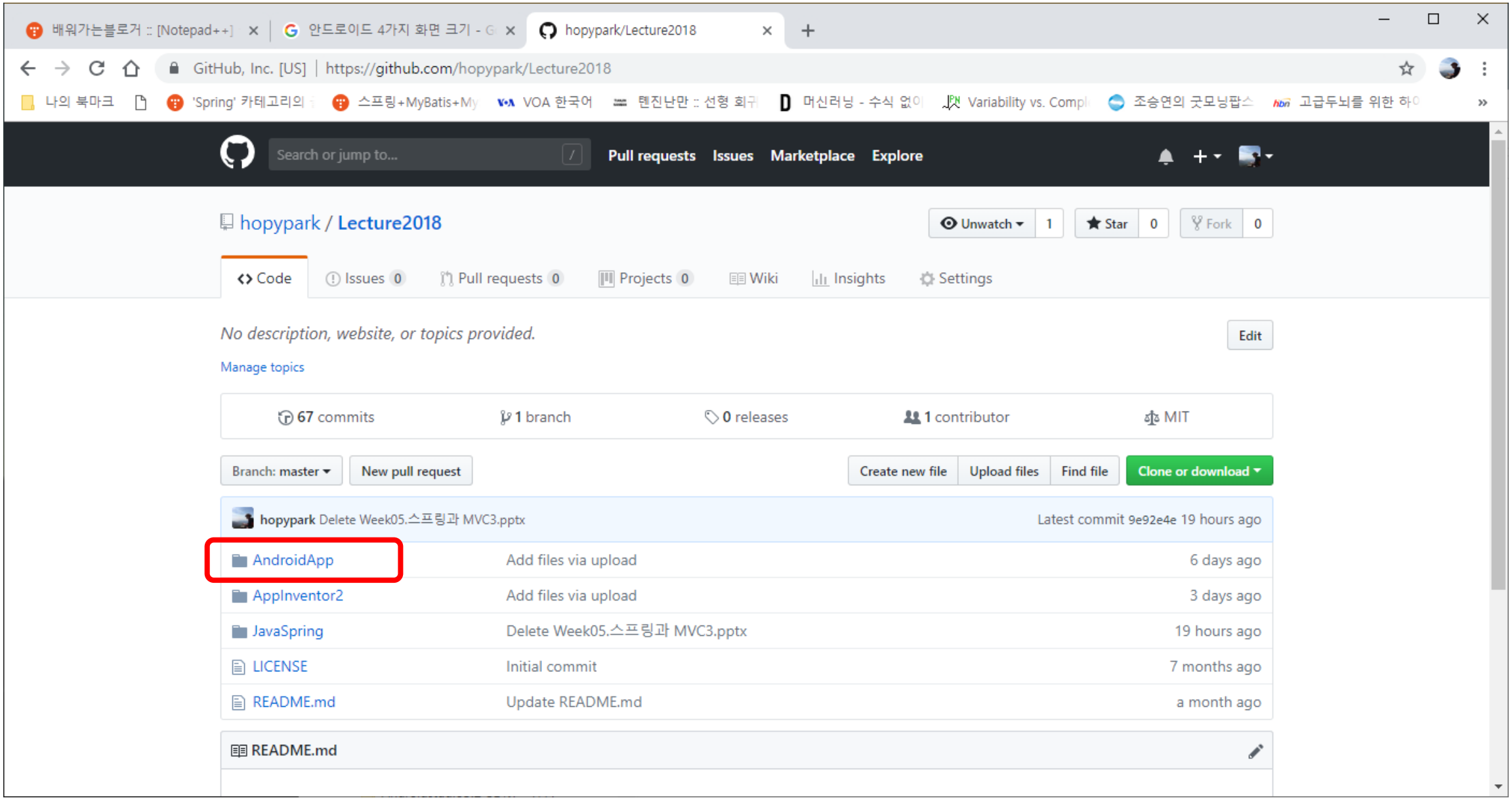
Week06.
오디오 재생

개발환경 구축 절차

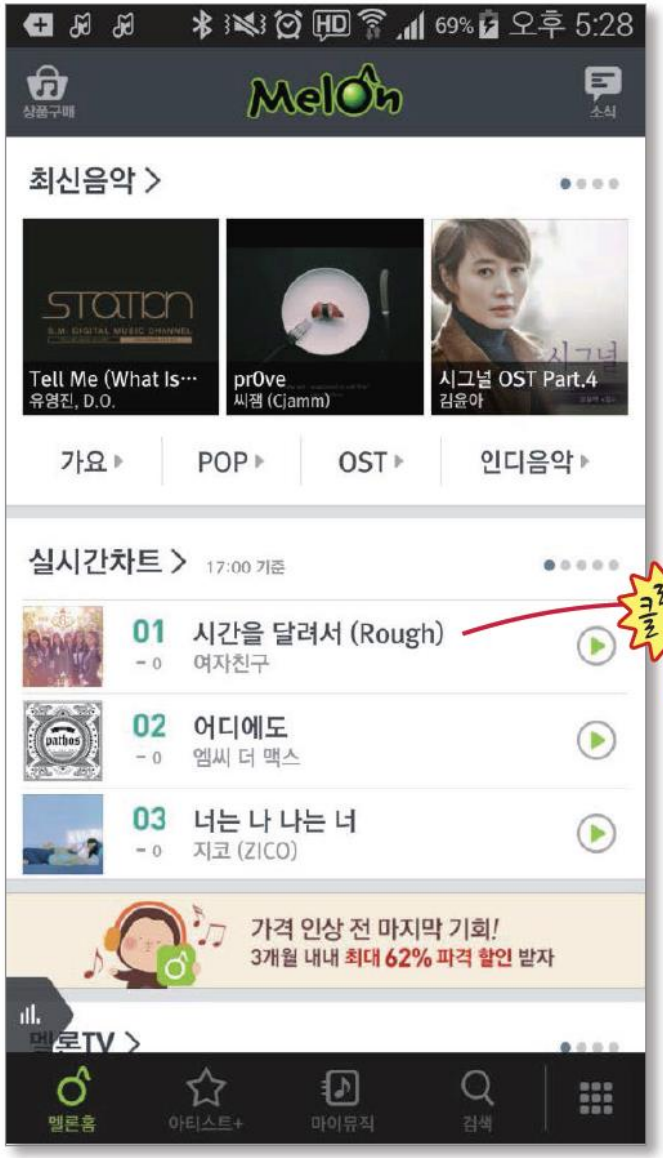
주 차	수 업 내 용
1	수업 소개
2	개발 환경 구축과 맛보기 프로젝트
3	텍스트 출력과 레이아웃
4	이미지의 출력
5	이벤트 처리와 액티비티 간 이동
6	**오디오 재생**
7	비디오 재생
8	**중간고사**
9	애니메이션
10	사물인터넷과 센서 – 터치 센서, 모션 센서
11	사물인터넷과 센서 – 위치 센서, 환경 센서
12	NFC 활용
13	공공 DB 오픈 API 활요
14	구글 맵과 위치 추적
15	**기말 고사**

강의 자료-https://goo.gl/4qhyv7




배워가는블로거 :: [Notepad++]
안드로이드 4가지 화면 크기 - G
hopypark/Lecture2018
GitHub, Inc. [US] | https://github.com/hopypark/Lecture2018
Search or jump to...
Pull requests
Issues
Marketplace
Explore
hopypark / Lecture2018
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
Projects 0
Wiki
Insights
Settings
No description, website, or topics provided.
Edit
Manage topics
67 commits
1 branch
0 releases
1 contributor
MIT
Branch: master
New pull request
Create new file
Upload files
Find file
Clone or download
hopypark Delete Week05.스프링과 MVC3.pptx
Latest commit 9e92e4e 19 hours ago
AndroidApp
Add files via upload
6 days ago
AppInventor2
Add files via upload
3 days ago
JavaSpring
Delete Week05.스프링과 MVC3.pptx
19 hours ago
LICENSE
Initial commit
7 months ago
README.md
Update README.md
a month ago
README.md

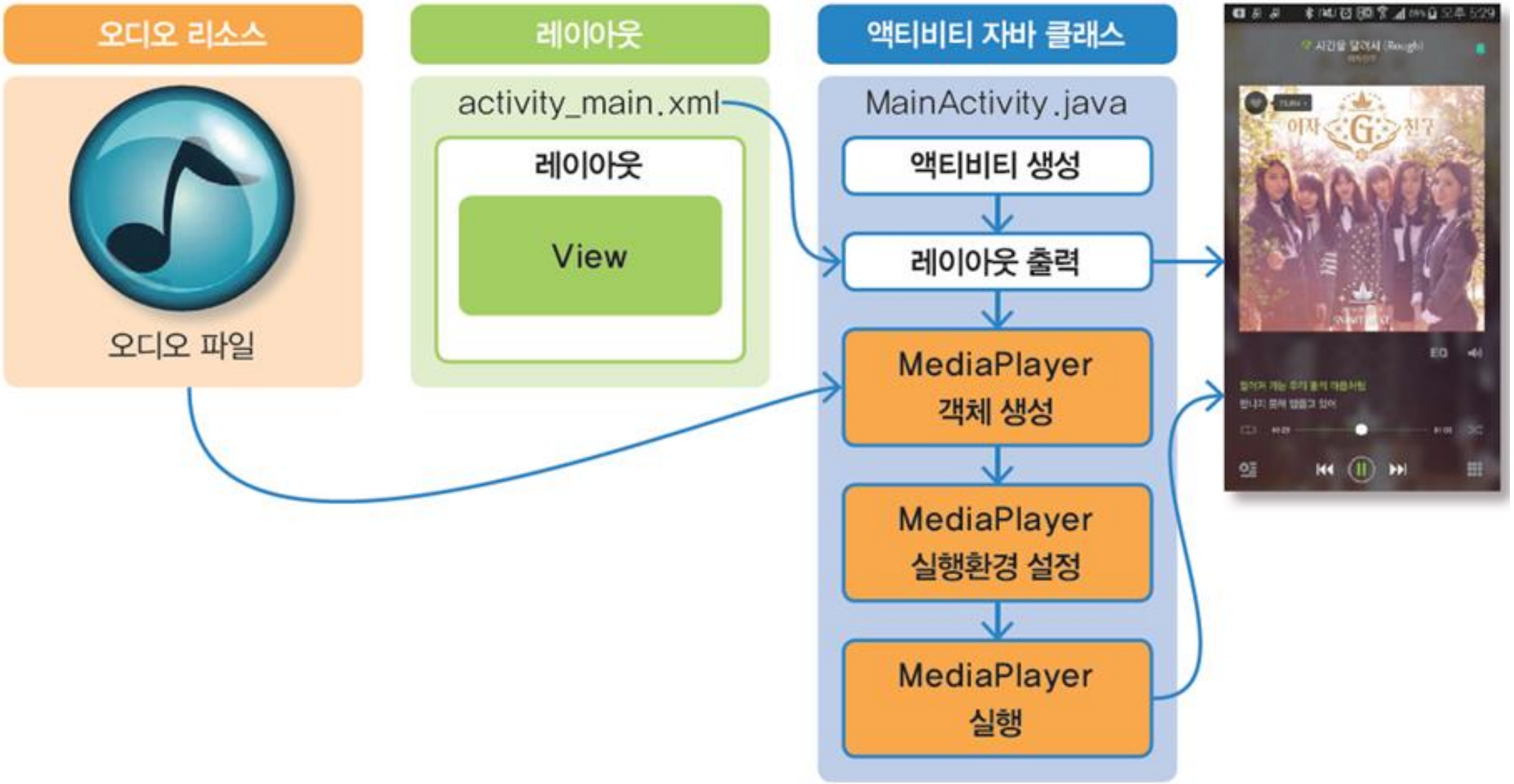
오디오 출력 앱의 예



(a) 초기 화면

(b) 노래 재생

오디오 재생 원리


오디오 리소스
오디오 파일
레이아웃
activity_main.xml
레이아웃
View
액티비티 자바 클래스
MainActivity.java
액티비티 생성
레이아웃 출력
MediaPlayer
객체 생성
MediaPlayer
실행환경 설정
MediaPlayer
실행

Step 0.프로젝트 개요

프로젝트 개요

오디오의 자동재생

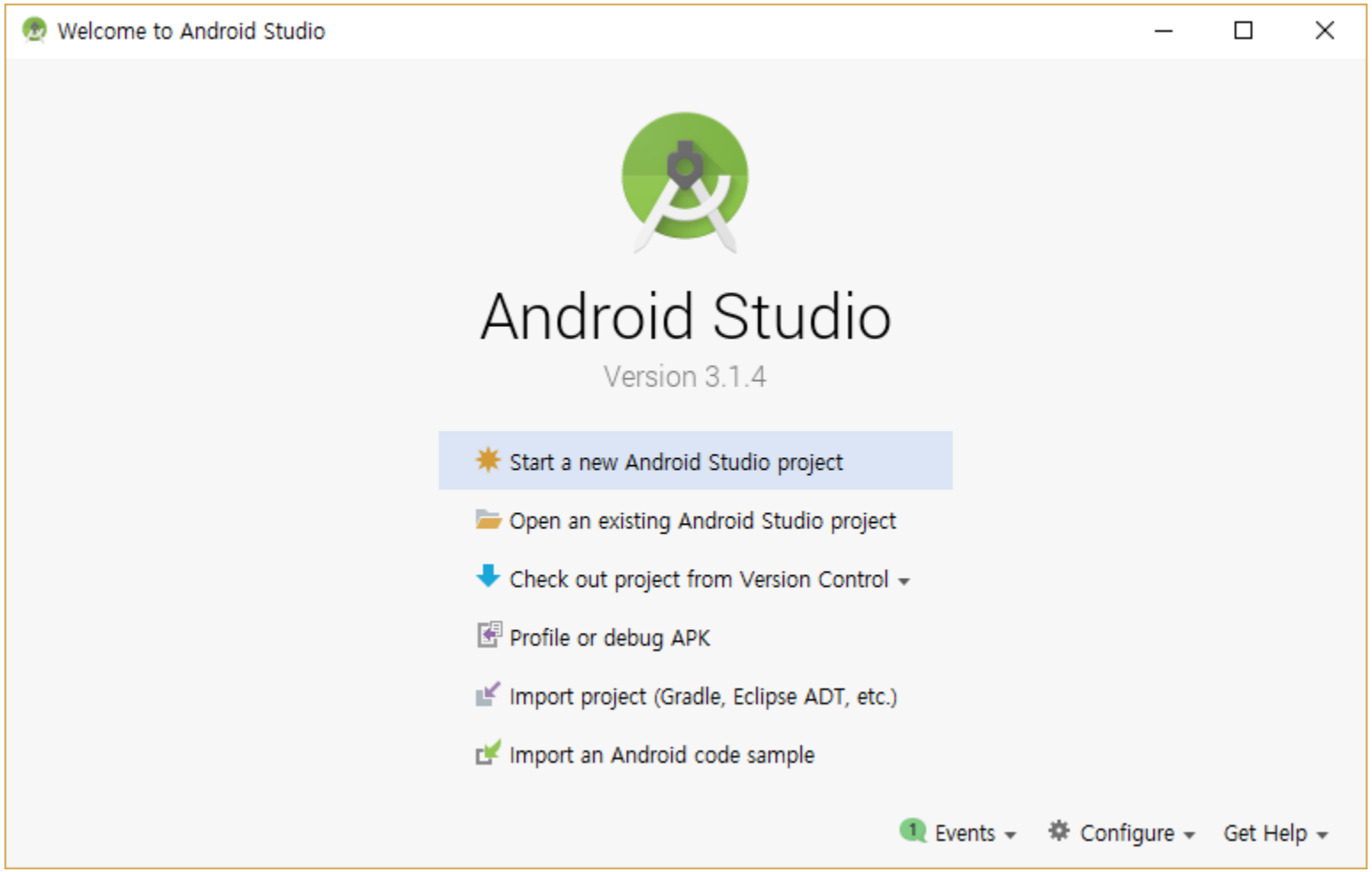
Application Name

Audio

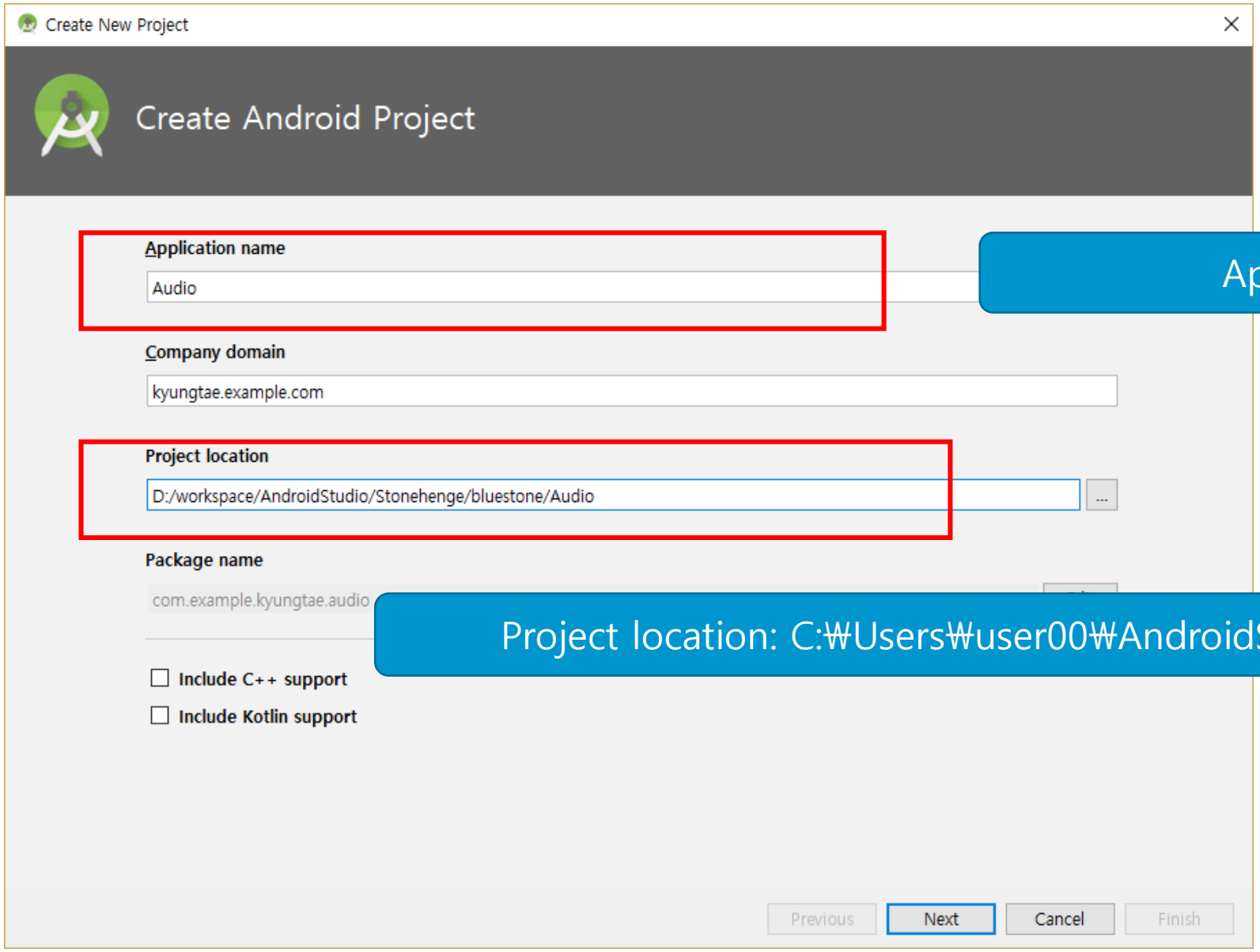
어플리케이션 라벨

노래 목록1

Start a new Android Studio project-type1



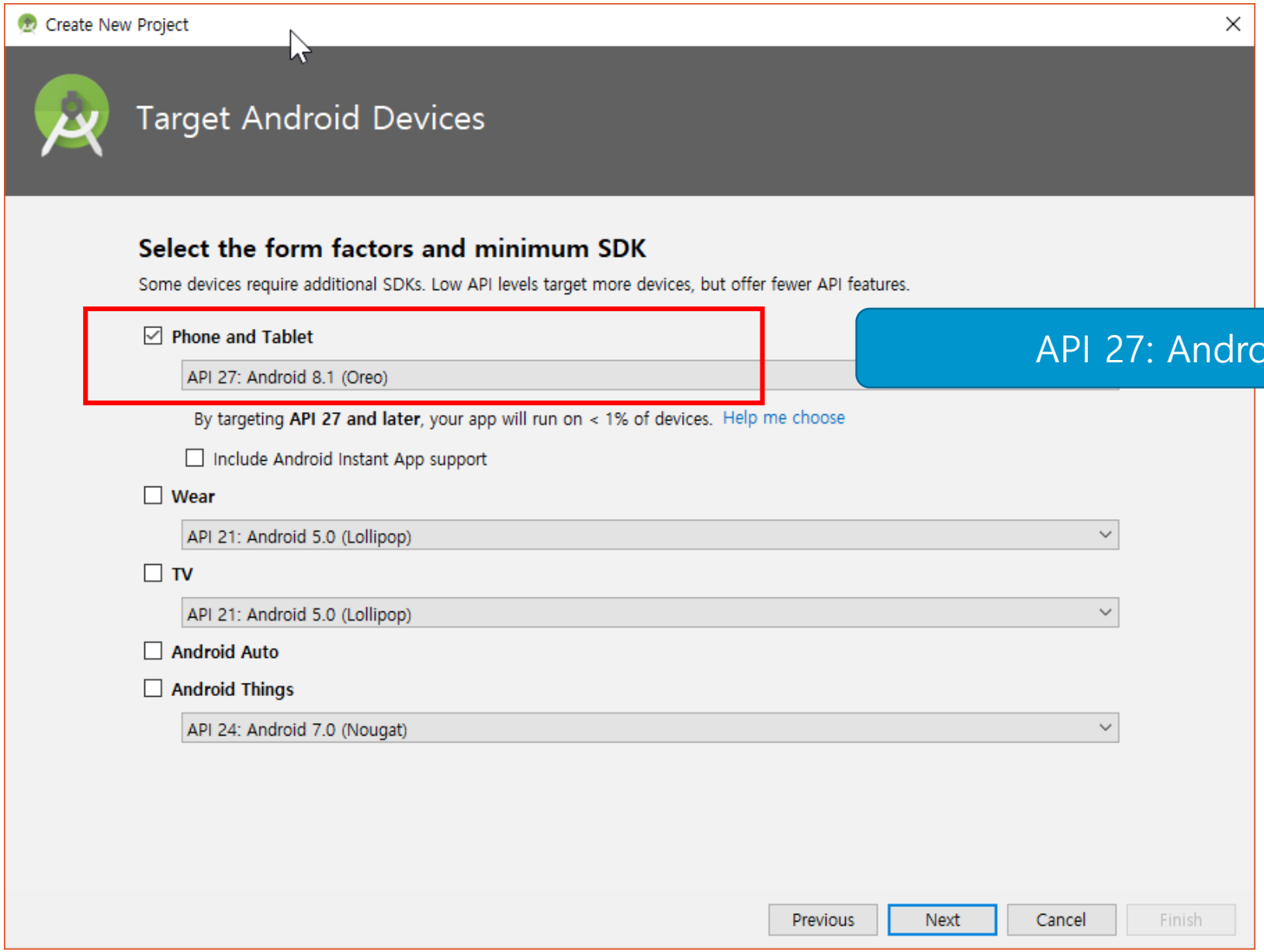
Configure your new project



Application name: Audio

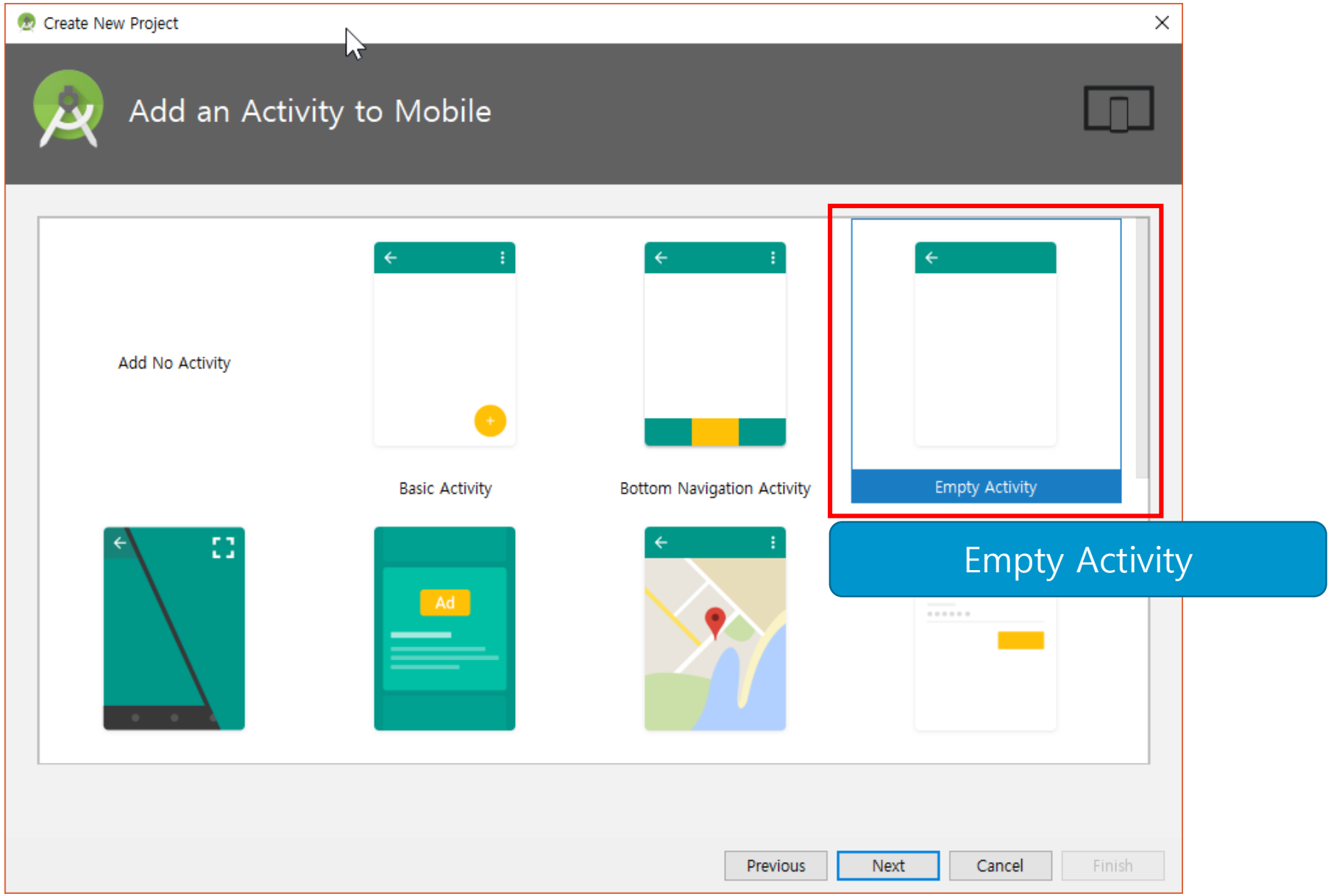
Project location: C:₩Users₩user00₩AndroidStudioProjects₩ktpark₩Audio

Configure your new project

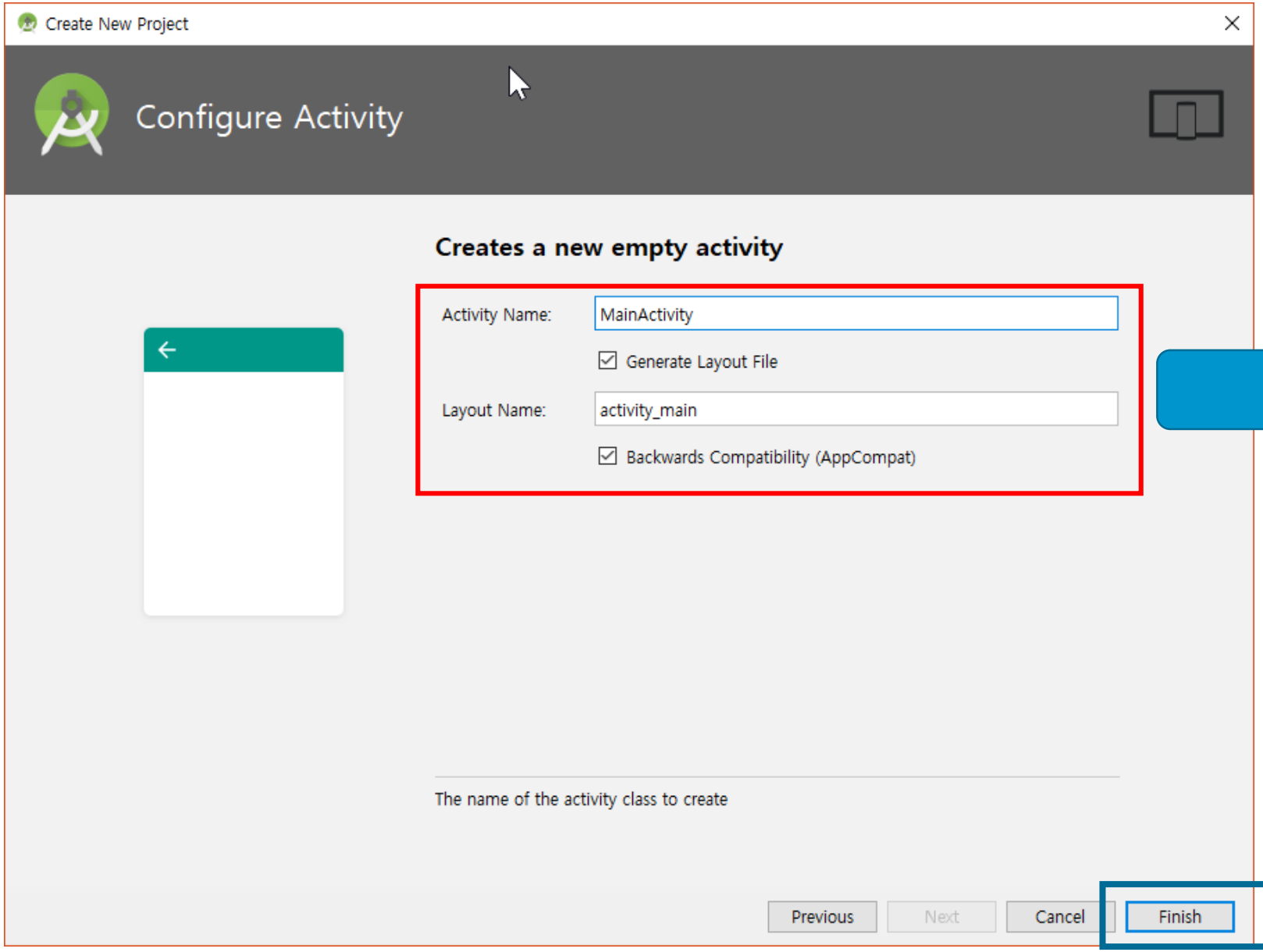


API 27: Android 8.1 (Oreo)

Configure your new project



Configure your new project



기본 입력값 사용

Step 1. 프로젝트 생성

절차	내 용
①프로젝트 시작	메뉴에서 'File → New Project' 클릭
②프로젝트 구성	Application Name: **Audio**
	Company Domain: 사용자계정.example.com(디폴트 사용)
	Project location: ~₩user00₩AndroidStudioProjects**₩ktpark₩Audio**
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

모듈	폴더	소스 파일	편집 내용
manifests		AndroidManifest.xml	
java	com.example.kyungtae.audio	MainActivity.java	• 오디오 목록 출력과 오디오 자동 재생
res	drawable	shape_list.xml	• 목록 아이템에 대한 출력 스타일 설계 (재생 시)
		shape_on.xml	• 목록 아이템에 대한 출력 스타일 설계 (재생 중지 시)
	layout	activity_main.xml	• 노래 목록의 화면 배치 • 목록 아이템에 출력 모양 적용(shape_list.xml)
	mipmap	ic_launcher.png	
	raw	schoolbell.mp3	• 노래 오디오 파일
	values	colors.xml	
		dimens.xml	
		strings.xml	• 어플리케이션 라벨(“노래 목록1”) • 노래에 대한 제목, 작사/작곡자, 재생 시간에 대한 텍스트 리소스 정의
		styles.xml	

Step 2.1 오디오 파일 복사

- res 폴더에 있는 **raw** 폴더에 schoolbell.mp3 파일 저장

모듈	폴더	소스 파일	내용
res	raw	schoolbell.mp3 	학교종 노래

목록 아이템의 제목 모양

모양

색

```
shape
    shape     rectangle
    color     #1061380B
```

shape_list.xml(drawable)

오디오 리소스

schoolbell.mp3(drawable)

앱 라벨

노래 제목

아티스트

재생시간

텍스트 자원

```
string
    app_name 노래 목록1
    title1 학교종
    artist1 김메리
    time1 00:00:19
```

strings.xml (values)

앱이 종료될 때 자동실행
(오디오 종료 설정 등)

```
LinearLayout
    background @drawable/shape_list
    orientation vertical
    LinearLayout
        TextView
            text @string/title1
        LinearLayout
            orientation horizontal
            TextView
            text @string/artist1
            TextView
            text @string/time1
```

노래 목록 아이템의 리니어 레이아웃에
shape_list 출력모양 적용

노래 제목

아티스트

재생시간

activity_main.xml (layout)

화면 출력 소스

액티비티 제어

```
onCreate()
    super onCreate()
    setContentView(R.layout.activity_main)
play()
    MediaPlayer
onDestory()
```

오디오 인식과 재생

MainActivity.java (layout)

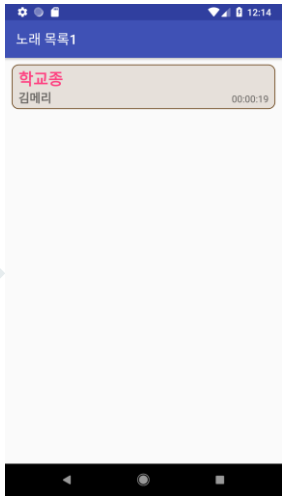
어플리케이션 구성
액티비티의 자바 클래스

어플리케이션 기본 정보

```
application
    icon  @mipmap/ic_launcher
    label @string/app_name
    theme @style/AppTheme
    activity
        name MainActivity
```

AndroidManifest.xml (manifest)

컴파일/빌더



컴파일/빌더 정보

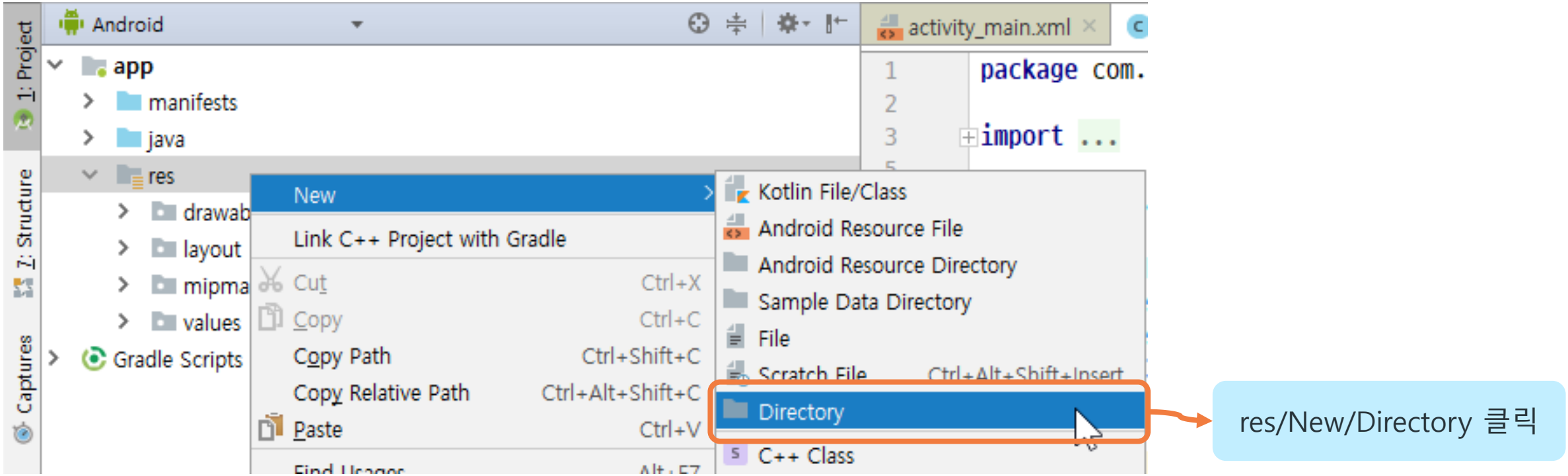
```
build gradle(Project)
build gradle(Module app)
gradle properties
settings gradle
local properties
```

(Gradle Scripts)

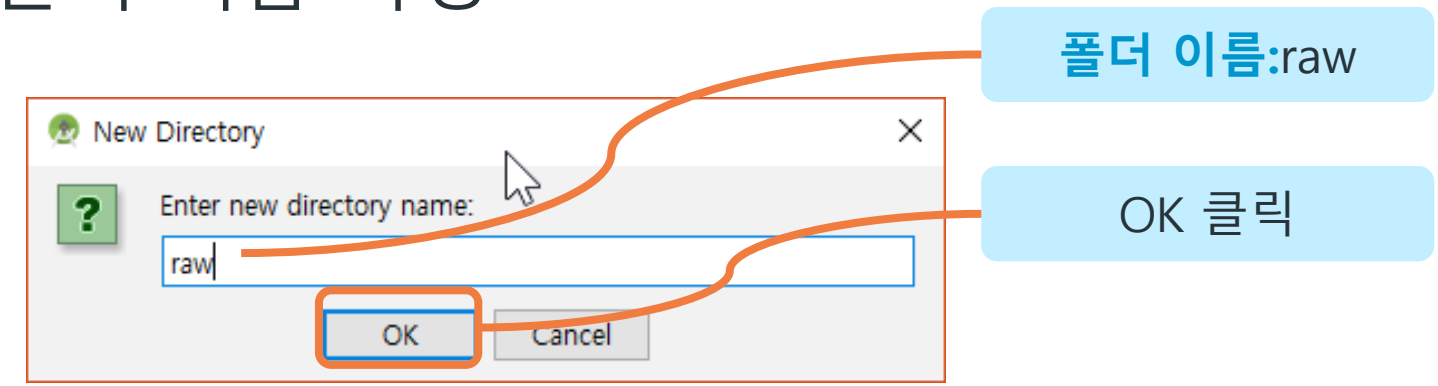
drawable/raw 폴더에 오디오 파일 올리기

drawable/raw 폴더에 오디오 파일 추가하기

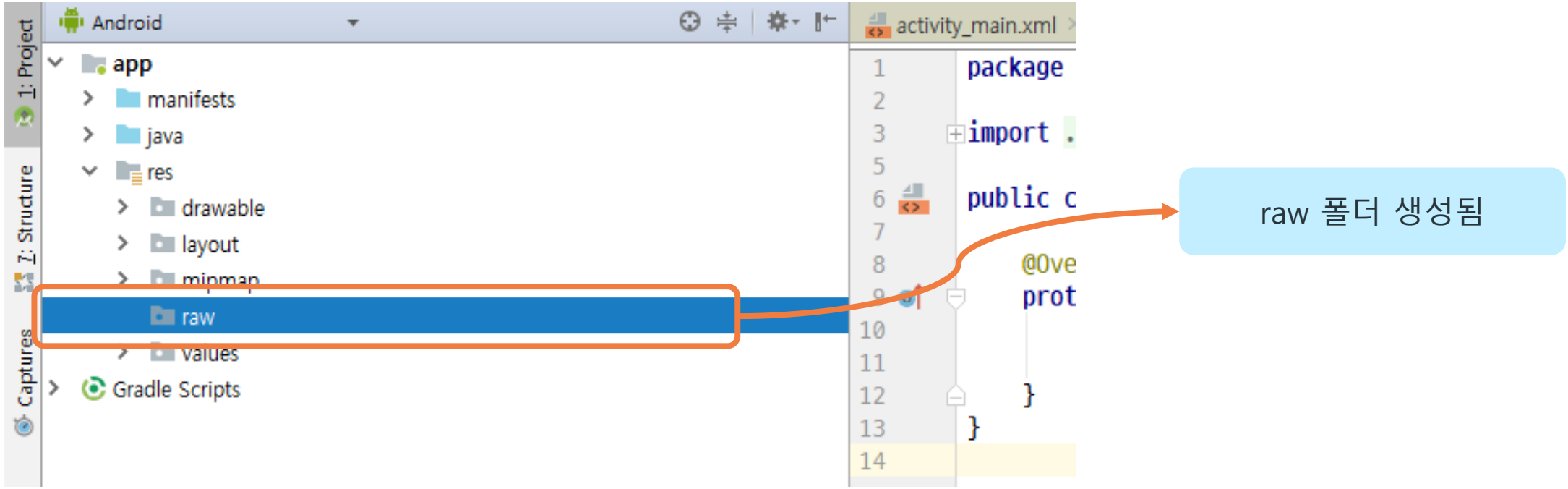
- app→res→New→Directory 클릭



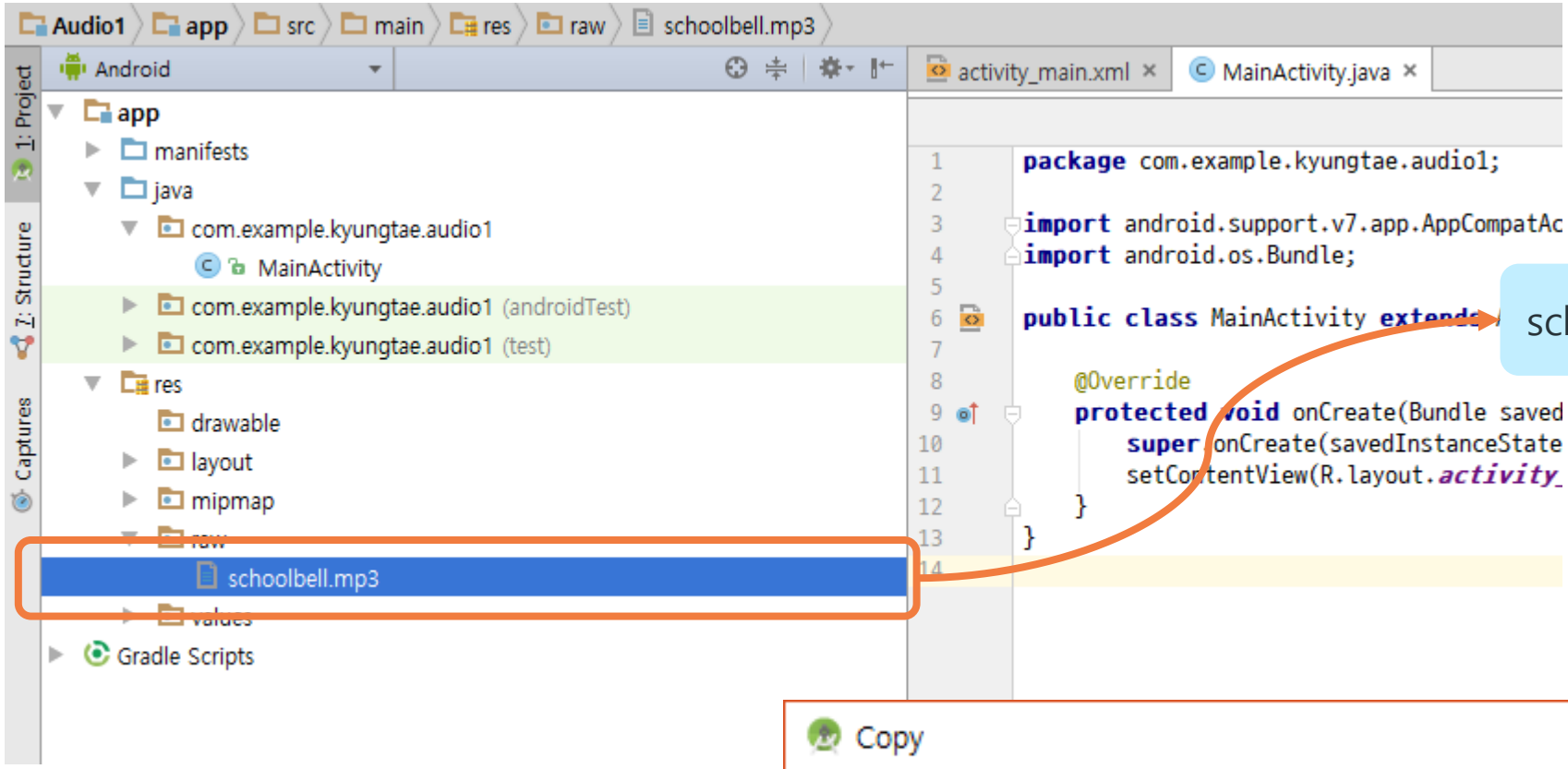
- 폴더 이름 작성

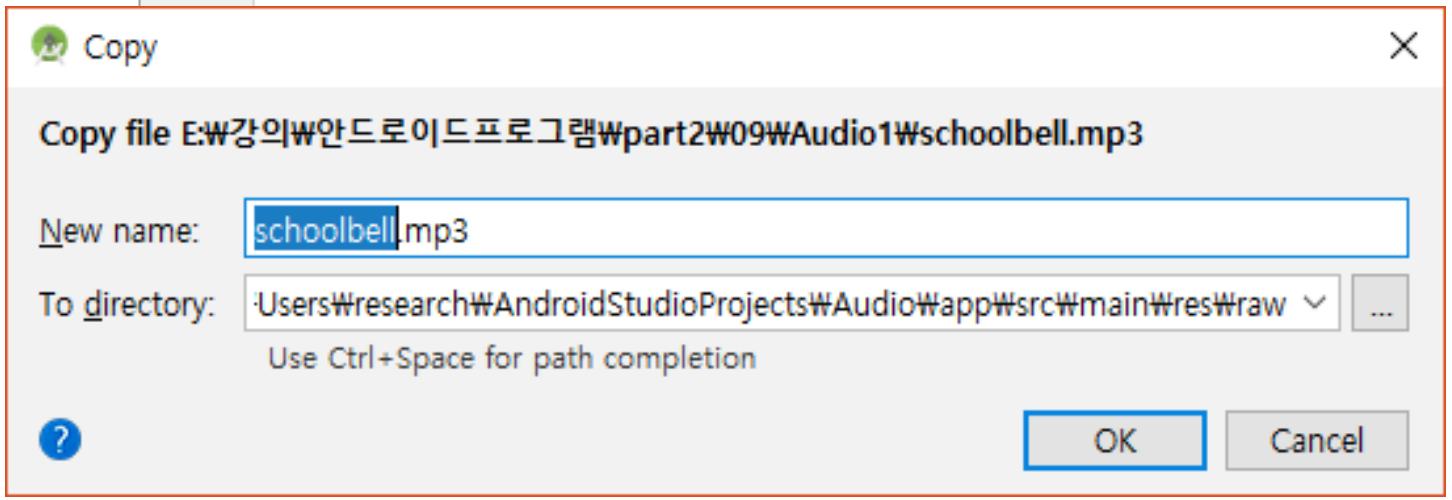

폴더 이름:raw
OK 클릭
New Directory
Enter new directory name:
raw
OK
Cancel

- 실행 결과


raw 폴더 생성됨
Android
app
manifests
java
res
drawable
layout
raw
values
Gradle Scripts
activity_main.xml
package
import .
public c
@Ove
prot
}
}
1: Project
7: Structure
Captures

- 오디오 파일(schoolbell.mp3) 복사하기





Step 2.2 텍스트 자원의 편집

- res>values>strings.xml

activity_main.xml | MainActivity.java | strings.xml

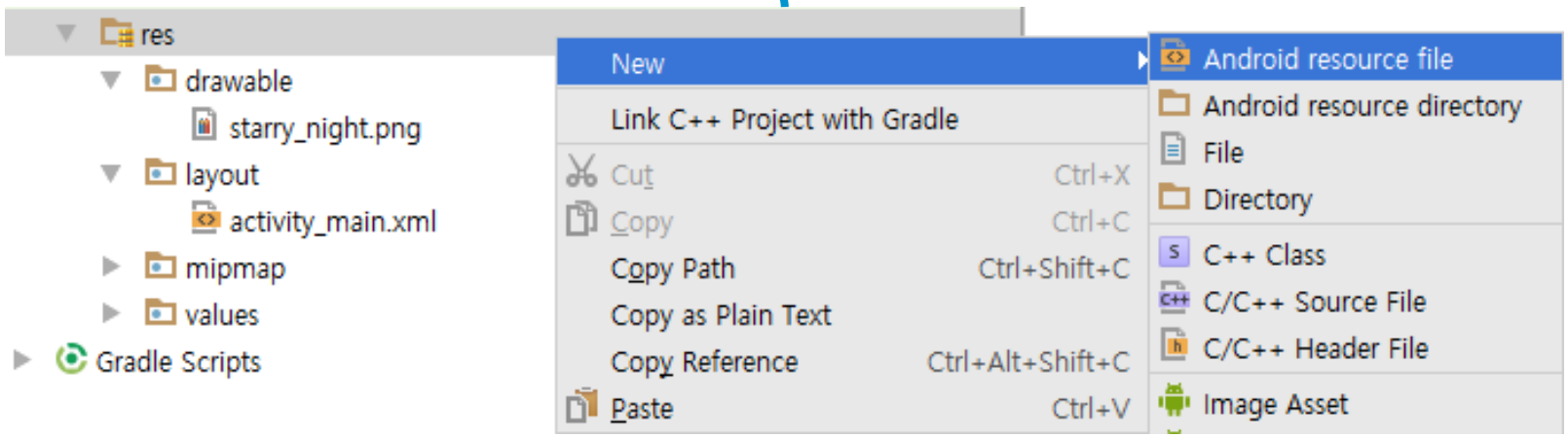
Edit translations for all locales in the translations editor.

```
1  <resources>
2      <string name="app_name">노래 목록1</string>
3
4      <string name="title">학교종</string>
5      <string name="artist">김메리</string>
6      <string name="time">00:00:19</string>
7  </resources>
8
9
```

Step 2.3 Drawable Resource 추가 및 편집

- **shape_list.xml** 생성(res/drawable 폴더)
  - drawable resource를 이용한 그림 출력
  - 노래 제목(title)에 대한 출력 모양을 지정
  - **drawable 폴더에는 화면에 그릴 수 있는 요소(도형)를 XML로 정의**
  - android:shape 속성을 사용하여 다른 XML 리소스에 적용할 수 있는 그래픽에 대한 일반적인 개념

XML 파일 생성



• Set New Resource File

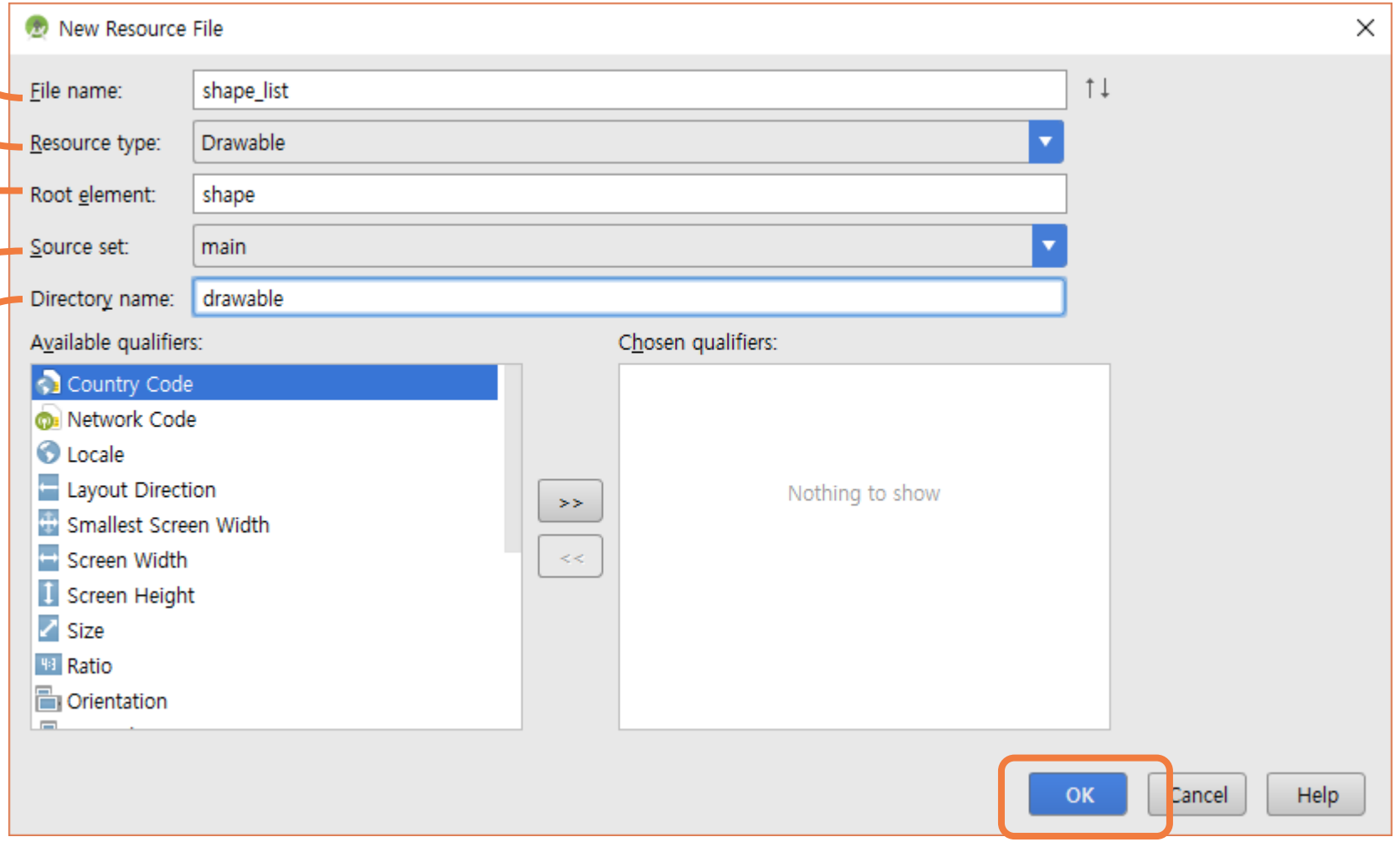
File name: shape_list

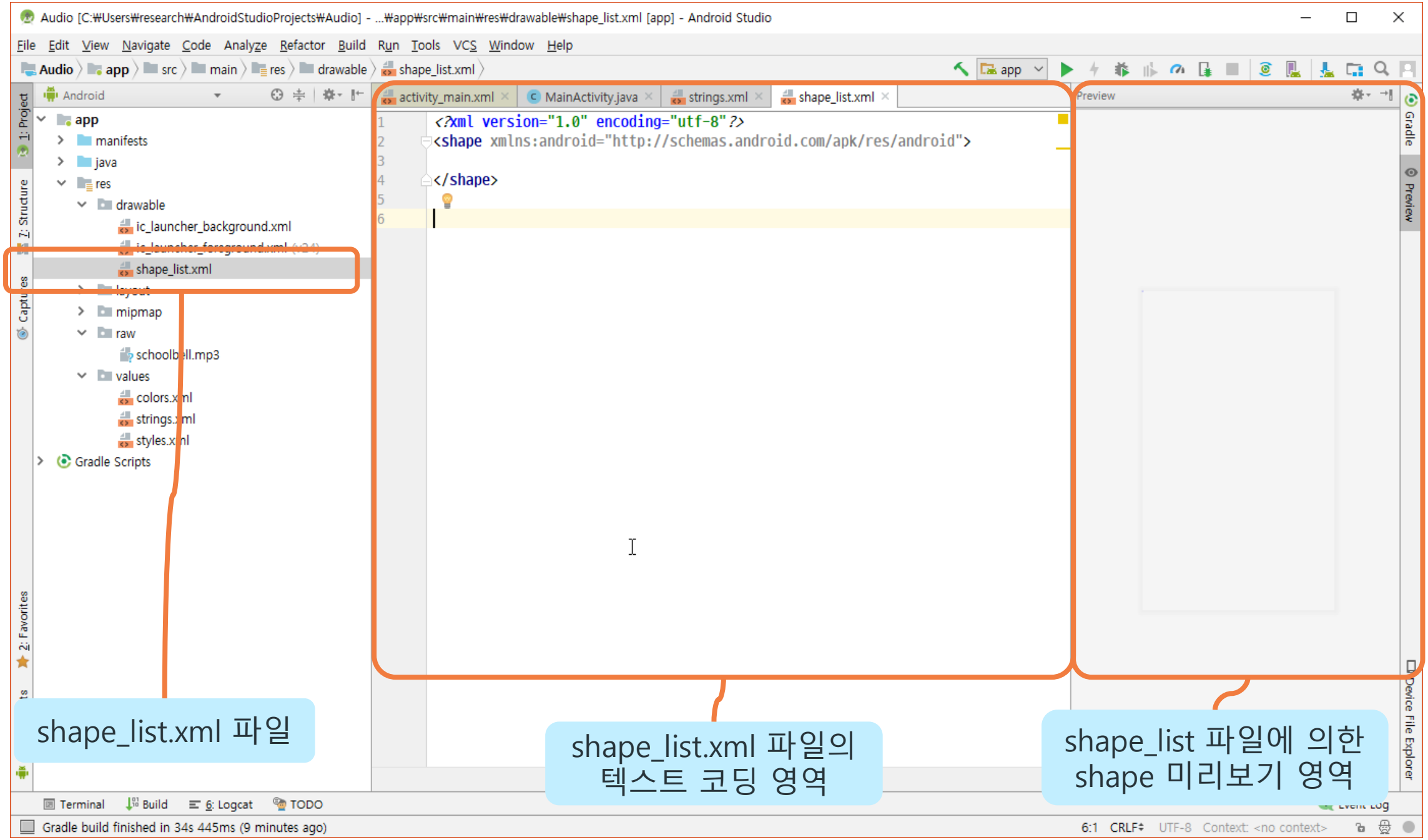
Resource type: Drawable

Root element: shape

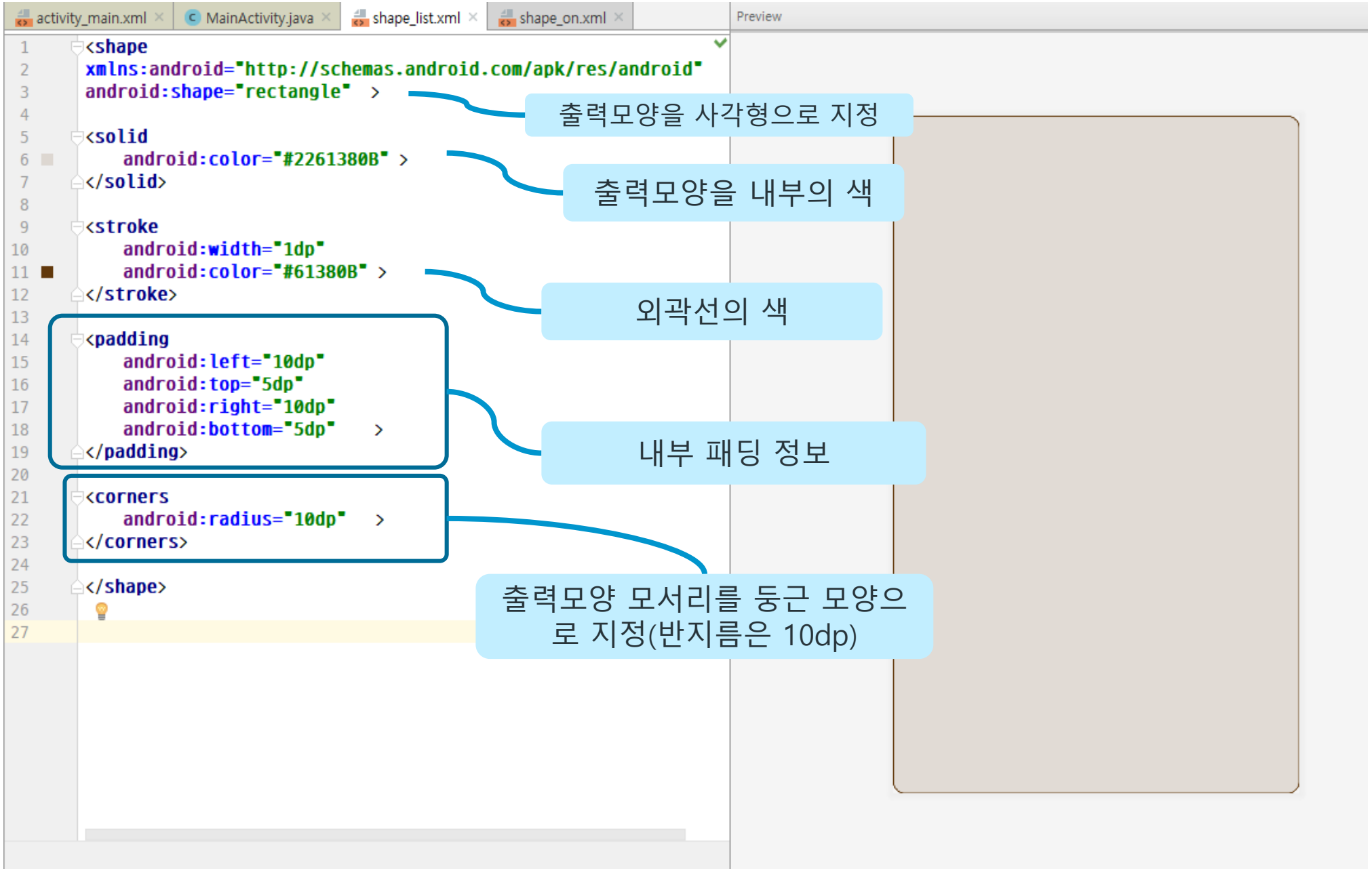
Source set: main

Directory name: drawable




Audio [C:₩Users₩research₩AndroidStudioProjects₩Audio] - ...₩app₩src₩main₩res₩drawable₩shape_list.xml [app] - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Audio app src main res drawable shape_list.xml
app
activity_main.xml
MainActivity.java
strings.xml
shape_list.xml
Preview
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
</shape>
manifests
java
res
drawable
ic_launcher_background.xml
shape_list.xml
mipmap
raw
values
colors.xml
styles.xml
Gradle Scripts
Terminal
Build
6: Logcat
TODO
Gradle build finished in 34s 445ms (9 minutes ago)
shape_list.xml 파일
shape_list.xml 파일의 텍스트 코딩 영역
shape_list 파일에 의한 shape 미리보기 영역

- shape_list.xml 소스

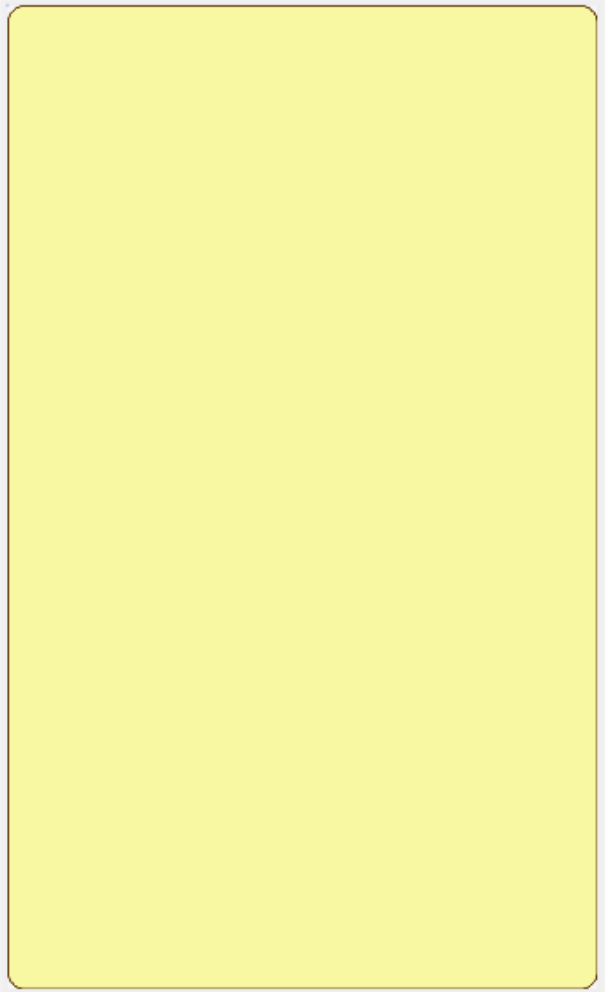


- shape_on.xml 소스

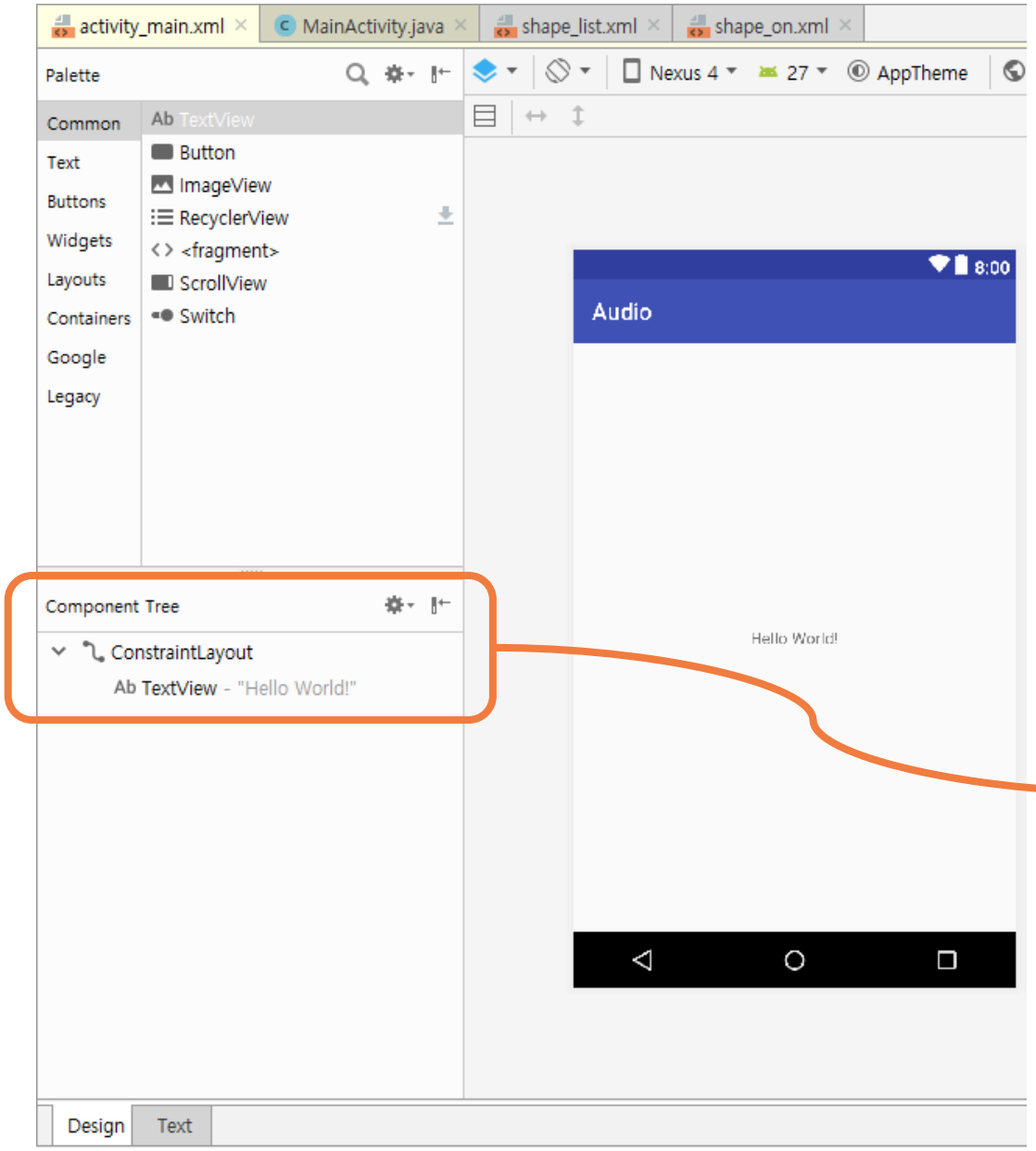
activity_main.xml | MainActivity.java | shape_list.xml | shape_on.xml | Preview

```xml
<shape
xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle" >

<solid
    android:color="#55ffff00" >
</solid>

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

<padding
    android:left="10dp"
    android:top="5dp"
    android:right="10dp"
    android:bottom="5dp" >
</padding>

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

</shape>
```

출력모양을 내부의 색

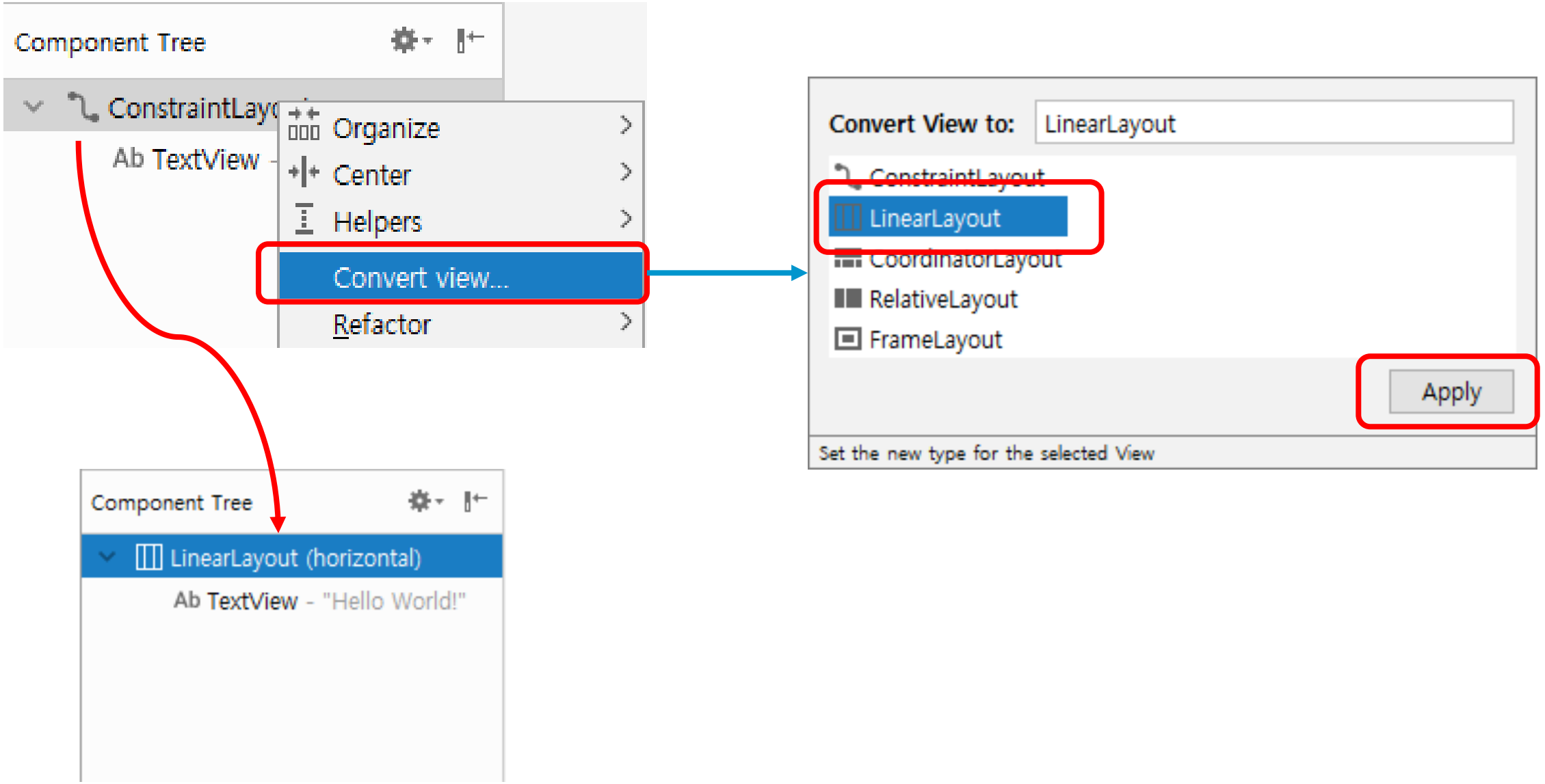


2.4 화면 설계

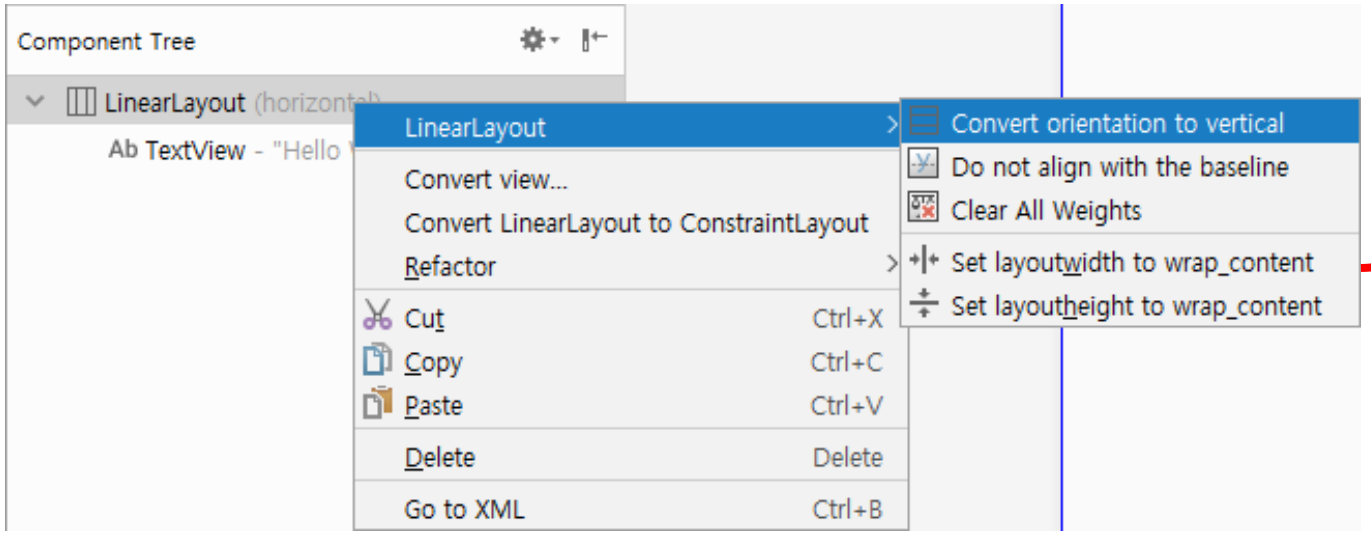


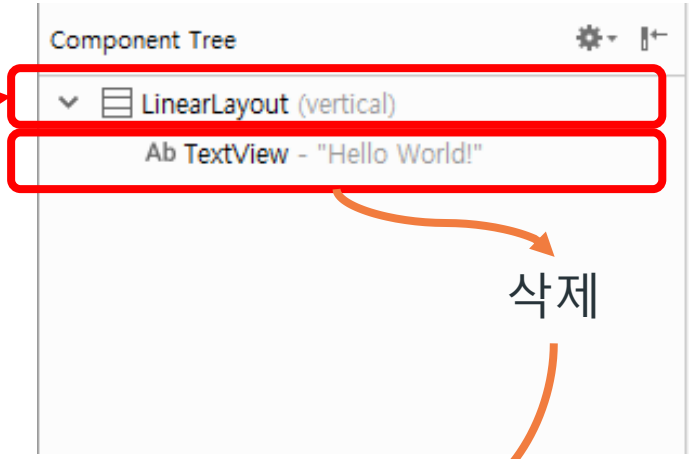
ConstraintLayout → LinearLayout으로 변경

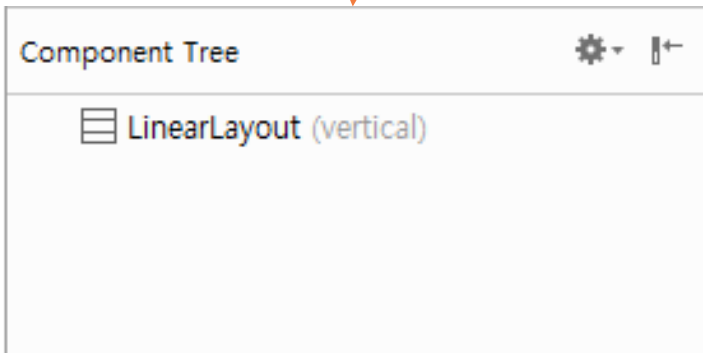
ConstraintLayout을 LinearLayout로 바꾸기



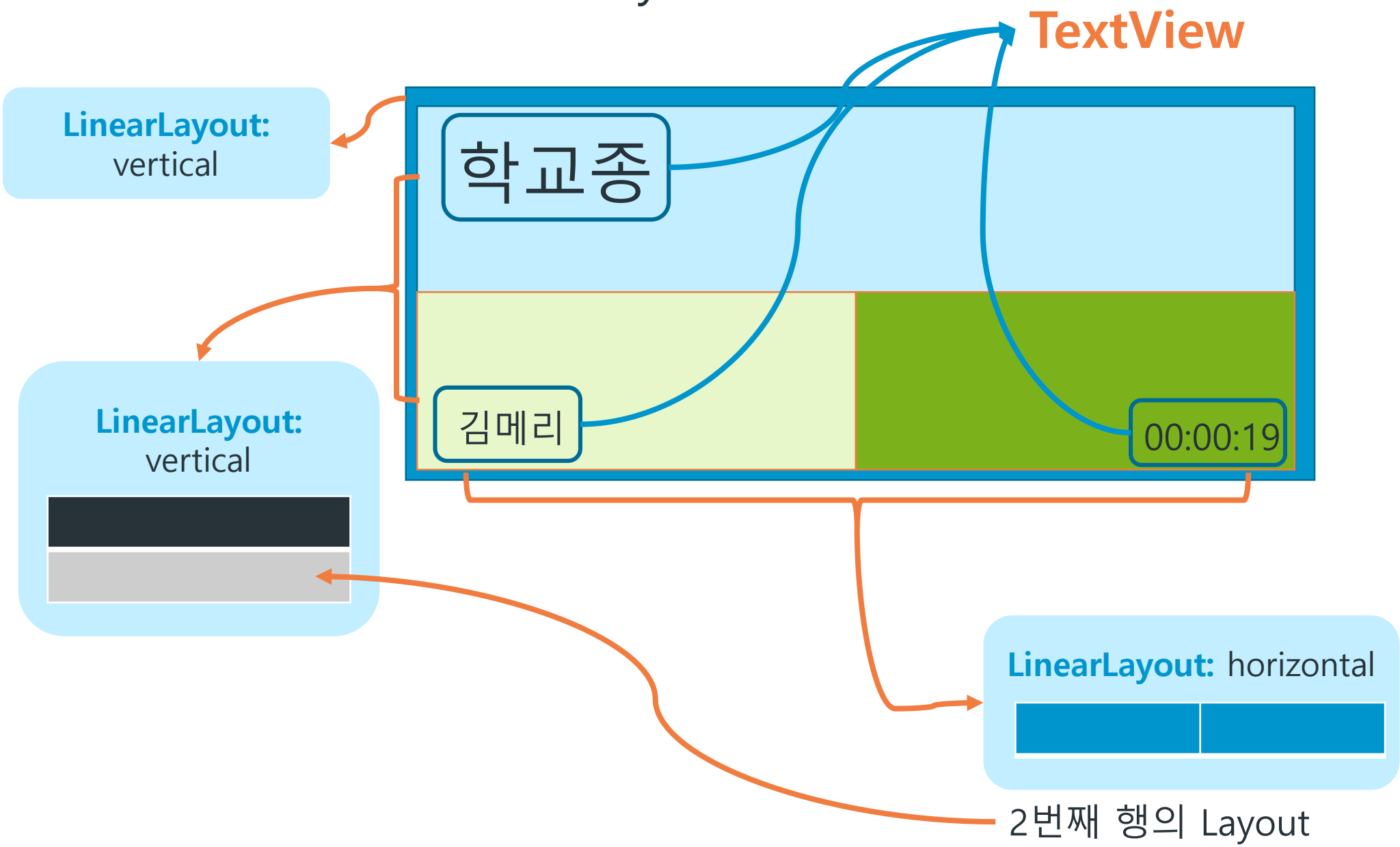
LinearLayout의 방향을 Horizontal → Vertical로 변경하기



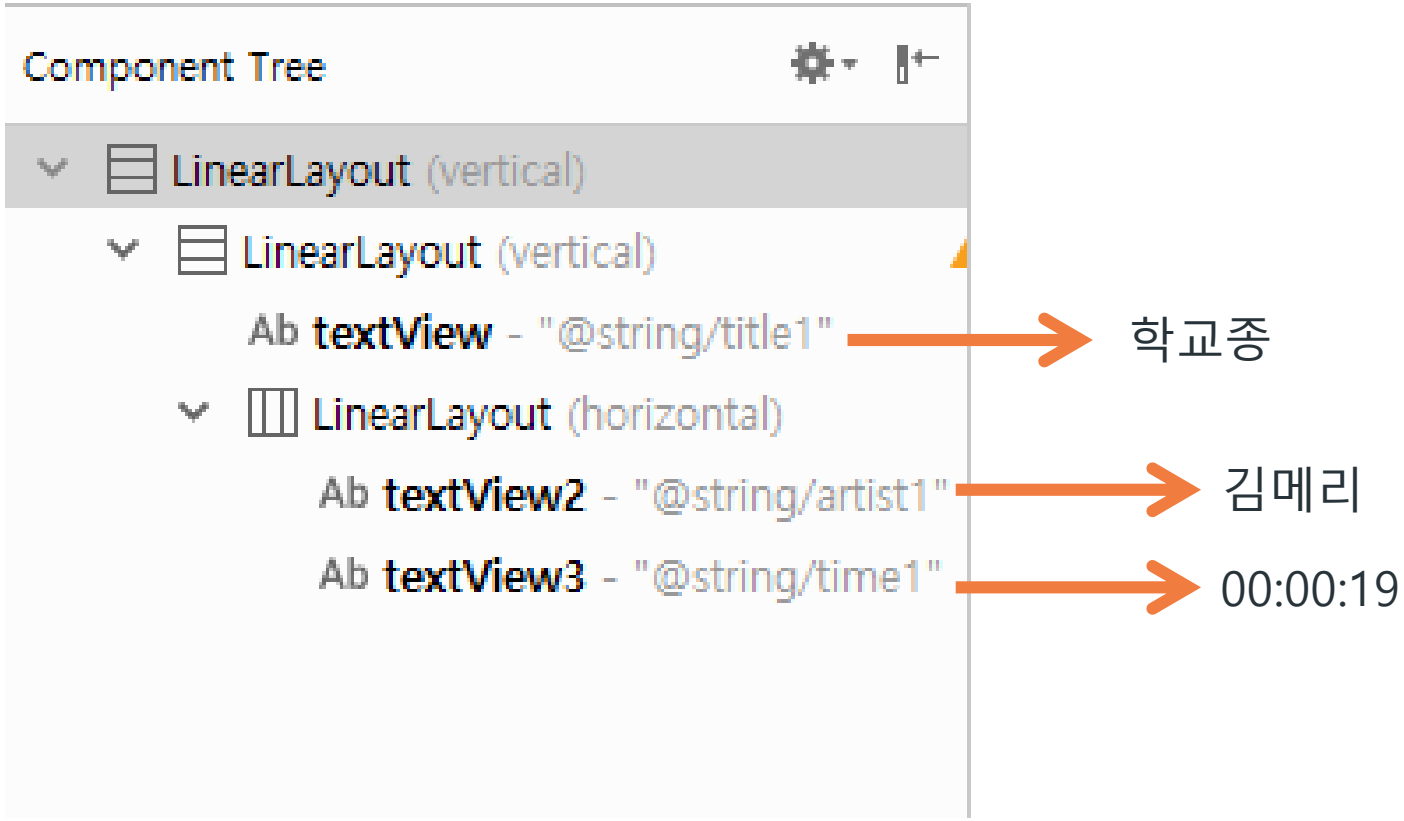




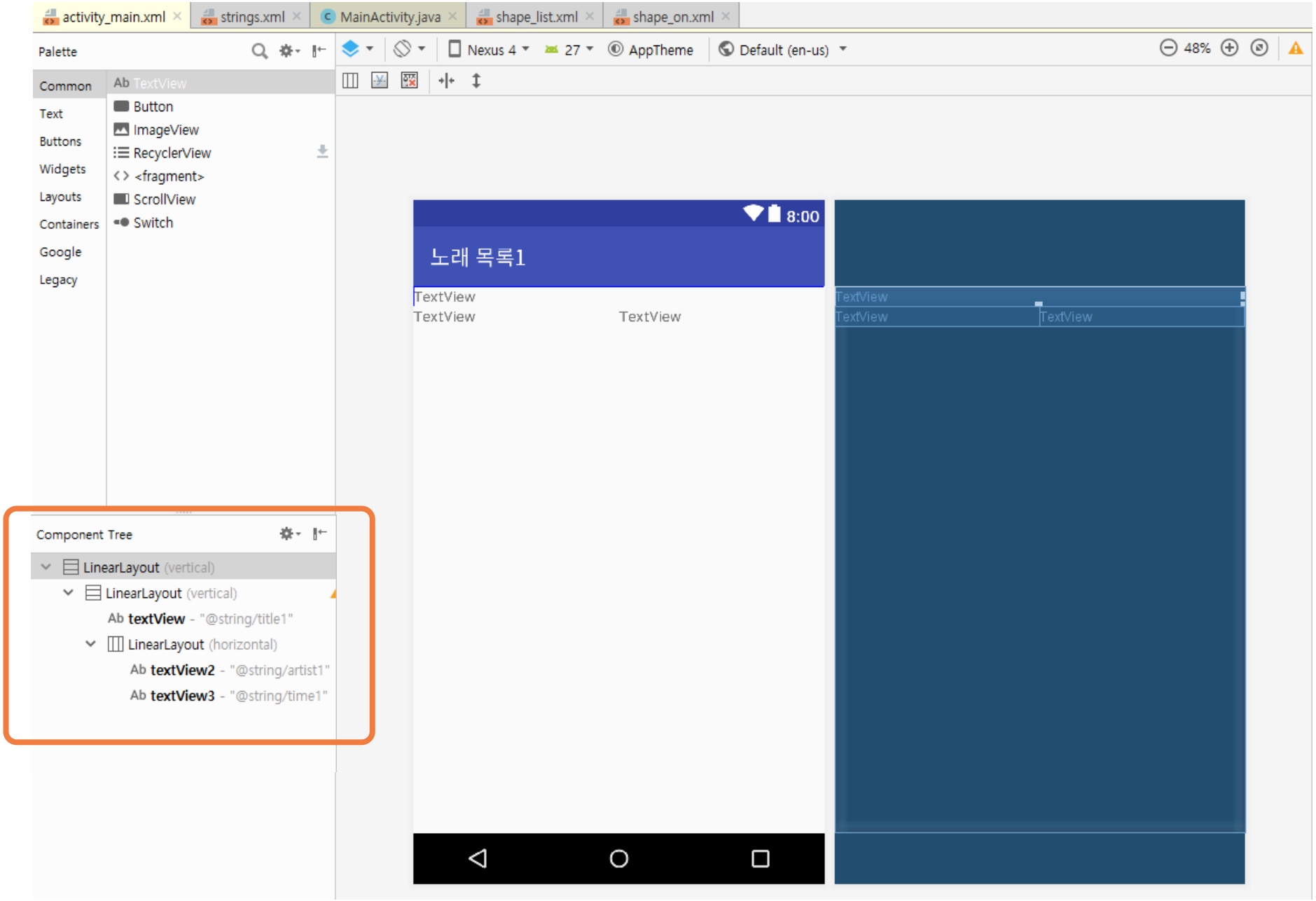
- 노래 목록 표시를 위한 Layout 구조

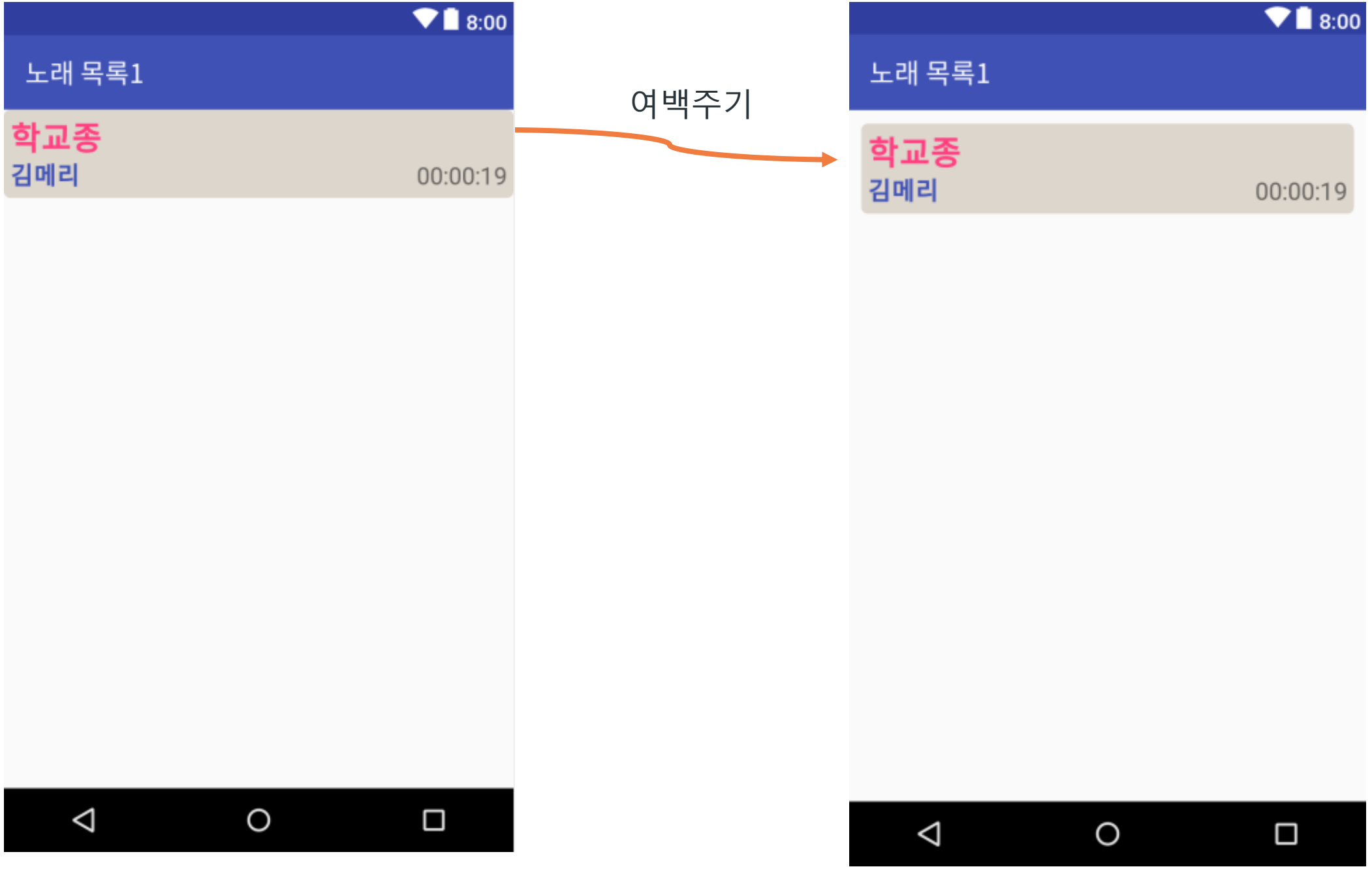


- 노래 목록 표시를 위한 Layout 구조-Component Tree

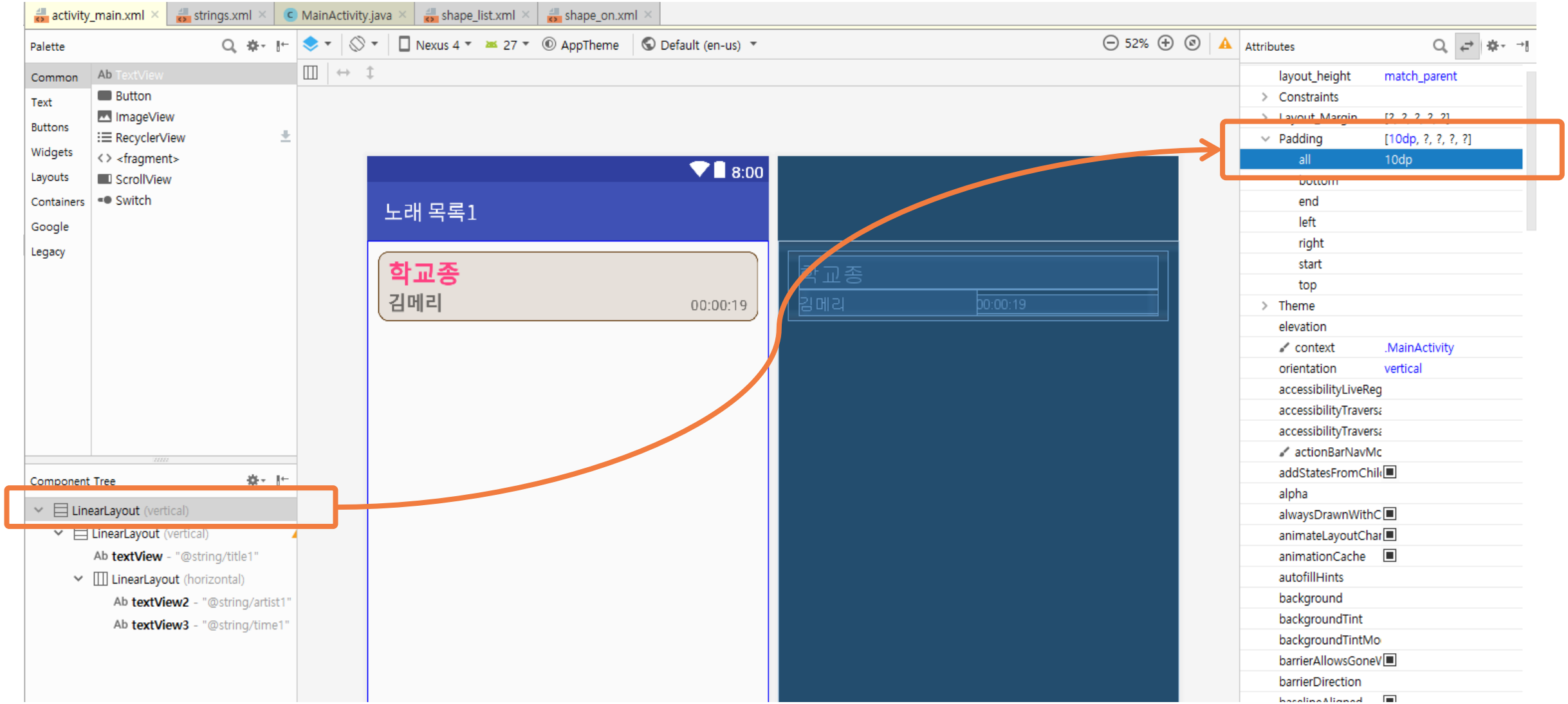


- 노래 목록 표시 Layout

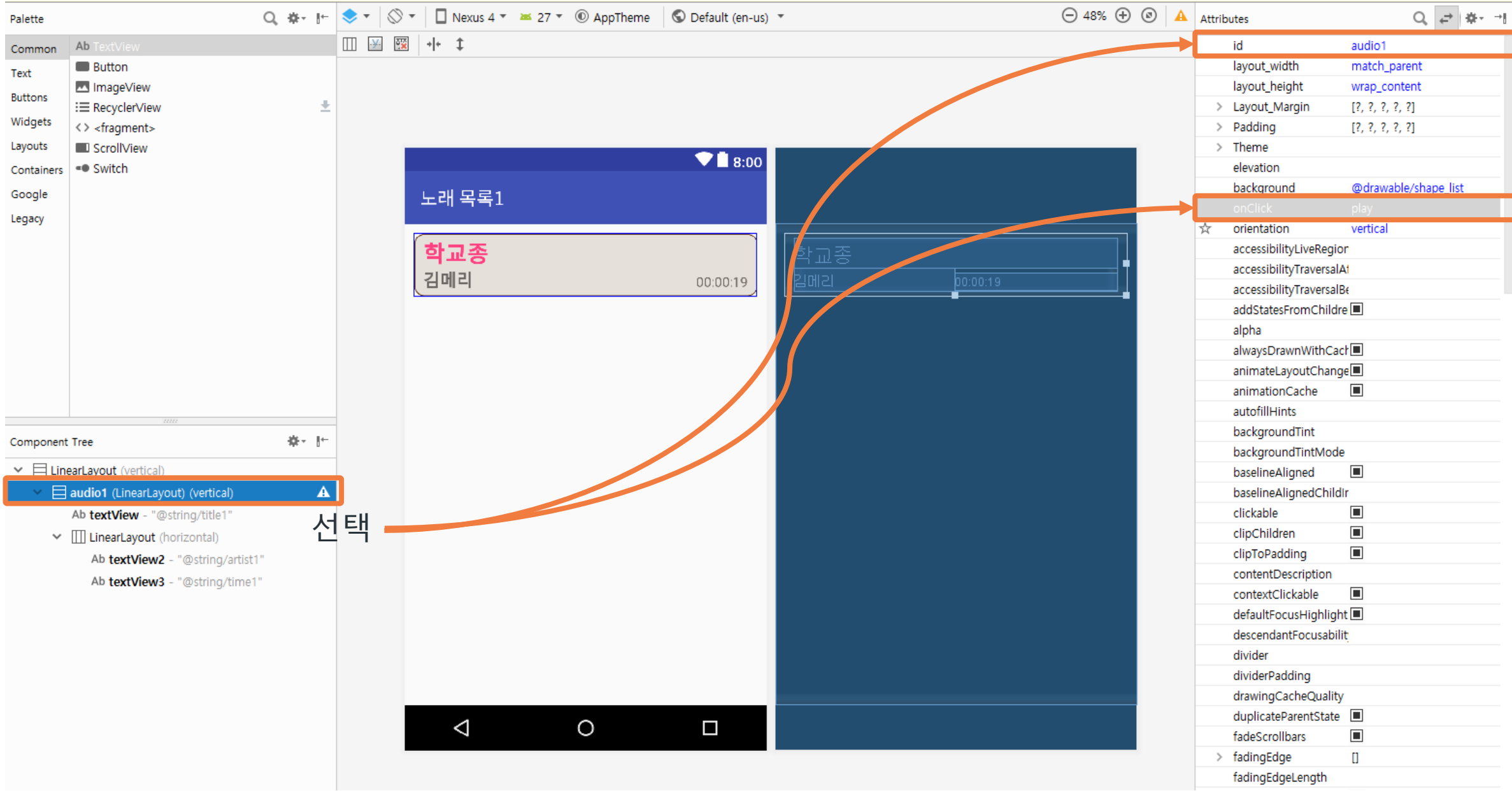



8:00
노래 목록1
학교종
김메리
00:00:19
여백주기
8:00
노래 목록1
학교종
김메리
00:00:19

- padding 속성을 이용한 여백 설정



• LinearLayout클릭 이벤트(onClick 속성)의 콜백함수(play) 설정



2.5 Activity 제어

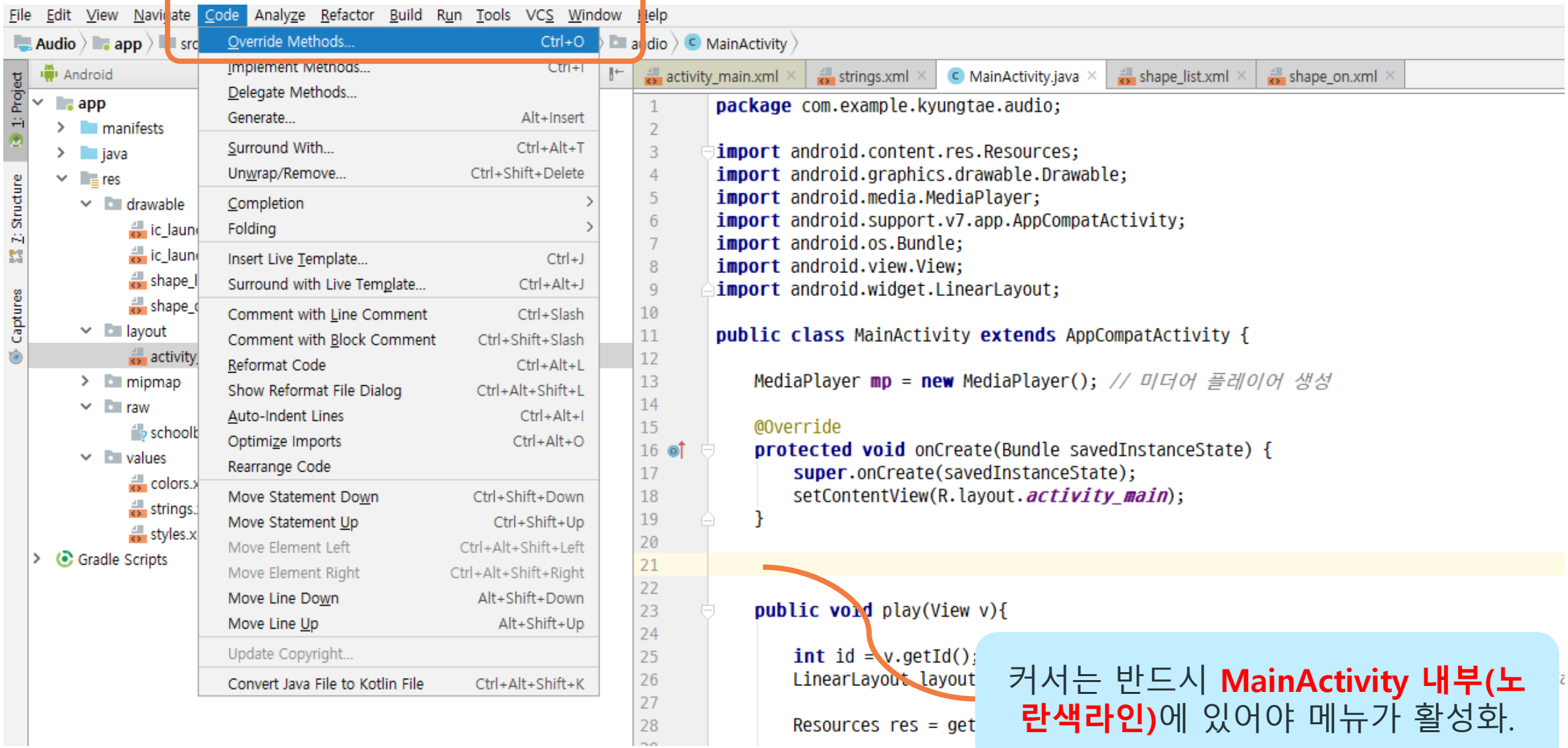
- 오디오 리소스에 대한 MediaPlayer를 생성하고 재생

activity_main.xml | strings.xml | MainActivity.java | shape_list.xml | shape_on.xml

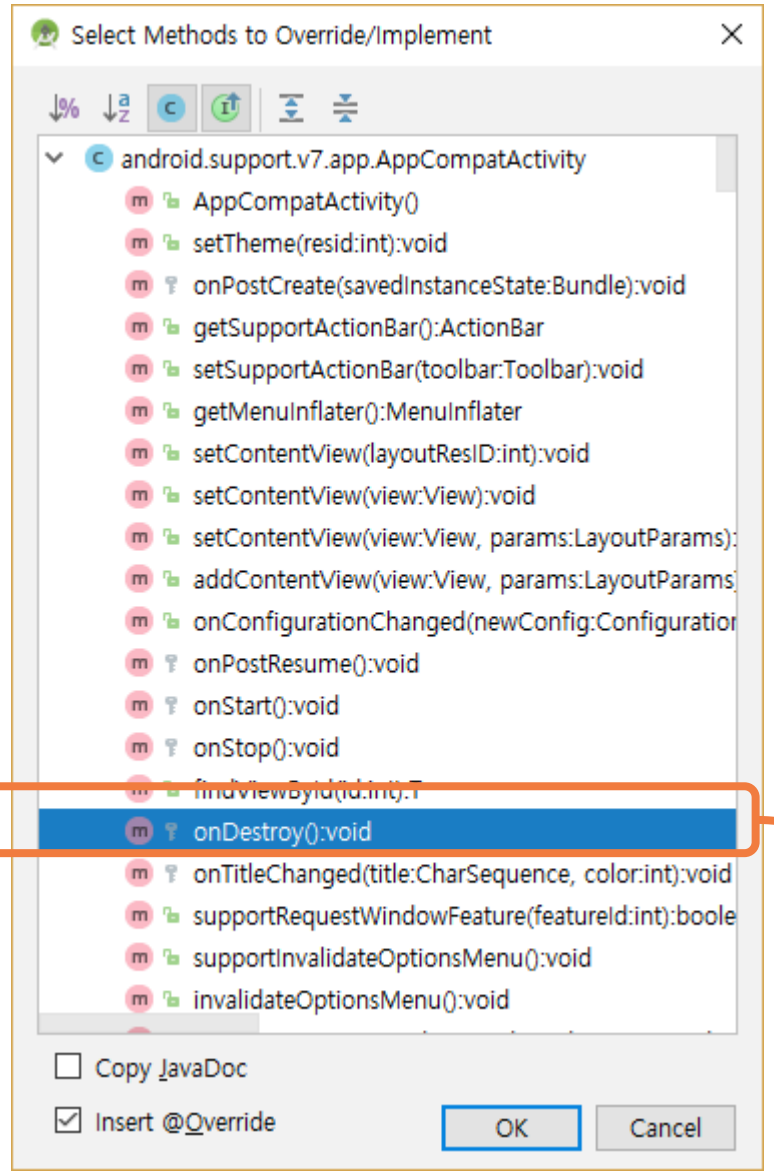
```
1  package com.example.kyungtae.audio;
2
3  import android.content.res.Resources;
4  import android.graphics.drawable.Drawable;
5  import android.media.MediaPlayer;
6  import android.support.v7.app.AppCompatActivity;
7  import android.os.Bundle;
8  import android.view.View;
9  import android.widget.LinearLayout;
10
11 public class MainActivity extends AppCompatActivity {
12
13     MediaPlayer mp = new MediaPlayer(); // 미더어 플레이어 생성
14
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19     }
20
21     public void play(View v){
22
23         int id = v.getId(); // 클릭한 뷰의 ID를 인식
24         LinearLayout layout = (LinearLayout) findViewById(id); // 클릭한 뷰의 ID에 해당하는 LinearLayout 인식
25
26         Resources res = getResources(); // 어플리케이션 리소스 객체 생성
27
28         if(mp.isPlaying()) { // 플레이중이면
29             mp.pause();     // 미디어 플레이어 실행중지
30             // 목록 배경으로 설정
31             Drawable drawable = res.getDrawable(R.drawable.shape_list, theme: null);
32             layout.setBackground(drawable);
```

```
33
34          }else{
35              // raw 폴더의 schoolbell 오디오 파일에 대한 미디어 플레이어 설정
36              mp = MediaPlayer.create( context: this, R.raw.schoolbell);
37              mp.setLooping(false);
38              mp.start();
39
40              // 재생 배경으로 설정
41              Drawable drawable = res.getDrawable(R.drawable.shape_on,  theme: null);
42              layout.setBackground(drawable);
43          }
44      }
45
46  }
47
```

• Activity 소멸될 때 호출되는 메소드(onDestroy()) 추가



- onDestroy()는 수퍼 클래스에 정의 되어 있으므로 Override 함



activity_main.xml | MainActivity.java | strings.xml | shape_list.xml

```java
package com.example.research.audio;

import android.media.MediaPlayer;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    MediaPlayer mp = new MediaPlayer(); // 미디어 플레이어 생성

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

        // raw 폴더의 schoolbell 오디오 파일에 대한 미디어 플레이어 설정
        mp = MediaPlayer.create( context: this, R.raw.schoolbell);
        mp.setLooping(false); // 반복재생 없음
        mp.start(); // Play 시작
    }

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

activity_main.xml | strings.xml | MainActivity.java | shape_list.xml | shape_on.xml

```java
package com.example.kyungtae.audio;

import android.content.res.Resources;
import android.graphics.drawable.Drawable;
import android.media.MediaPlayer;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.LinearLayout;

public class MainActivity extends AppCompatActivity {

    MediaPlayer mp = new MediaPlayer(); // 미더어 플레이어 생성

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

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

    public void play(View v){
```

Activity가 소멸될 때 호출되는 메소드
(Activity 클래스에 정의된 메소드를 재정의함)

미디어 플레이어 실행 중지

미디어 플레이어에 할당된 자원을 해제

Activity 클래스에 정의된
onDestroy() 메소드로 Activity 종료

클래스와 속성/메소드

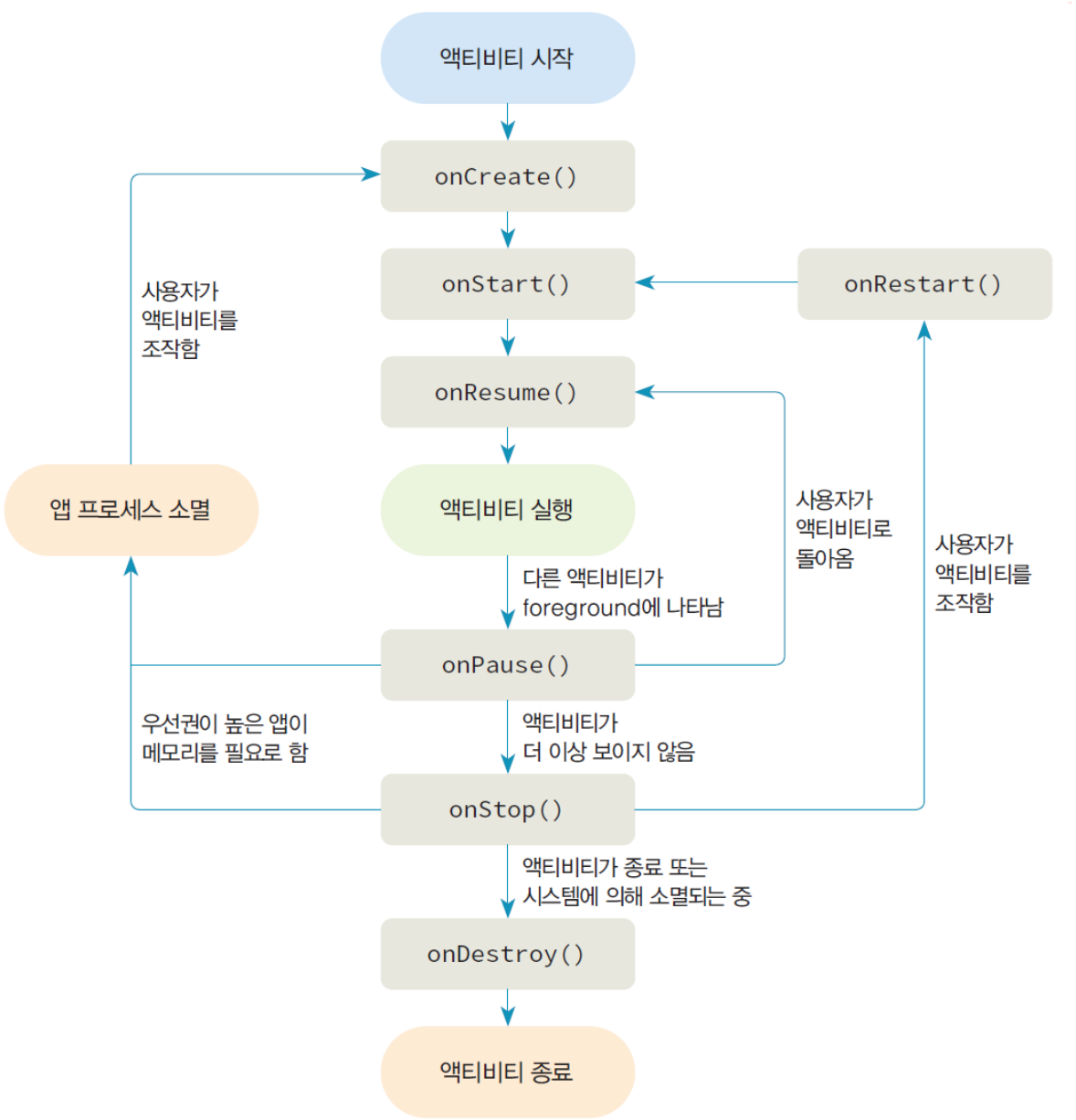
- 클래스

클래스	설명
MediaPlayer	오디오/비디오 재생 및 제어를 위해 사용됨.

- 메소드

클래스	메소드	설명
Activity	void onDestroy()	Activity가 소멸될 때 불려짐
MediaPlayer	static MediaPlayer.create(Context context, int resid)	주어진 리소스 ID에 대한 MediaPlayer를 생성함
	MediaPlayer()	MediaPlayer 클래스의 생성자
	void setLooping(boolean looping)	미디어 재생을 반복(true) 또는 비반복(false)으로 정함
	void start()	미디어를 재생함
	void stop()	미디어 재생을 중지함
	void release()	MediaPlayer 객체에 할당된 자원을 해제함

Activity의 생명주기



메소드	설명
onCreate()	액티비티가 생성될 때
onRestart()	액티비티가 중지되었다가 다시 시작하기 전
onStart()	액티비티가 사용자가 보여질 때
onResume()	액티비티가 사용자와 상호작용할 때
onPause()	다른 액티비티를 시작할 때
onStop()	액티비티가 사용자에게 더 이상 보여지지 않을 때
onDestroy()	액티비티가 소멸될 때

출처: www.android.com

쉼터

다양한 화면 크기를 지원하기 위한 화면 출력 단위
(화면 출력 용어와 개념)

<table>
<tr><th>용어</th><th>개념</th></tr>
<tr><td>화면 크기
(screen size)</td><td>

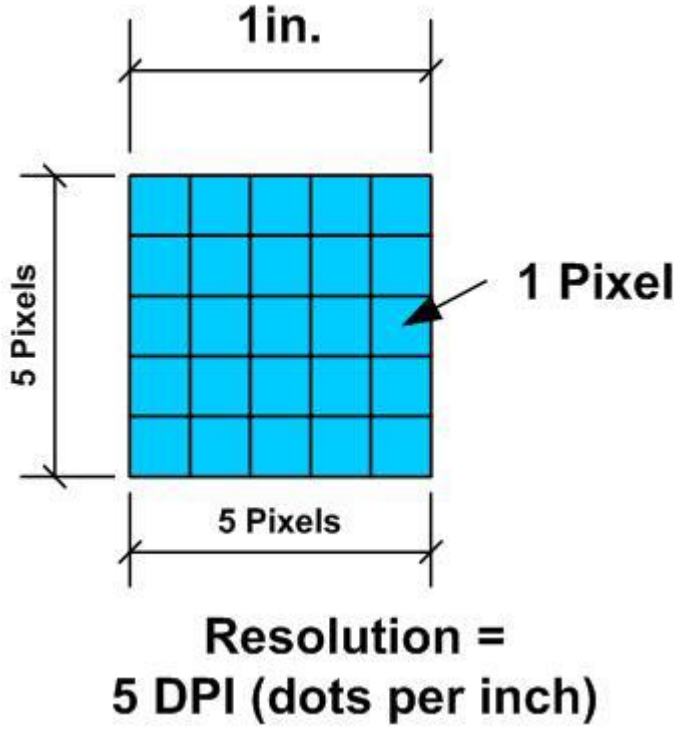
화면 대각선의 실제 길이
안드로이드의 4가지 분류: small, normal, large, extra-large

<table>
<tr><th>화면크기 분류</th><th>최소 화면밀도</th></tr>
<tr><td>small</td><td>426dp x 320dp</td></tr>
<tr><td>normal</td><td>470dp x 320dp</td></tr>
<tr><td>large</td><td>640dp x 480dp</td></tr>
<tr><td>extra-large</td><td>960dp x 720dp</td></tr>
</table>
</td></tr>
<tr><td>화면 밀도
(screen density)</td><td>

화면 면적 당 픽셀 수, 대개 1인치 당 픽셀 수를 나타내는 dpi(dot per inch) 또는 pd(pixel density)를 사용함
안드로이드의 6가지 화면밀도 분류와 dpi 수

<table>
<tr><th>화면크기 분류</th><th>최소 화면밀도</th></tr>
<tr><td>ldpi(low)</td><td>120</td></tr>
<tr><td>mdpi(medium)</td><td>160</td></tr>
<tr><td>hdpi(high)</td><td>240</td></tr>
<tr><td>xhdpi(extra-high)</td><td>320</td></tr>
<tr><td>xxhdpi(extra-extra-high)</td><td>480</td></tr>
<tr><td>xxxhdpi(extra-extra-extra-high)</td><td>640</td></tr>
</table>
</td></tr>
</table>

- DPI(Dots Per Inch)라는 단위



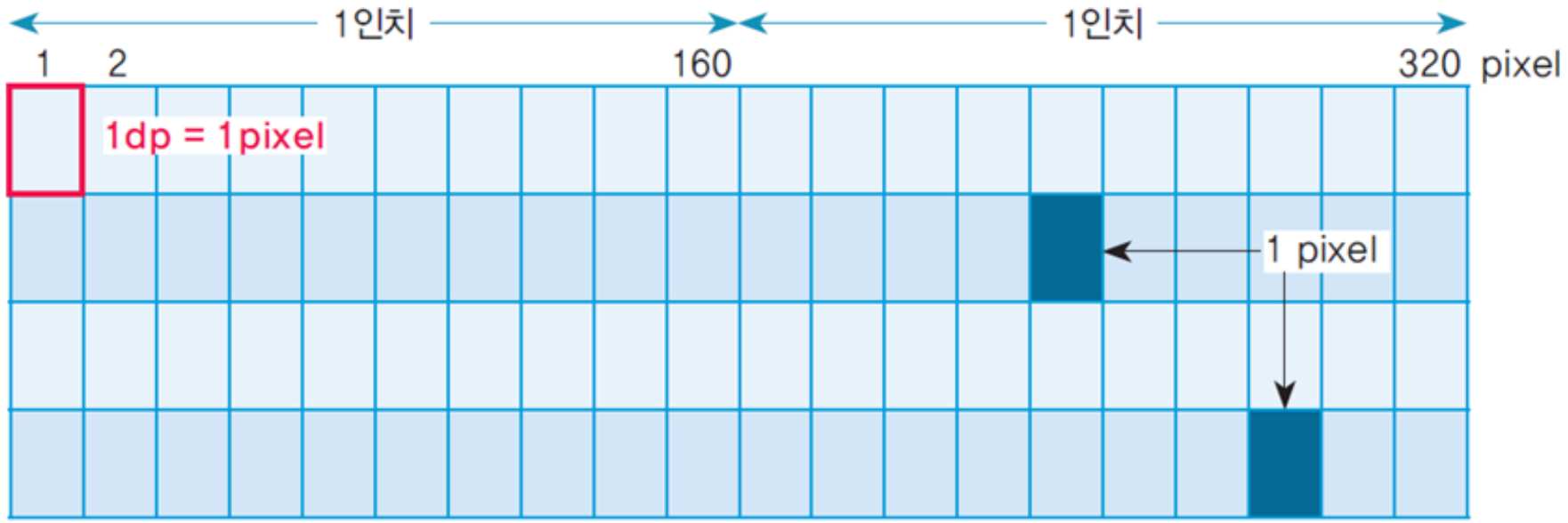
- 출처: http://solarisailab.com/archives/179

용어	개념
방향 (orientation)	▪ 사용자 관점의 화면 방향 ▪ landscape(가로 방향이 길게 보임), portrait(세로 방향이 길게 보임)
해상도 (resolution)	▪ 화면 상의 물리적인 픽셀 수(장비의존적)

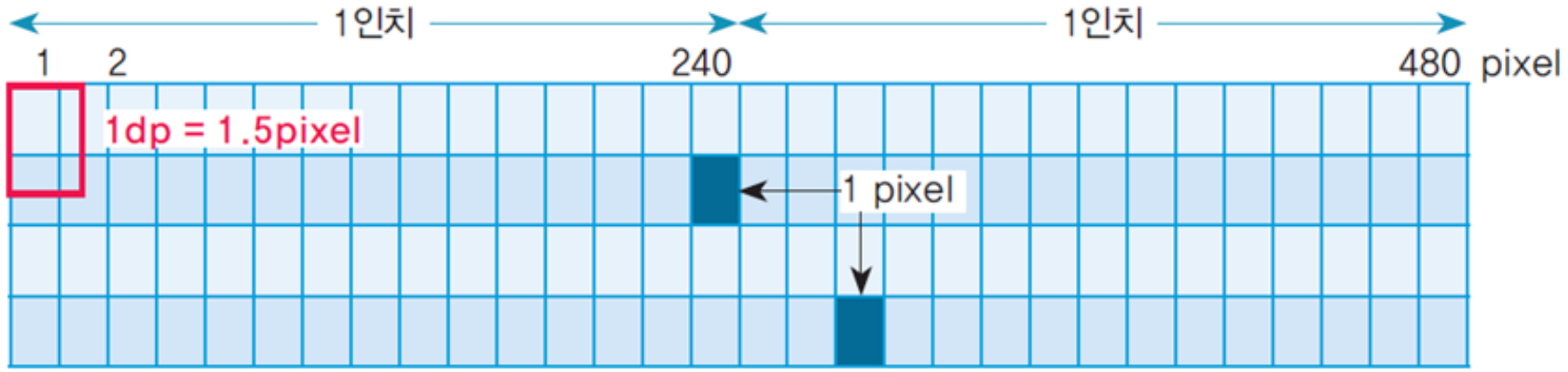
dp
(density-independent pixel, dip)
	▪ **밀도와 무관한 가상 픽셀**
	▪ **이론상 어떠한 해상도에서도 같은 크기를 보여 주는 것이 핵심 개념**
	▪ 1인치 당 160 픽셀 수를 가진 medium 화면 밀도 유형을 기준으로 볼 때 1dp는 1pixel에 대응됨
	▪ 픽셀과 dp의 관계
	  ▪ px = dp x dpi / 160
	  ▪ 예) dpi가 160인 경우 1dp은 1pixel과 같음. 그러나, dpi가 240인 경우는 1dp는 1.5pixel이 됨.
	  즉, 모바일 기기의 해상도는 다르더라도 dp를 사용하면 같은 배율로 화면에 출력됨

- dp와 pixel의 관계(px = dp x dpi / 160)

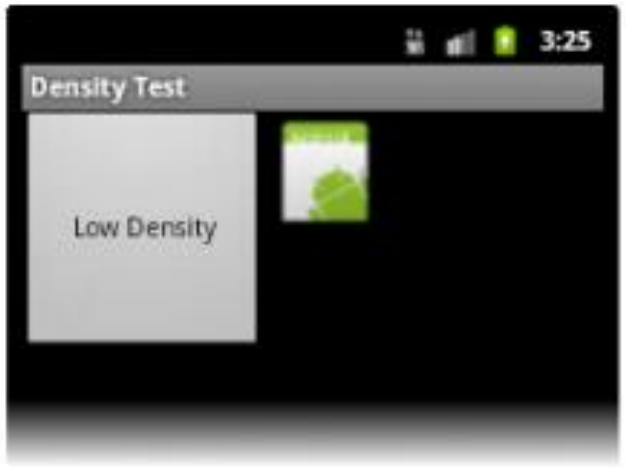
① 160 dpi의 경우, 1pixel = 1dp × 160dpi / 160



② 240 dpi의 경우, 1.5pixel = 1dp × 240dpi / 160



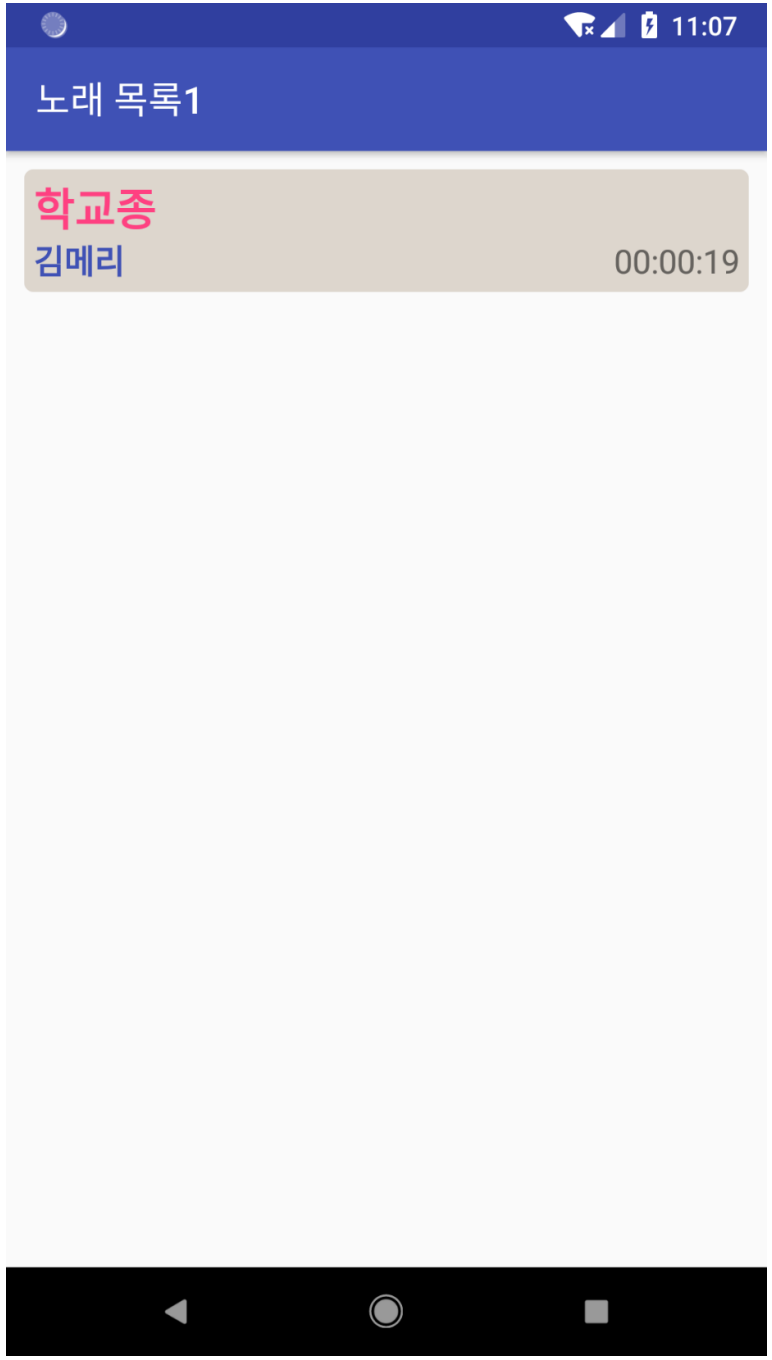
Low, Medium, High-density 화면밀도에 **dpi 단위**로 크기를 지정했을 때



Low, Medium, High-density 화면밀도에 **DP 단위**를 지원했을 때

- 출처: http://solarisailab.com/archives/179

Outputs



Question & Answer