

앱의 프로젝트 구조와 실행원리



- 프로젝트 예제: Hello Android 프로젝트
- 프로젝트 개발
- 프로젝트 파일 구조
- 앱의 실행 원리
- 프로젝트 소스 간의 연관성
- 어플리케이션 아이콘의 변경





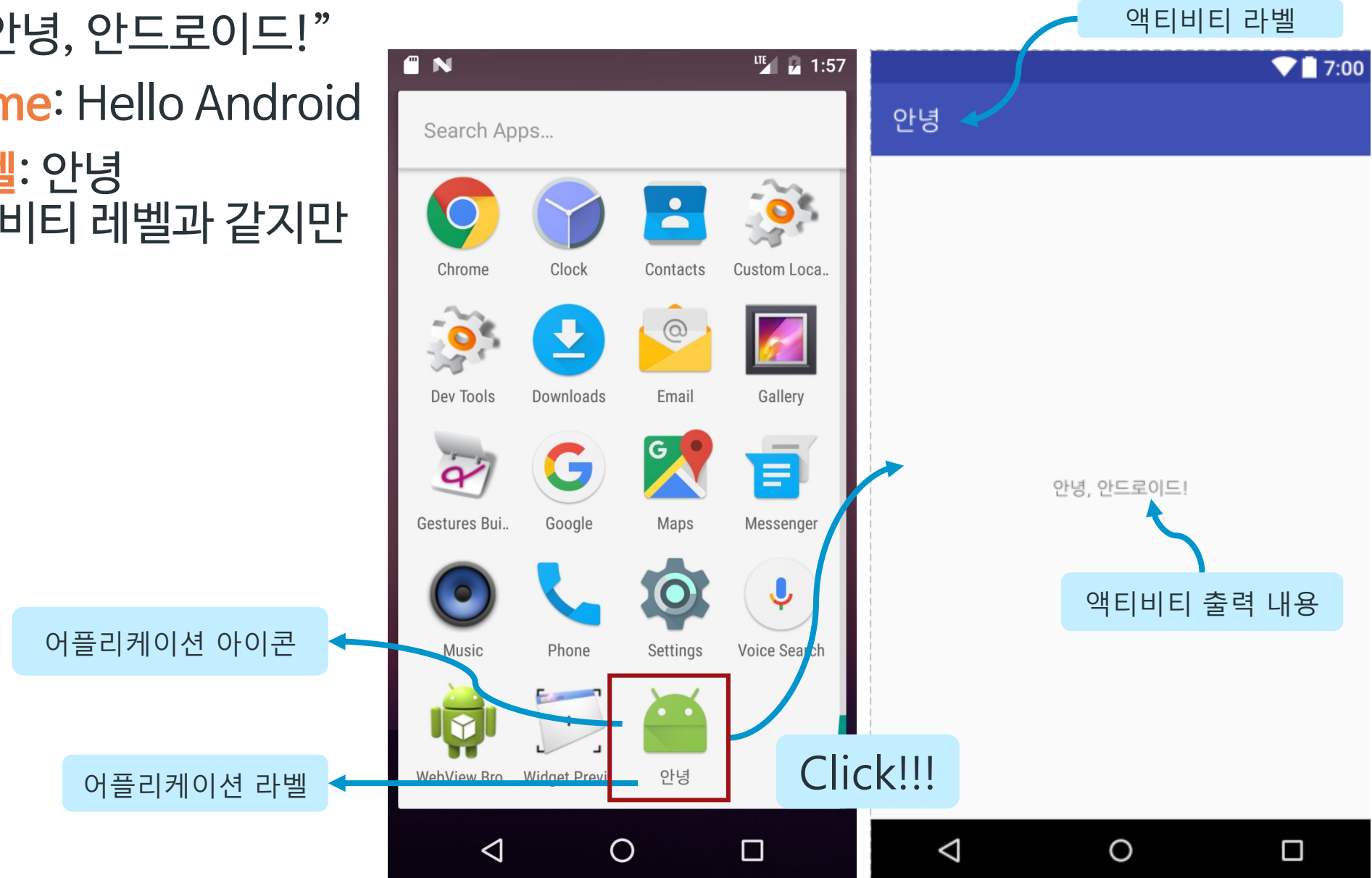
3.1 프로젝트 예제

Hello Android 프로젝트(프로젝트의 시작)

Step 1. Hello Android 프로젝트

4

- **프로젝트 개요:** “안녕, 안드로이드!”
- **Application Name:** Hello Android
- **어플리케이션 라벨:** 안녕
(기본적으로 액티비티 레벨과 같지만 다르게 지정가능)

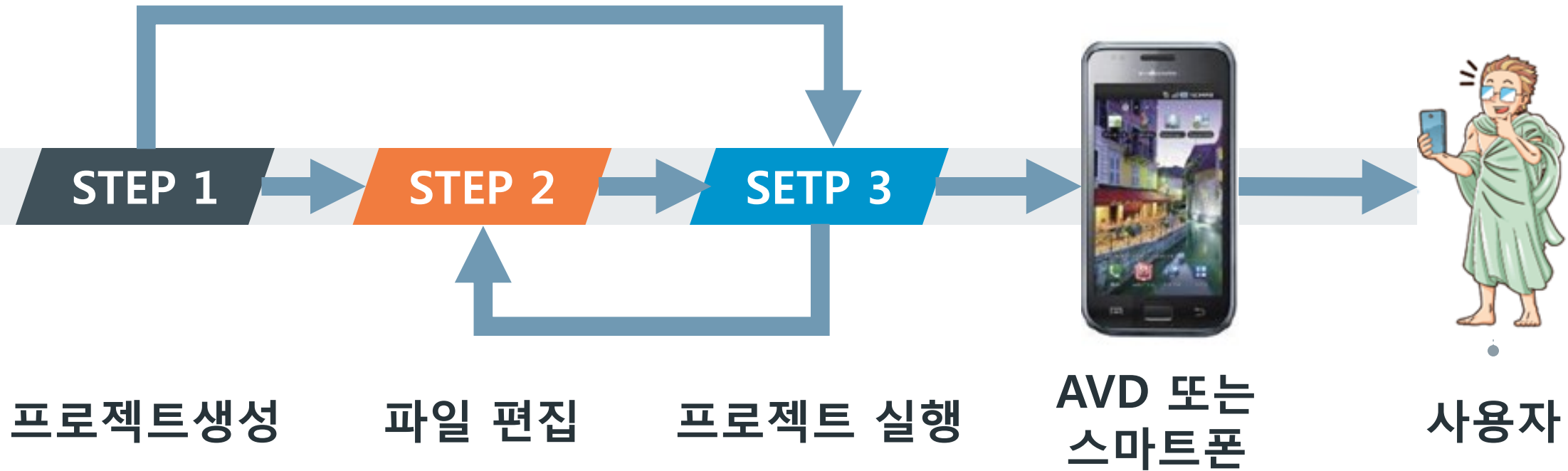




3.2 프로젝트 개발

Step 0.프로젝트 개발 과정

6



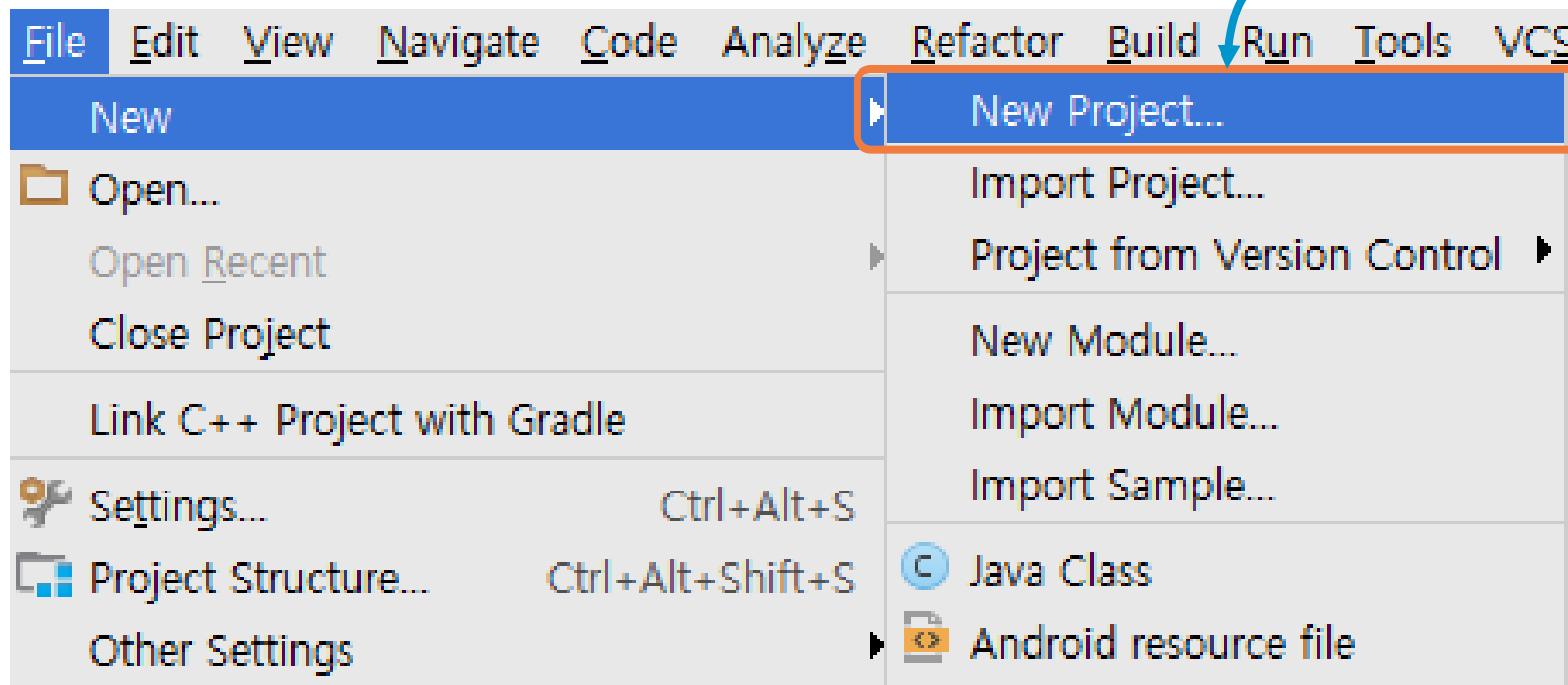
Step 1. 프로젝트 생성

7

- 프로젝트는 다음 5단계 과정을 거쳐 만들게 된다.

① 프로젝트 시작

Create new project



② 프로젝트 구성

8

Create New Project

New Project
Android Studio

Configure your new project

Application name: My Application

Company domain: kyungtae.example.com

Package name: com.example.kyungtae.myapplication

☐ Include C++ support

Project location: C:\Users\Kyungtae\AndroidStudioProjects\MyApplication2

Previous Next Cancel Finish

Application name:
"My Application"

Company Domain:
"USERID.example.com"으
로 지정됨

폴더 위치 변경 버튼

Project location:
저장할 폴더\MyApplication

③ 실행 디바이스

9

Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK

API 15: Android 4.0.3 (IceCreamSandwich)

API 15: Android 4.0.3 (IceCreamSandwich)

API 16: Android 4.1 (Jelly Bean)

API 17: Android 4.2 (Jelly Bean)

API 18: Android 4.3 (Jelly Bean)

API 19: Android 4.4 (KitKat)

☐ Wear

Minimum SDK

API 20: Android 4.4W (KitKat Wear)

API 21: Android 5.0 (Lollipop)

API 22: Android 5.1 (Lollipop)

☐ TV

Minimum SDK

API 21: Android 5.0 (Lollipop)

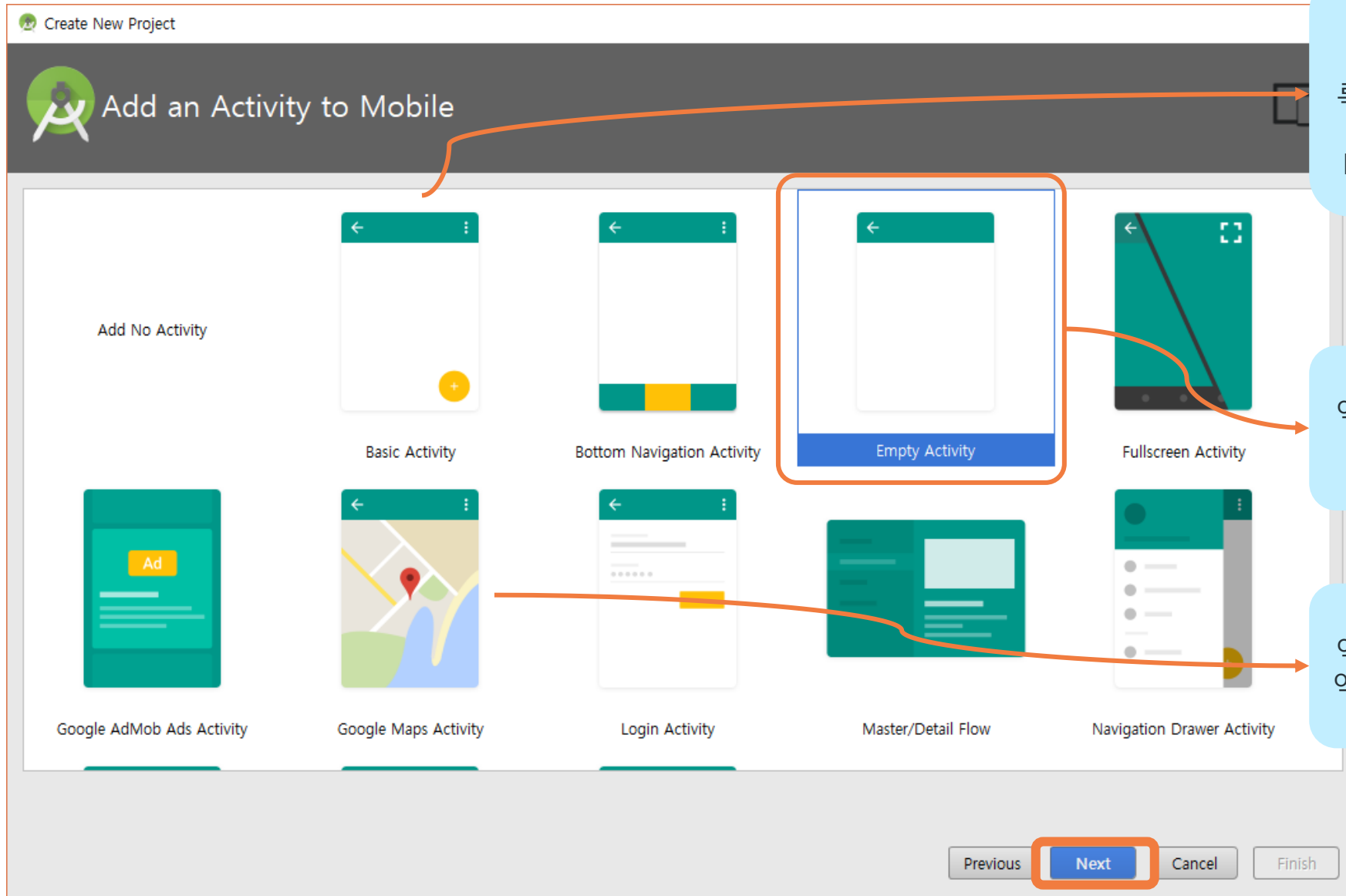
☐ Android Auto

Previous Next Cancel Finish

실행할 스마트폰의 안드로이드
버전을 고려한 SDK 버전 설정:
Android 4.4 (KitKat)

④ 액티비티 유형(Empty Activity)

10



Basic Activity

액티비티 라벨을 표시하는 타이틀 바(좌측 상단)와 네비게이션이 가능하도록 메뉴가 나타나는 액션 바(우측 상단) 외에 화면 위에 떠다니는 버튼인 Floating Action Button(우측하단)으로 구성

Empty Activity

액티비티 라벨을 표시하는 타이틀 바만 나타나는 가장 단순한 화면 구성(대부분의 예제에서 사용)

Google Map Activity

액티비티 라벨을 표시하는 타이틀 바와 액션바가 나타나며, 구글맵을 표시할 수 있는 화면 구성

⑤ 액티비티 구성 파일 이름 설정

11

Create New Project

Customize the Activity

Creates a new empty activity

Activity Name: MainActivity

☒ Generate Layout File

Layout Name: activity_main

☒ Backwards Compatibility (AppCompat)

Empty Activity

The name of the activity class to create

Previous Next Cancel Finish

앱이 실행 시 처음 실행되는 액티비티
이름(그대로 됨): MainActivity

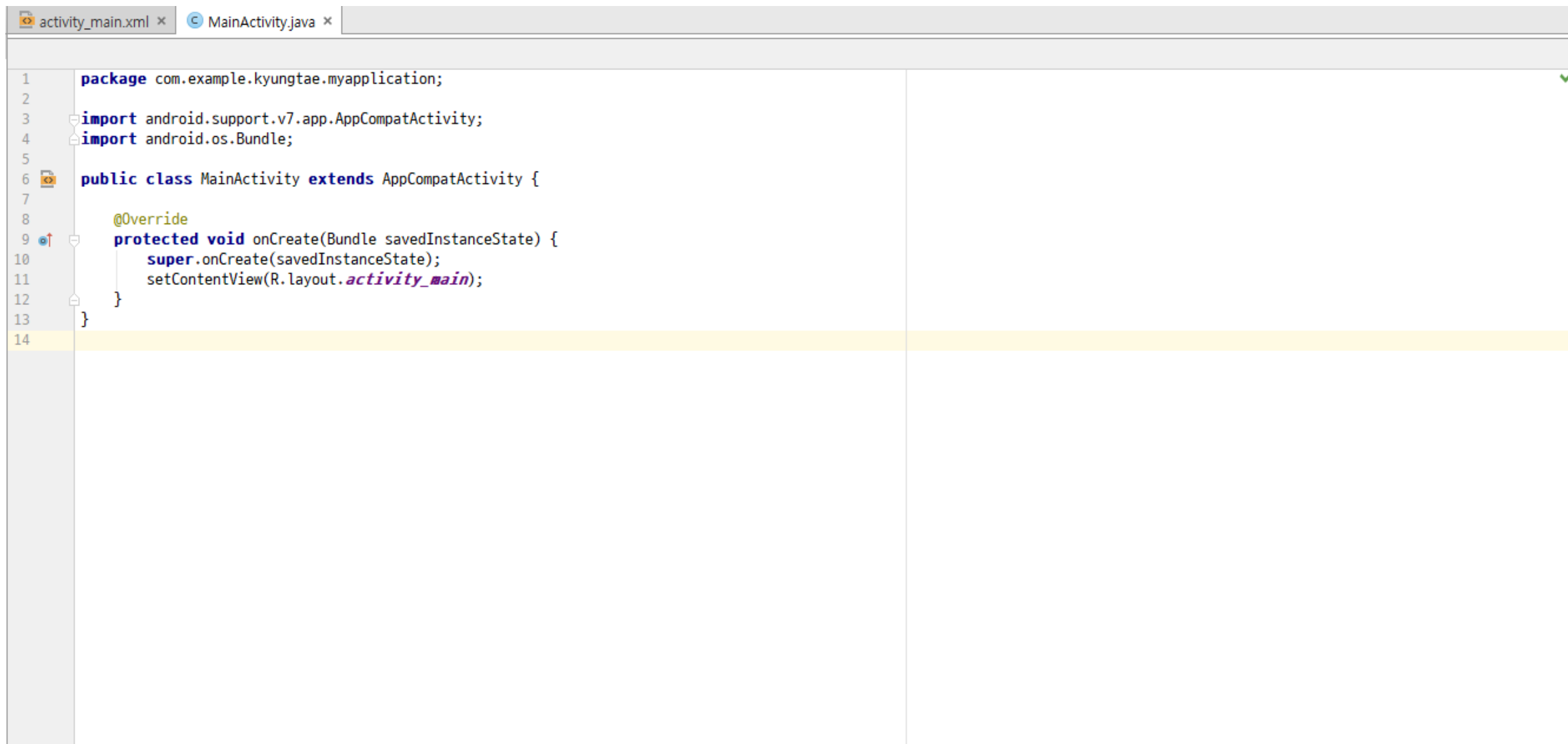
액티비티에 출력될 레이아웃 이름
(그대로 됨): activity_main



프로젝트 생성 완료 – 프로젝트 보기

MainActivity.java 클래스

13



The screenshot shows an IDE window with two tabs: 'activity_main.xml' and 'MainActivity.java'. The 'MainActivity.java' tab is active, displaying the following code:

```
1 package com.example.kyungtae.myapplication;
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
```

The code is syntactically correct, as indicated by a green checkmark in the top right corner. The line numbers 1 through 14 are visible on the left side of the editor.

activity_main.xml – Text view

14

The screenshot displays the Android Studio environment with two main panels. The left panel shows the XML code for `activity_main.xml`, and the right panel shows a visual preview of the layout.

XML Code (activity_main.xml):

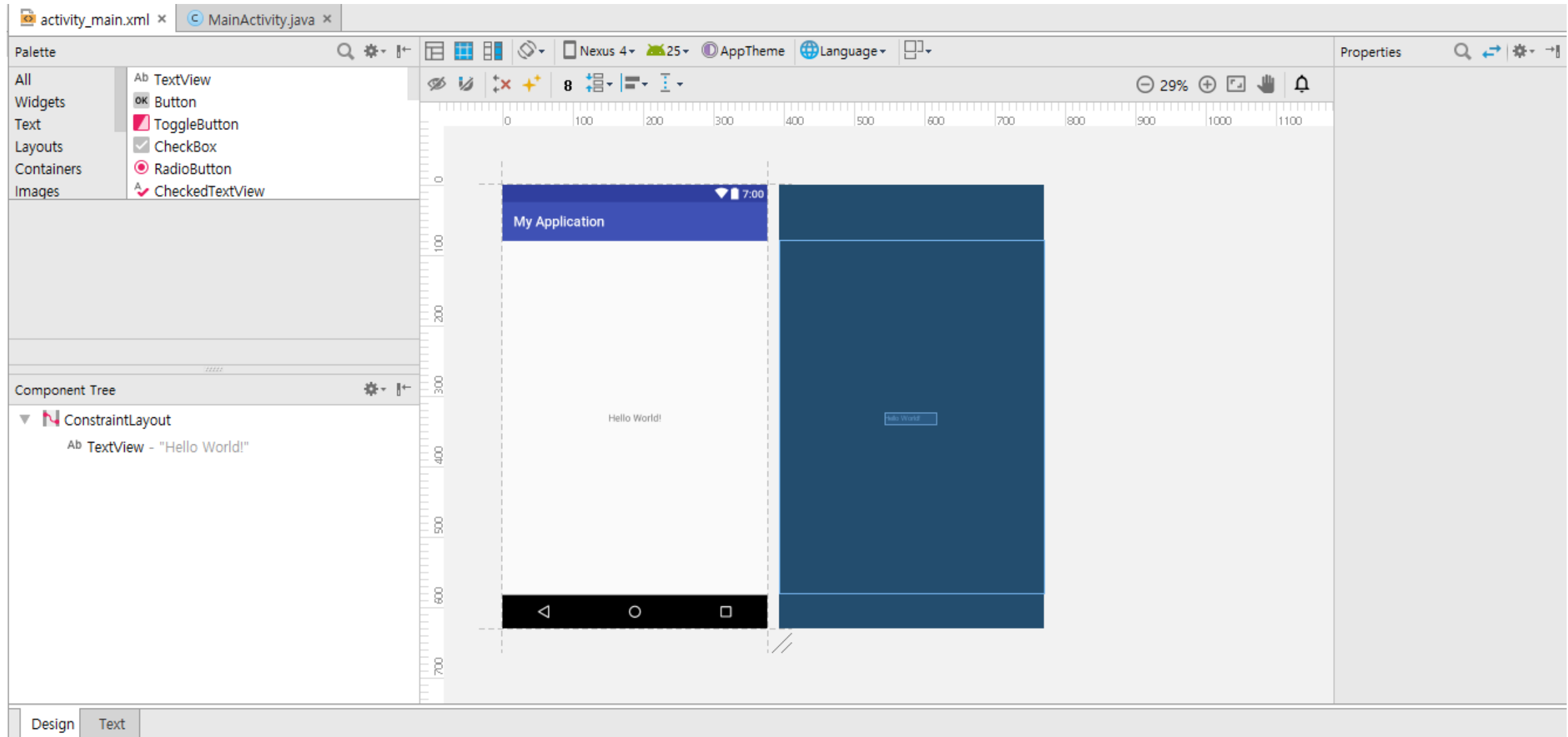
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     tools:context="com.example.kyungtae.myapplication.MainActivity">
8
9     <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Hello World!"
13         app:layout_constraintBottom_toBottomOf="parent"
14         app:layout_constraintLeft_toLeftOf="parent"
15         app:layout_constraintRight_toRightOf="parent"
16         app:layout_constraintTop_toTopOf="parent" />
17
18 </android.support.constraint.ConstraintLayout>
19
20
```

Visual Preview:

The preview window shows a mobile device screen with a blue header bar containing the text "My Application". Below the header, the text "Hello World!" is displayed in the center. The screen is framed by a black navigation bar at the bottom. The interface includes a toolbar with various design tools and a palette on the left side of the preview area.

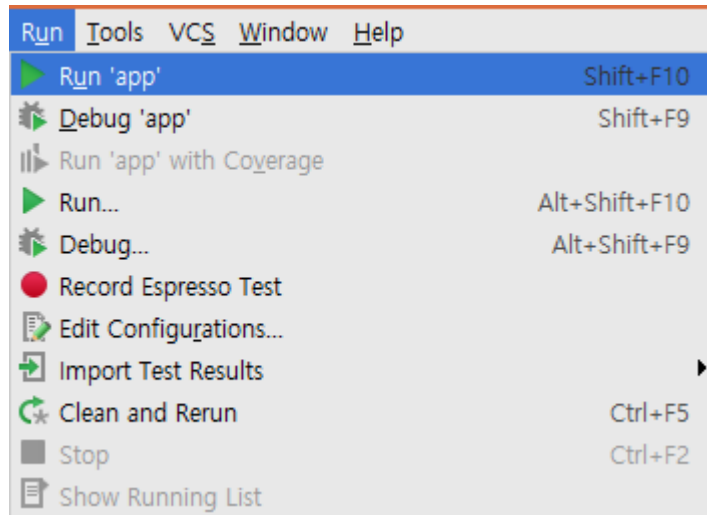
activity_main.xml – Design view

15



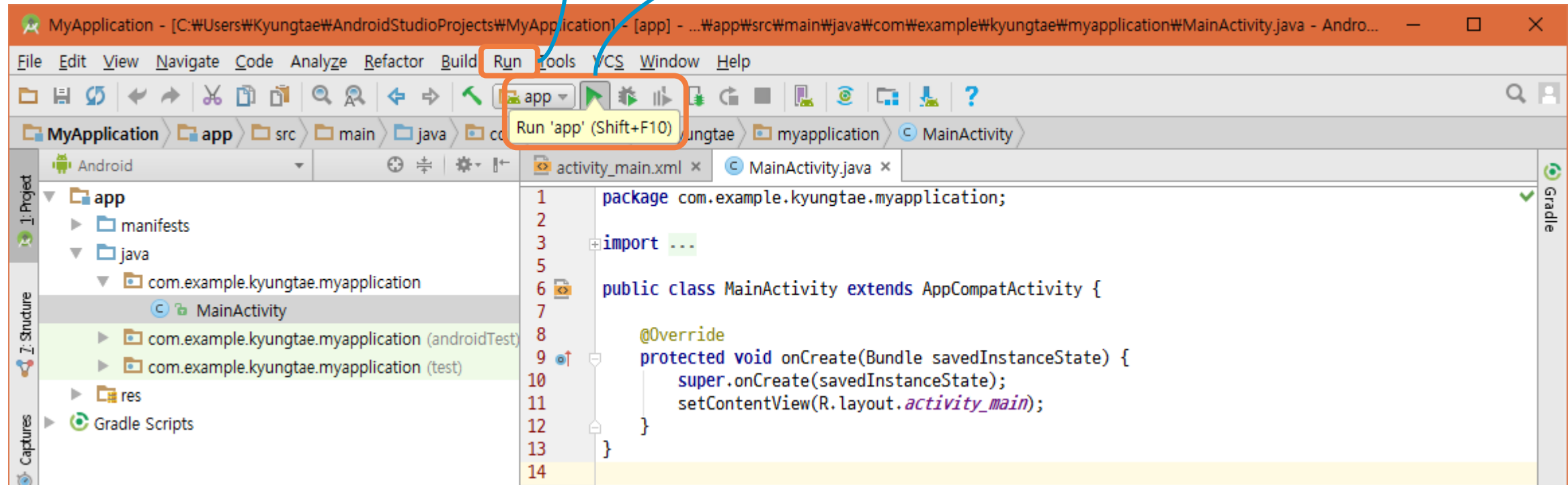
프로젝트 실행[단, 편집 전 실행]

16

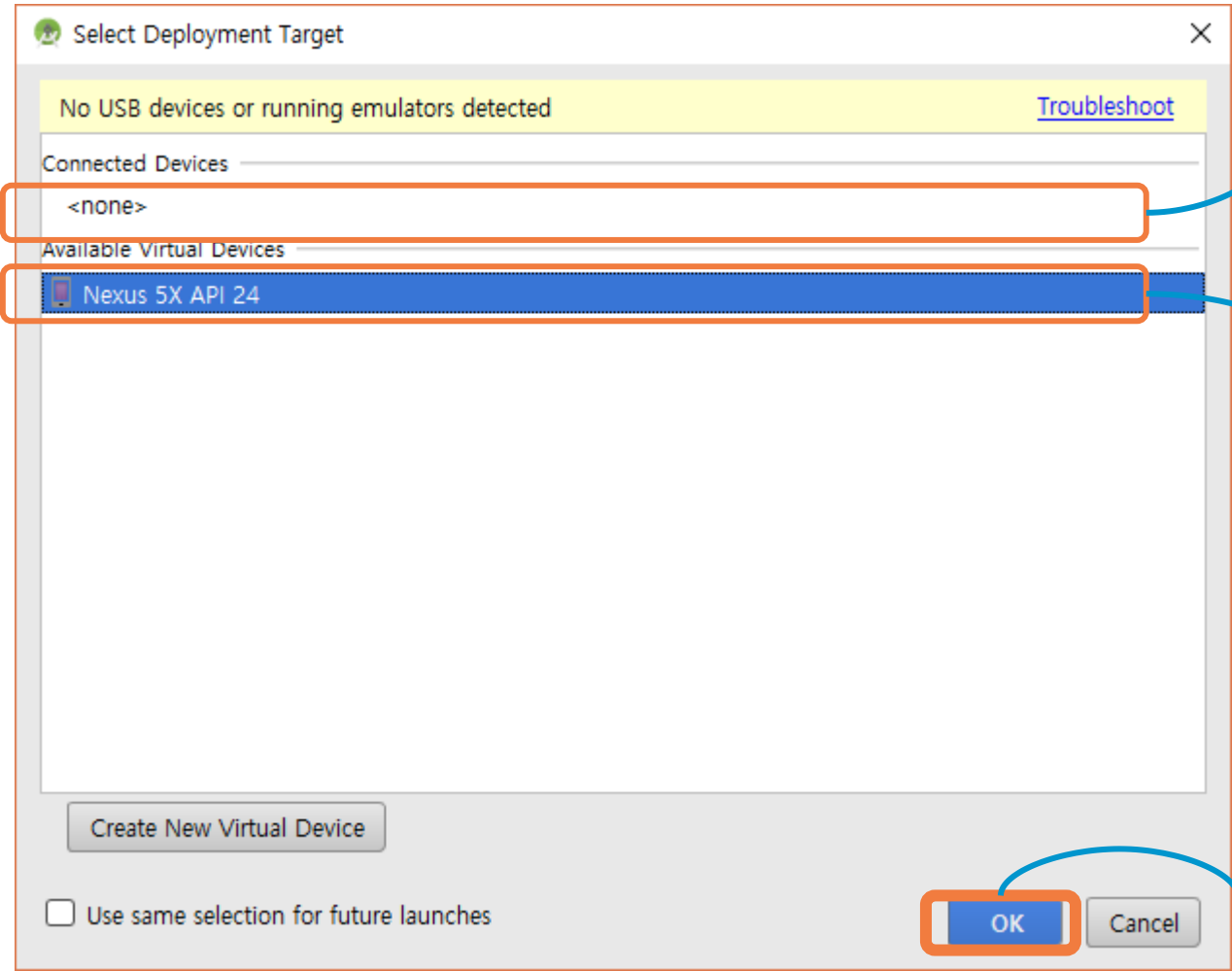


Run → Run 'app' 메뉴 클릭

앱 실행 아이콘 클릭



- Select Deployment Device



The screenshot shows the 'Select Deployment Target' dialog box. It has a title bar with an Android icon and a close button. The main content area is divided into two sections: 'Connected Devices' and 'Available Virtual Devices'. The 'Connected Devices' section shows '<none>' and is highlighted with an orange rectangle. A blue arrow points from this rectangle to a light blue callout box containing the text '데이터 케이블로 연결된 스마트폰'. The 'Available Virtual Devices' section shows 'Nexus 5X API 24' and is highlighted with a blue rectangle. A blue arrow points from this rectangle to a light blue callout box containing the text 'AVD'. At the bottom of the dialog, there is a 'Create New Virtual Device' button, a checkbox labeled 'Use same selection for future launches', and 'OK' and 'Cancel' buttons. The 'OK' button is highlighted with an orange rectangle, and a blue arrow points from it to a light blue callout box containing the text '스마트폰 또는 AVD를 선택하고 'OK' 버튼을 클릭'.

Select Deployment Target

No USB devices or running emulators detected [Troubleshoot](#)

Connected Devices

<none>

Available Virtual Devices

Nexus 5X API 24

Create New Virtual Device

☐ Use same selection for future launches

OK Cancel

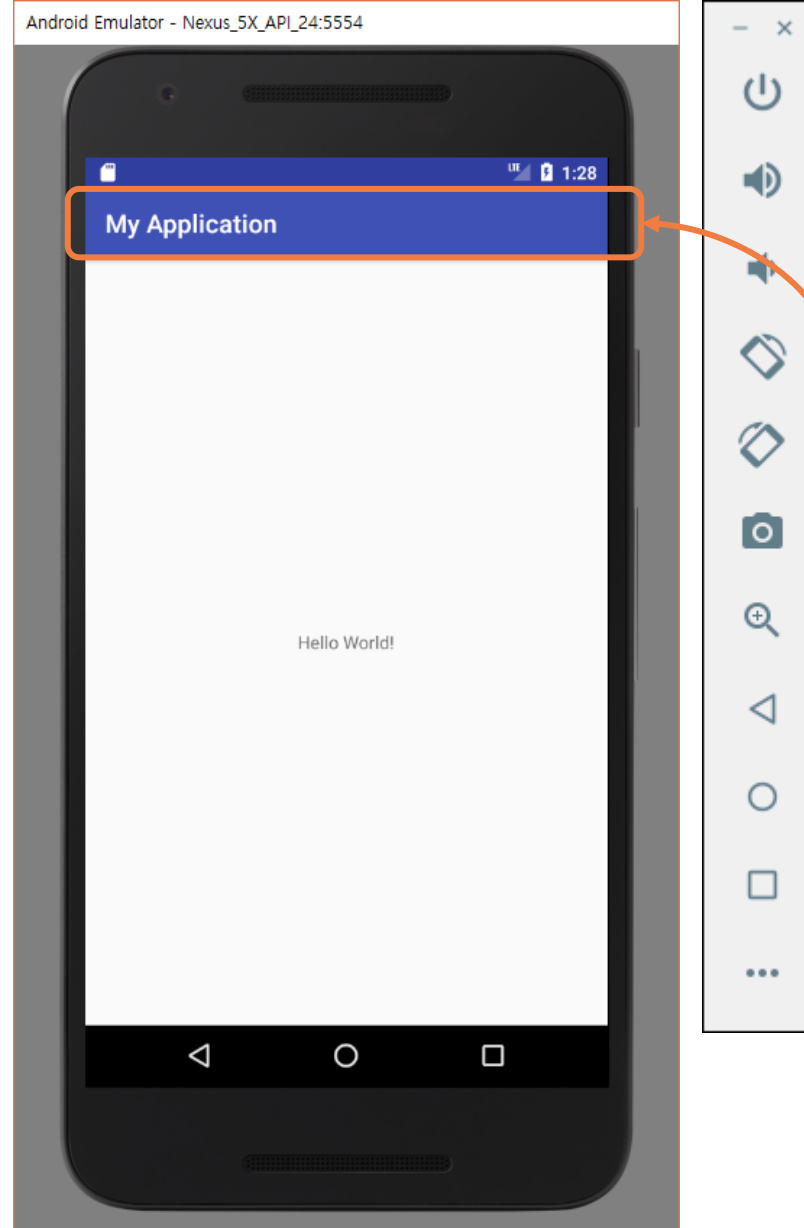
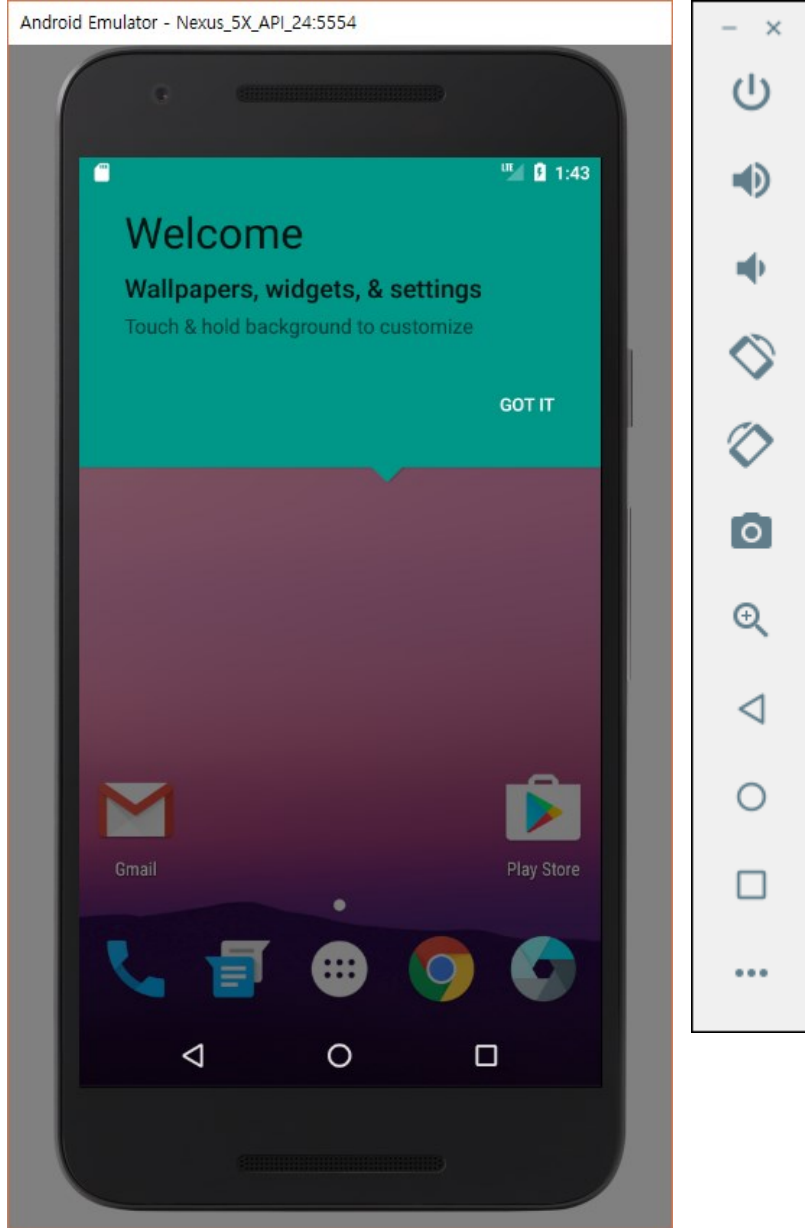
데이터 케이블로 연결된
스마트폰

AVD

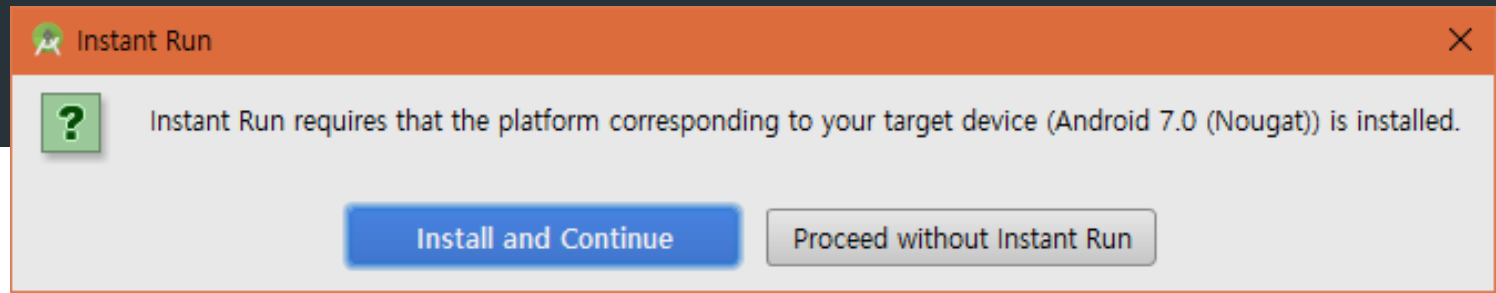
스마트폰 또는 AVD를 선택하고
'OK' 버튼을 클릭

AVD 실행 결과

18



액티비티 라벨:
"My Application"



- 앱의 업데이터터 간 시간을 대폭 줄여준다. 즉 코드를 수정하는 즉시 적용되는 기능으로 코드를 수정하는 순간 빌드와 배포(deploy)없이 바로 결과를 확인
- 첫 빌드를 완료하는 데 오랜 시간이 걸릴 수도 있지만 Instant Run은 APK를 빌드하지 않고 후속 업데이트를 앱에 푸시하므로 변경 사항을 빨리 확인할 수 있다.
- hot swap, warm swap, cold swap 3가지 종류가 있다.
- 사용 요건
 - Android Plugin for Gradle 버전 2.0.0 이상을 사용
 - 앱의 모듈 레벨(build.gradle)이 minSdkVersion 15 (Ice Cream Sandwich) 이상을 설정해야만 지원하며 최고 성능을 얻으려면 minSdkVersion을 21 (Lollipop) 이상으로 설정해야 한다.

HOT SWAP

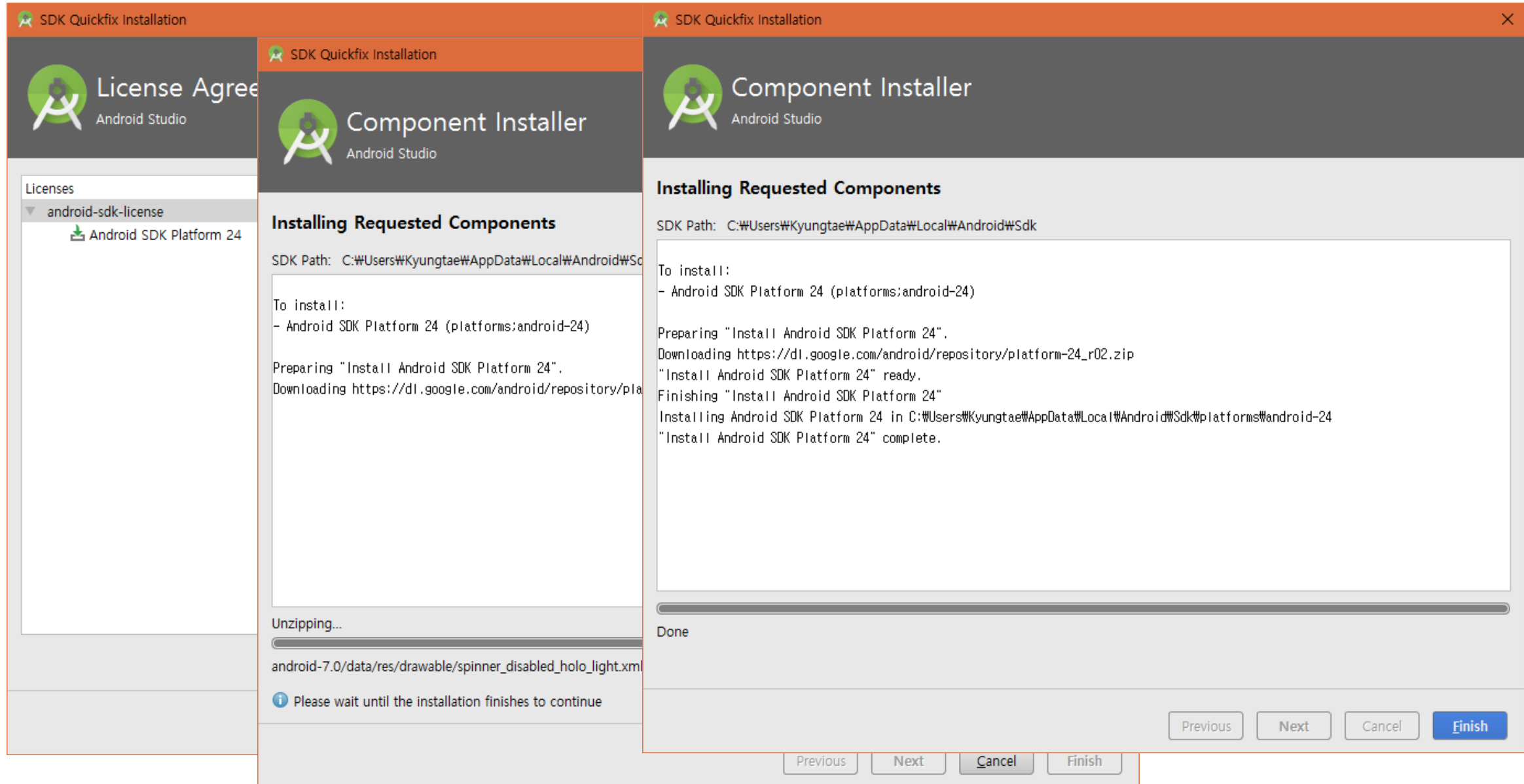
- 가장 빠른 방식
- 이미 존재하는 method의 코드를 수정할 경우 바로 확인 가능
- object 자체를 재초기화하지 않으므로 필요하다면 activity나 app 자체를 다시 실행해야 할 수도 있다.

WARM SWAP

- 기존에 존재하는 resource를 변경하거나 삭제하는 경우
- 빠르게 동작하며, 현재 동작 중인 activity를 재시작해야만 한다.
- 잠시 깜빡거림을 볼 수도 있지만 정상적인 동작이다.

COLD SWAP

- API level 21 이상(Lollipop)
- Structural Code가 변경된 경우
 - annotation
 - instance field
 - static field
 - static method signature
 - instance method signature
 - 상속한 부모 클래스가 변경
 - 구현된 interface list가 변경
 - class의 static initialize가 변경
 - dynamic resource ID를 사용 중인 layout elements가 reorder 되었을 경우



사용자 인터페이스 (User Interface)



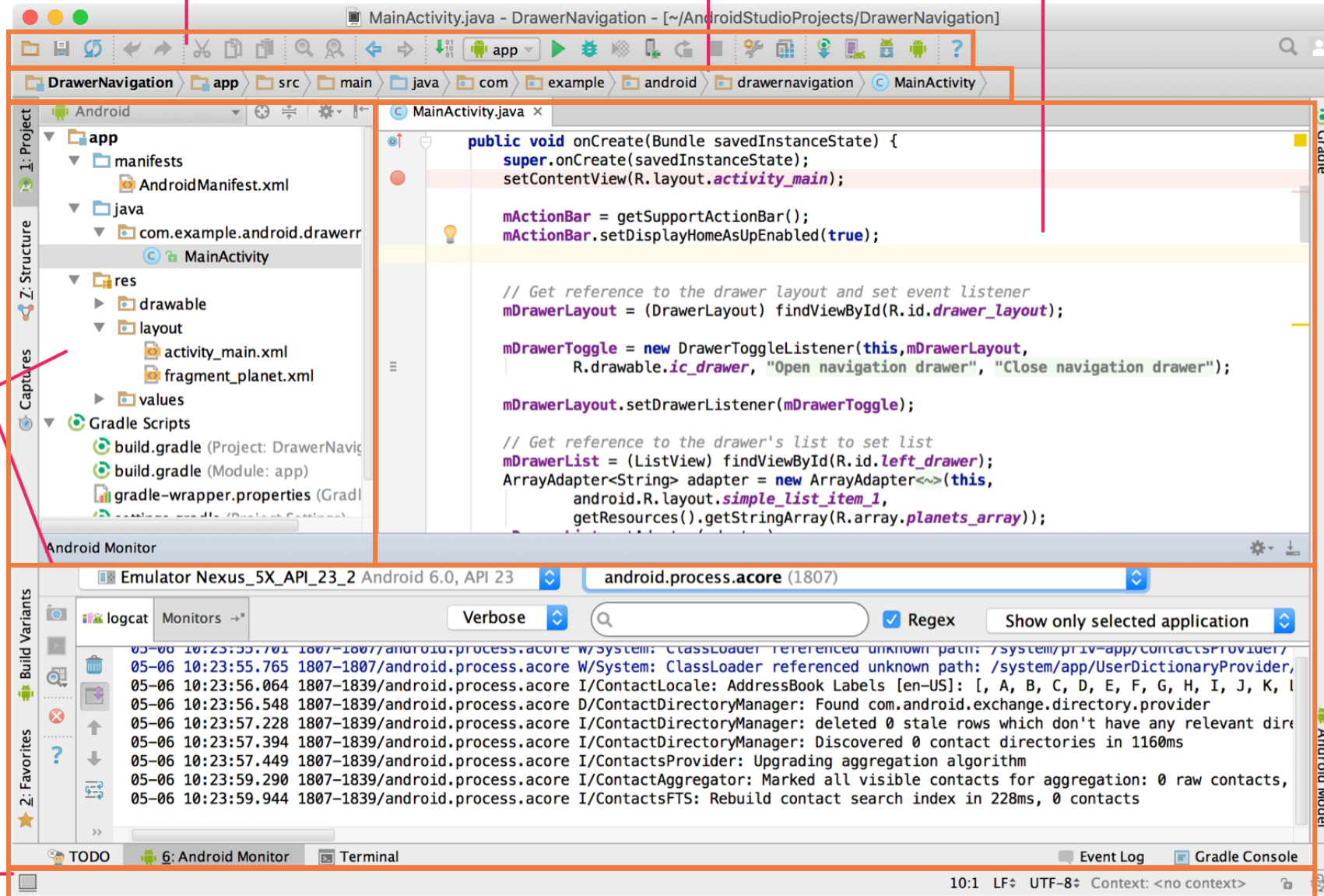
1 툴바

2 탐색모음

3 편집기 창

도구 창

4



상태 표시줄

5

- ① **툴바**를 사용하면 앱 실행, Android 도구 시작과 같은 다양한 작업을 수행할 수 있습니다.
- ② **탐색 모음**을 사용하면 **프로젝트를 탐색하고 편집할 파일을 열 수 있습니다**. 탐색 모음은 **프로젝트** 창에 나타나는 구조를 보다 간략하게 표시합니다.
- ③ **편집기 창**에서는 **코드를 작성**하고 수정할 수 있습니다. **현재의 파일 유형에 따라 편집기가 바뀔 수 있습니다**. 예를 들어, 레이아웃 파일을 볼 때 편집기는 Layout Editor를 표시합니다.
- ④ **도구 창**에서는 **프로젝트 관리, 검색, 버전 제어** 등의 특정 작업에 액세스할 수 있습니다. 창을 확대/축소할 수 있습니다.
- ⑤ **상태 표시줄**은 **프로젝트와 IDE의 상태**를 표시하며, 또한 **경고나 메시지도 표시**합니다.

STEP 2. 파일 편집 – strings.xml

25

- 어플리케이션 라벨과 액티비티 라벨로 사용되는 텍스트 리소스를 관리하는 **strings.xml**에 있는 **app_name** 속성의 데이터를 '안녕'으로 수정한다.

The screenshot illustrates the steps to modify the application name in an Android project:

- Project View:** The 'Project' tab on the left shows the file structure. The 'res' folder is expanded, and 'strings.xml' is selected. A blue box labeled '클릭' (click) points to this file.
- Main Editor:** The 'strings.xml' file is open. It contains the following XML code:

```
1 <resources>
2   <string name="app_name">My Application</string>
3 </resources>
```

A blue arrow points from the text '수정할 부분' (part to be modified) to the value 'My Application' on line 2.
- Secondary Editor:** A secondary editor window shows the modified code:

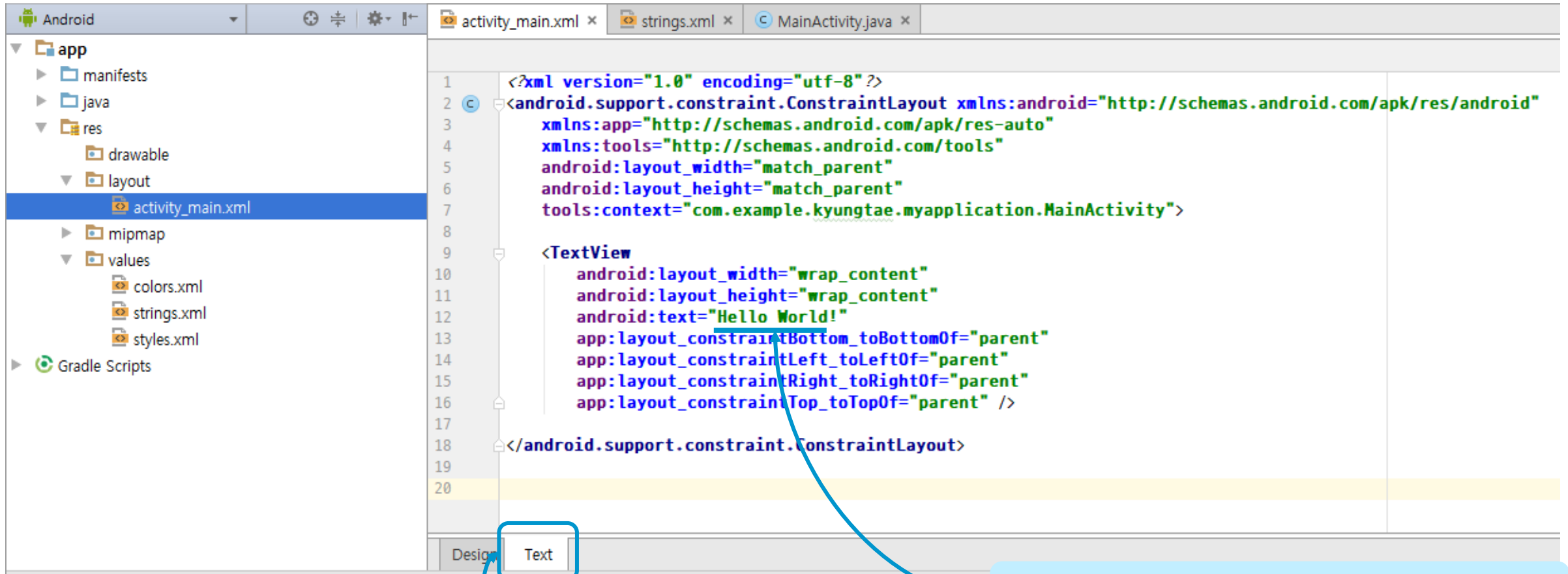
```
1 <resources>
2   <string name="app_name">안녕</string>
3 </resources>
```

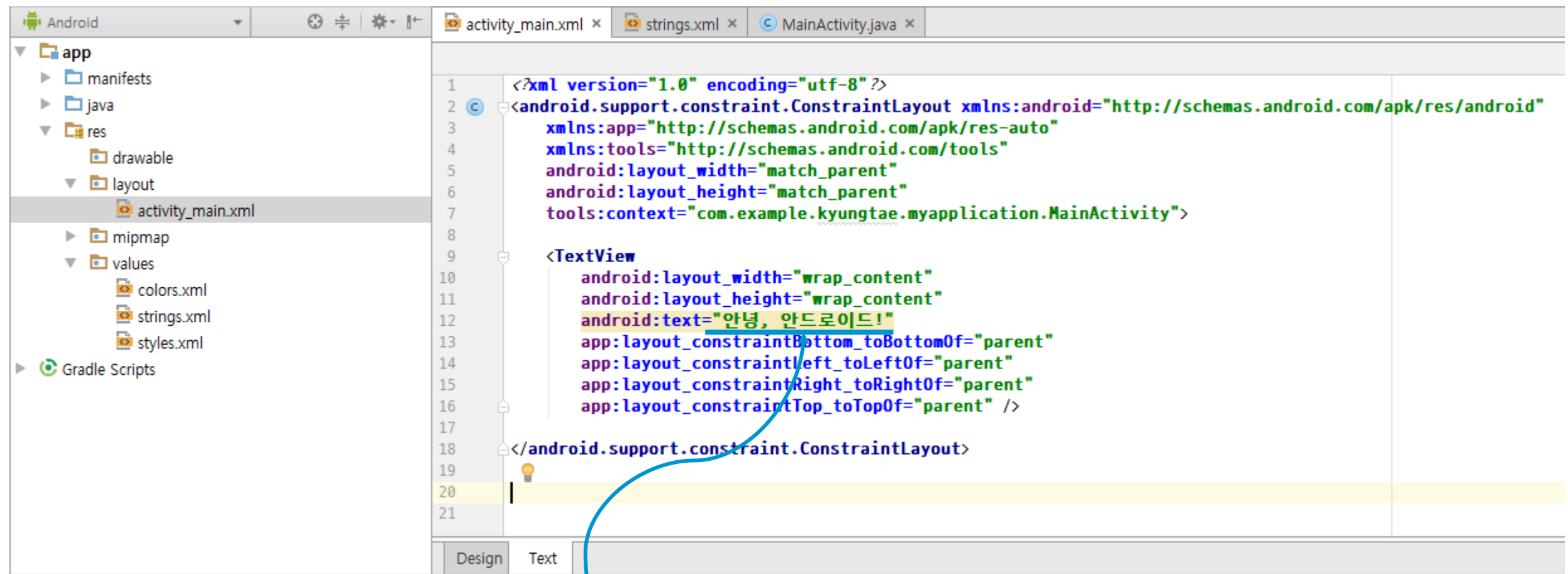
A blue arrow points from the text '‘ Hello Android ’ 를 ‘ 안녕 ’ 으로 수정' (modify 'Hello Android' to '안녕') to the new value '안녕' on line 2.

Continued-activity_main.xml

26

- 액티비티 화면에 나타나는 텍스트는 **activity_main.xml**에 있는 **TextView**의 **text** 속성의 데이터를 ‘**안녕, 안드로이드**’로 수정.





Step 3. 프로젝트 실행(편집 후)

28

- “Run” 메뉴에서 “Run ‘app’” 을 클릭하거나 메뉴 아이콘 “Run ‘app’”을 클릭하면 나타나는 디바이스 선택 창의 목록에서 실행할 디바이스 (AVD 또는 스마트폰)를 선택하고, ‘OK’ 버튼을 클릭한다



AVD Layout Preview에서 한글 깨짐

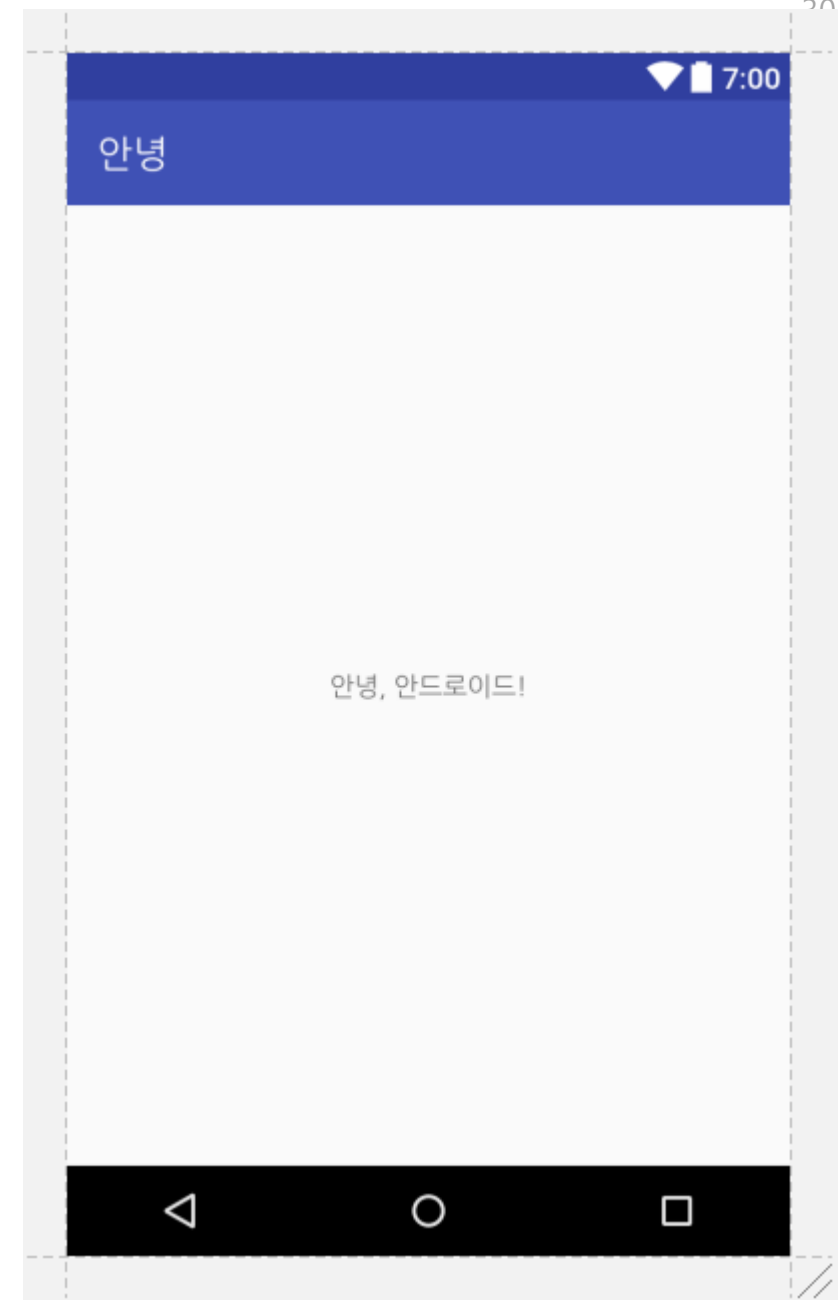


> H0_ROOT (C:) > Program Files > Android > Android Studio > plugins > android > lib > layoutlib > data > fonts

| 이름 | 수정한 날짜 | 유형 | 크기 |
|-----------------------------|------------------|------------|---------|
| AndroidClock.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 5KB |
| CarroisGothicSC-Regular.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 40KB |
| ComingSoon.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 58KB |
| CutiveMono.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 68KB |
| DancingScript-Bold.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 108KB |
| DancingScript-Regular.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 109KB |
| DroidSans.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 303KB |
| DroidSans-Bold.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 303KB |
| DroidSansFallback.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 4,020KB |
| DroidSansMono.ttf | 2017-06-06 오후... | 트루타입 글꼴 파일 | 107KB |
| fonts.xml | 2017-06-06 오후... | XML 문서 | 17KB |
| fontsInSdk.txt | 2017-06-06 오후... | 텍스트 문서 | 4KB |

```
337 <family lang="zh-Hans">
338     <font weight="400" style="normal" index="2">NotoSansCJK-Regular.ttc</font>
339 </family>
340 <!-- TODO: Add Bopo -->
341 <family lang="zh-Hant">
342     <font weight="400" style="normal" index="3">NotoSansCJK-Regular.ttc</font>
343 </family>
344 <family lang="ja">
345     <font weight="400" style="normal" index="0">NotoSansCJK-Regular.ttc</font>
346 </family>
347 <family lang="ko">
348     <font weight="400" style="normal" index="1">NotoSansCJK-Regular.ttc</font>
349 </family>
350 <family lang="und-Zsye">
351     <font weight="400" style="normal">NotoColorEmoji.ttf</font>
352 </family>
353 <family>
354     <font weight="400" style="normal">NotoSansSymbols-Regular-Subsetted2.ttf</font>
355 </family>
356 <family>
357     <font weight="400" style="normal">DroidSansFallback.ttf</font>
358 </family>
```

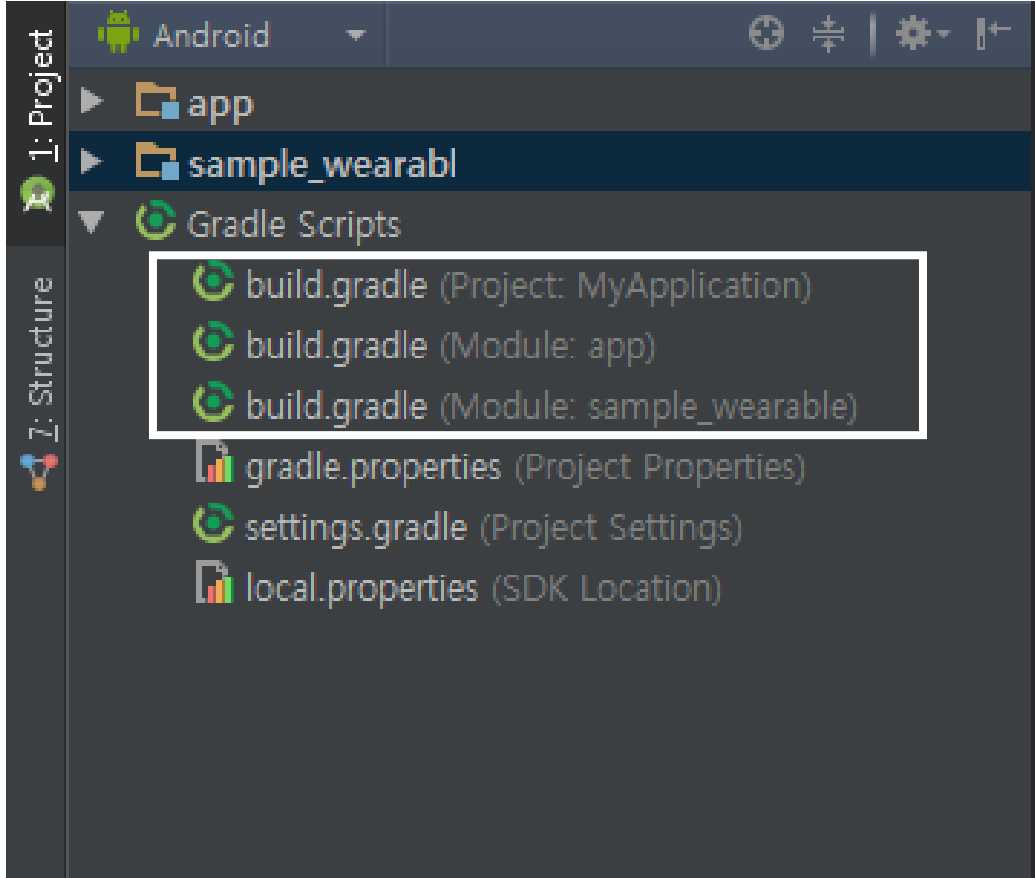
```
347 <family lang="ko">
348     <font weight="400" style="normal" index="1">NanumGothic.ttf</font>
349 </family>
```



3.3 프로젝트 파일 구조

소스 코드 및 리소스(자산)에서 테스트 코드 및 빌드 구성에 이르기까지 앱에 대한 작업 영역을 정의하는 모든 항목이 포함되고 최소 1개 이상의 모듈을 가진다.

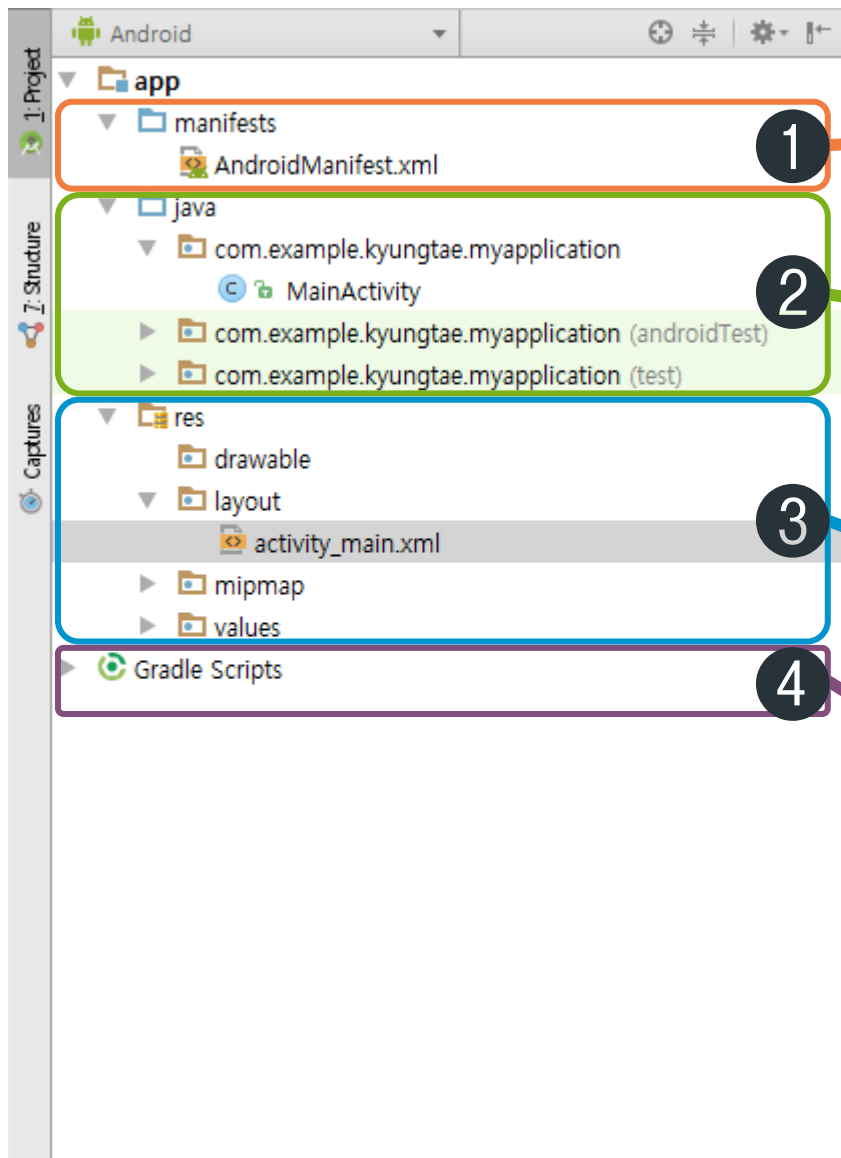
- 독립되어 있는 하나의 소프트웨어 또는 하드웨어 단위를 지칭
- 스마트폰과 스마트시계와 통신을 하는 앱을 만든다고 예를 들었을 때의 화면
- app 모듈은 스마트폰 디바이스에 올라갈 수 있는 앱
- sample_wearable 모듈은 스마트와치에 올라갈 앱



출처:

<http://banaworldco.tistory.com/entry/%EA%B2%81%EC%9F%81%EC%9D%B4%EB%93%A4%EC%9D%84-%EC%9C%84%ED%95%9C-%EC%9D%B4%ED%81%B4%EB%A6%BD%EC%8A%A4%EC%97%90%EC%84%9C-Android-Studio%EB%A1%9C-%EB%84%98%EC%96%B4%EA%B0%80%EA%B8%B0>

프로젝트 모듈의 구조와 기능



모듈에 관한 기본 정보 관리
(Application 라벨, theme 등)

모듈을 위한 자바 소스 파일 관리
(화면 출력, 사용자의 이벤트 처리 등)

모듈에 사용되는 리소스 파일 관리
(화면 레이아웃 설계, 텍스트와 이미지 리소스 등)

실행 가능 파일을 만들기 위한 기본
정보
(초보자는 거의 수정 필요 없음)

| 모듈분류 | 폴더(위치) | 소스 파일 | 기능 |
|----------------|--|----------------------|--|
| ① manifests | | AndroidManifest.xml | <p>어플리케이션에 관한 정보들이 설정되어 있다. 어플리케이션 라벨, 어플리케이션 아이콘, 액티비티 라벨, 처음 실행 될 액티비티명(자바 클래스)을 포함한다.</p> <p>또한, 어플리케이션이 실행될 때 필요한 권한 등이 기술된다. (인터넷 접속 허용 설정 등)</p> |
| ② java | com.example.kyungtae.myapplication | MainActivity.java | <p>어플리케이션을 구성하는 액티비티를 구현하는 자바 클래스 (어플리케이션 실행 시에 처음 실행되는 자바 클래스로 사용됨)</p> |
| | com.example.kyungtae.myapplication (androidTest) | ApplicationTest.java | <p>실행 테스트를 위한 자바 클래스(삭제해도 무관함)</p> |

| 소스 | 폴더(위치) | 소스 파일 | 기능 |
|----------|----------|--------------------------|---|
| ③ res | drawable | | 화면에 그려지는 그래픽을 위한 Drawable resource를 저장 (png, jpg, gif 이미지 파일과 XML 파일) |
| | layout | activity_main.xml | 액티비티 실행 시에 화면에 나타나는 레이아웃을 설계함. strings.xml에 정의된 텍스트 리소스나 drawable 폴더에 있는 이미 리소스들을 출력할 위치에 배치함 |
| | mipmap | ic_launcher.png(hdpi) | 홈 화면의 아이콘(72 x 72 픽셀) , 240dpi 화면밀도에서 사용 |
| | | ic_launcher.png(mdpi) | 홈 화면의 아이콘(48 x 48 픽셀) , 160dpi 화면밀도에서 사용 |
| | | ic_launcher.png(xhdpi) | 홈 화면의 아이콘(96 x 96 픽셀) , 320dpi 화면밀도에서 사용 |
| | | ic_launcher.png(xxhdpi) | 홈 화면의 아이콘(144 x 144 픽셀) , 480dpi 화면밀도에서 사용 |
| | | ic_launcher.png(xxxhdpi) | 홈 화면의 아이콘(192 x 192 픽셀) , 640dpi 화면밀도에서 사용 |
| | values | colors.xml | 화면 테마에 사용되는 색상 정의 |
| | | dimens.xml | 화면 구성 자원의 크기(여백, 글자 크기 등)를 정의 |
| | | strings.xml | 어플리케이션 라벨과 액티비티 라벨을 포함하며, 액티비티 화면에 출력될 여러 텍스트 리소스를 정의 |
| | | styles.xml | 화면의 스타일(화면 테마, 텍스트 폰트 등)을 정의 |

| 소스 | 폴더(위치) | 소스 파일 | 기능 |
|------------------------|--------|-----------------------------|--|
| ④ Gradle Scripts | | build.gradle(project:프로젝트명) | 모듈 전체에 적용되는 빌드 정보 |
| | | build.gradle(Module app) | app 모듈에 적용되는 빌드 정보 |
| | | proguard-rule.pro | 사용되지 않는 코드를 지워 사이즈를 줄여 최적화하거나 코드를 난독화하여 안전하게 보호 하기 위해 사용 |
| | | gradle.properties | 빌드 관련 속성 관리 |
| | | settings.gradle | 빌드 관련 환경설정, 빌드 될 하위 모듈 정보 포함 |
| | | local.properties | 빌드 진행 시, 필요한 환경변수 정보를 저장하는 파일 임. 안드로이드 SDK의 경로가 이곳에 저장됨. |



3.4 앱의 실행 원리

- 자바와 xml 파일로 구성되는 기본 파일들은 서로 연관성을 가지고 있다.



| 파일 | 내용 |
|----------------------------|---|
| activity_main.xml | 화면을 설계하며, <code>dimens.xml</code> 파일에 정의된 여백의 크기를 설정 |
| MainActivity.java | 액티비티를 생성하고 <code>activity_main.xml</code> 에 정의된 레이아웃을 출력 |
| AndroidManifest.xml | 어플리케이션 아이콘과 라벨, 화면 스타일 등을 지정하고, 액티비티를 구성하는 자바 클래스를 지정한다. |
| Gradle Scripts | 컴파일/빌더 정보를 담고 있으며 실행파일을 만드는 역할 |

화면 구성 자원 크기

`dimen``activity_horizontal_margin: 16dp`

dimens.xml (values)

아이콘 이미지



ic_launcher.png (mipmap)

텍스트 자원

`string``app_name: 안녕`

strings.xml (values)

화면 테마

`style``AppTheme:``Theme.AppCompat.Light.DarkActionBar``item``colorPrimary:``@color/colorPrimary`

styles.xml (values)

화면 테마 구성 색상

`color``colorPrimary: #3F51B5`

colors.xml (values)

화면 레이아웃

`RelativeLayout``android:paddingLeft:``@dimen/activity_horizontal_margin``TextView``text: 안녕, 안드로이드`

activity_main.xml (layout)

화면 출력 소스

액티비티 제어

`onCreate``super.onCreate()``setContentView(R.layout.activity_main)`

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`

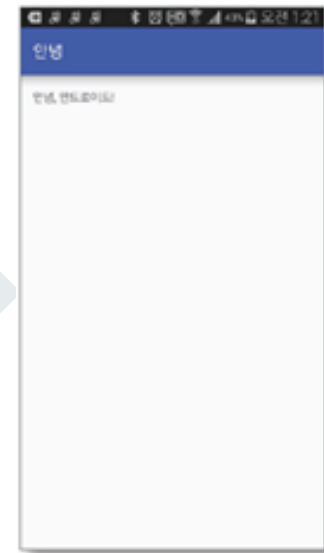
AndroidManifest.xml (manifest)

앱이 실행될 때 처음 불러짐

액티비티 생성

화면 출력

컴파일/빌더



AVD/스마트폰

컴파일/빌더 정보

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

(Gradle Scripts)

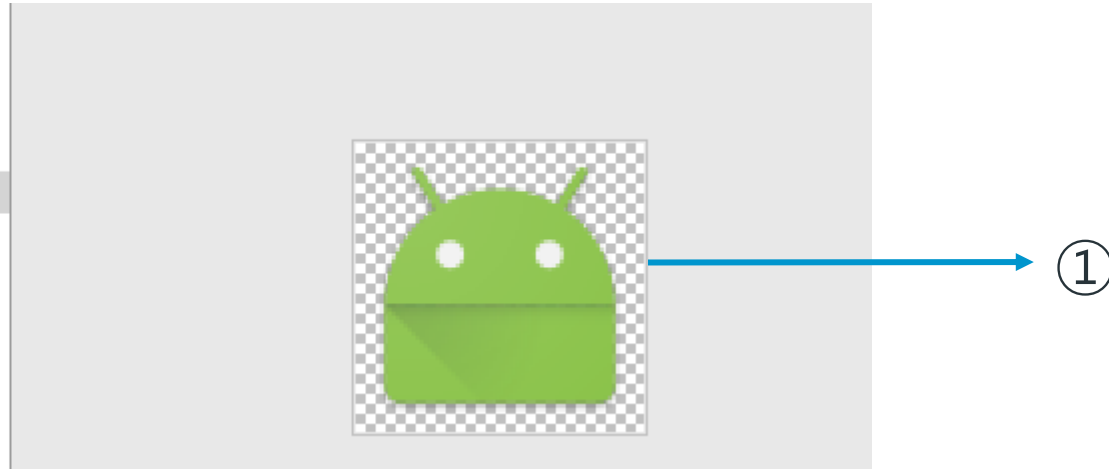
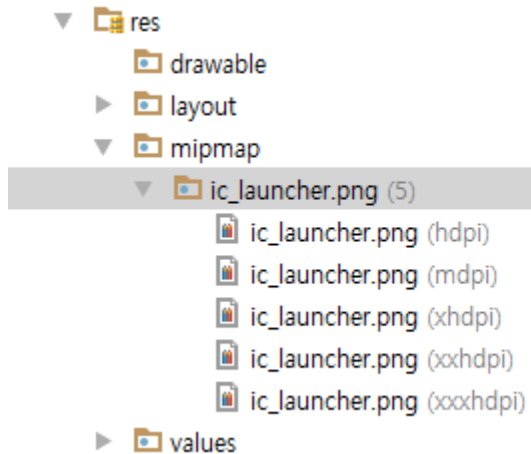
앱 라벨



3.5 프로젝트 소스 간의 연관성

- 프로젝트 모듈을 구성하는 주요 파일의 소소내의 연관 부분 간 흐름을 살펴 보자. 구체적인 문법은 다음 4장부터 설명한다.

- res/mipmap 폴더



- res/values 폴더의 colors.xml

```
activity_main.xml x MainActivity.java x colors.xml x strings.xml x (화면 테마에 사용될 색)
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3   <color name="colorPrimary">#3F51B5</color>
4   <color name="colorPrimaryDark">#303F9F</color>
5   <color name="colorAccent">#FF4081</color>
6 </resources>
```

②

- res/values 폴더의 styles.xml

```
activity_main.xml x MainActivity.java x colors.xml x styles.xml x strings.xml x (화면 테마)
Edit all themes in the project in the theme editor.
1 <resources>
2
3   <!-- Base application theme. -->
4   <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
5     <!-- Customize your theme here. -->
6     <item name="colorPrimary">@color/colorPrimary</item>
7     <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
8     <item name="colorAccent">@color/colorAccent</item>
9   </style>
10
11 </resources>
```

AppTheme 속성값을 '어두운 색의 액션 바를 가진 밝은 테마'로 할당

상태 바(status bar)의 색

앱 바(application bar)의 색

체크박스, 텍스트 입력 필드와 같은 UI 컨트롤의 색

화면 테마의 추가적인 정보

• res/values 폴더의 dimens.xml과 strings.xml

activity_main.xml x MainActivity.java x dimens.xml x (화면을 구성하는 자원들의 크기)

```
1 <resources>
2   <!-- Default screen margins, per the Android Design guidelines. -->
3   <dimen name="activity_horizontal_margin">16dp</dimen>
4   <dimen name="activity_vertical_margin">16dp</dimen>
5 </resources>
```

activity_main.xml x MainActivity.java x dimens.xml x strings.xml x (텍스트 리소스)

③

Edit translations for all locales in the translations editor.

```
1 <resources>
2   <string name="app_name">안녕</string>
3 </resources>
```

app_name을 '안녕'으로 수정

• res/layout 폴더의 activity_main.xml

④

activity_main.xml x strings.xml x MainActivity.java x (화면 레이아웃)

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:app="http://schemas.android.com/apk/res-auto"
4   xmlns:tools="http://schemas.android.com/tools"
5   android:layout_width="match_parent"
6   android:layout_height="match_parent"
7   tools:context="com.example.kyungtae.myapplication.MainActivity">
8
9   <TextView
10    android:layout_width="wrap_content"
11    android:layout_height="wrap_content"
12    android:text="안녕, 안드로이드!"
13    app:layout_constraintBottom_toBottomOf="parent"
14    app:layout_constraintLeft_toLeftOf="parent"
15    app:layout_constraintRight_toRightOf="parent"
16    app:layout_constraintTop_toTopOf="parent" />
17
18 </android.support.constraint.ConstraintLayout>
```

화면의 하단,좌측,우측,상단 패딩의 크기

- java/com.example.Kyungtae.myapplication 패키지

⑤

activity_main.xml × MainActivity.java × dimens.xml × strings.xml × (액티비티 제어)

```
1 package com.example.kyungtae.myapplication;
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 }
```

MainActivity 자바 클래스가 호출될 때 처음 실행되는 메소드

액티비티 생성

activity_main.xml에서 정의된 화면 레이아웃을 액티비티에 출력

④

• Manifests 폴더

(어플리케이션 기본 정보)

```
activity_main.xml x AndroidManifest.xml x strings.xml x MainActivity.java x
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.example.kyungtae.myapplication">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="안녕"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportRtl="true"
11        android:theme="@style/AppTheme">
12        <activity android:name=".MainActivity">
13            <intent-filter>
14                <action android:name="android.intent.action.MAIN" />
15
16                <category android:name="android.intent.category.LAUNCHER" />
17            </intent-filter>
18        </activity>
19    </application>
20
21 </manifest>
22
```

① 어플리케이션 아이콘

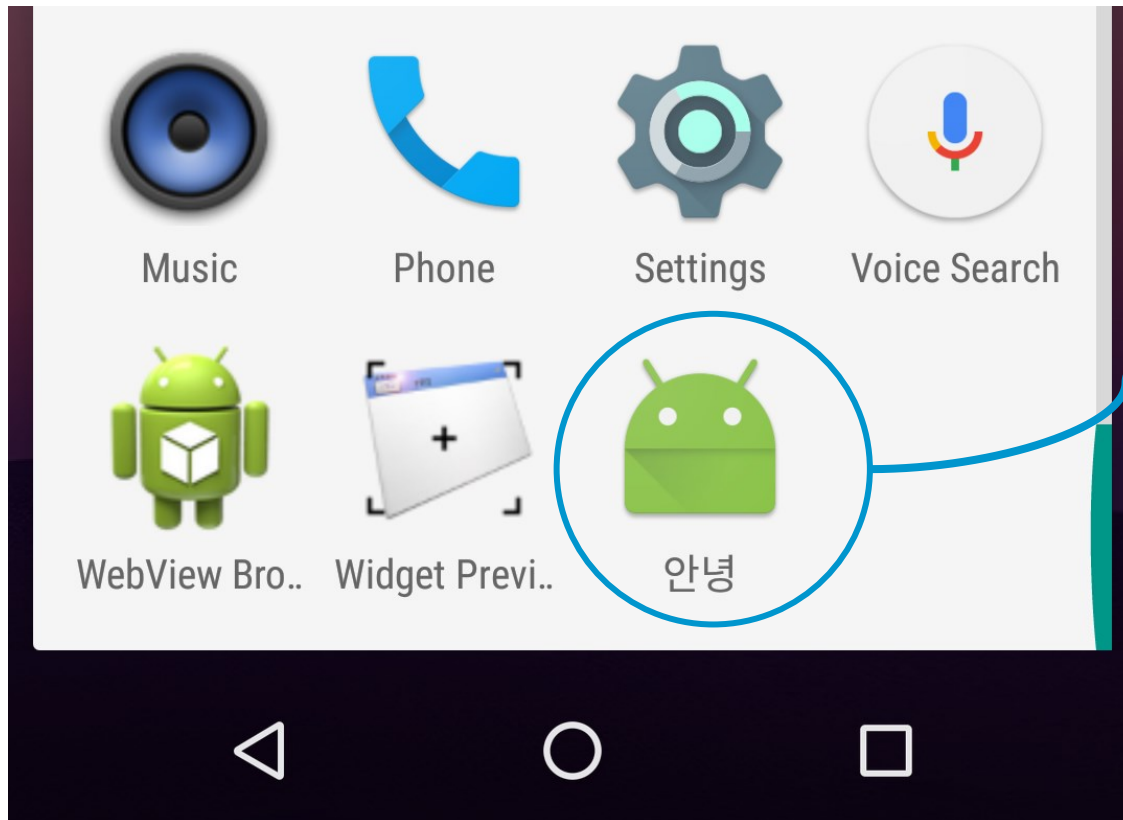
③ 어플리케이션 라벨

② 화면 테마

⑤ 어플리케이션을 구성하는
액티비티 자바 클래스로
앱 실행시 처음 실행됨

3.6 어플리케이션의 아이콘 변경

- 어플리케이션 아이콘을 바꾸고 싶으면, **mipmap** 폴더에 **아이콘 이미지**를 저장하고, **AndroidManifest.xml**에 **아이콘 파일 이름**을 수정하면 된다.

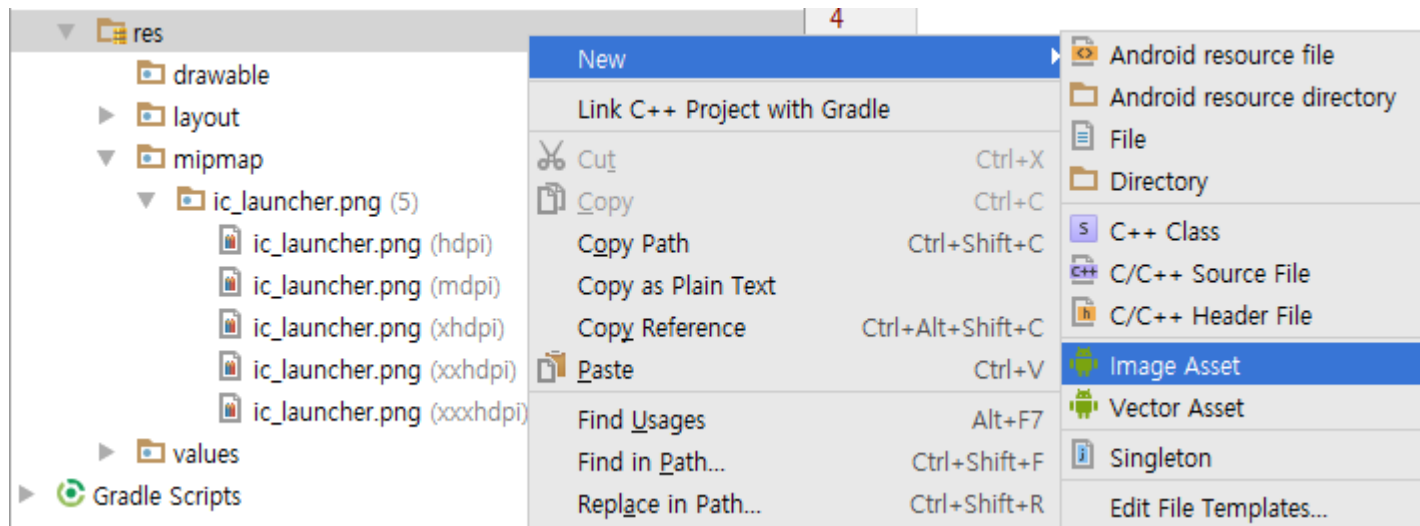


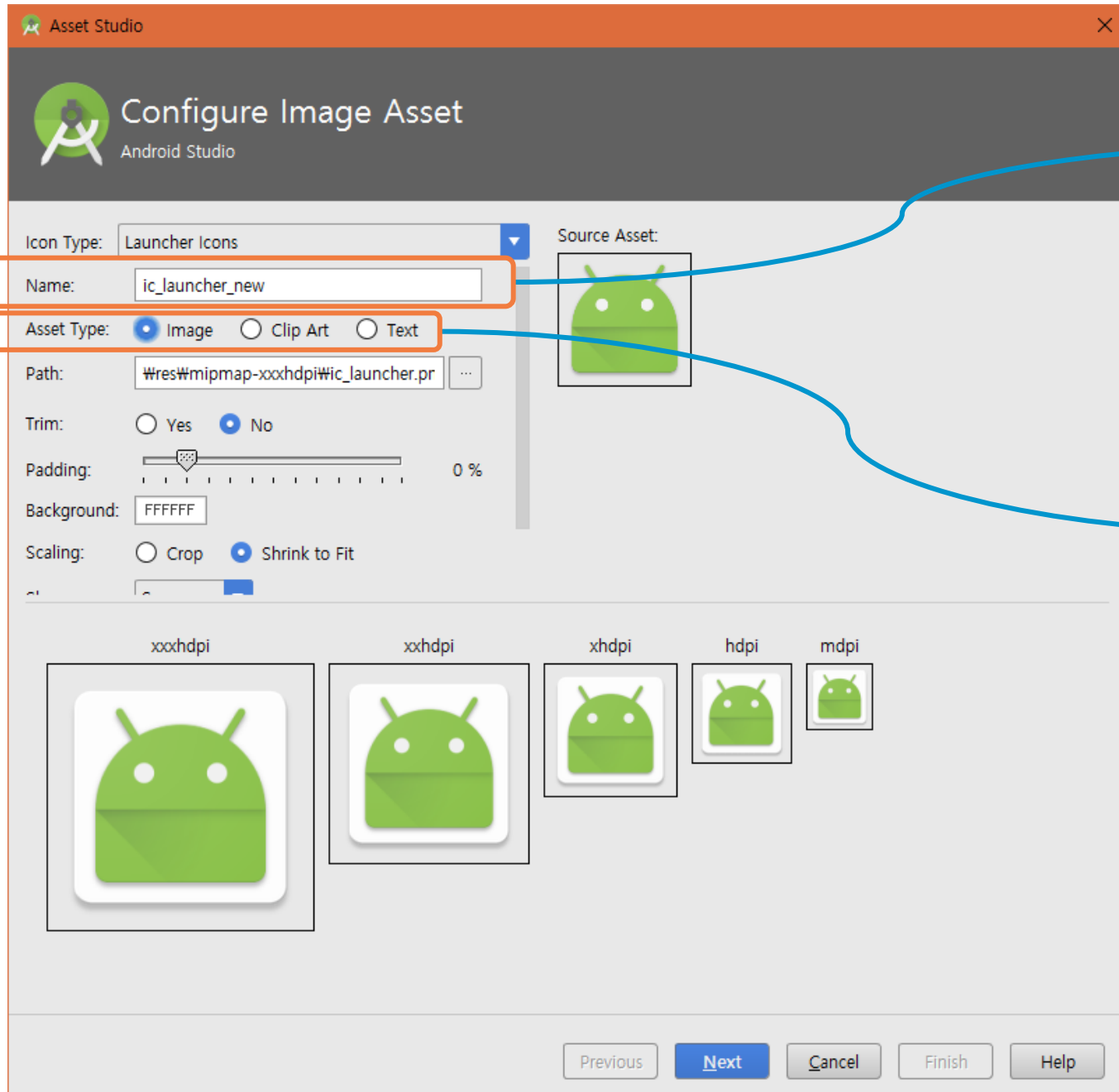
아이콘을 변경



Step 1. 이미지 추가

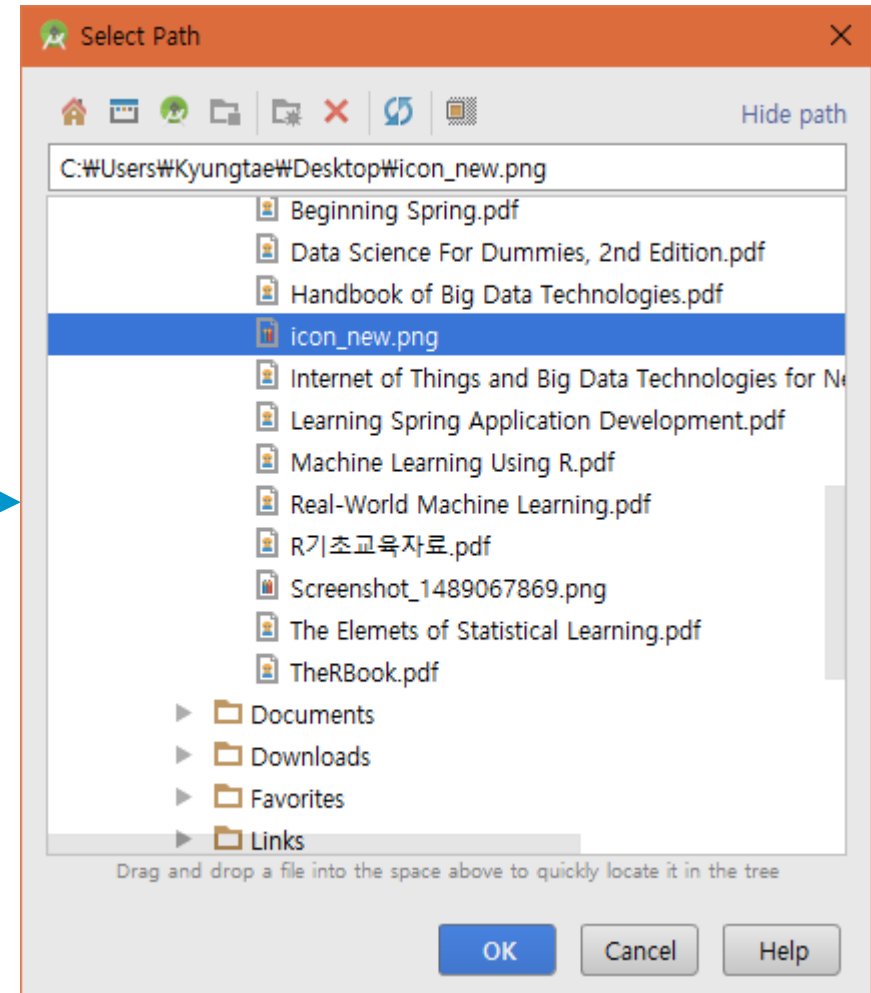
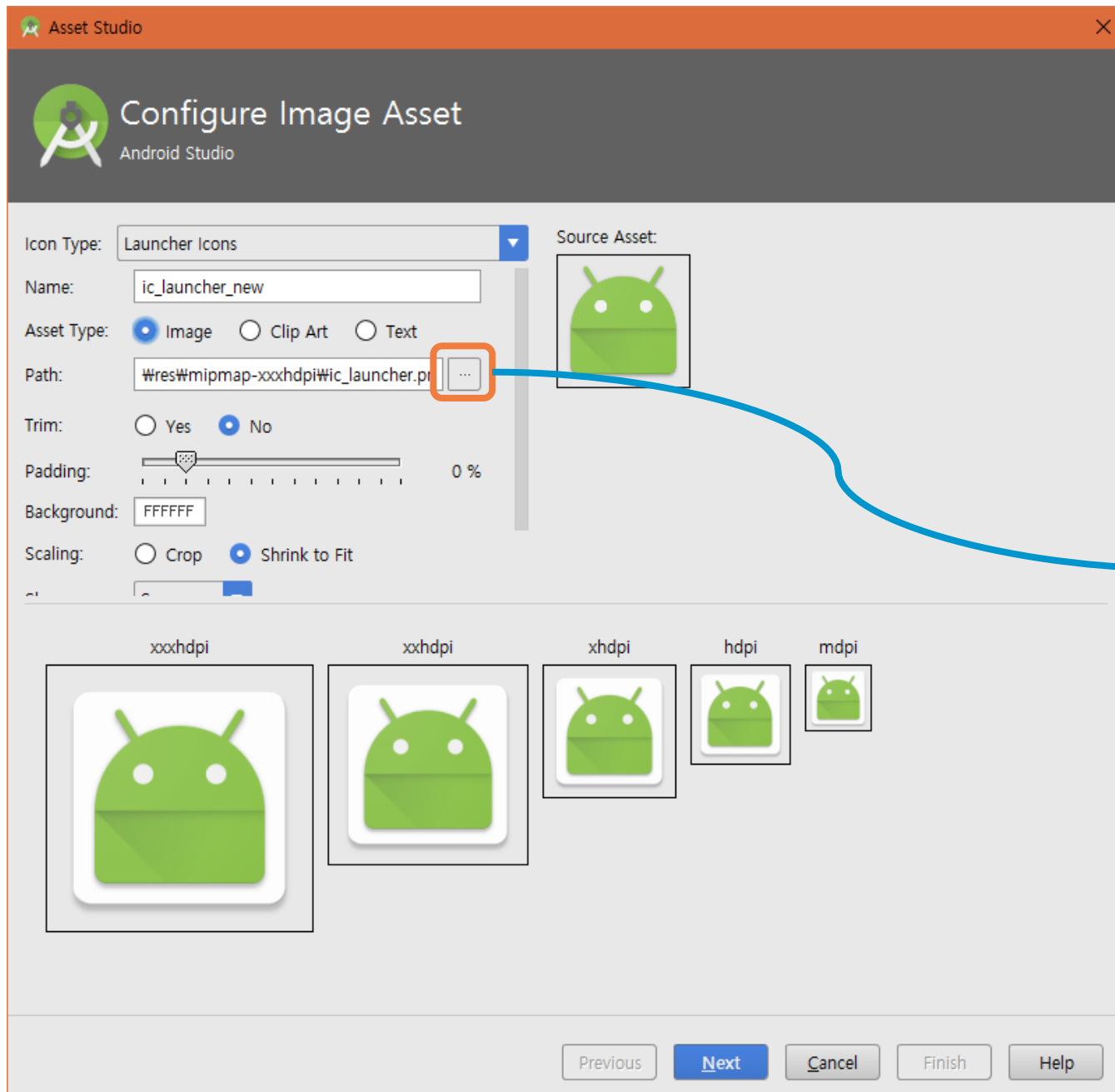
- res폴더를 선택한 후 팝업 메뉴 New → Image Asset 선택

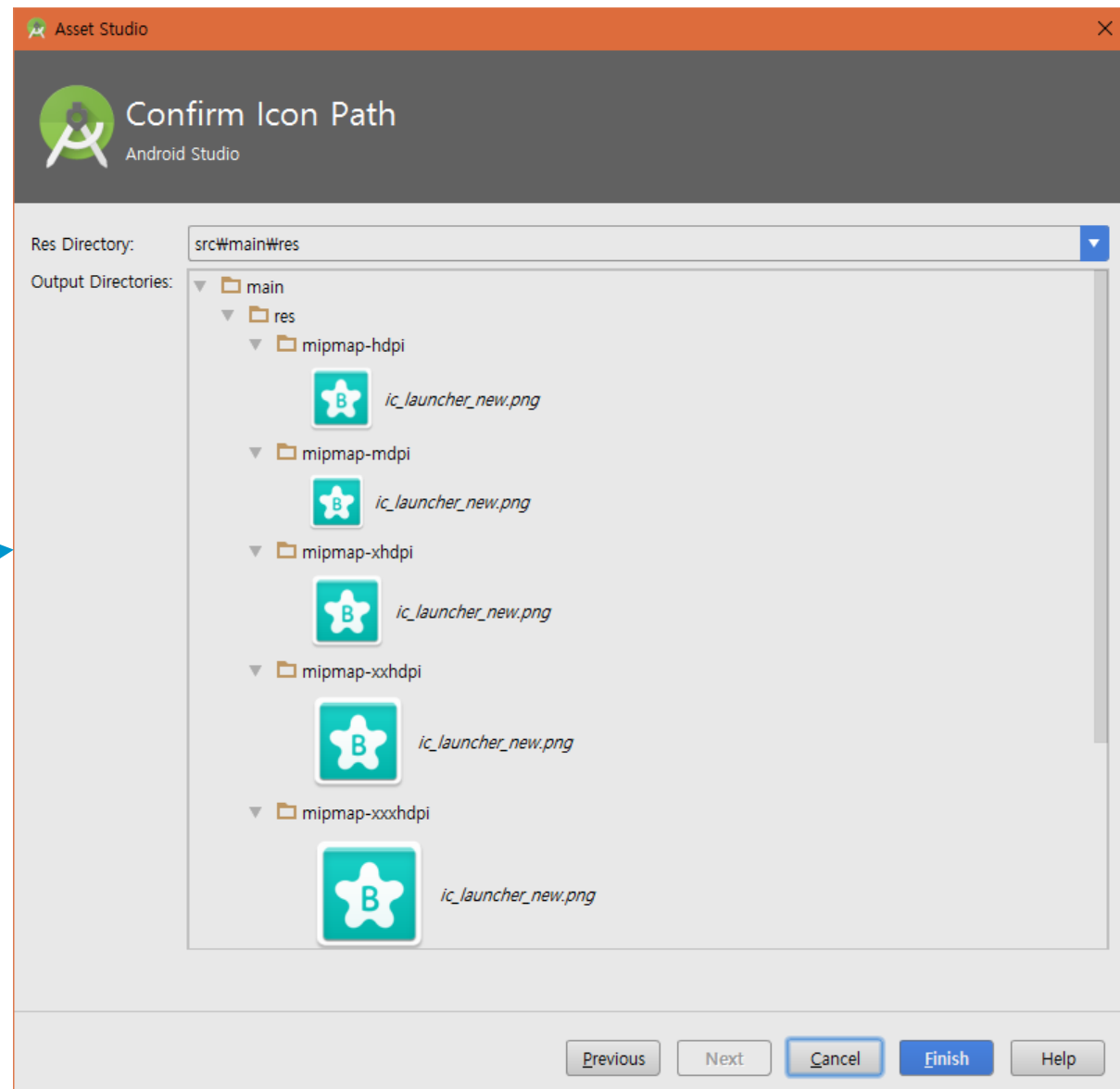
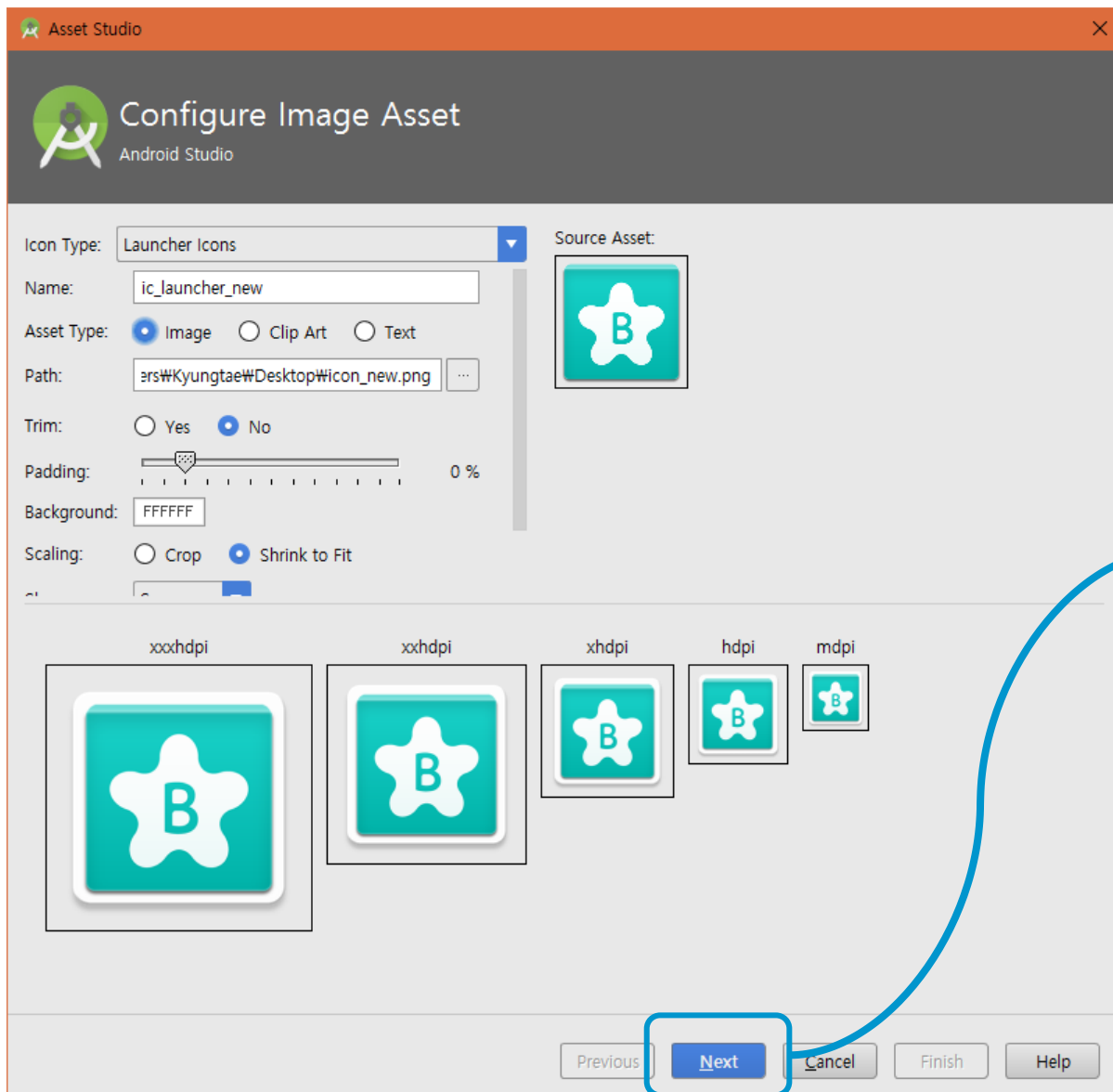


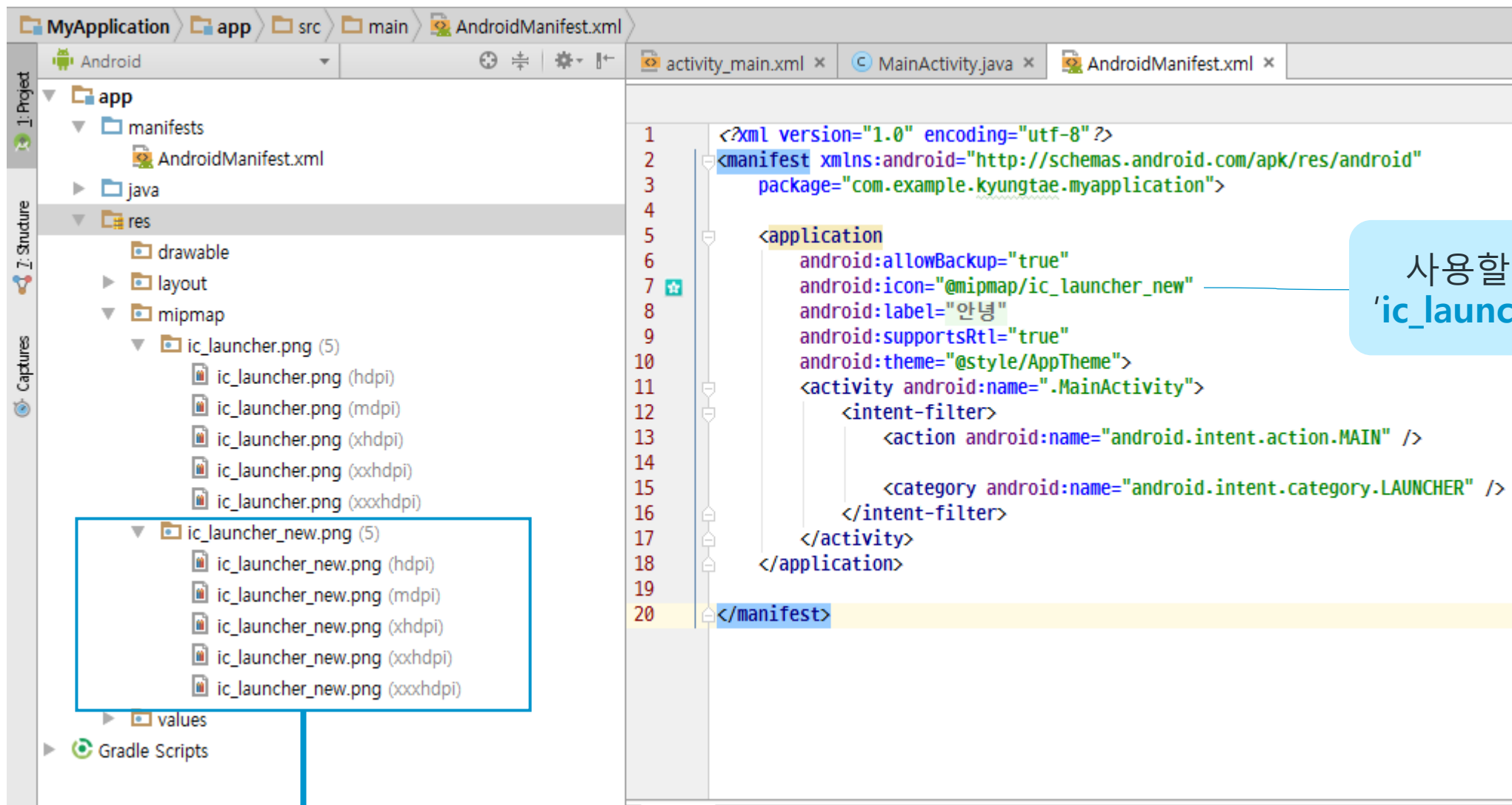


'ic_launcher_new' 로
파일 이름 변경

Asset Type을 'Image' 로
변경

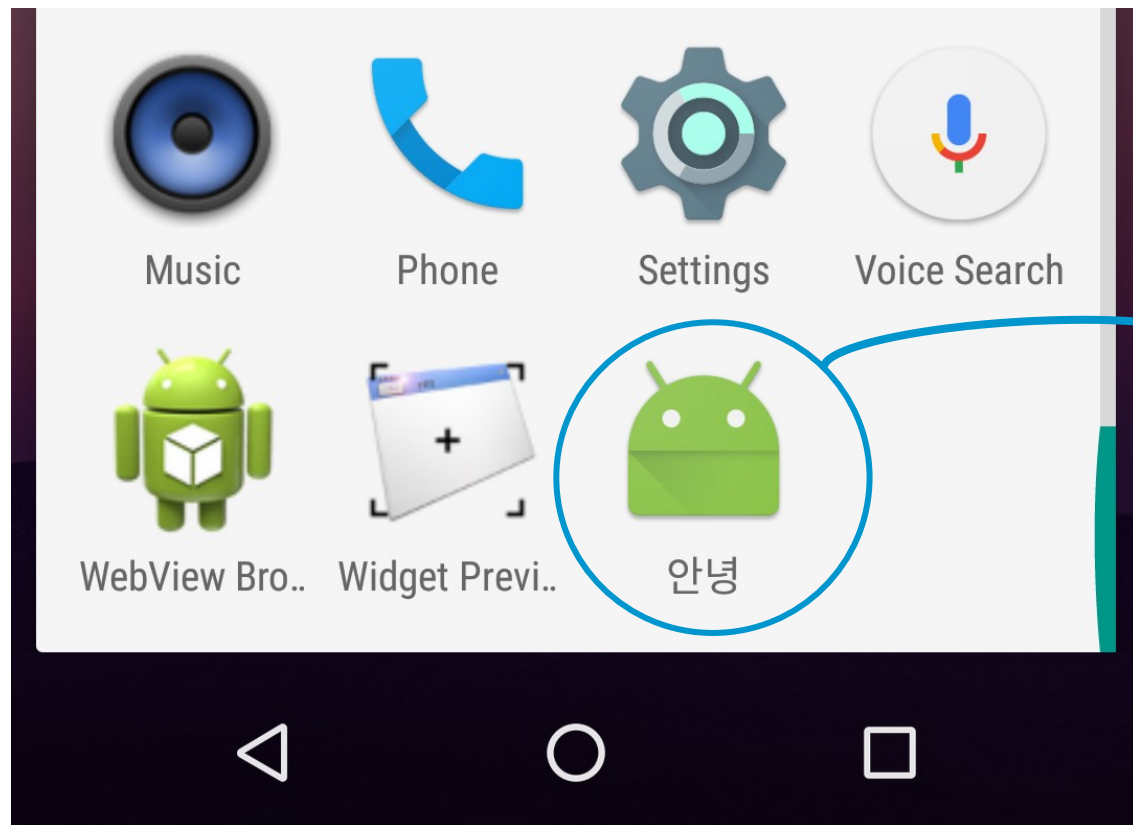




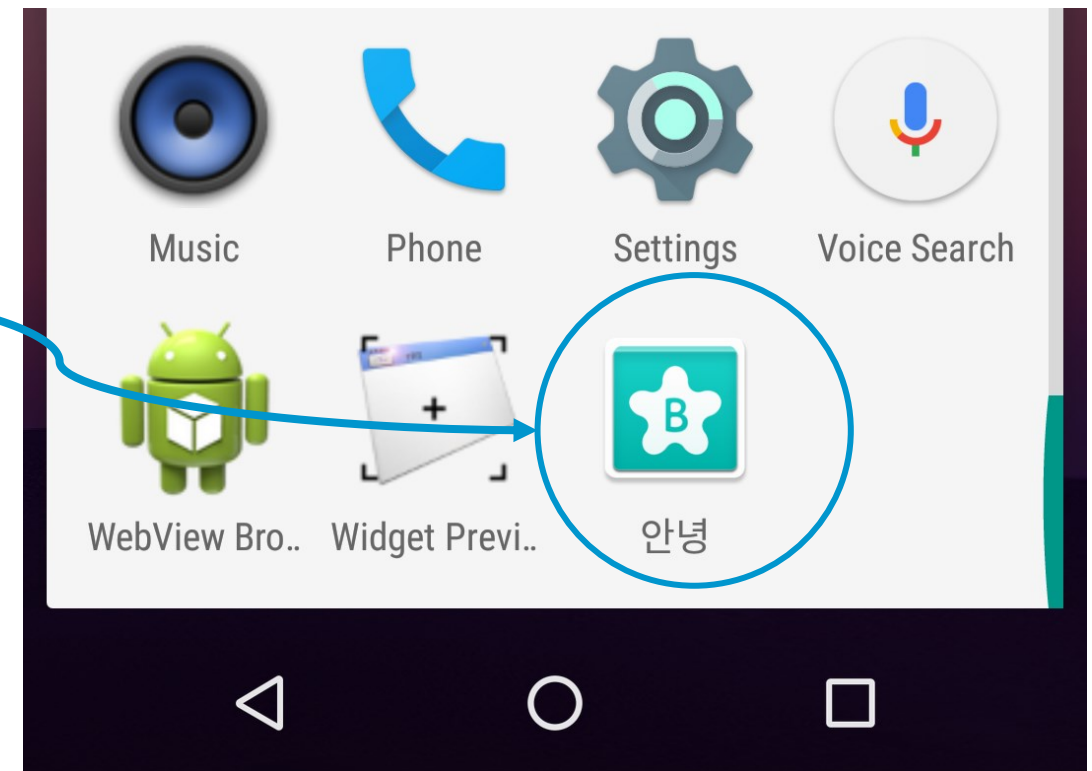


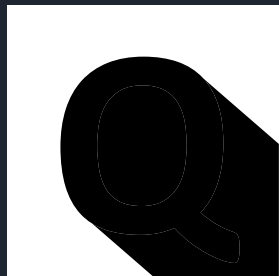
새롭게 추가된
아이콘 이미지

사용할 아이콘 이름을
'ic_launcher_new'로 변경



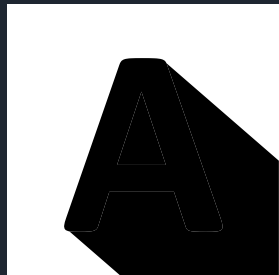
아이콘이 변경됨





uestion

&



nswer

