

<자바 기말 레포트 - 안드로이드>

12171588 김민석



목차

1. 기본 개념

1-1 정의

1-2 역사

1-3 특징 및 장점

2. 개발환경 구축

2-1 안드로이드 스튜디오 설치

2-2 프로젝트 생성

3. 프로젝트 실행

3-1 실제 기기로 실행

3-2 가상 디바이스로 실행

4. 안드로이드 스튜디오의 구조 및 동작 방식

4-1 프로젝트 구조

4-1-1 필수 설정 파일

4-1-2 java 폴더

4-1-3 res 폴더

5. 추억을 되새겨주는 갤러리(간단한 어플)

5-1 apk 추출

5-2 결과물 공유(Github)

1. 기본 개념

1-1 정의

안드로이드는 스마트폰 모바일 운영체제 중 하나입니다. 스마트폰의 어플리케이션이나, 미들웨어, 사용자 인터페이스 등을 통합해서 관리하고 제공하는 소프트웨어 플랫폼으로 볼 수 있습니다.

여러 스마트폰 제조사는 서로 사용하는 운영체제가 다르며, 그에 따라 구성도 다릅니다. 대표적으로 국내 기업인 삼성이나 LG 등은 안드로이드 운영체제를 선택했지만, 애플의 아이폰은 IOS 운영체제를 채택하여 자체적으로 개발하고 있습니다.

1-2 역사

2005 년 7 월, 구글이 앤디 루빈의 안드로이드 사를 인수한 것이 안드로이드의 시작입니다. 이 때 당시만 해도 작은 소프트웨어 회사였던 안드로이드 사는 운영체제 개발을 통해 규모가 커집니다. 이후 구글은 2007 년 11 월에 안드로이드 플랫폼을 무료로 공개한다고 발표하였으나, 그 해 초창기에 스티븐 잡스가 발표한 아이폰에 대한 반응이 너무 좋아서 위협을 느꼈습니다. 이에 따라 구글은 삼성, 노키아, 모토로라 등 애플을 제외한 여러사와 함께 모바일기기의 공개 표준을 개발하기 위한 OHA(Open Handset Alliance)를 결성하게 되었습니다. 이 OHA 에서 리눅스 커널 2.6 을 바탕으로 첫 모바일 플랫폼인 안드로이드를 발표했습니다. 구글이 이후에 아파치

라이선스로 소스를 무료공개 함으로써 안드로이드는 개인 사용자와 많은 회사의 협력을 받아 현재까지 독보적인 모바일 운영체제로 자리잡고 있습니다.

1-3 특징 및 장점

안드로이드는 JAVA 언어를 사용합니다. 안드로이드 개발사들은 대부분 이클립스나 안드로이드 스튜디오를 사용하는데, 이들은 모두 JAVA 를 기반으로 구동됩니다. 안드로이드는 JAVA 라는 개발자에게 친숙한 언어를 사용함으로써 빠른 성장의 발판이 되었습니다.

또한 개방형 플랫폼이라는 장점이 있습니다. 소스 코드를 모두 공개함으로써 누구라도 이를 통해 소프트웨어를 만들고 판매할 수 있고, 응용도 가능하기 때문에 풍부한 서비스가 가능합니다.

그리고 자바기반의 특성 상 이식성이 뛰어납니다. 이에 따라 스마트폰 외에도 스마트워치, TV 등 여러 통신기기에 접목되어 사용될 가능성이 높습니다.

또한 연동이 쉽습니다. 어플리케이션끼리 만든 기능을 쉽게 공유하고 사용할 수 있도록 제작되어 있어서 개발자 입장에서 편리하고 빠른 개발을 할 수 있습니다.

2. 개발환경 구축

안드로이드 앱을 제작하려면 JDK와 Android Studio를 설치해야 합니다. JDK는 기존에 설치했던

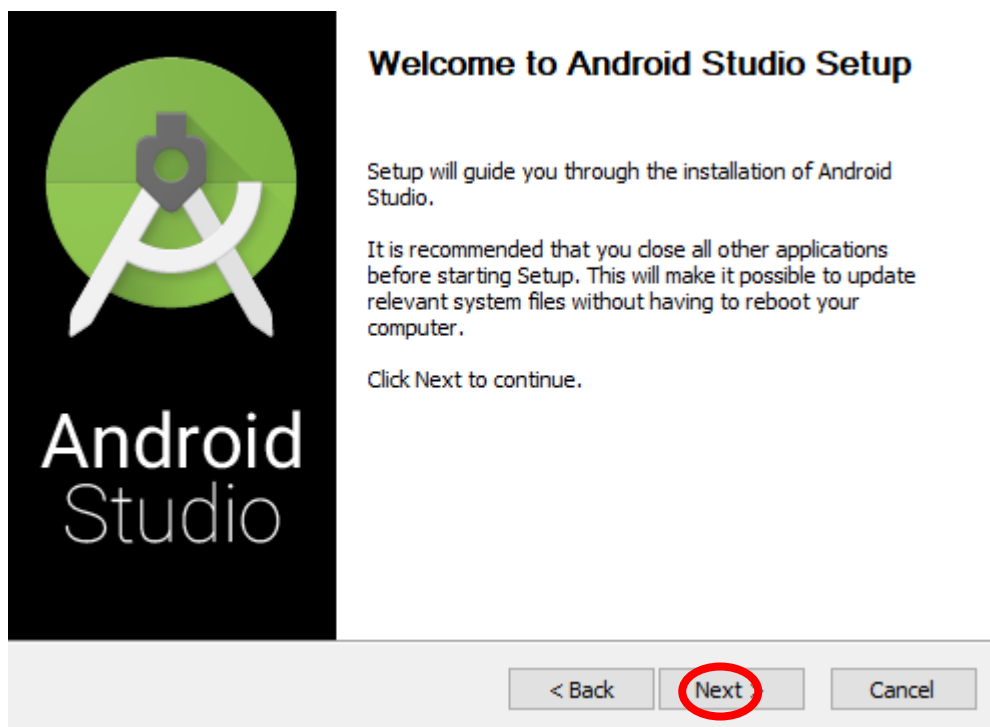
JDK가 있으므로, 안드로이드 스튜디오 설치를 살펴보겠습니다.

2-1 안드로이드 스튜디오 설치

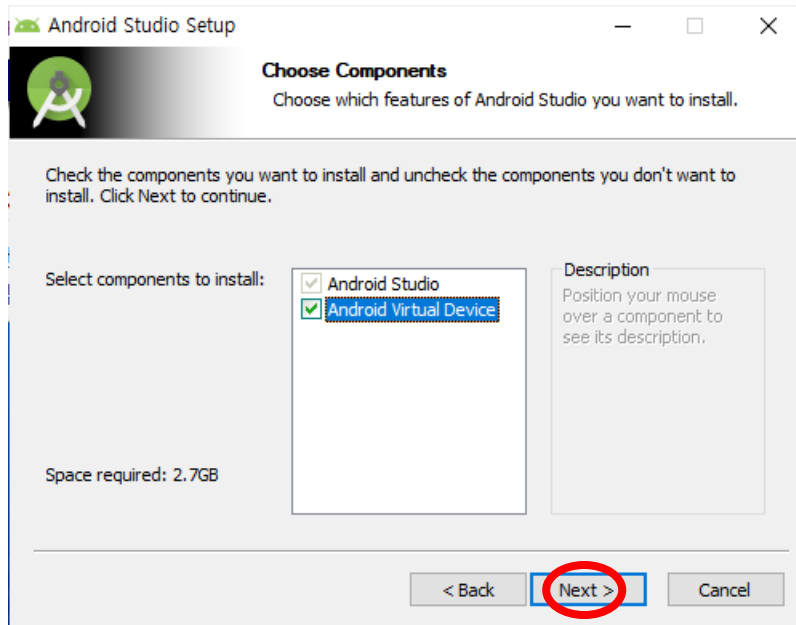
-안드로이드 스튜디오 다운로드

<https://developer.android.com/studio?hl=ko>

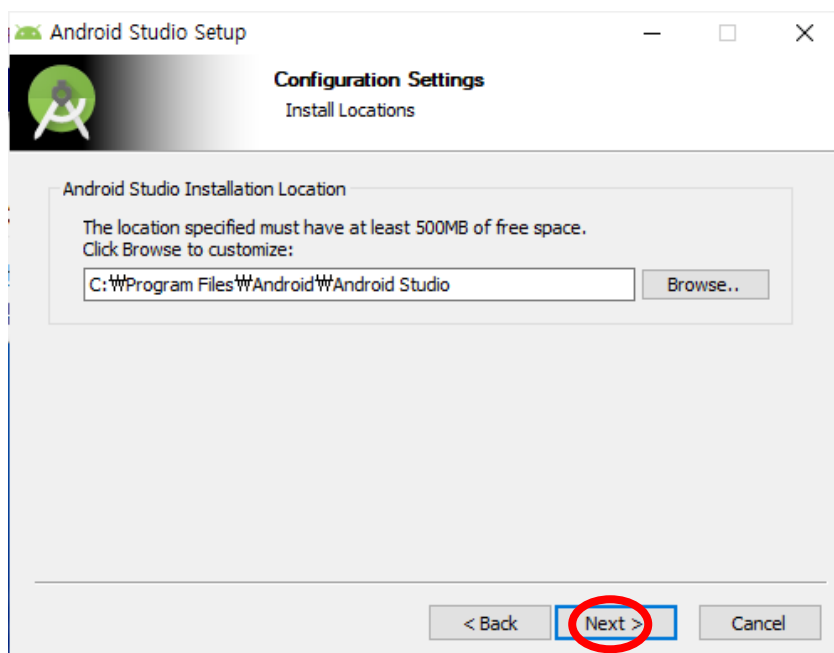
-설치 처음 화면



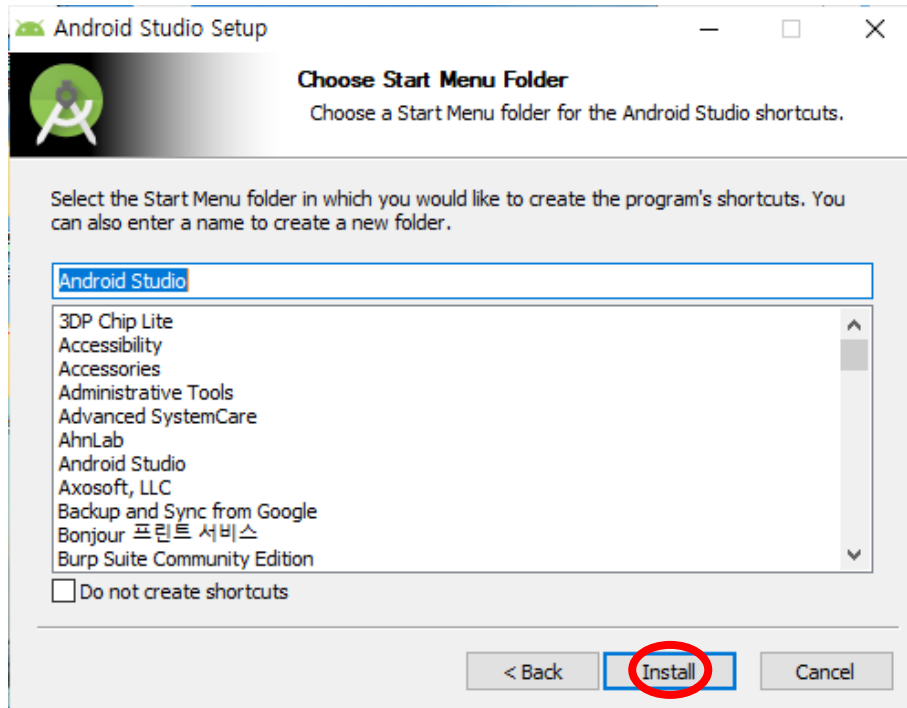
-구성요소 선택 : Android Virtual Device 는 어플 테스트를 위한 가상의 장치를 의미합니다. 직접 스마트폰을 이용하거나, 가상의 장치를 설치해서 테스트에 이용할 수 있습니다.



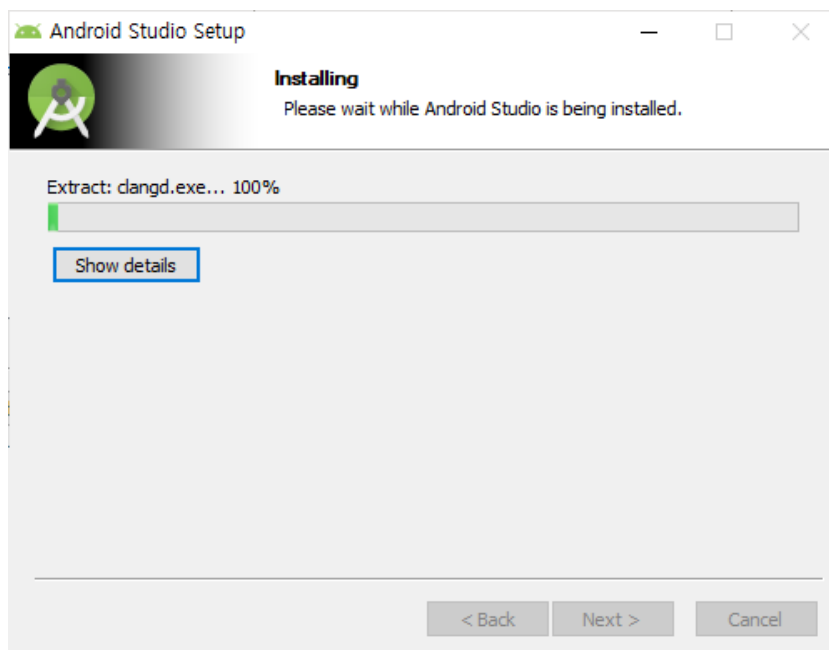
-설치경로 지정



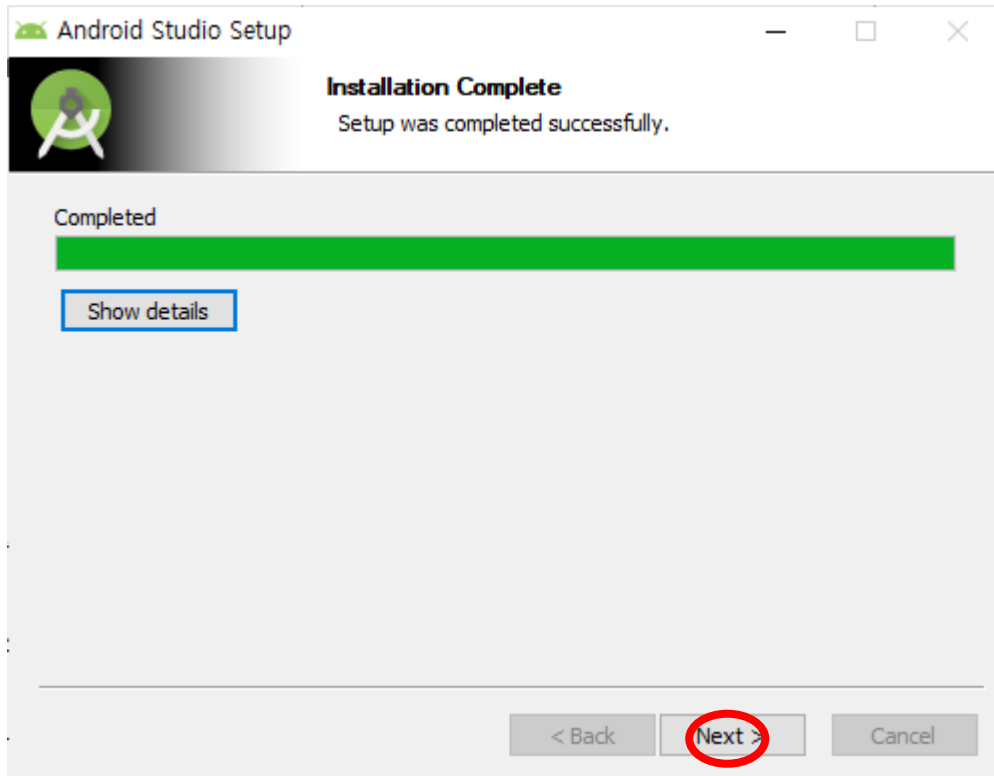
-시작메뉴 폴더 지정



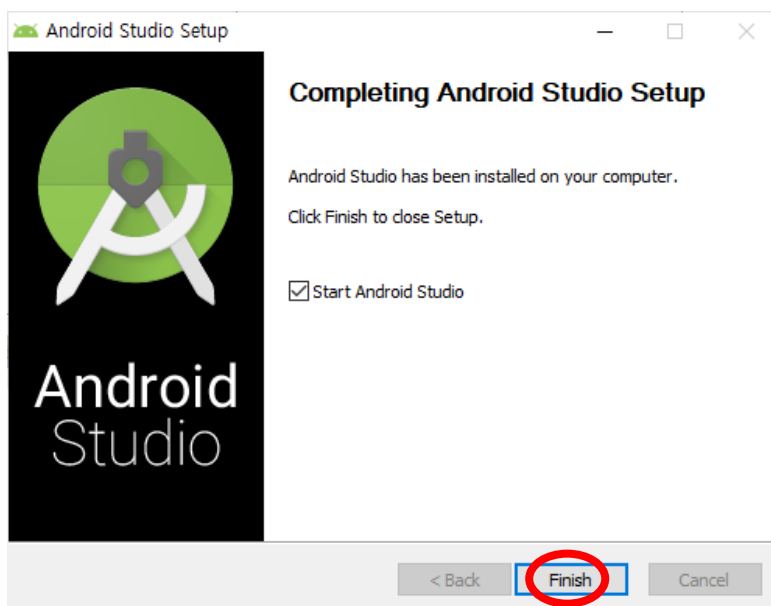
-설치 중 화면



-설치 완료화면

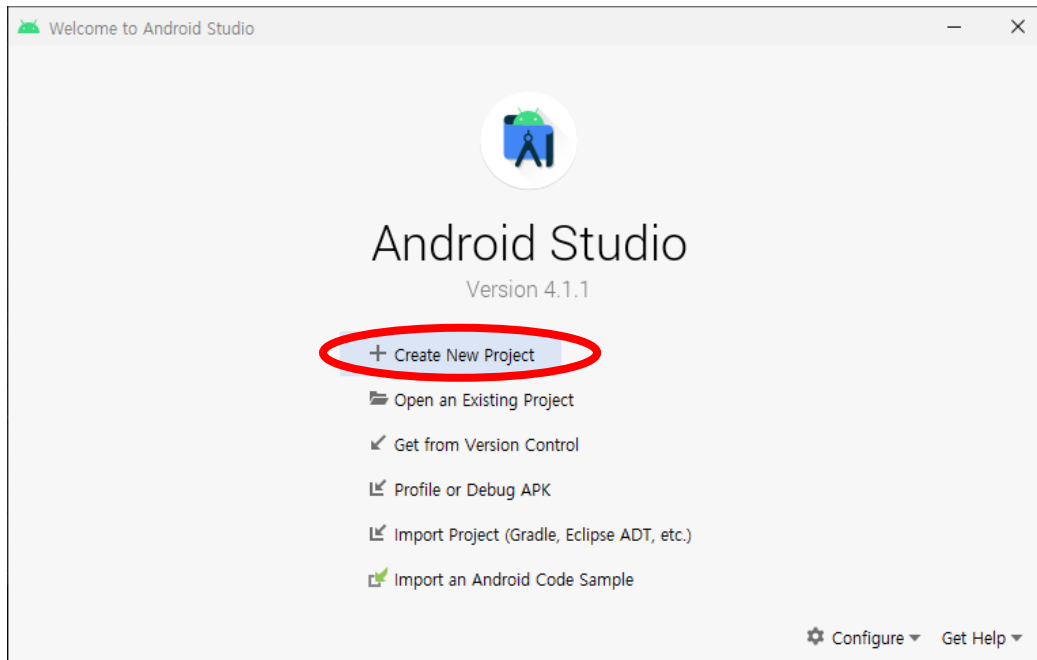


-정상적으로 설치 완료

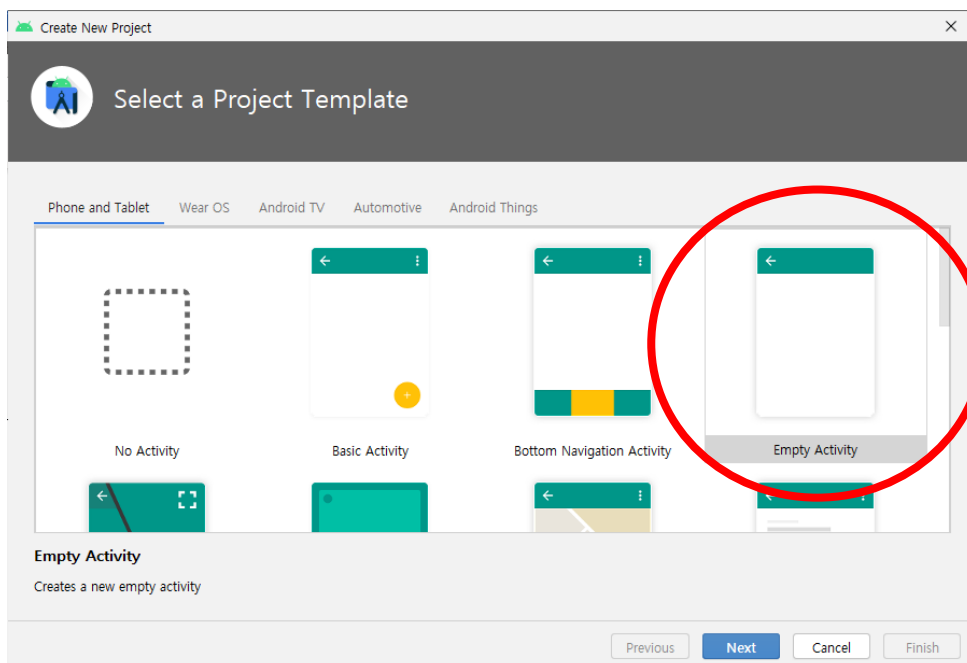


2-2 프로젝트 생성

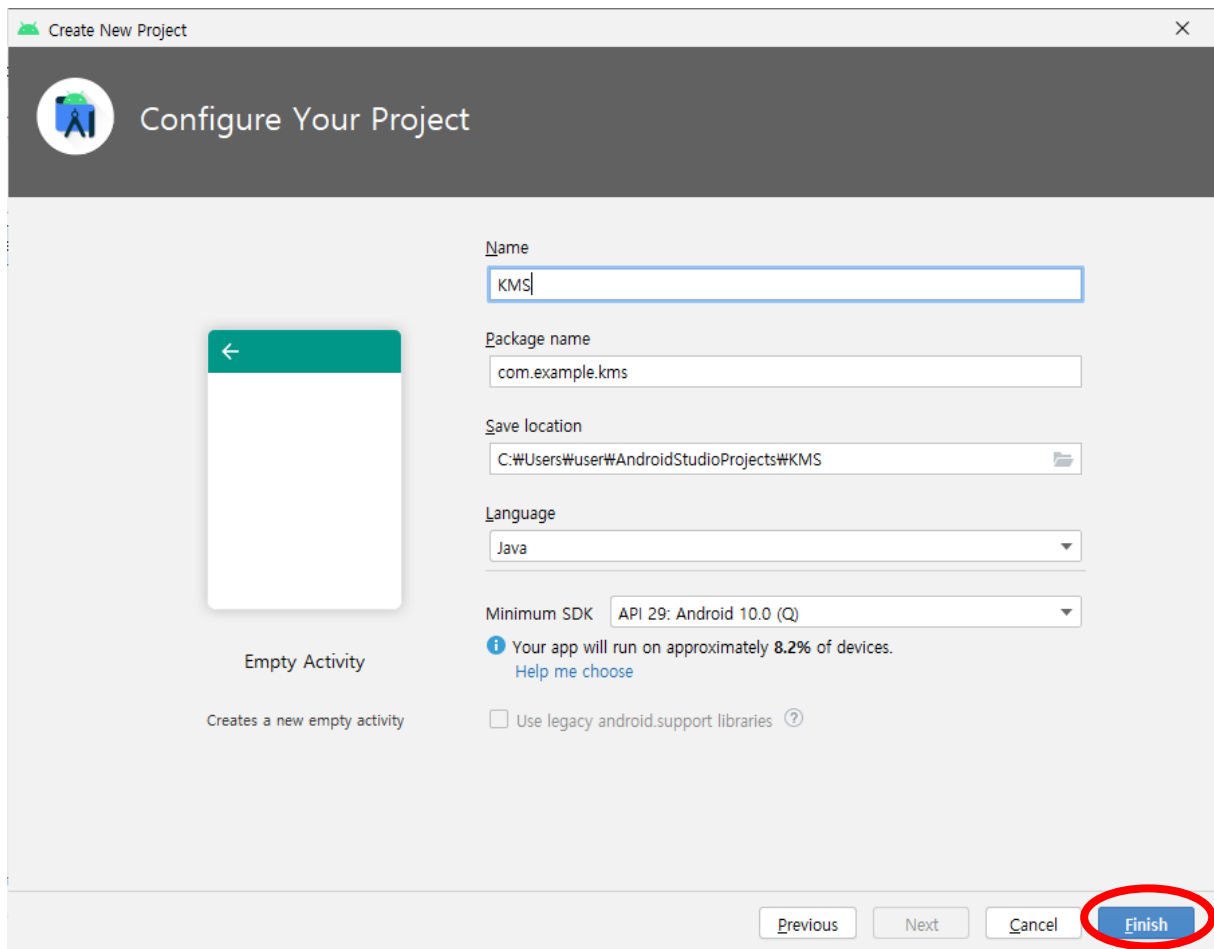
-안드로이드 스튜디오 초기화면 : 새 프로젝트를 생성하기 위해 Create New Project 를 클릭합니다.



-프로젝트 템플릿 선택 : 테스트를 위해 Empty Activity 를 선택 후, Next 를 클릭합니다.



-프로젝트 구성



Name : 프로젝트의 이름을 설정합니다.

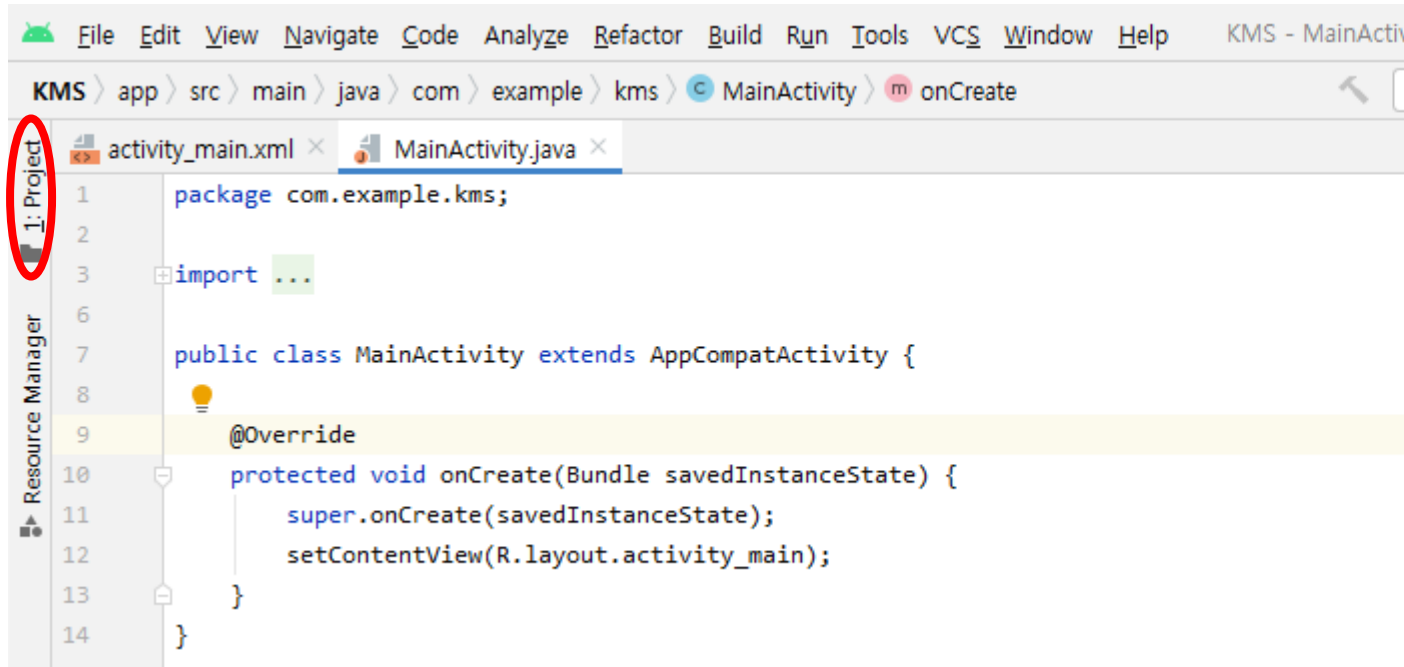
Package name : 메인 액티비티 패키지의 이름을 설정합니다.

Save location : 저장 위치를 설정합니다

Language : 사용할 언어를 설정합니다. Kotlin 은 2011 년에 JetBrains 에서 공개된 프로그래밍 언어이며, JAVA 에 비해 문법이 간결하고 JAVA 와 100% 대응이 되는 언어입니다.

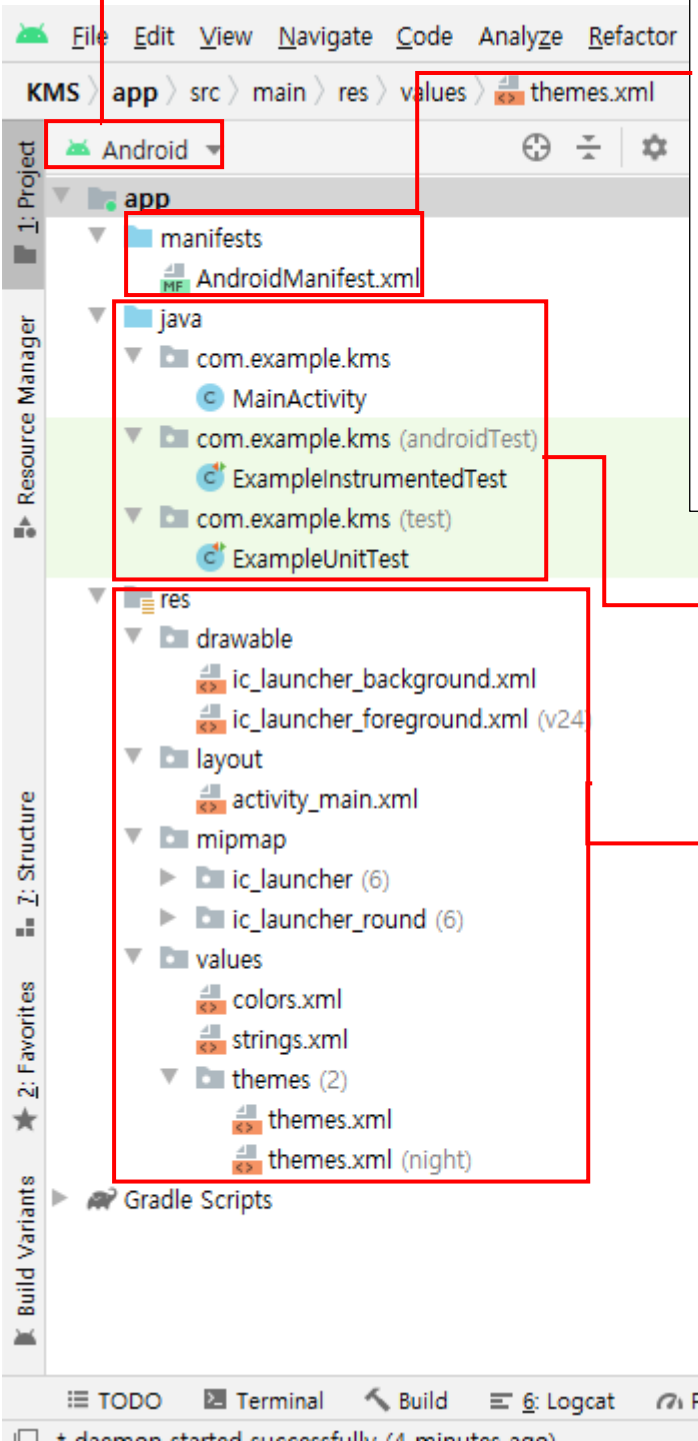
Minimum SDK(Software Development Kit) : 호환 가능한 안드로이드 장치의 OS 의 최소 버전을 설정합니다. 나중에 출시된 어플은 여기서 설정된 버전 이상의 OS 에서 작동합니다. 'Help me choose' 를 통해 현재 안드로이드의 OS 점유율 등 다양한 정보를 확인할 수 있습니다.

-프로젝트 생성 후 초기화면



프로젝트를 생성하고 나면 다음과 같은 화면이 나타납니다. 표시한 부분을 클릭하면 프로젝트의 파일들을 볼 수 있습니다.

이 부분을 통해 파일들을 보는 기준을 정할 수 있습니다.



manifests : 어플리케이션에 대한 다양한 기본정보가 담겨 있는 곳입니다. 한 화면을 구성하는 액티비티가 여러 개인 경우 처음 실행될 액티비티를 설정할 수 있습니다. 또한, 어플의 다양한 설정이나 권한을 부여할 때 사용됩니다.

manifests : 어플리케이션에 대한 다양한 기본정보가 담겨 있는 곳입니다. 한 화면을 구성하는 액티비티가 여러 개인 경우 처음 실행될 액티비티를 설정할 수 있습니다. 또한, 어플의 다양한 설정이나 권한을 부여할 때 사용됩니다.

java : 자바 코드를 넣는 곳으로,
클래스를 관리합니다.

drawable : 어플리케이션에 사용될 오디오,
이미지 파일 등을 넣는 곳입니다.

layout : 어플리케이션 화면을 구성하기 위한
xml 파일들이 있는 곳입니다.

mipmap : 어플리케이션 실행 아이콘 관련
이미지를 넣는 곳입니다.

drawable : 어플리케이션에 사용될 오디오,
이미지 파일 등을 넣는 곳입니다.

layout : 어플리케이션 화면을 구성하기 위한
xml 파일들이 있는 곳입니다.

mipmap : 어플리케이션 실행 아이콘 관련
이미지를 넣는 곳입니다.

drawable : 어플리케이션에 사용될 오디오,
이미지 파일 등을 넣는 곳입니다.

layout : 어플리케이션 화면을 구성하기 위한
xml 파일들이 있는 곳입니다.

mipmap : 어플리케이션 실행 아이콘 관련
이미지를 넣는 곳입니다.

drawable : 어플리케이션에 사용될 오디오,
이미지 파일 등을 넣는 곳입니다.

layout : 어플리케이션 화면을 구성하기 위한
xml 파일들이 있는 곳입니다.

mipmap : 어플리케이션 실행 아이콘 관련
이미지를 넣는 곳입니다.

drawable : 어플리케이션에 사용될 오디오,
이미지 파일 등을 넣는 곳입니다.

layout : 어플리케이션 화면을 구성하기 위한
xml 파일들이 있는 곳입니다.

mipmap : 어플리케이션 실행 아이콘 관련
이미지를 넣는 곳입니다.

drawable : 어플리케이션에 사용될 오디오,
이미지 파일 등을 넣는 곳입니다.

layout : 어플리케이션 화면을 구성하기 위한
xml 파일들이 있는 곳입니다.

mipmap : 어플리케이션 실행 아이콘 관련
이미지를 넣는 곳입니다.

values : 여러 값들을 수정하는 곳입니다.

특히, strings.xml 에 들어가서

어플리케이션의 이름을 수정할 수

있습니다.

values : 여러 값들을 수정하는 곳입니다.

특히, strings.xml 에 들어가서

어플리케이션의 이름을 수정할 수

있습니다.

values : 여러 값들을 수정하는 곳입니다.

특히, strings.xml 에 들어가서

어플리케이션의 이름을 수정할 수

있습니다.

values : 여러 값들을 수정하는 곳입니다.

특히, strings.xml 에 들어가서

어플리케이션의 이름을 수정할 수

있습니다.

-프로젝트의 기본 파일 구성요소

MainActivity.java : 안드로이드 앱에서의 메인 함수 역할을 수행하는 메인 액티비티 클래스 입니다.

activity_main.xml : main activity 의 화면을 구성하는 파일입니다.

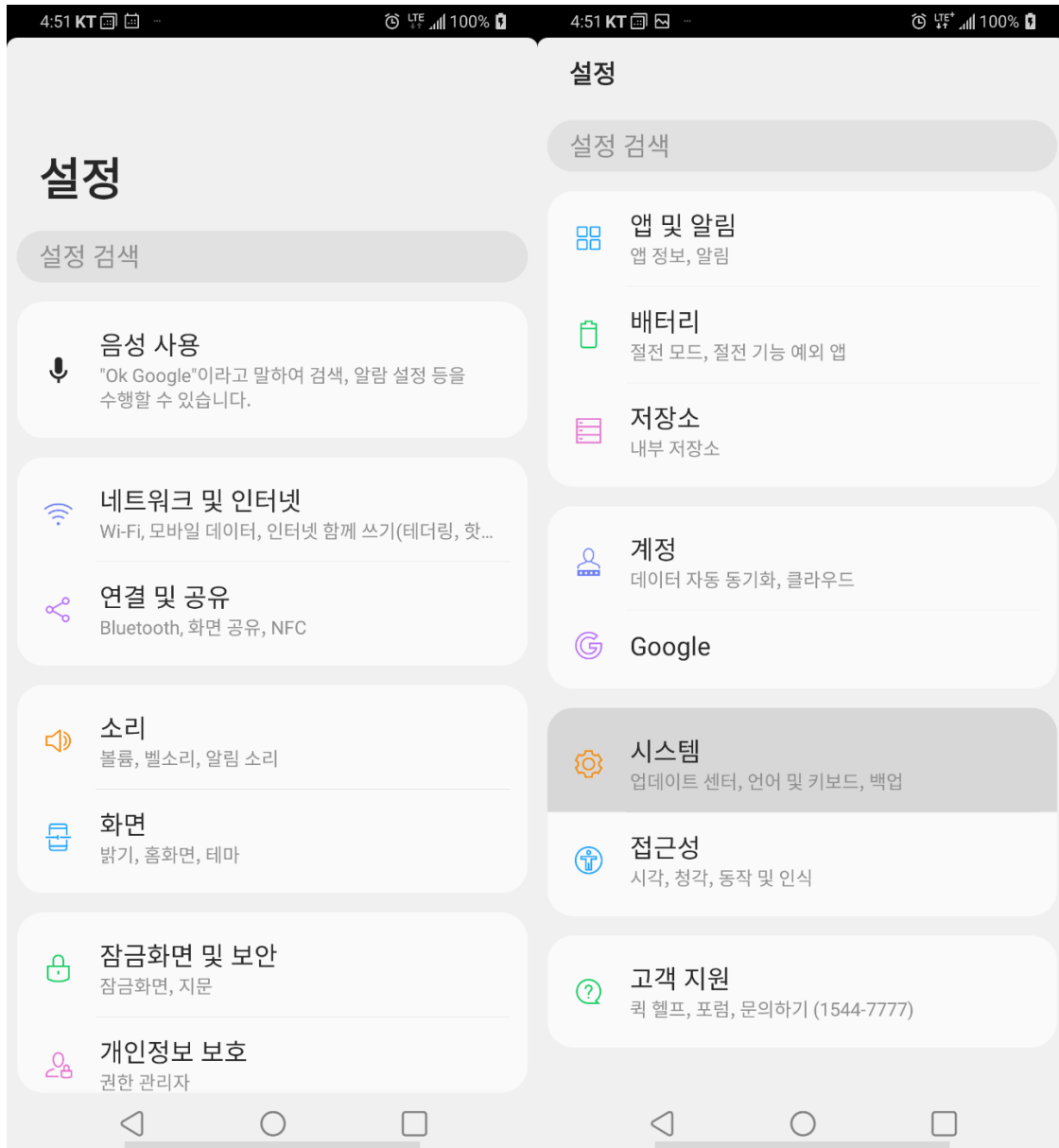
AndroidManifest.xml : 어플의 다양한 정보들을 담고 있는 설정 파일입니다.

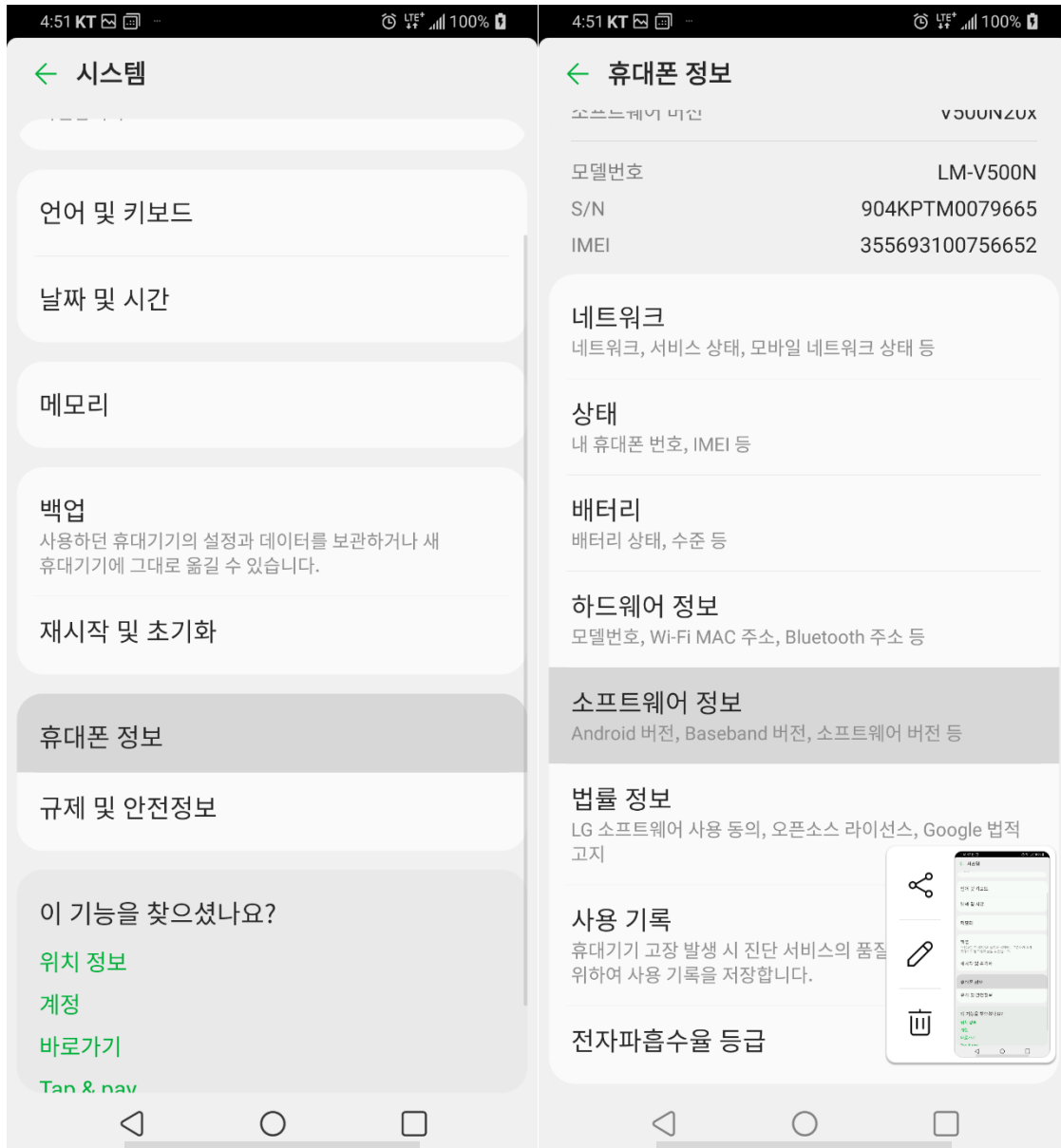
3. 프로젝트 실행

생성된 프로젝트를 실행시켜 보겠습니다. 두 가지의 방법으로 실행이 가능한데, 실제 스마트폰을 컴퓨터와 연결해서 앱을 실행시켜 볼 수 있고, 가상의 디바이스인 에뮬레이터를 통해 앱을 실행할 수도 있습니다.

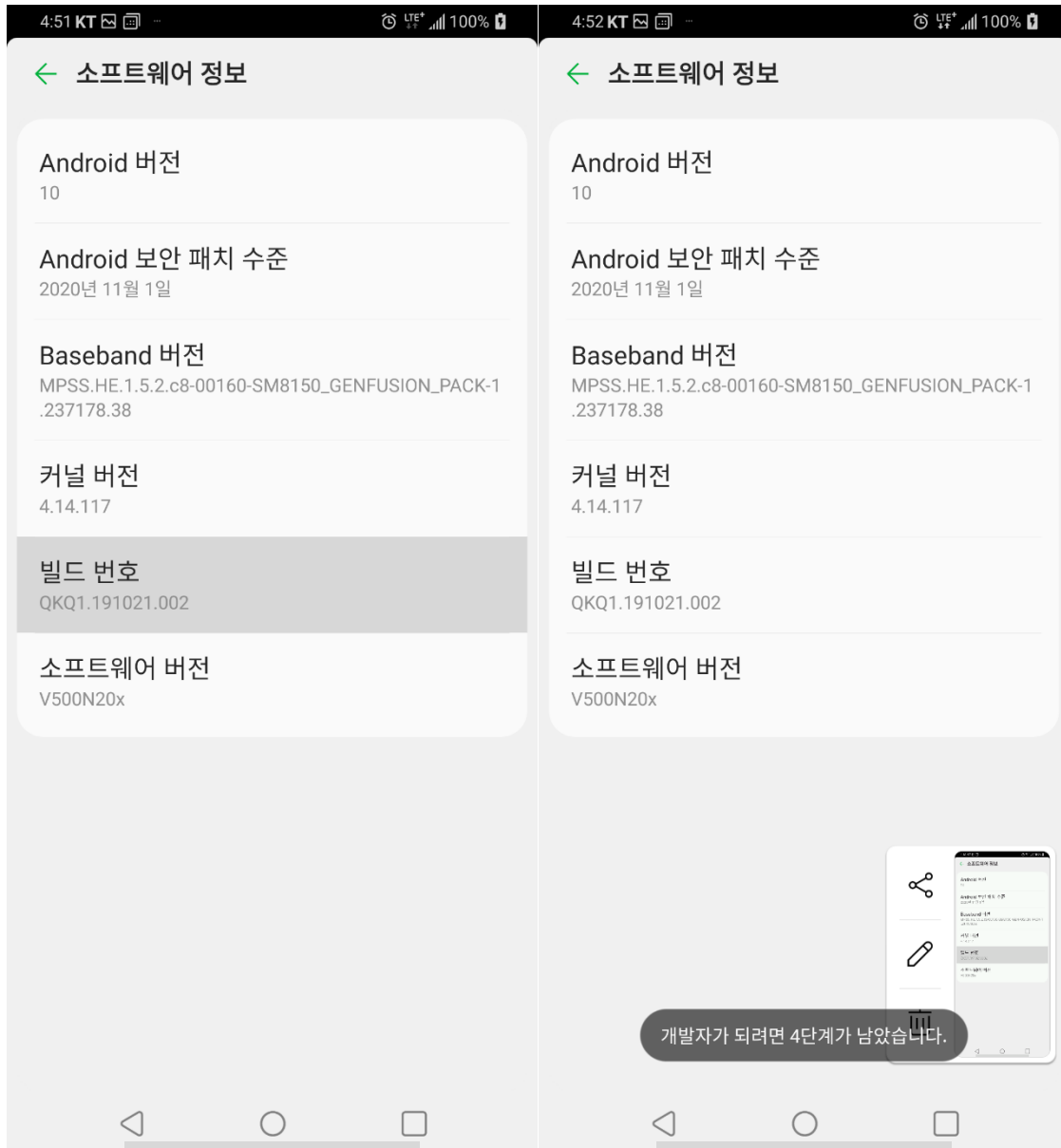
3-1 실제 기기로 실행

먼저 실제로 스마트폰을 연결해서 실행시켜 보았습니다. 테스트한 스마트폰의 기종은 V50 입니다. 실제 스마트폰 기기에서 안드로이드 스튜디오로 개발을 테스트하기 위해서는 USB 디버깅을 활성화해야 합니다. 테스트를 위해서는 추가적으로 각 기종에 맞는 Mobile Driver 가 설치되어 있어야 합니다. 진행 과정은 다음과 같습니다.





-소프트웨어 정보에서 빌드 번호를 연속적으로 터치하면 개발자옵션이 활성화됩니다.



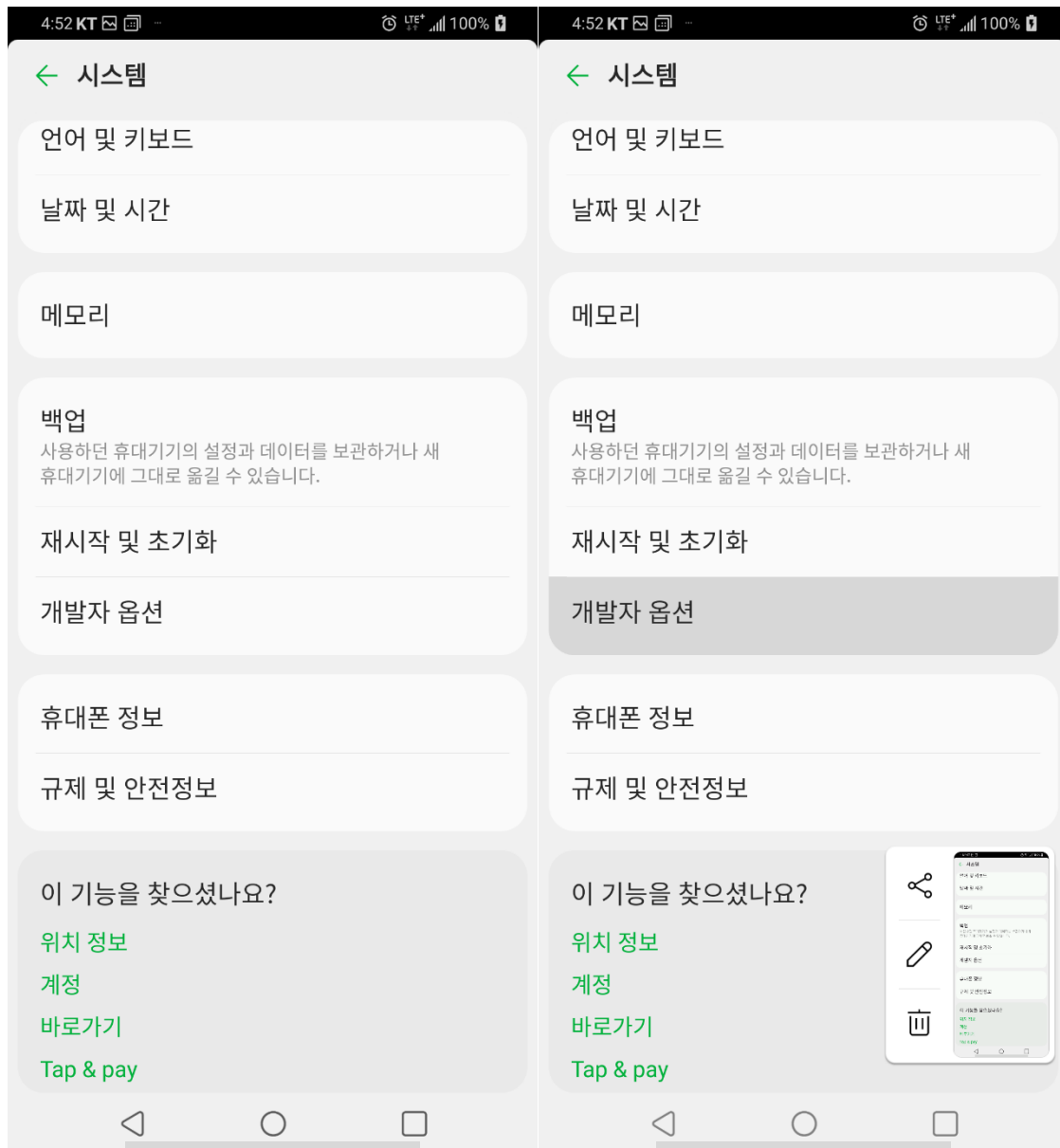
개발자가 되려면 3단계가 남았습니다.

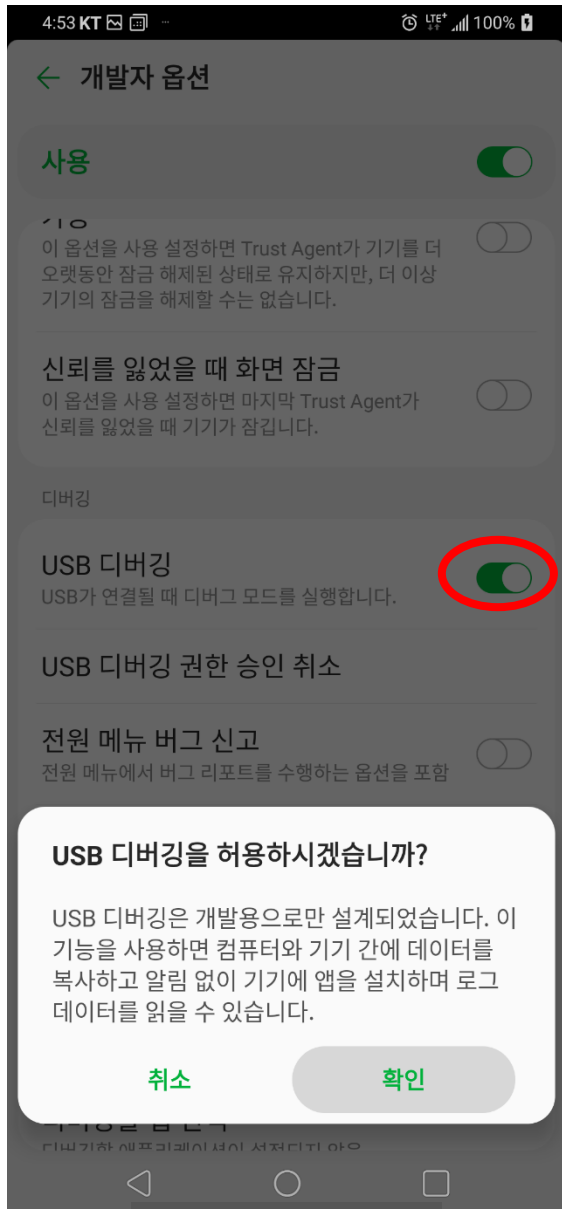
개발자가 되려면 2단계가 남았습니다.

개발자가 되려면 1단계가 남았습니다.

개발자가 되었습니다.

-개발자 옵션이 활성화되면, 시스템 탭에서 개발자 옵션 항목을 볼 수 있습니다. 이 곳에서 USB 디버깅 모드를 활성화합니다.





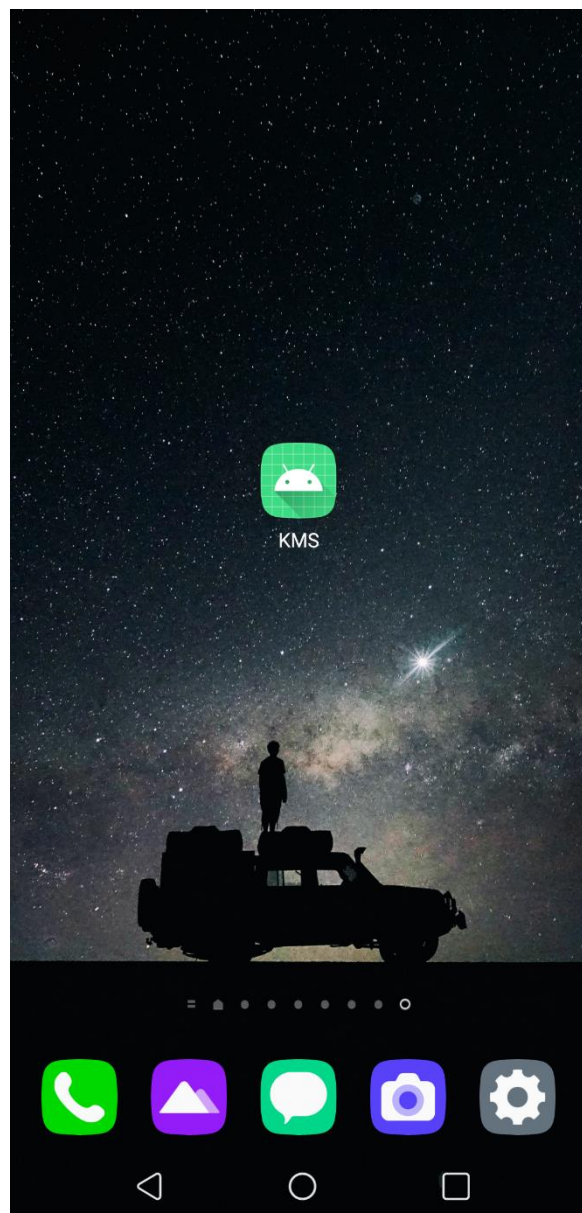
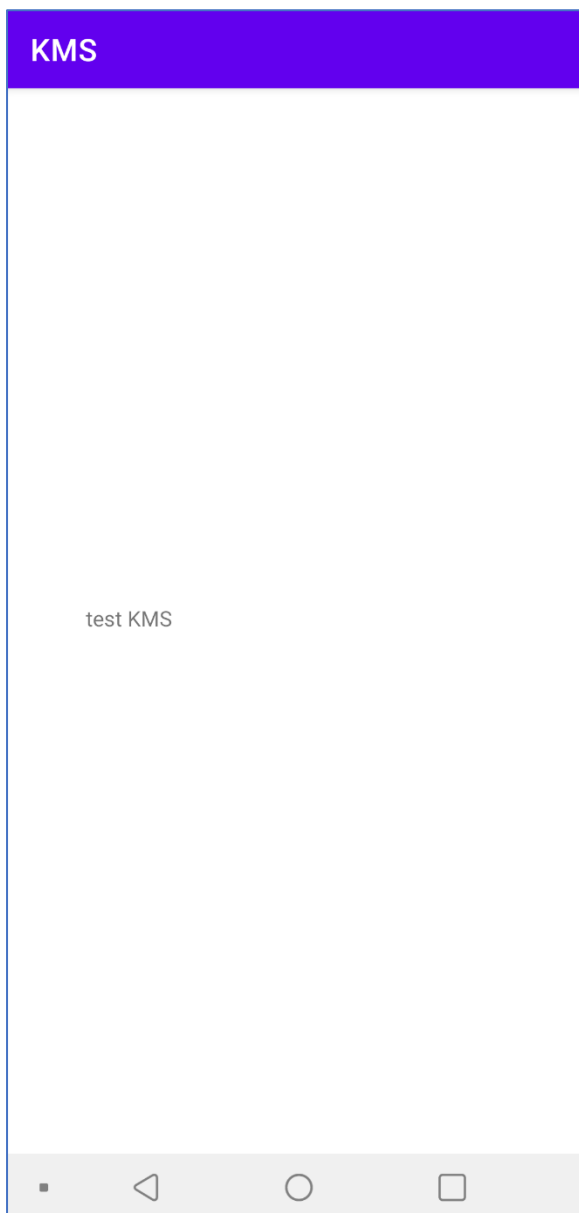
이제 안드로이드 스튜디오로 개발을 진행하는 컴퓨터와 해당 스마트폰을 연결하면, 테스트 준비가 완료됩니다.

연결한 결과, 다음과 같이 안드로이드 스튜디오에서 휴대폰을 인식한 것을 볼 수 있습니다.



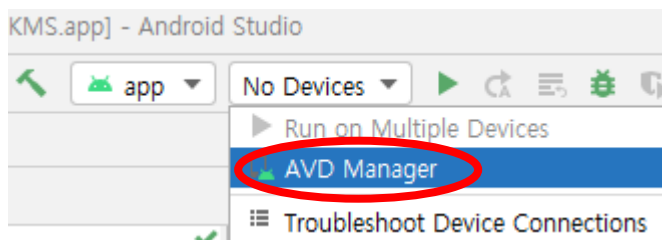
표시한 실행버튼을 클릭하면 개발한 어플이 스마트폰에서 실행되는 것을 확인할 수 있습니다.

또한, 앱을 종료하고 나면 홈 화면에 테스트용 앱이 생성되어 있는 것을 볼 수 있습니다.



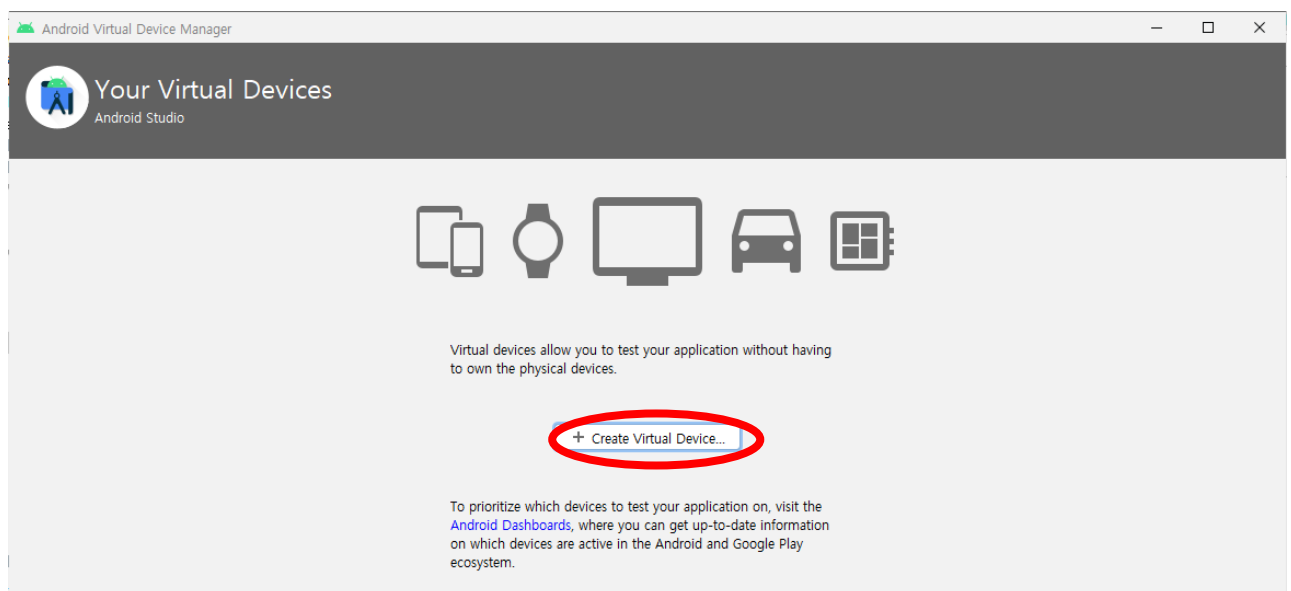
3-2 가상 디바이스로 실행

이번에는 가상 디바이스를 통한 테스트를 진행해보겠습니다.



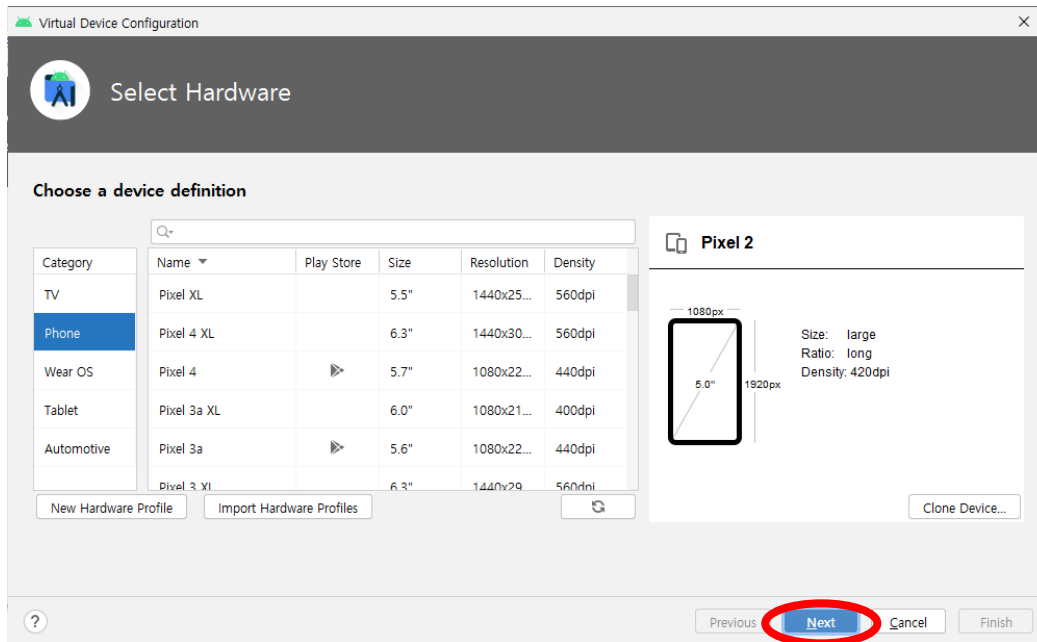
표시된 부분을 클릭하여 AVD Manager 설정 창으로 이동합니다. 이는 프로젝트가 실행될 디바이스를 설정하는 과정입니다.

-AVD Manager



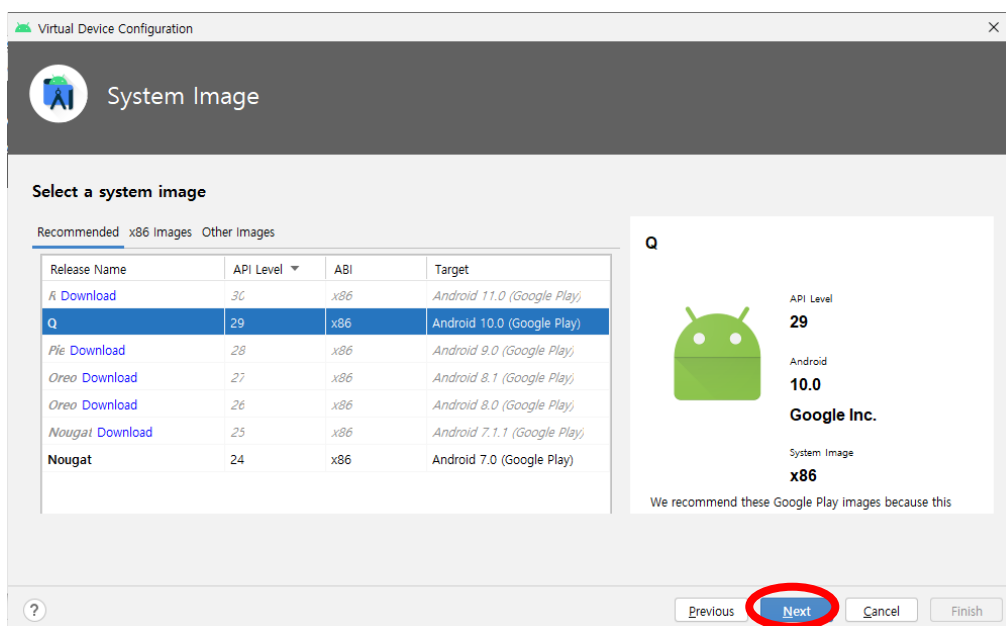
Create Virtual Device 를 클릭하여 새로운 디바이스를 만듭니다.

-가상 디바이스 구성



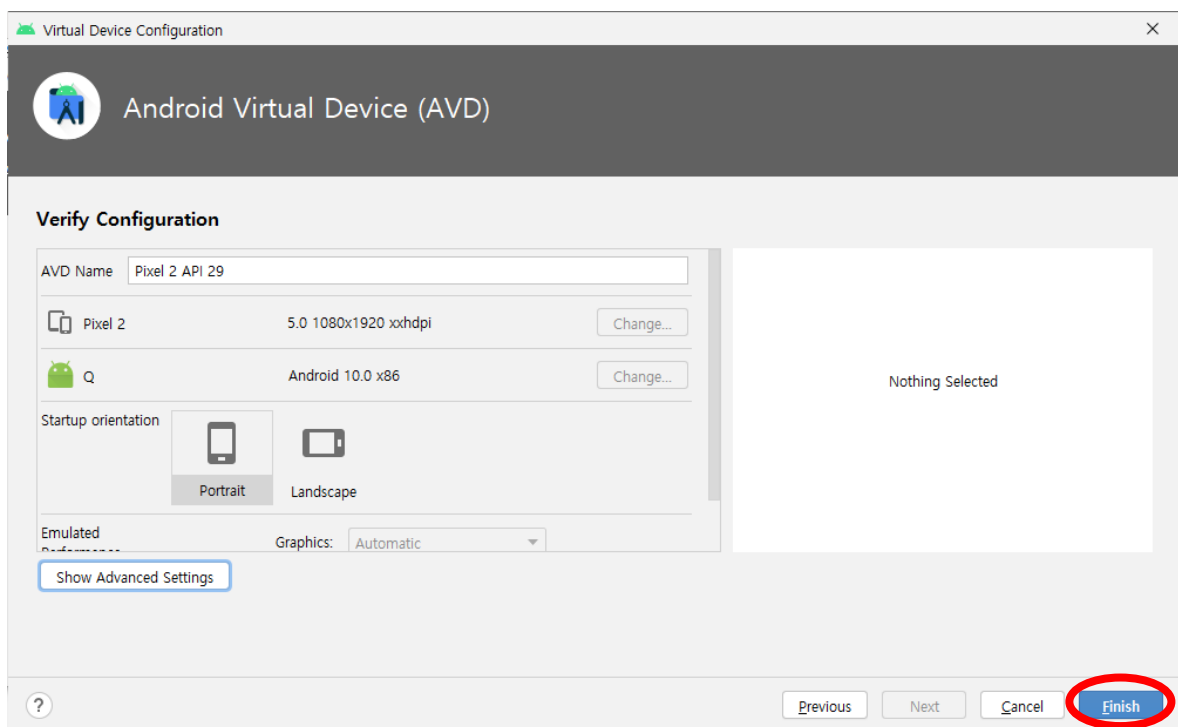
장치의 종류, 이름, 해상도, 크기 등을 설정할 수 있는 화면입니다. 설정 후 Next를 눌러 다음으로 넘어갑니다.

-시스템 이미지 설정



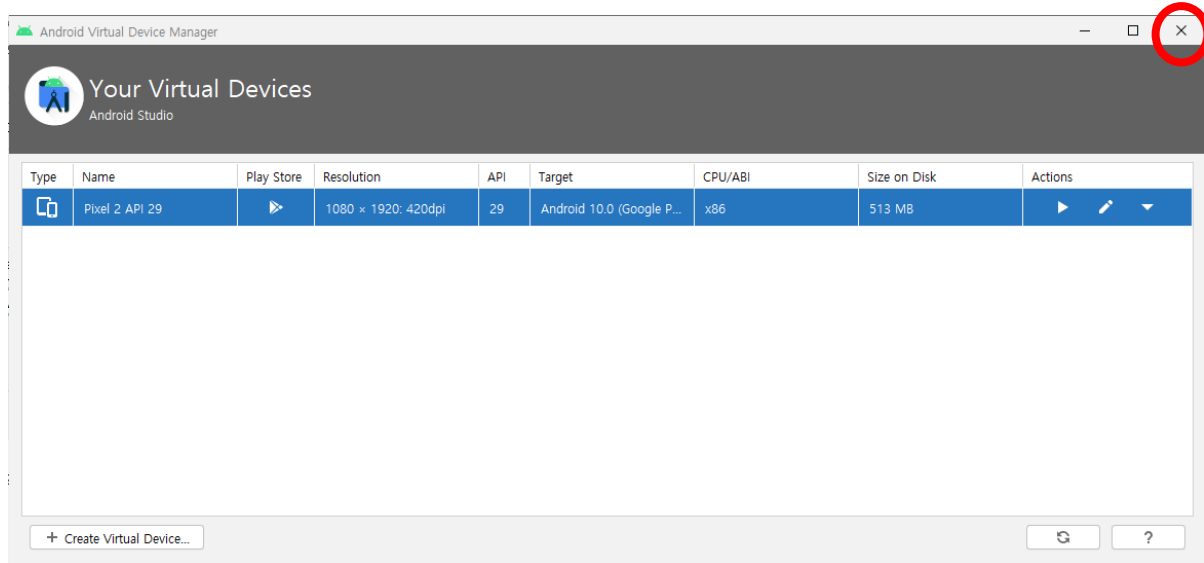
운영체제의 버전(이미지)을 설정하는 부분입니다. 프로젝트를 생성할 때 설정했던 Mininum SDK의 API 보다 같거나 높도록 이미지를 설정해야 합니다. 필요에 따라 Download 를 클릭하여 이미지를 다운로드합니다.

-설정 확인



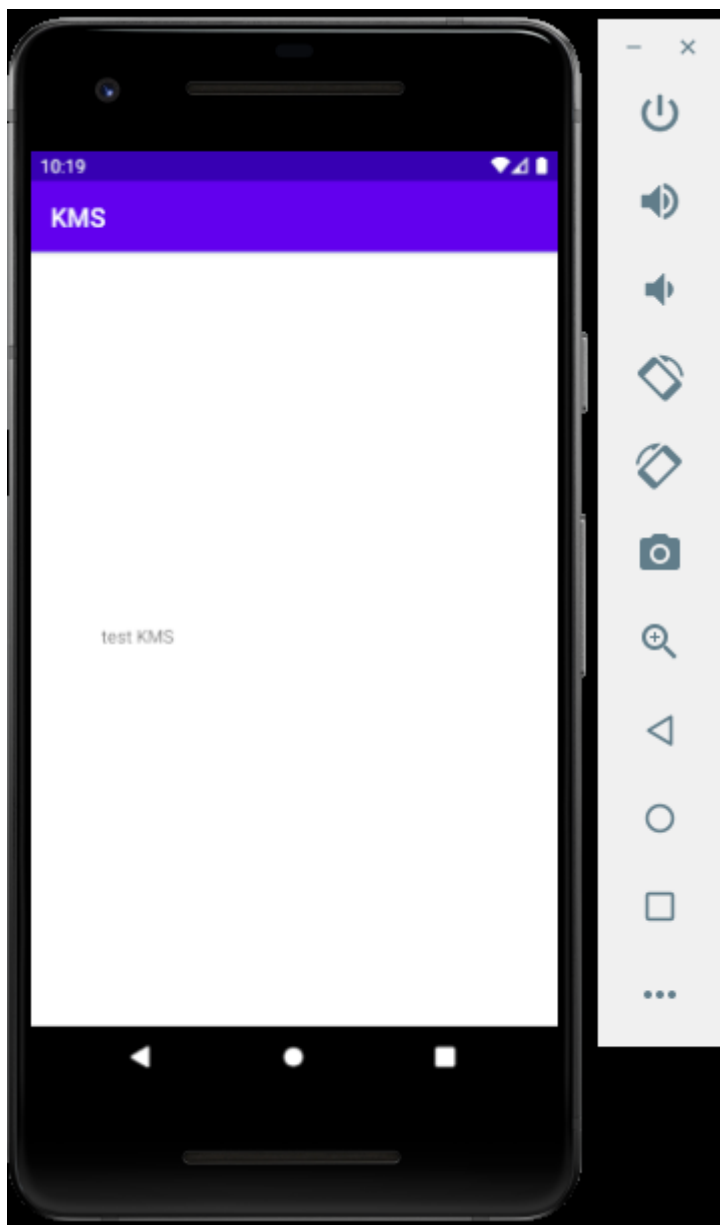
기존의 설정을 확인하고, 가로로 실행할지, 세로로 실행할지 등을 결정합니다. 'Show advanced Settings'를 통해 카메라, 네트워크, 성능, 메모리 등의 다양한 설정을 할 수 있습니다.

-AVD Manager : 새로운 가상 디바이스가 생긴 것을 확인할 수 있습니다.



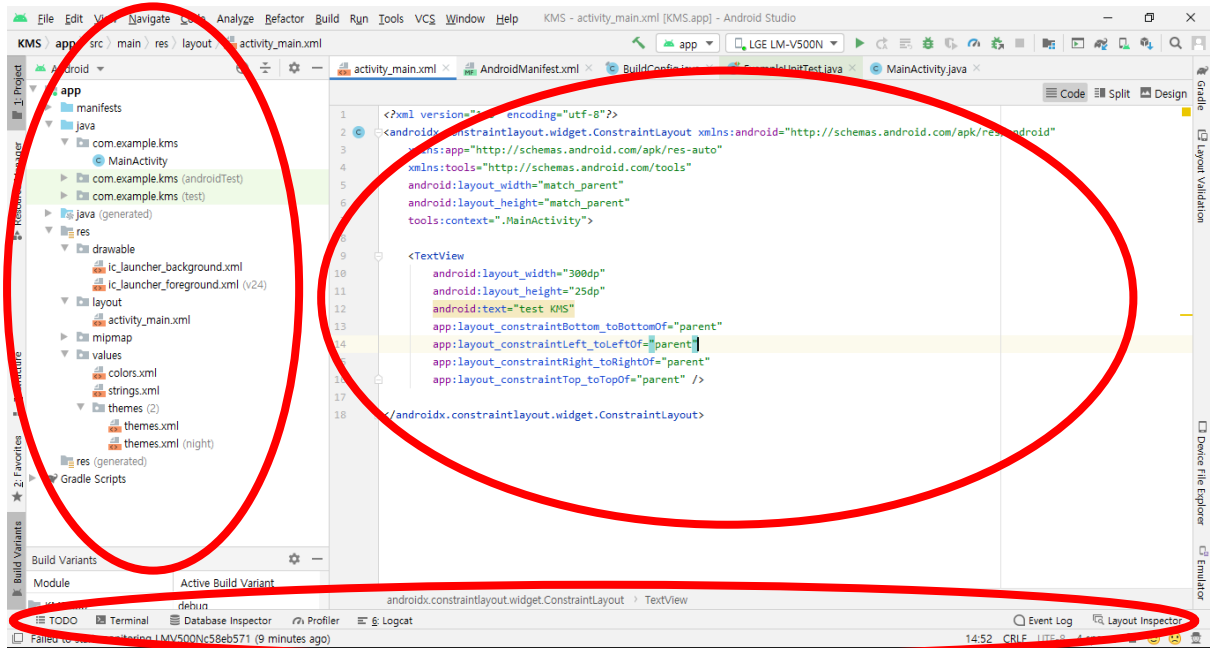
이후 현재 app 을 생성한 가상 디바이스로 실행하면 에뮬레이터가 실행되면서 개발한 어플을 테스트할 수 있습니다.





4. 안드로이드 스튜디오의 구조 및 동작 방식

안드로이드 스튜디오의 전반적인 구조와 동작 방식을 살펴보겠습니다.



안드로이드 스튜디오는 먼저 크게 좌측, 우측, 하단의 3 가지 영역으로 나누어 집니다. 좌측은 프로젝트의 구조를, 우측은 파일이나 코드의 내용을, 하단은 실행 모드나 디버깅 등을 나타냅니다.

4-1 프로젝트 구조

프로젝트의 구조는 app 폴더와 Gradle Scripts 폴더로 이루어져 있습니다. Gradle Scripts 폴더는 어플리케이션에 필요한 라이브러리나, 빌드에 필요한 옵션 등을 포함합니다.

app 폴더는 아래에 크게 manifests, java, res 세 개의 폴더가 존재합니다.

manifests : 전반적인 제어를 하는 컨트롤타워로 볼 수 있습니다. 이 곳에는 AndroidManifest.xml

파일이 있는데, 여기서 어플리케이션에 필요한 여러 설정 값들을 관리합니다.

java : 자바코드가 들어 있는 폴더 입니다. 이 곳에서 여러 이벤트 처리나 자바 관련 클래스 들을 관리합니다.

res : 자원들을 가지고 있는 resource 폴더이며, 디자인, 문자열 리소스나 UI 관련 파일들을 포함하는 곳입니다.

4-1-1 필수 설정 파일

manifests 폴더의 AndroidManifest.xml 파일을 통해 필요한 설정들을 해주지 않으면 잠재적인 에러가 발생할 수 있으므로 필요할 때마다 수정을 해야 합니다. AndroidManifest.xml 파일은 다음과 같습니다.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3         package="com.example.kms">
4
5      <application
6          android:allowBackup="true"
7          android:icon="@mipmap/ic_launcher"
8          android:label="KMS"
9          android:roundIcon="@mipmap/ic_launcher_round"
10         android:supportRtl="true"
11         android:theme="@style/Theme.KMS">
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>
```

태그의 종류가 <manifest>, <application>, <activity>, <intent-filter>, <action>, <category> 등 총 6 개 정도가 있습니다.

<manifest> : 패키지를 표시합니다.

<application> : 어플리케이션 관련 설정을 진행합니다. 백업허용, 아이콘 위치, 앱 이름 등 다양한 설정을 할 수 있습니다.

나머지 태그들은 자동으로 생성되는 부분들입니다.

이처럼 안드로이드 프로젝트의 구조는 application 위에 activity 가 실행되는 구조이며, Activity 파일은 항상 xml 파일과 짝을 이루는 경우가 대부분입니다.

4-1-2 java 폴더

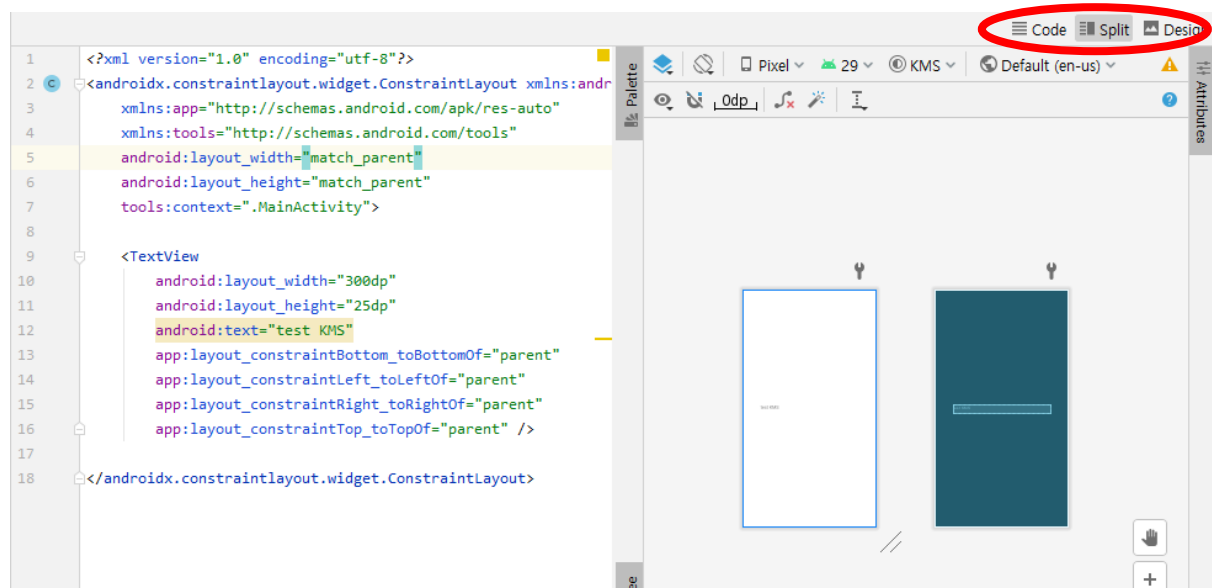
이 부분에서 안드로이드 관련 개발을 진행한다고 볼 수 있습니다. 자바 파일을 생성하고, 수정함으로써 어플리케이션의 전반적인 작동 방식을 정의합니다. 여기서 첫번째 패키지 안에 MainActivity.java 라는 파일이 있는데, 이 파일이 모든 프로젝트에 필수적으로 존재하는 기본 파일입니다. 처음 코드를 보면 다음과 같습니다.

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

본문의 onCreate()함수 내부에서 setContentView(R.layout.activity_main); 이 있는 것을 볼 수 있습니다. 이 코드는 layout 폴더 아래에 activity_main.xml 이라는 파일을 뷰로 연결하는 코드로, 이 연결을 통해 UI 를 구현할 수 있습니다. 여기서 xml 파일에 대해 나오는데, 이렇게 액티비티에 해당하는 자바 파일 하나에 xml 파일 하나를 연결해서 관리하는 것이 일반적입니다.

이제 MainActivity.java 에 대응하는 layout 폴더 아래의 activity_main.xml 파일을 살펴보겠습니다.

4-1-3 res 폴더



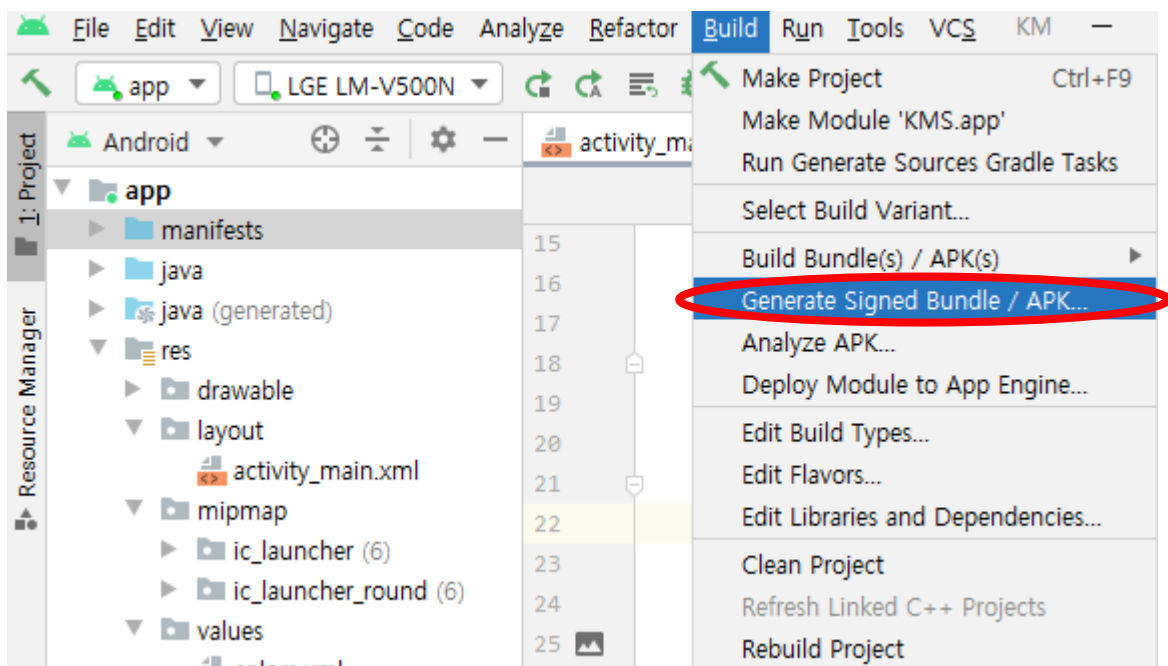
위 그림은 res/layout 에 있는 activity_main.xml 파일입니다. 그림에서 표시된 부분을 통해 파일을 코드형태, 디자인형태, 또는 두 가지 모두의 형태로 볼 수 있습니다. 이처럼 레이아웃과 관련된 화면을 구성하는 xml 파일들은 모두 layout 폴더에 존재합니다. 이 layout 폴더가 뼈대를 담당하며, drawable, mipmap, values 폴더들은 수치 값이나 이미지파일을 저장하고 있는 부수적인 곳입니다.

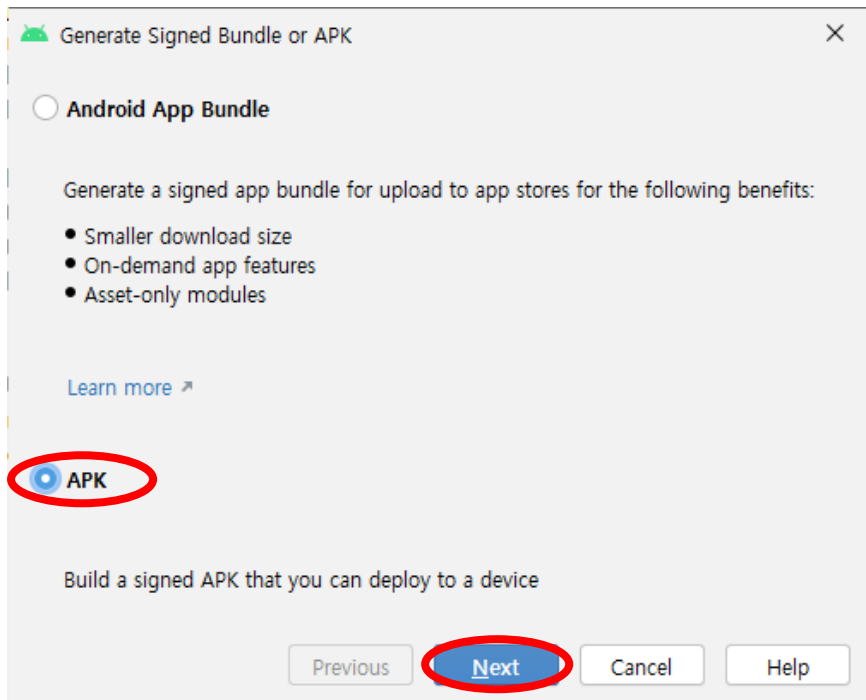
5. 추억을 되새겨주는 갤러리(간단한 어플)

지금까지 프로젝트의 구조, 동작방식, 개발환경 구축 등의 학습한 내용들을 바탕으로 실제로 간단한 어플리케이션을 만들어서 테스트해보았습니다. 또한 결과물을 Github 에 업로드하여 누구나 확인해볼 수 있도록 구현했습니다. 이 과정을 살펴보겠습니다.

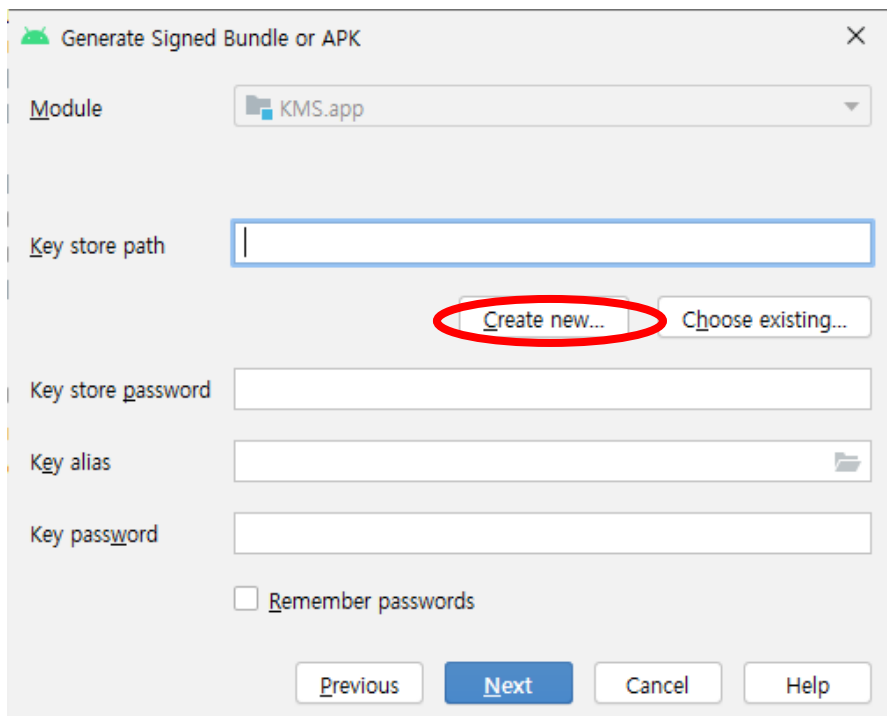
5-1 apk 추출

앱을 배포하기 위해 아래의 과정을 거쳐 apk 파일을 추출하였습니다.

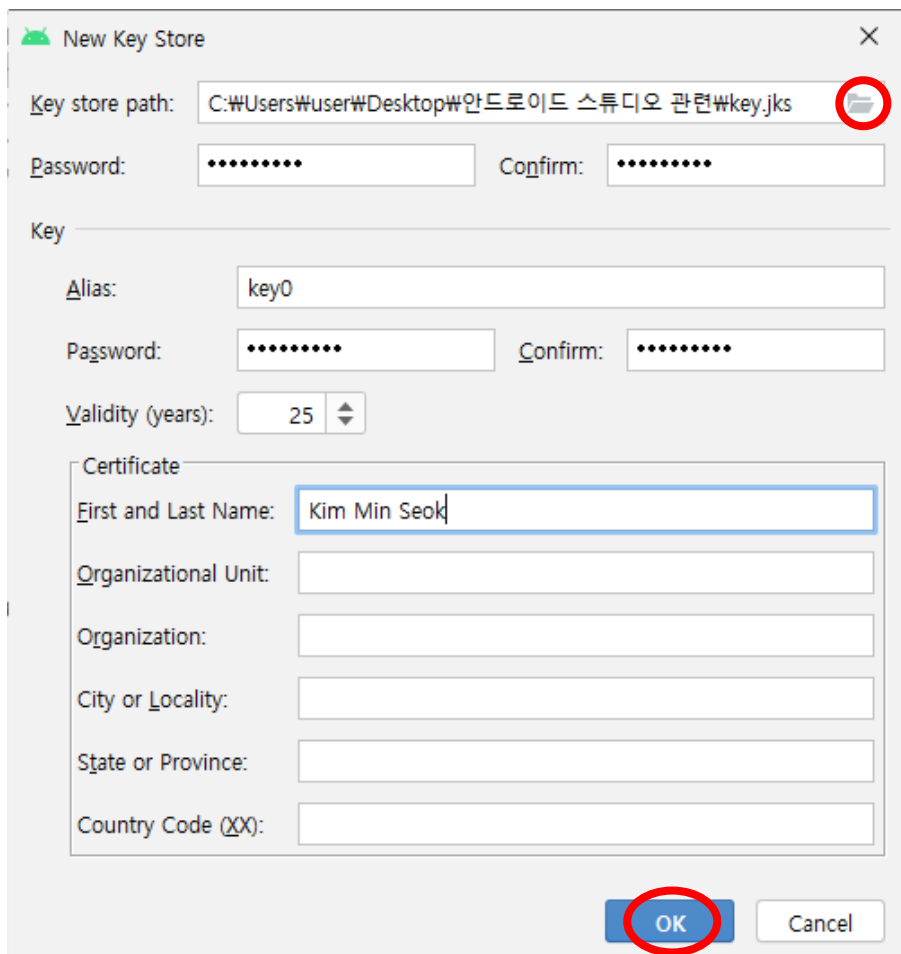




APK 를 선택한 후 다음으로 넘어갑니다.

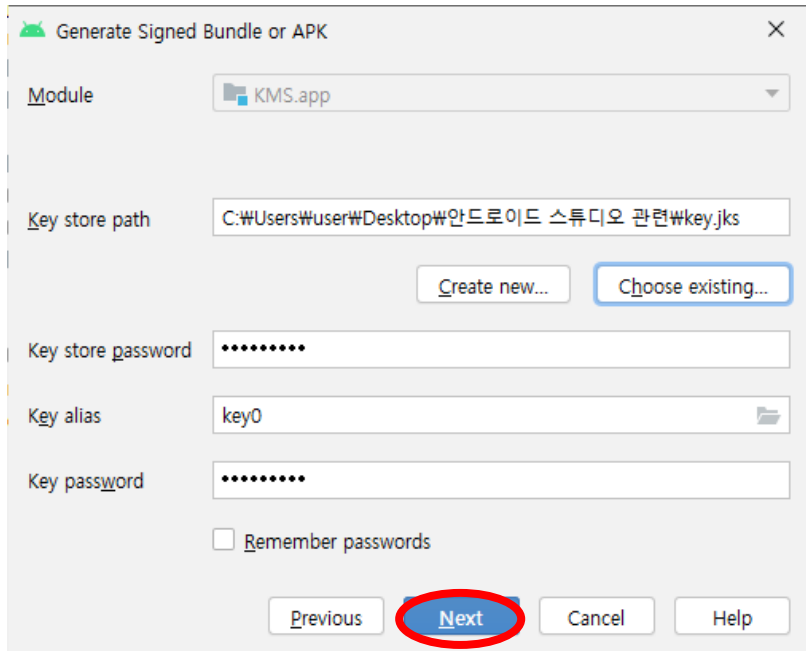


키를 설정하는 창이 나오는데, 기존의 키가 없으므로 새로 생성합니다. 여기서 키는 jks(Java Key Store)파일로 저장하는데, 이는 암호화를 위한 키들을 모아 놓은 파일입니다. 이 파일은 프로젝트를 인증해주는 역할을 하기 때문에 함부로 누군가에게 공개되거나 변조되어서는 안됩니다.

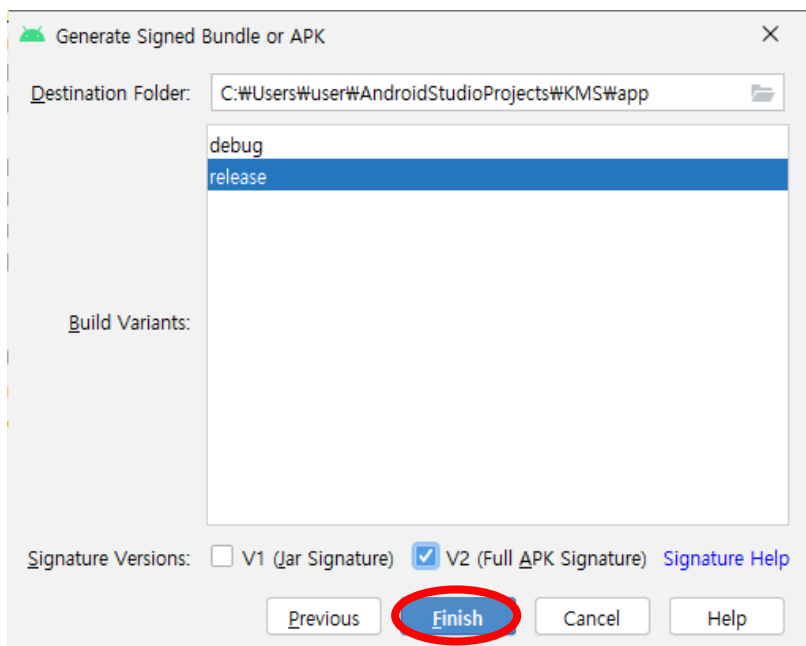


The screenshot shows the 'New Key Store' dialog box. The 'Key store path' field is set to 'C:\Users\User\Desktop\안드로이드 스튜디오 관련\key.jks' and is circled in red. Below it are 'Password' and 'Confirm' fields, both masked with dots. The 'Key' section has an 'Alias' field set to 'key0', and 'Password' and 'Confirm' fields, also masked. The 'Validity (years)' field is set to 25. The 'Certificate' section is expanded, showing fields for 'First and Last Name' (set to 'Kim Min Seok'), 'Organizational Unit', 'Organization', 'City or Locality', 'State or Province', and 'Country Code (XX)'. At the bottom, the 'OK' button is circled in red, next to a 'Cancel' button.

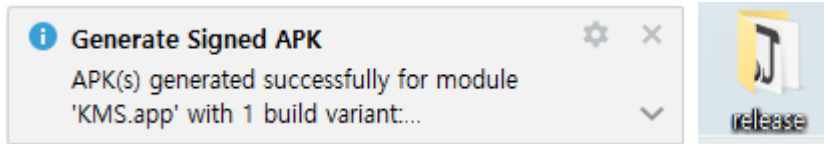
키 저장경로를 설정하고, 패스워드, 별칭(Alias) 등을 설정합니다 certificate 부분은 키의 정보를 입력하는 부분으로, 최소 하나 이상만 필수로 입력해주면 되고, 나머지는 선택 사항입니다.



키를 생성한 후에, 생성한 키에 대한 정보를 입력하고 다음으로 이동합니다.



저장 위치를 설정하고 release, V2(Full APK Signature)를 선택한 후에 Finish 를 누릅니다. 'Signature Help'를 통해 V1 과 V2 의 차이를 더 자세히 알 수 있습니다.



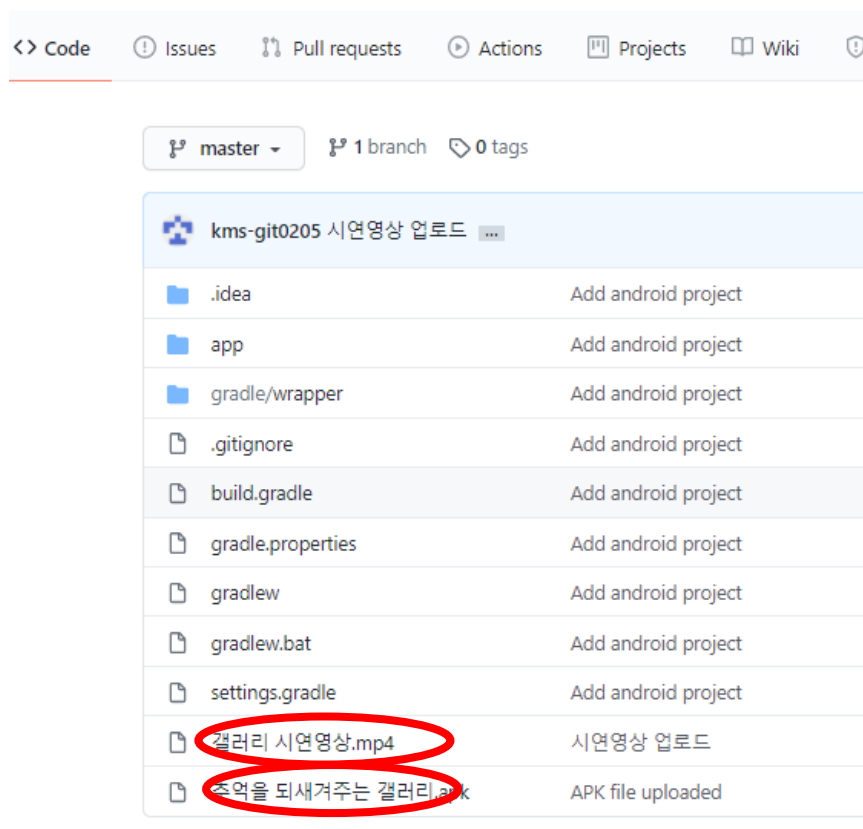
생성이 완료되면 위의 메시지와 함께 release 폴더가 설정한 위치에 생성된 것을 확인할 수 있습니다. release 폴더 안에 있는 apk 파일이 스마트폰에서 실행될 어플리케이션 파일입니다.

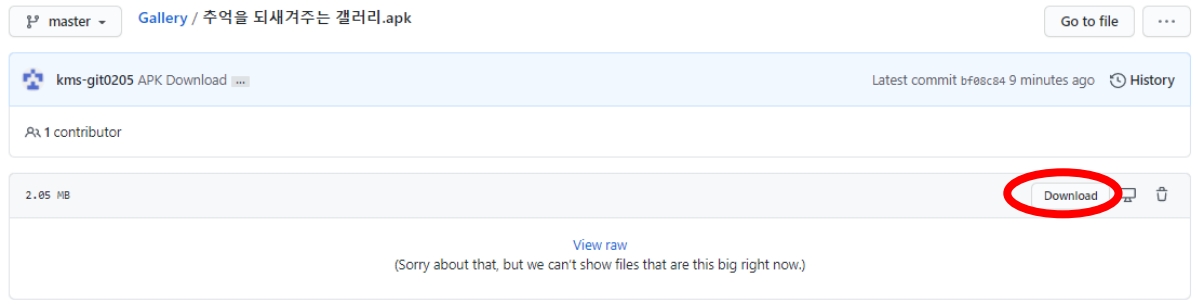
5-2 결과물 공유(Github)

안드로이드 스튜디오의 프로젝트 결과물과 간단한 시연영상을 Github 에 업로드해서 누구나 확인해보고 어플리케이션을 다운받아 볼 수 있도록 하였습니다.

링크 : <https://github.com/kms-git0205/Gallery.git>

아래의 과정을 통해 스마트폰에서 어플리케이션을 실행해볼 수 있습니다.





다운받은 apk 파일을 모바일에서 실행하면 어플리케이션을 테스트할 수 있습니다.

같은 방식으로 갤러리 시연영상을 다운받아서 볼 수 있습니다.