



# Homework 1

---

Game Graphics  
Prof. Hyeong Yeop Kang  
siamiz@khu.ac.kr

# Goal

---

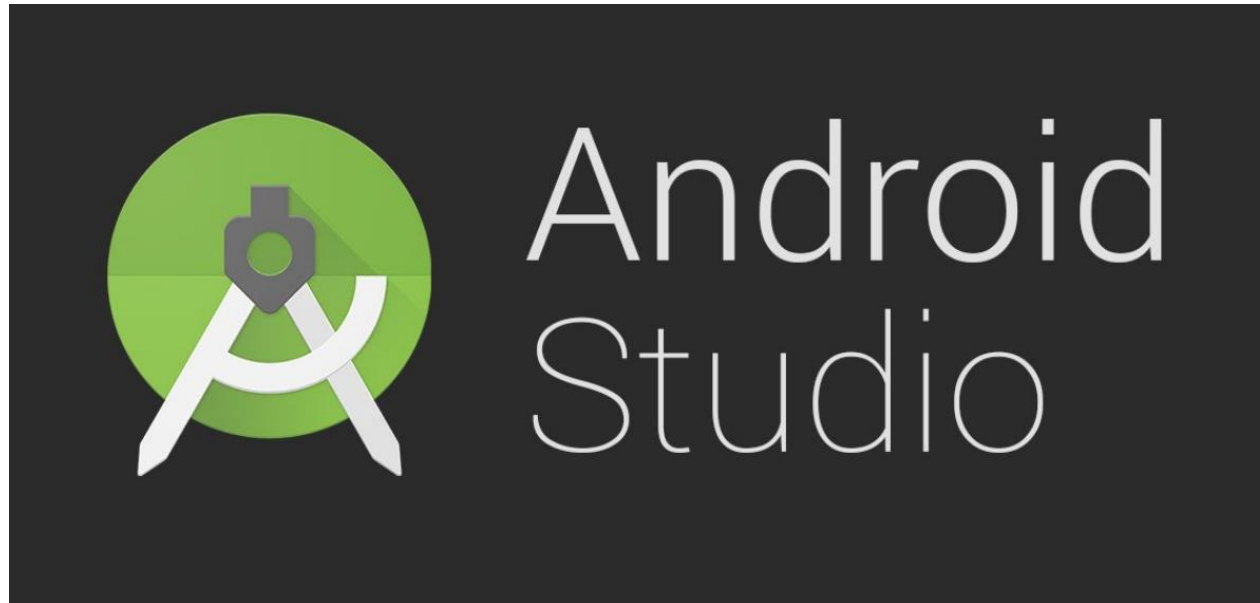


- Setting up Android Studio development environment
- Filling in code lines in the vertex shader.
  - Vertex position
  - Vertex normal
  - Texture coordinates
- Filling in code lines in the functions of the scene class in GL program.
- +10 points homework.

# Android Studio



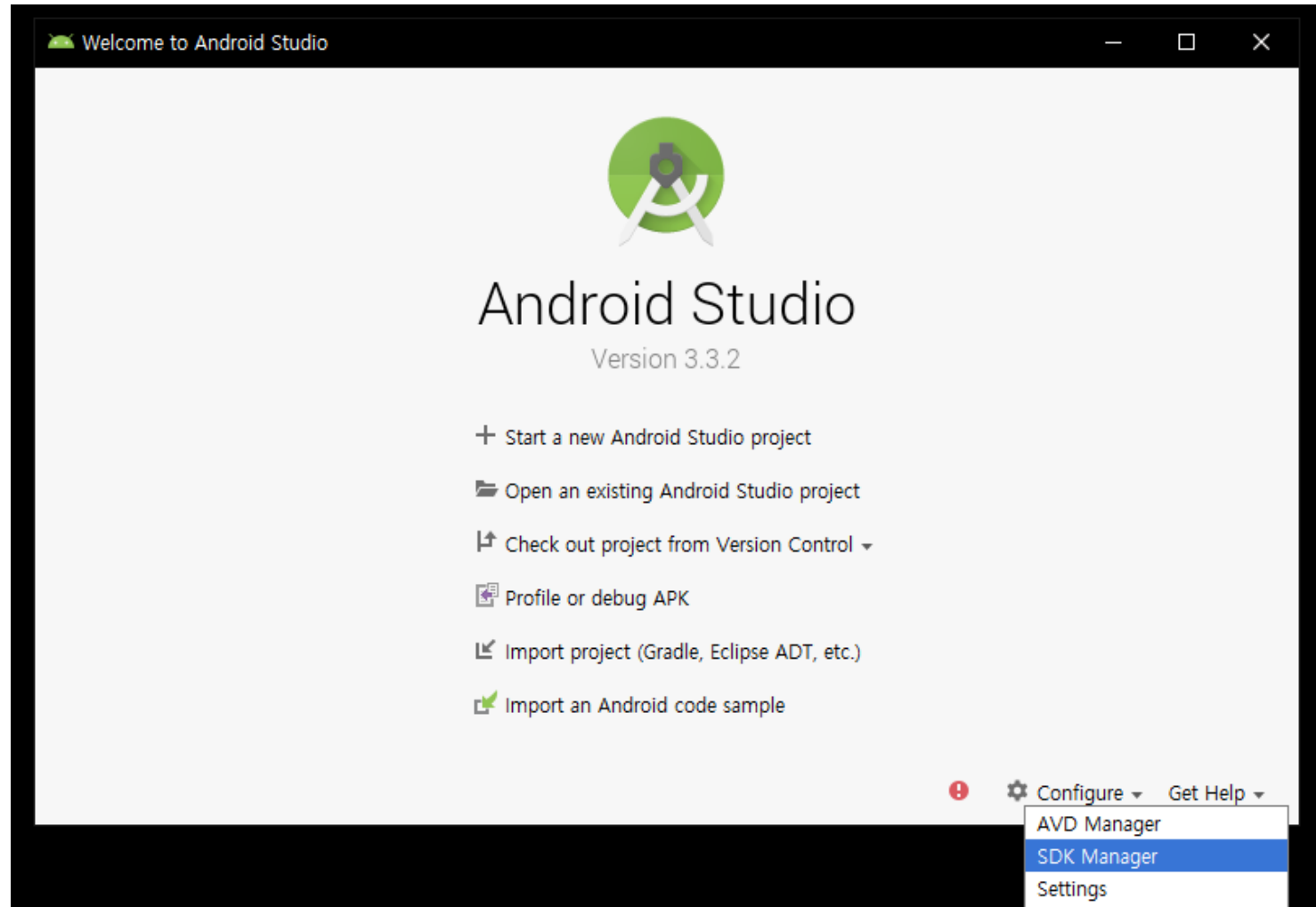
- **Android Studio** is the official integrated development environment (IDE) for the Android platform.
- Android Studio can be downloaded from the official website.



# Android SDK



- **Android SDK** can be installed through the SDK Manager in Android Studio.



# Android SDK



- The **latest platform** (2020.03.26. Android 10.0 Q) will be installed automatically.

SDK Platforms    SDK Tools    SDK Update Sites

Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, Android Studio will automatically check for updates. Check "show package details" to display individual SDK components.

	Name	API Level	Revision	Status
<input type="checkbox"/>	Android R Preview	R	2	Not installed
<input checked="" type="checkbox"/>	Android 10.0 (Q)	29	3	Update available
<input checked="" type="checkbox"/>	Android 9.0 (Pie)	28	6	Installed
<input checked="" type="checkbox"/>	Android 8.1 (Oreo)	27	3	Installed
<input checked="" type="checkbox"/>	Android 8.0 (Oreo)	26	2	Installed
<input type="checkbox"/>	Android 7.1.1 (Nougat)	25	3	Not installed
<input type="checkbox"/>	Android 7.0 (Nougat)	24	2	Not installed
<input type="checkbox"/>	Android 6.0 (Marshmallow)	23	3	Not installed
<input type="checkbox"/>	Android 5.1 (Lollipop)	22	2	Not installed
<input type="checkbox"/>	Android 5.0 (Lollipop)	21	2	Not installed
<input type="checkbox"/>	Android 4.4W (KitKat Wear)	20	2	Not installed
<input type="checkbox"/>	Android 4.4 (KitKat)	19	4	Not installed
<input type="checkbox"/>	Android 4.3 (Jelly Bean)	18	3	Not installed
<input type="checkbox"/>	Android 4.2 (Jelly Bean)	17	3	Not installed
<input type="checkbox"/>	Android 4.1 (Jelly Bean)	16	5	Not installed
<input type="checkbox"/>	Android 4.0.3 (IceCreamSandwich)	15	5	Not installed
<input type="checkbox"/>	Android 4.0 (IceCreamSandwich)	14	4	Not installed
<input type="checkbox"/>	Android 3.2 (Honeycomb)	13	1	Not installed
<input type="checkbox"/>	Android 3.1 (Honeycomb)	12	3	Not installed
<input type="checkbox"/>	Android 3.0 (Honeycomb)	11	2	Not installed
<input type="checkbox"/>	Android 2.3.3 (Gingerbread)	10	2	Not installed
<input type="checkbox"/>	Android 2.3 (Gingerbread)	9	2	Not installed
<input type="checkbox"/>	Android 2.2 (Froyo)	8	3	Not installed

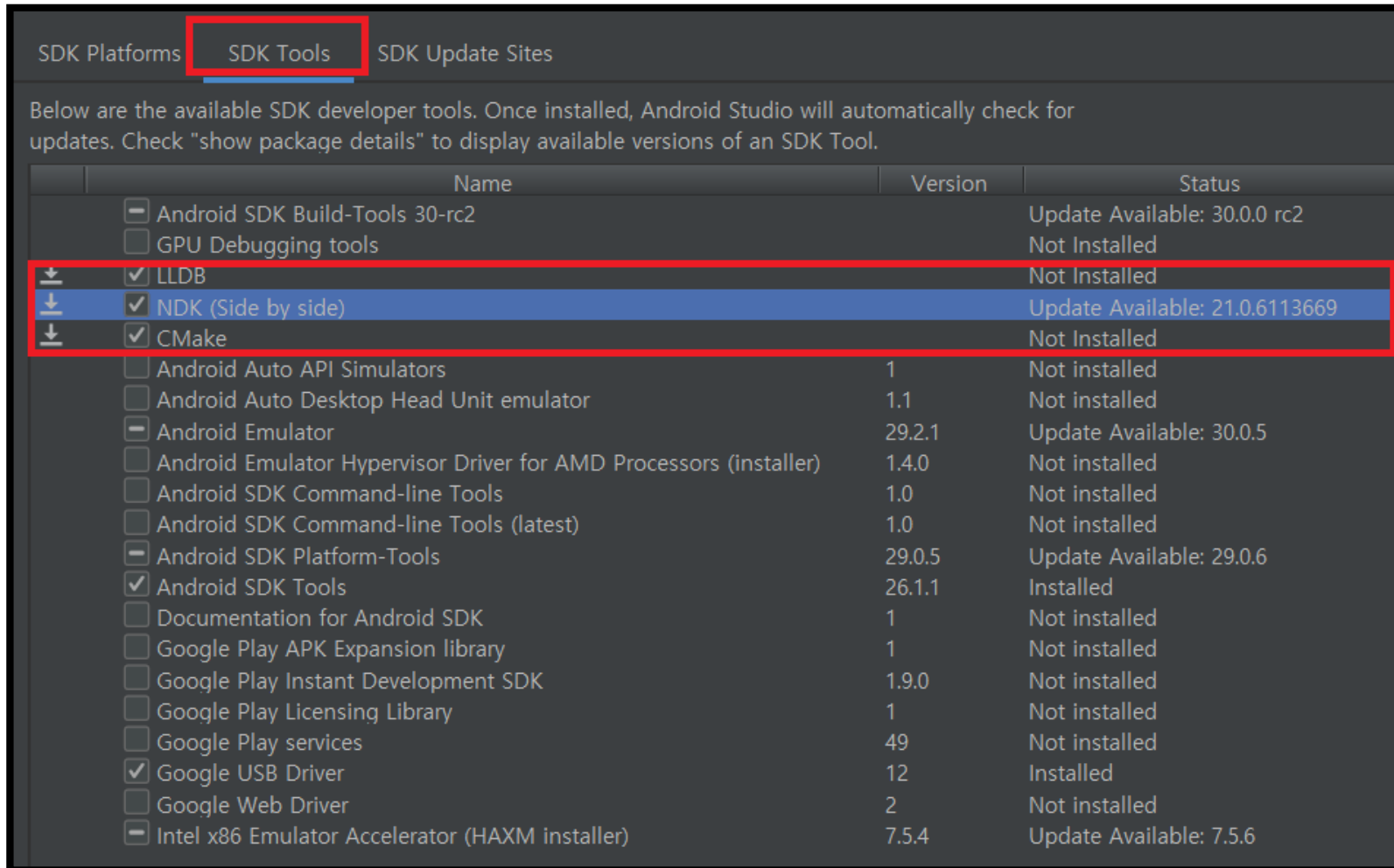
The image and the actual screen can be different.

# Android SDK



To use C++ native language on Android, you need to install the following three tools.

- CMake
- LLDB
- NDK

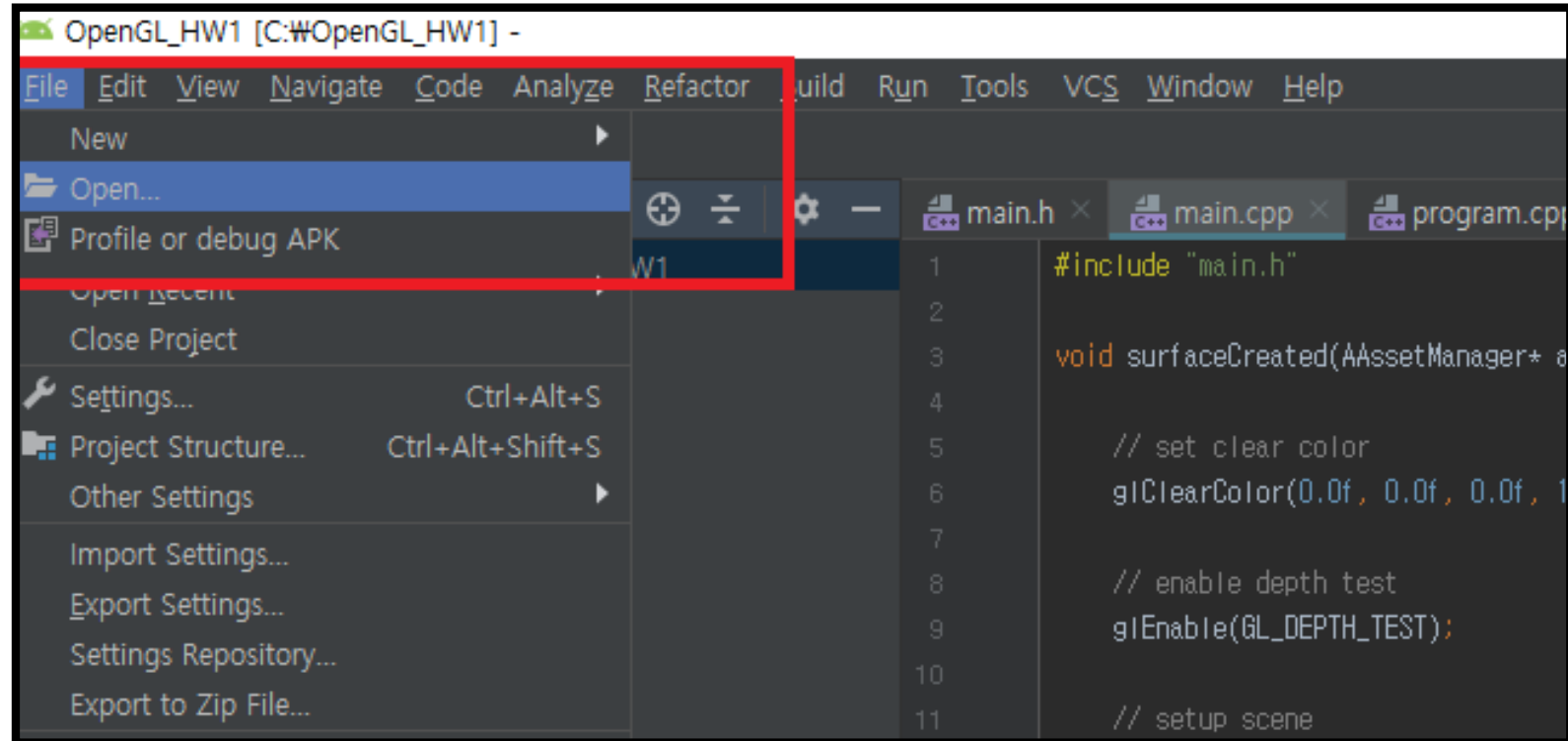


The image and the actual screen can be different.

# Open Project Folder



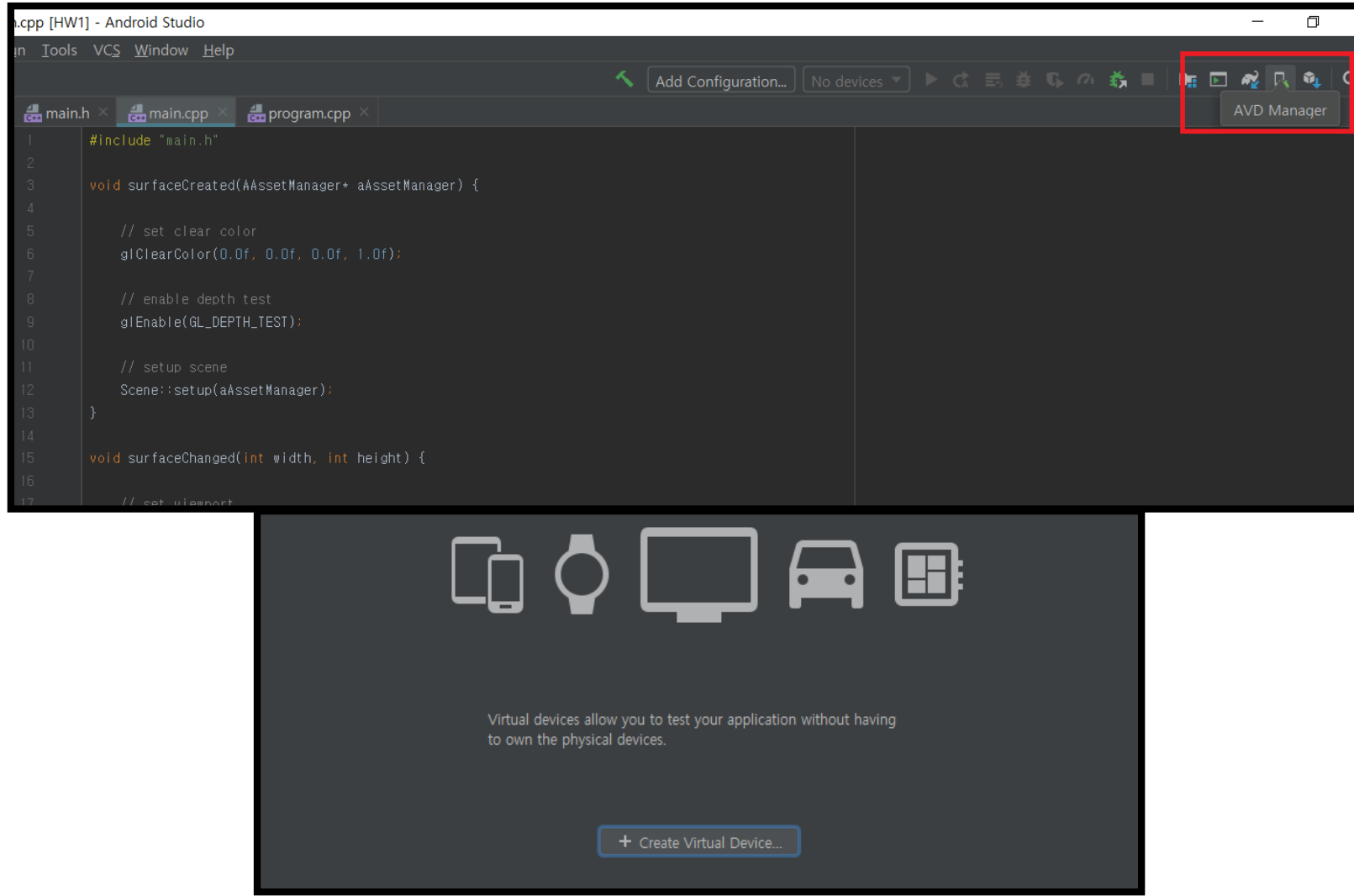
Click the [File] – [Open].



# Create Virtual Device



- After opening the project, let's create virtual device.
- Click the AVD Manager and create virtual device.



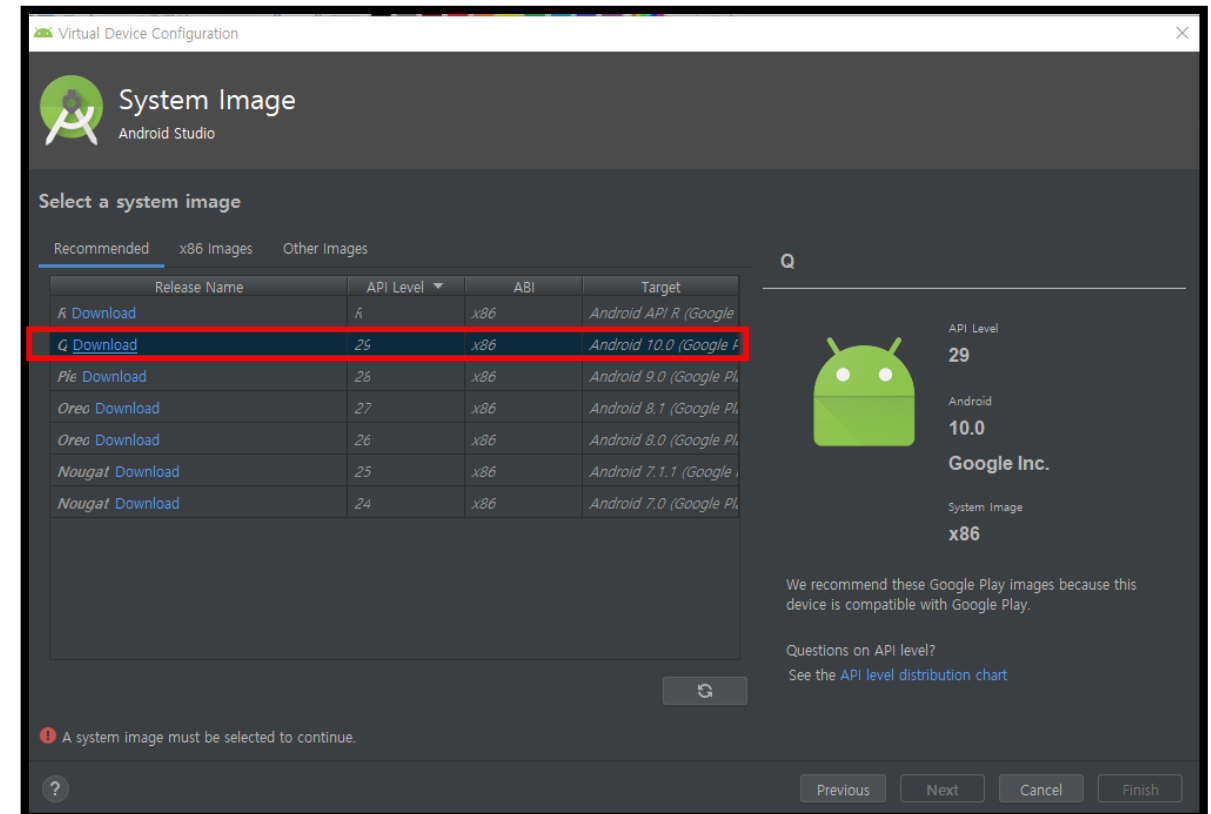
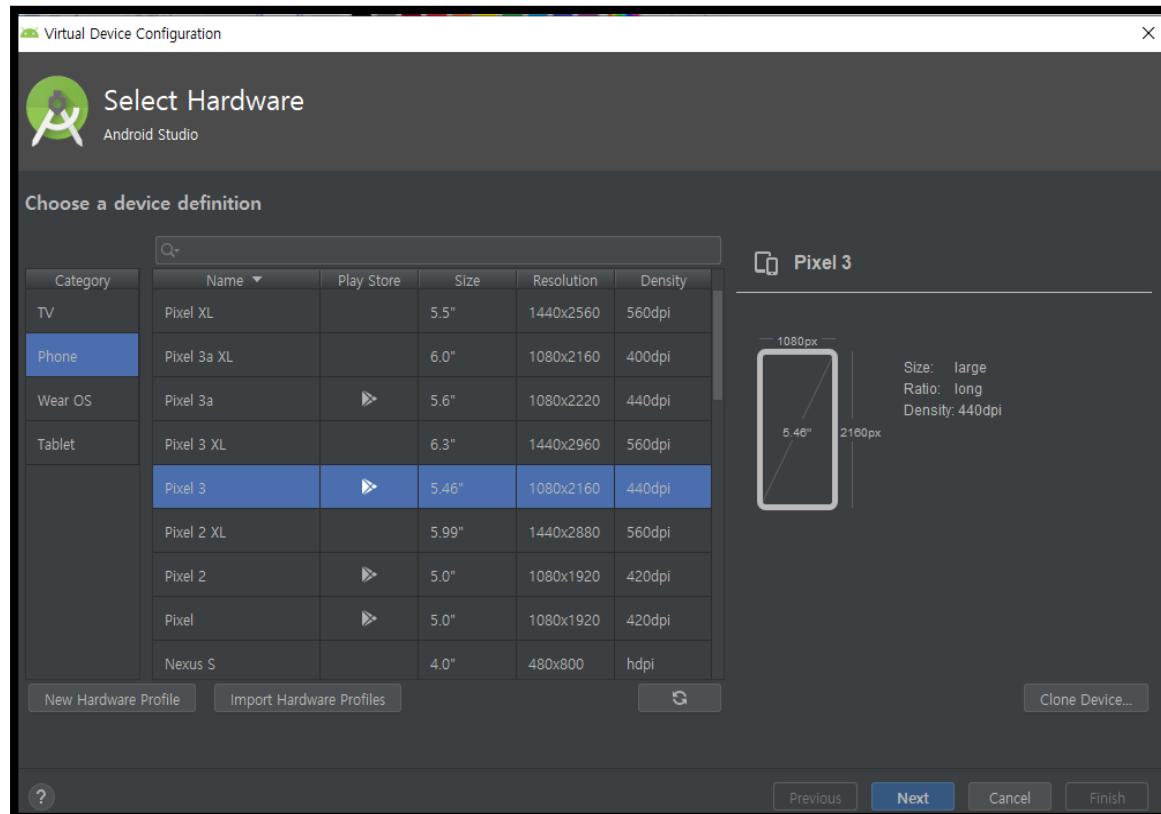


# Create Virtual Device



Choose the device you prefer. (In my case, Pixel 3 with system Q.)

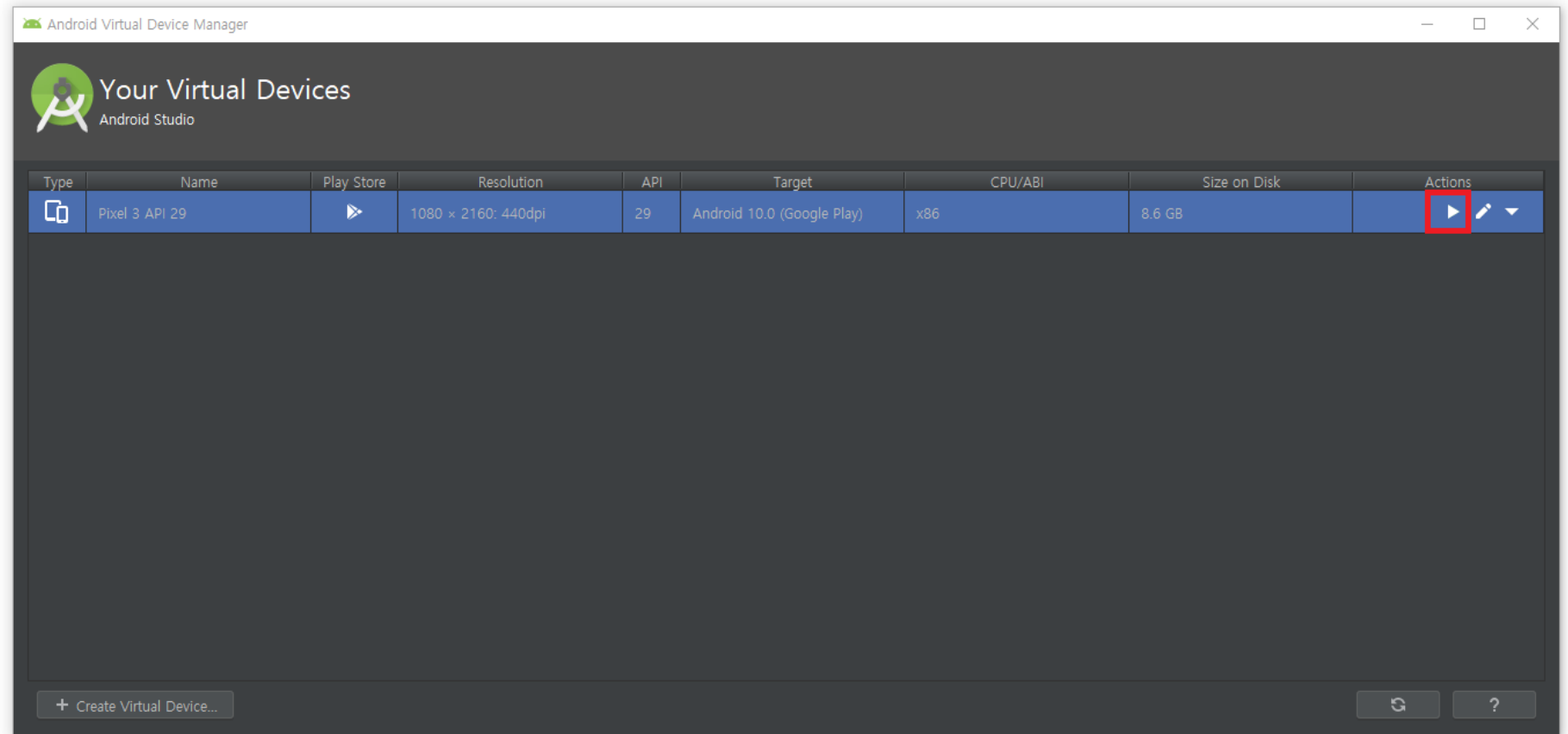
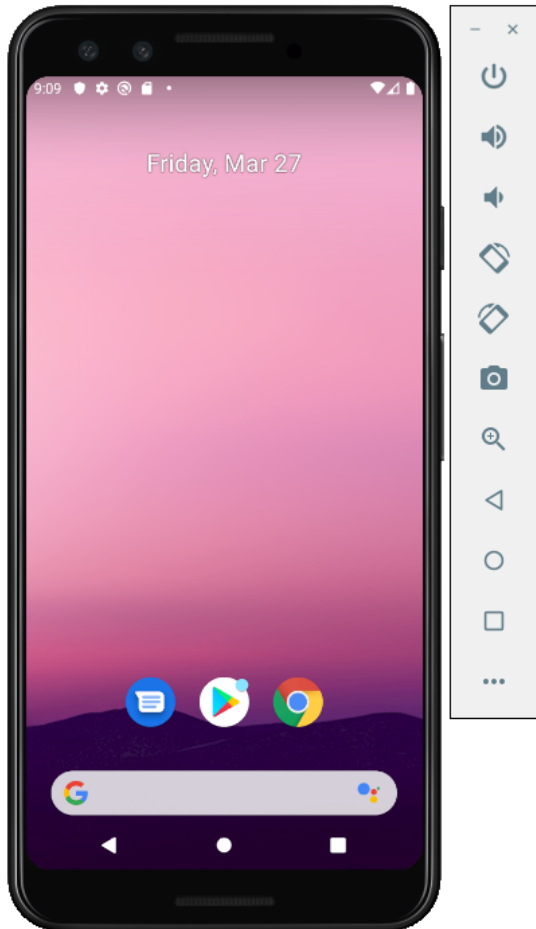
- If required, download the API.



# Create Virtual Device



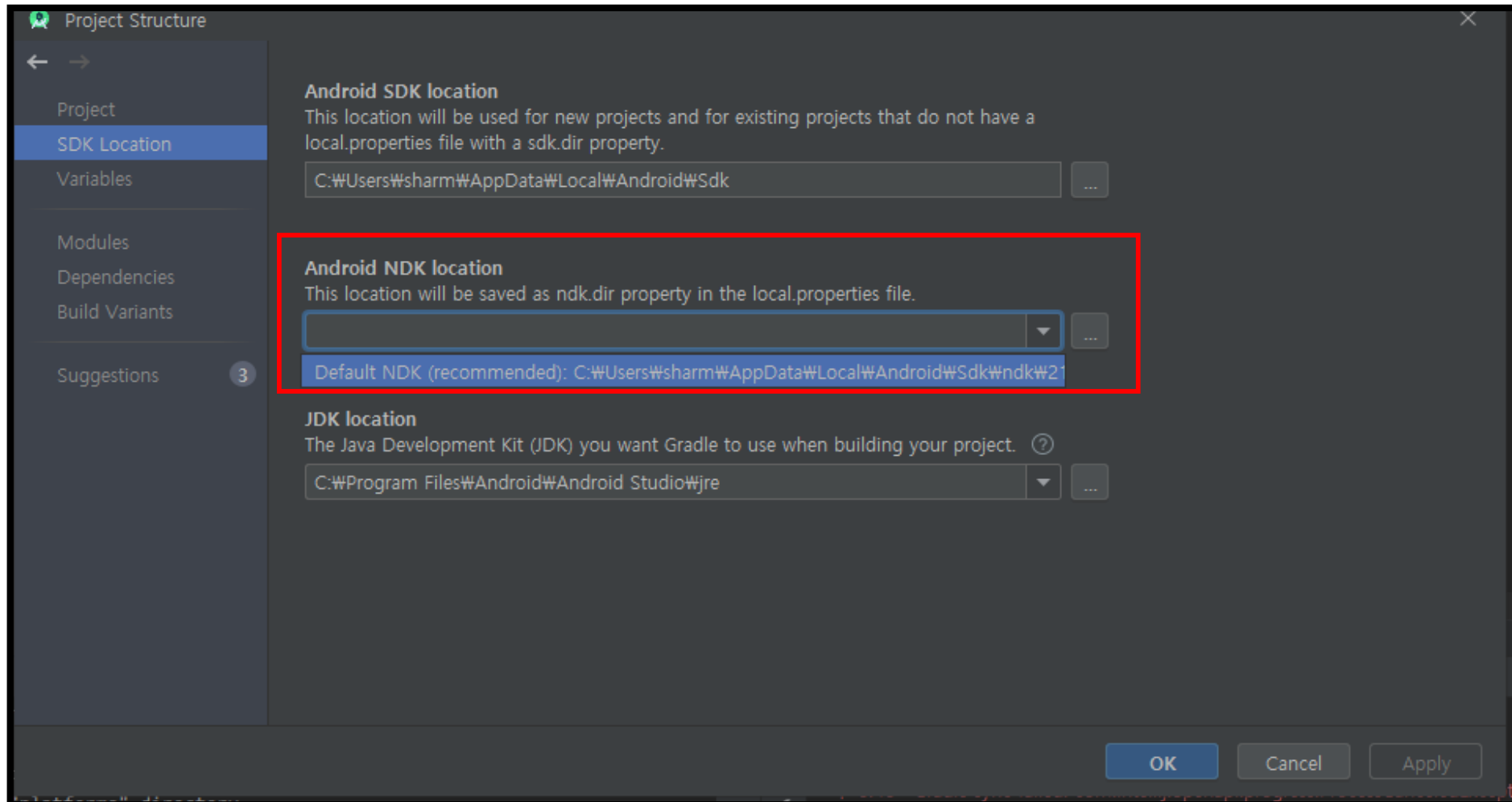
After finishing the creation, click the launch button on the Actions tab. Then, the virtual device shown on the left will appear.



# NDK Location



Set a proper Android NDK location. (It varies depending on your computer setting!)



# Ready?



Then click the [Run] – [Run] (or green triangle shown below.)

- If required, several installation process will be performed (It takes several minutes.)

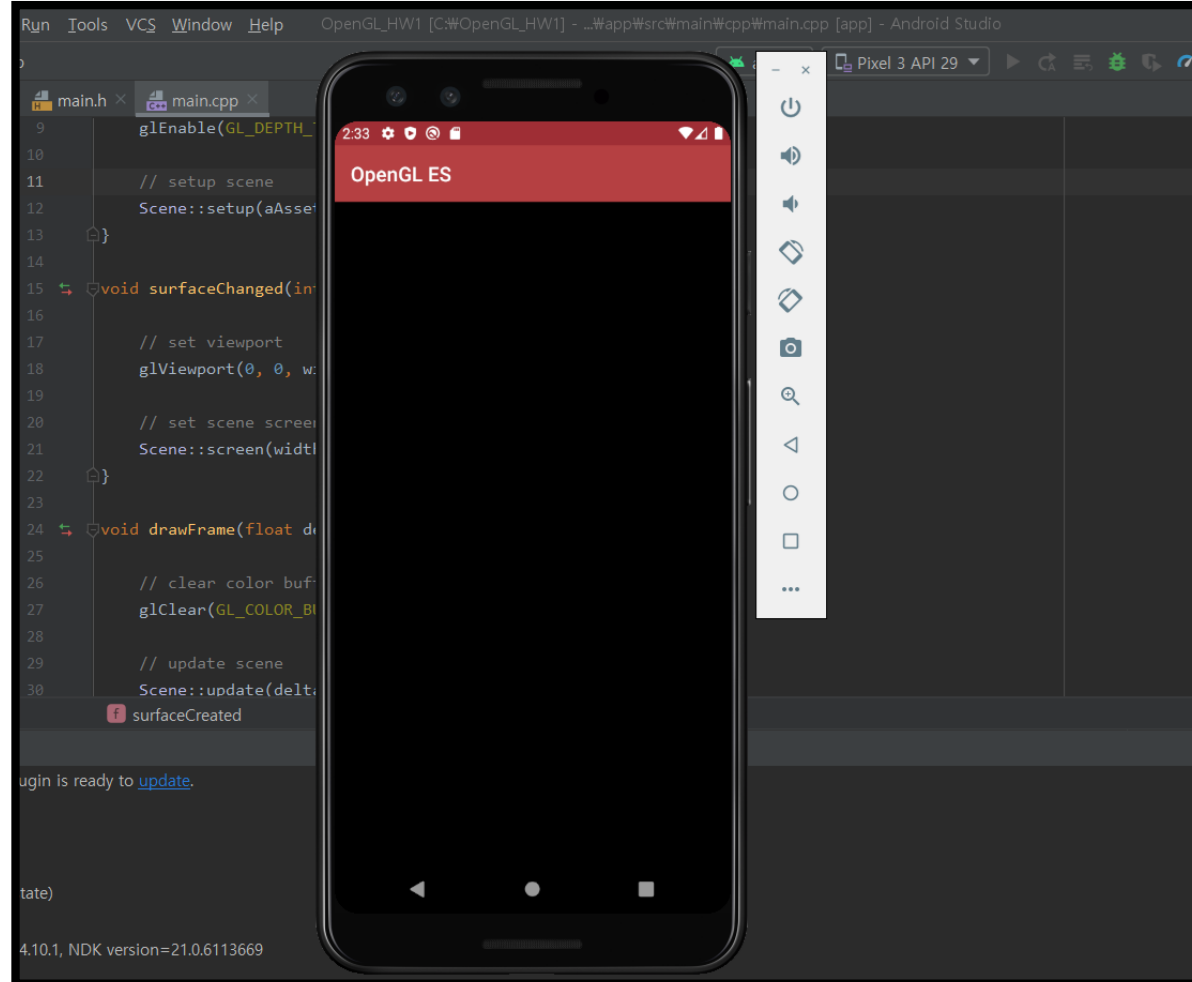
```
Run  Tools  VCS  Window  Help  OpenGL_HW1 [C:\OpenGL_HW1] - ...Wapp\src\main\cpp\main.cpp [app] - Android Studio

main.h x  main.cpp x
9      glEnable(GL_DEPTH_TEST);
10
11     // setup scene
12     Scene::setup(aAssetManager);
13 }
14
15 void surfaceChanged(int width, int height) {
16
17     // set viewport
18     glViewport(0, 0, width, height);
19
20     // set scene screen
21     Scene::screen(width, height);
22 }
23
24 void drawFrame(float deltaTime) {
25
26     // clear color buffer and depth buffer
27     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
28
29     // update scene
30     Scene::update(deltaTime);
31 }
32
33 surfaceCreated
```

# Ready?



The emulator shown bellow will appear. Now, you are ready.



# Problem



Fill in lines.

- App/src/main/cpp/scene.cpp (setup and update)
- App/src/main/assets/vertex.glsl (main)

```
camera->eye = vec3(60.0f, 00.0f, 0.0f);

// create light
light = new Light(program);
light->position = vec3(100.0f, 0.0f, 0.0f);

// create floral texture
flower = new Material(program, texFlowerData, texFlowerSize);

// create teapot object
teapot = new Object(program, flower, objTeapotVertices, objTeapotIndices,
                    objTeapotVerticesSize, objTeapotIndicesSize);
//Set world matrix
//teapot->worldMatrix;

void Scene::screen(int width, int height) {

    // set camera aspect ratio
    camera->aspect = (float) width / height;
}
```

```
void Scene::update(float deltaTime) {
    static float angle = 0.0f;

    // use program
    program->use();

    // rotate the camera relative to the object
    // camera->eye = ;

    // setup camera and light
    camera->setup();
    light->setup();

    // draw teapot
    teapot->draw();
}
```

```
uniform vec3 lightPos;

layout(location = 0) in vec3 position;
layout(location = 1) in vec3 normal;
layout(location = 2) in vec2 texCoord;

out vec3 v_normal;
out vec2 v_texCoord;
out vec3 v_lightDir;

void main() {

    // fill in the lines below
    //gl_Position = ;
    //v_normal = ;
    //v_texCoord = ;

    // do not touch below
    vec3 posWS = (worldMat * vec4(position, 1.0)).xyz;
    v_lightDir = normalize(lightPos - posWS);
}
```

# Problem



Goal 1 (3points). Write your code at the `setup()` to transform the teapot as follows:

- Scale the teapot by 1.0, 2.0 and 0.5 along the x, y and z axis respectively.
- Rotate the teapot by  $50^\circ$  around the axis, (1, 1, 0).

Goal 2 (3points). Write your code at the `update()` to rotate the camera around the y-axis

- Rotate the teapot by  $2^\circ$  for every update.

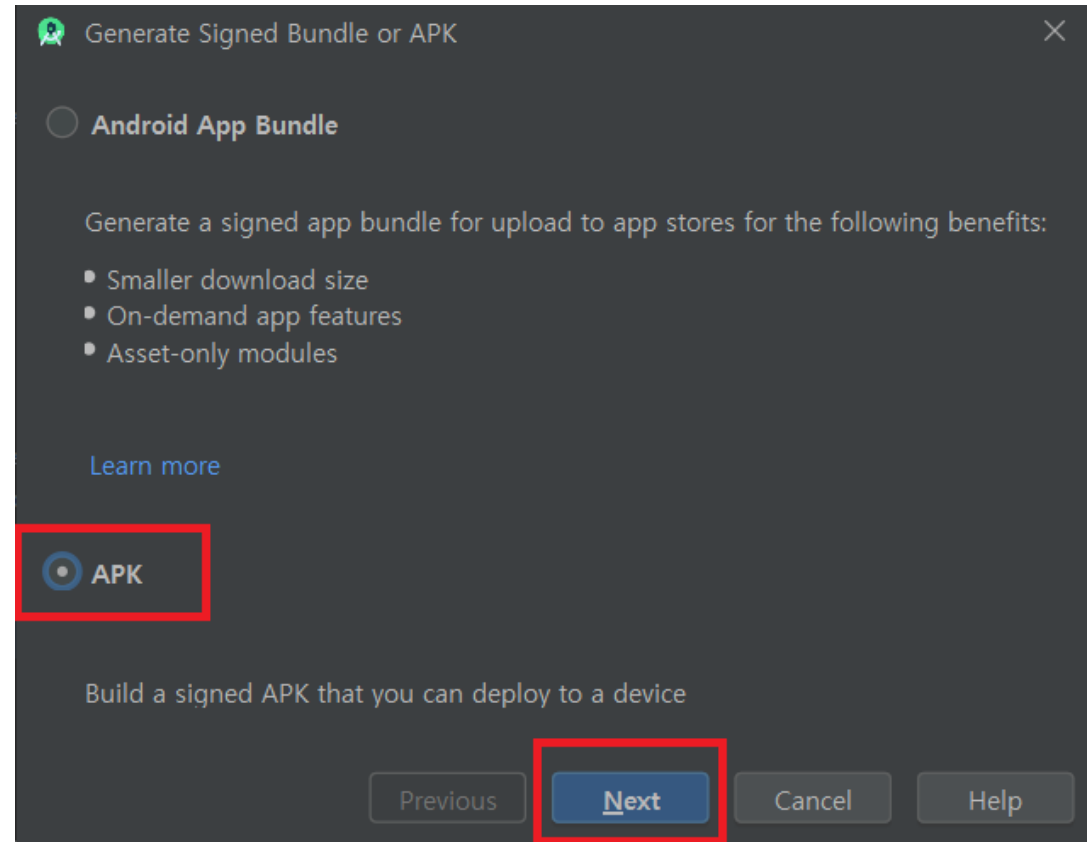
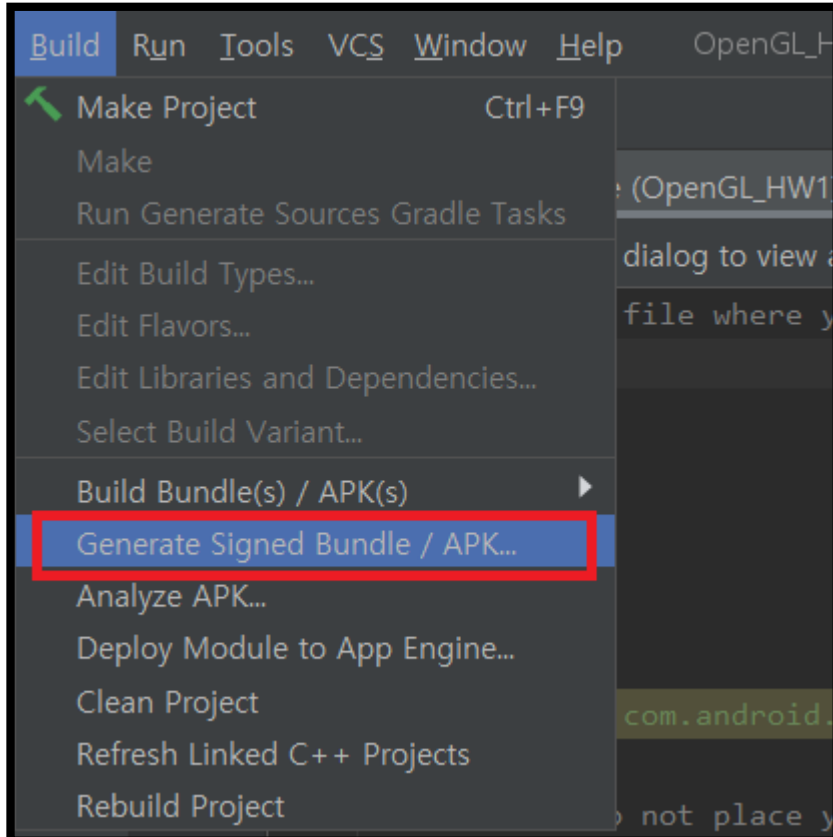
Goal 3 (4points). Write your code at the `.glsl main()`.

- The clip-space vertex position must be calculated.
- The world-space vertex normal must be calculated accurately considering a non-uniform scaling (note that the lighting will look weird if vertex normals are incorrect.)
- The texture coordinates will be sent to the rasterizer stage with no change.

Note: do not use `rotate()` function!



# Generate APK





# Generate APK



Generate Signed Bundle or APK

Module: app

Key store path:

Key store password:

Key alias:

Key password:

☐ Remember passwords

New Key Store

Key store path: C:\Users\sharm\HW1Key.jks

Password:  Confirm:

Key:

Alias: key0

Password:  Confirm:

Validity (years): 25

Certificate

First and Last Name: Siamiz

Organizational Unit:

Organization:

City or Locality:

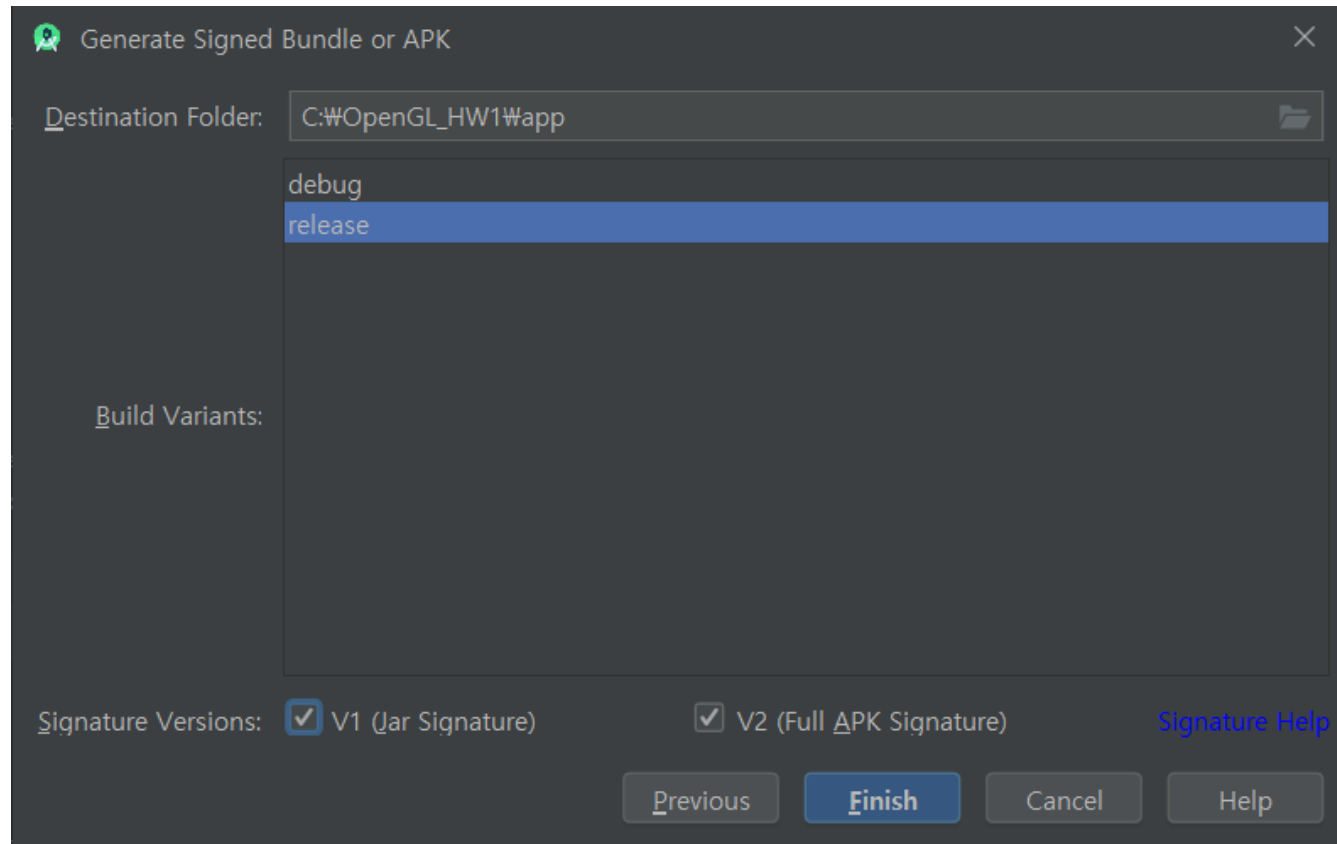
State or Province:

Country Code (XX):

# Generate APK



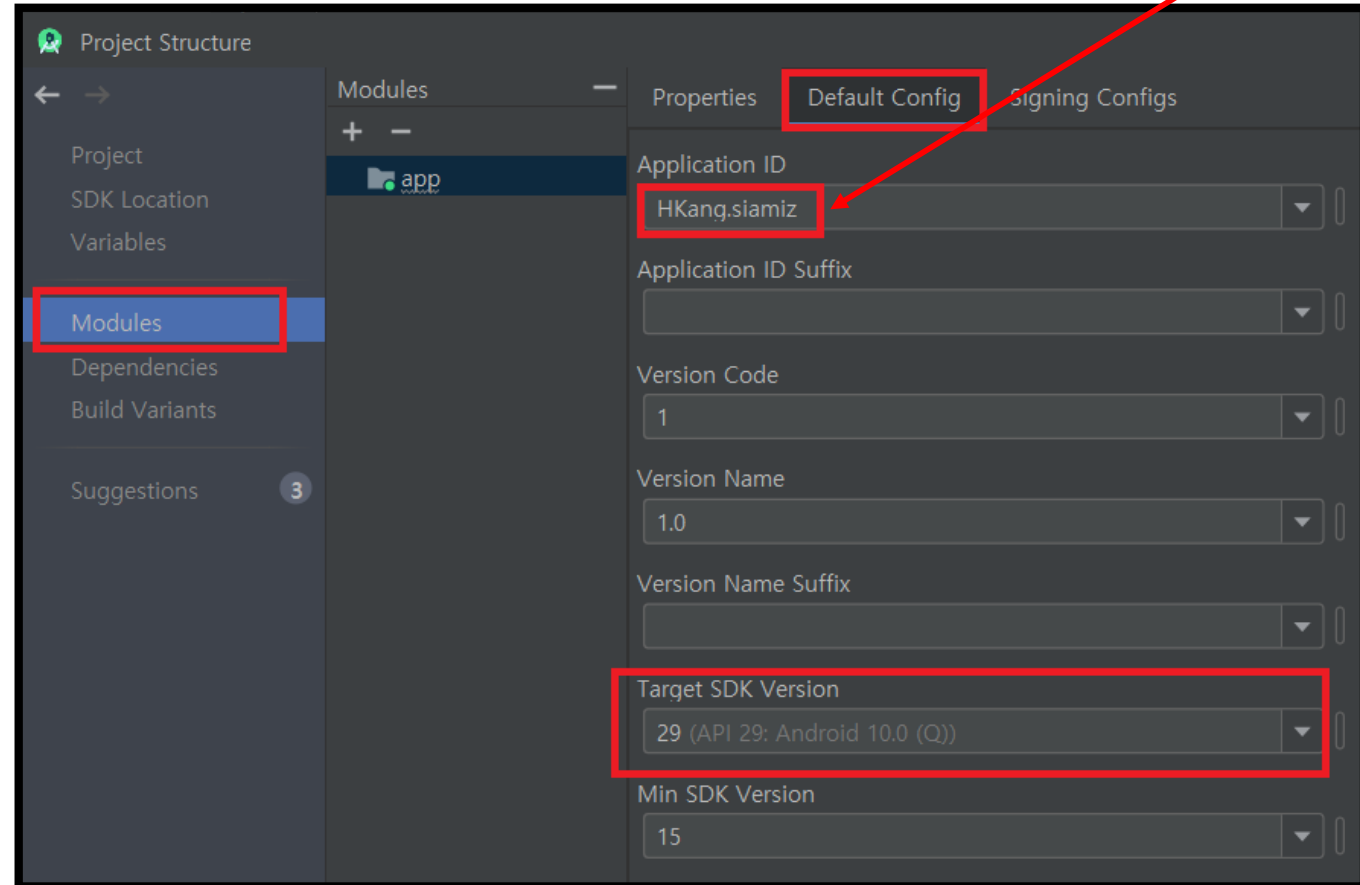
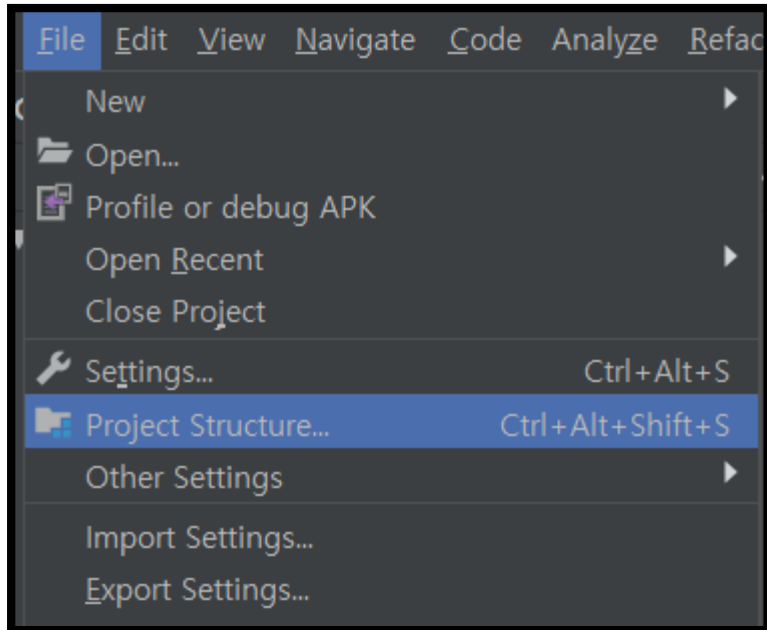
You have an error? Google may help you to generate APK 😊



# Project Setting



Add your student\_number  
to the end of the ID



app-release.apk

output.json

2020-03-30 오후 9:24

압축(APK) 파일

1,755KB

2020-03-30 오후 9:24

JSON 파일

1KB

**Upload these to your git repository.**

# Submission



## Deadline

- 4. 24. 23:50 (important!)

## Submit followings to klas.

- Make an apk file and upload it to your git repository. Then, share your git URL.
  - ✓ APK does not work: 0 point!
- {student\_number}\_{name}.zip including vertex.glsl and scene.cpp.
  - ✓ does not submit code: 0 point!

## TA

- 이정은 (jeunlee0306@khu.ac.kr),

## Office hour

- Monday, Wednesday 2:00 PM ~ 5:00 PM
- Contact TA by email before you visit.