



# Homework 2

---

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

# Setting



Please see the setting procedure presented in homework 1.

- Please install both versions (10.0 and 9.0)

Settings

Appearance & Behavior > System Settings > Android SDK

Manager for the Android SDK and Tools used by Android Studio

Android SDK Location: C:\Users\sharm\AppData\Local\Android\Sdk [Edit](#) [Optimize disk space](#)

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.

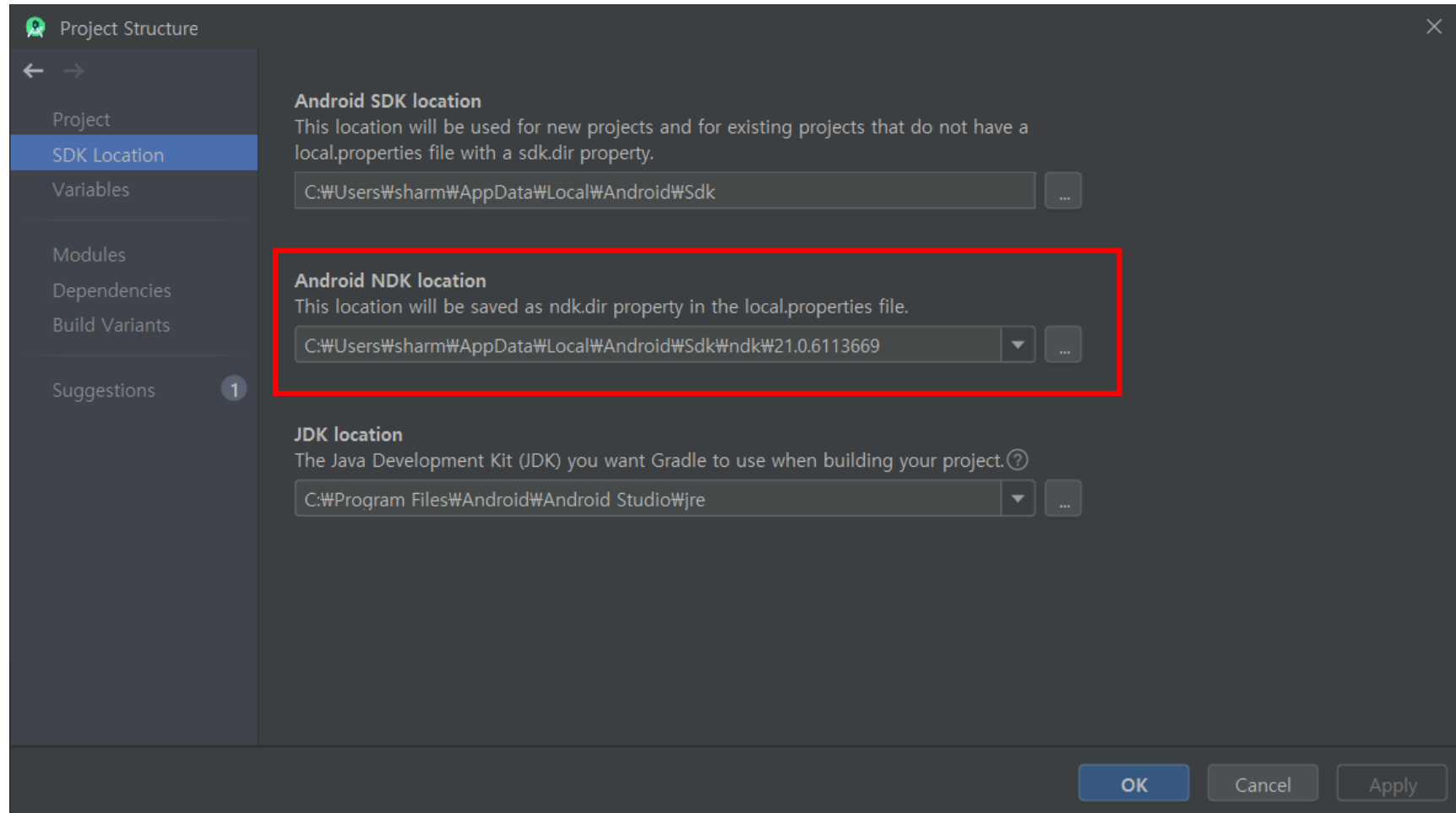
	Name	API Level	Revision	Status
<input type="checkbox"/>	Android R Preview	R	3	Not installed
<input checked="" type="checkbox"/>	Android 10.0 (Q)	29	4	Installed
<input checked="" type="checkbox"/>	Android 9.0 (Pie)	28	6	Installed
<input checked="" type="checkbox"/>	Android 8.1 (Oreo)	27	3	Installed
<input 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

# Setting



Please see the setting procedure presented in homework 1.

- Check your NDK path. (Figure shows my NDK path.)

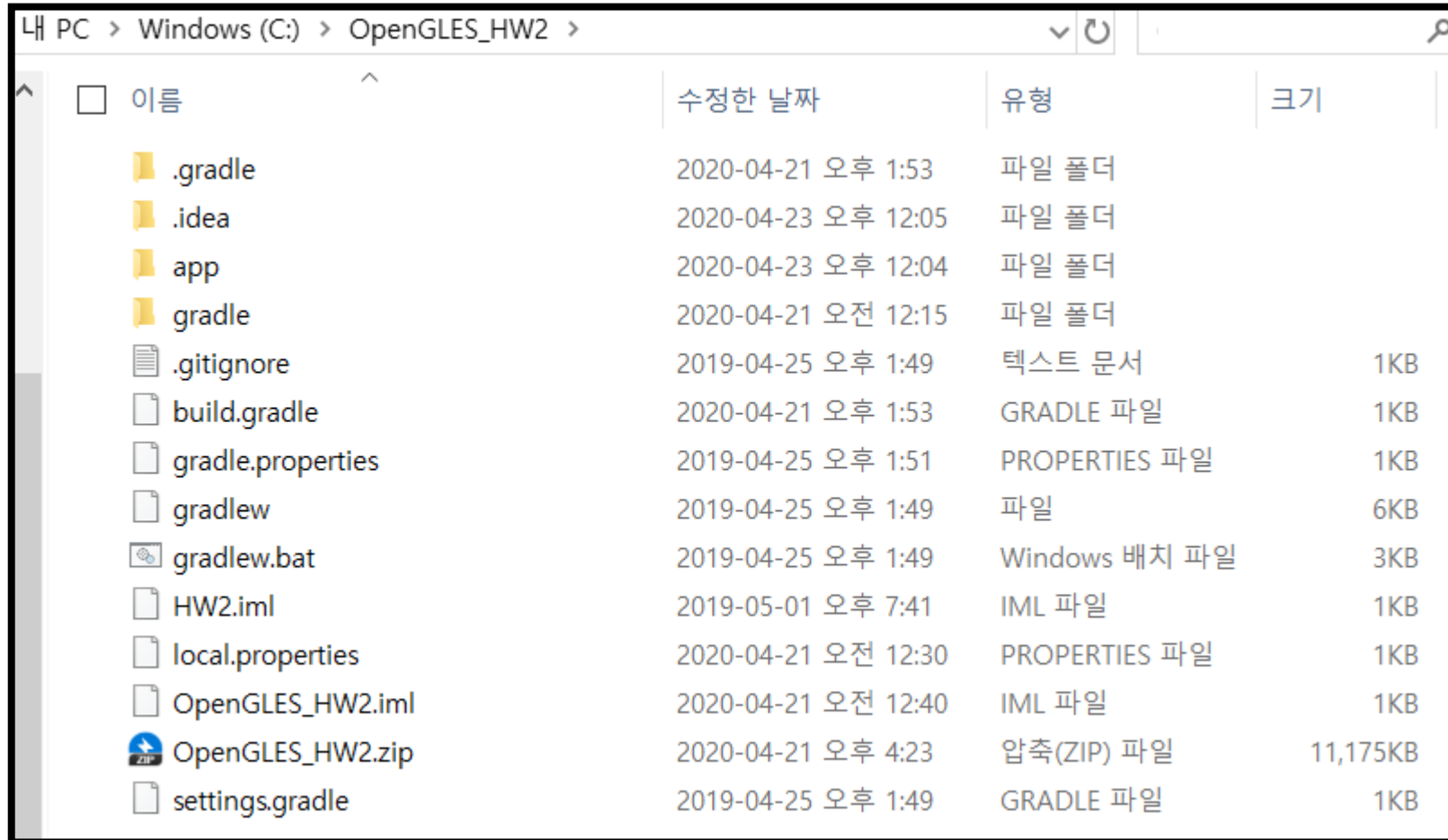


# Setting



Please see the setting procedure presented in homework 1.

- Following figure shows the root folder.



이름	수정한 날짜	유형	크기
.gradle	2020-04-21 오후 1:53	파일 폴더	
.idea	2020-04-23 오후 12:05	파일 폴더	
app	2020-04-23 오후 12:04	파일 폴더	
gradle	2020-04-21 오전 12:15	파일 폴더	
.gitignore	2019-04-25 오후 1:49	텍스트 문서	1KB
build.gradle	2020-04-21 오후 1:53	GRADLE 파일	1KB
gradle.properties	2019-04-25 오후 1:51	PROPERTIES 파일	1KB
gradlew	2019-04-25 오후 1:49	파일	6KB
gradlew.bat	2019-04-25 오후 1:49	Windows 배치 파일	3KB
HW2.iml	2019-05-01 오후 7:41	IML 파일	1KB
local.properties	2020-04-21 오전 12:30	PROPERTIES 파일	1KB
OpenGLES_HW2.iml	2020-04-21 오전 12:40	IML 파일	1KB
OpenGLES_HW2.zip	2020-04-21 오후 4:23	압축(ZIP) 파일	11,175KB
settings.gradle	2019-04-25 오후 1:49	GRADLE 파일	1KB

# HW2 Goal

---

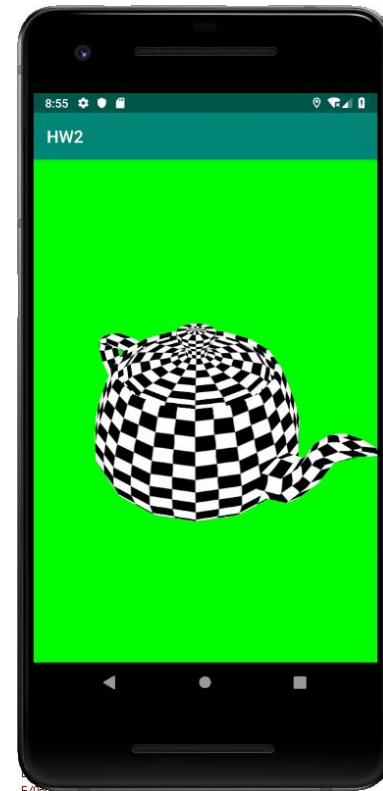


- 1) To change the texture (+2 points)
  - Load PNG file (down\_pusheen.png) and applied this to the teapot. (+0.5 points)
- 2) To implement Phong lighting with 2 point lights (+2.5 points)
- 3) To implement vertex displacement (+2.5 points)
- 4) To implement alpha blending using an extra texture (+2.5 points)

# Problem 1



Change the texture.



# Problem 1



In texture.cpp, you can find the texture setting.

- The data type is an 8-bit unsigned integer.

```
void Texture::load(const vector<Texel> &data, const GLsizei size) {  
    LOG_PRINT_DEBUG("Load texture data");  
  
    glBindTexture(GL_TEXTURE_2D, id);  
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, size, size, 0, GL_RGB, GL_UNSIGNED_BYTE, data.data());  
    glGenerateMipmap(GL_TEXTURE_2D);  
}
```

- Therefore, you can use vector<Texel> to store the texture data. Texel is a structure defined in the texel.h.

```
struct Texel {  
    GLubyte red;  
    GLubyte green;  
    GLubyte blue;  
};
```

# Problem 1



The texture data must be in an 1D array of texels.

- In the floral.h, the 1D array of floralTexels is defined.

```
GLuint floralSize = 128;  
vector<Texel> floralTexels = {  
    {0xC4, 0xC3, 0x92}, {0xAC, 0xA8, 0x5F}, {0xB7, 0xAF, 0x65}, {0xCA, 0xBC, 0x75},  
    {0xD7, 0xC6, 0x80}, {0xB9, 0xAF, 0x64}, {0xB7, 0xB5, 0x7B}, {0xFF, 0xFF, 0xFF},  
    {0xFF, 0xFF, 0xFF}, {0xEB, 0xE9, 0xD1}, {0xBD, 0xB8, 0x78}, {0xA6, 0x9F, 0x4E},  
    {0x92, 0x91, 0x49}, {0x6F, 0x73, 0x32}, {0x7E, 0x80, 0x3F}, {0xB7, 0xB8, 0x8C},  
    {0xFF, 0xFF, 0xFF}, {0xF0, 0xE6, 0xC3}, {0xDC, 0xCA, 0x86}, {0xE8, 0xE0, 0xBE},  
    {0xEE, 0xE4, 0xC4}, {0xEB, 0xE2, 0xBF}, {0xFB, 0xFA, 0xF4}, {0xFF, 0xFF, 0xFF},  
    {0xFF, 0xFF, 0xFE}, {0xFF, 0xFF, 0xFF}, {0xE5, 0xE2, 0xCC}, {0xFA, 0xFA, 0xF3},  
    {0xFD, 0xF8, 0xFE}, {0xF1, 0xD5, 0xEF}, {0xF1, 0xC7, 0xE6}, {0xEF, 0xC1, 0xDB},  
    {0xD2, 0xBA, 0xBA}, {0xFB, 0xEF, 0xF9}, {0xF7, 0xE6, 0xF5}, {0xF7, 0xE5, 0xF5},  
    {0xF8, 0xEE, 0xF8}, {0xF0, 0xDF, 0xF0}, {0xF3, 0xE5, 0xF3}, {0xF9, 0xF3, 0xF9},  
}
```



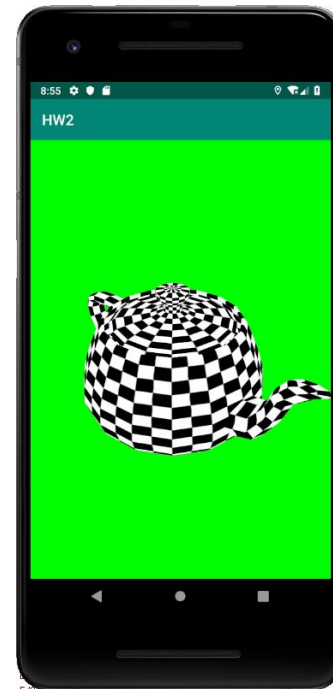
# Problem 1



Use the given texture class, replace the texture data with new one.

- First of all, load the checker.h file.
- Then, change the part of the following code to load the checkerTexel.

```
Scene::diffuse = new Texture(Scene::program, 0, "textureDiff", floralTexels, floralSize);  
// Scene::dissolve = ;  
Scene::material = new Material(Scene::program, diffuse, nullptr);  
Scene::teapot = new Object(program, material, teapotVertices, teapotIndices);
```



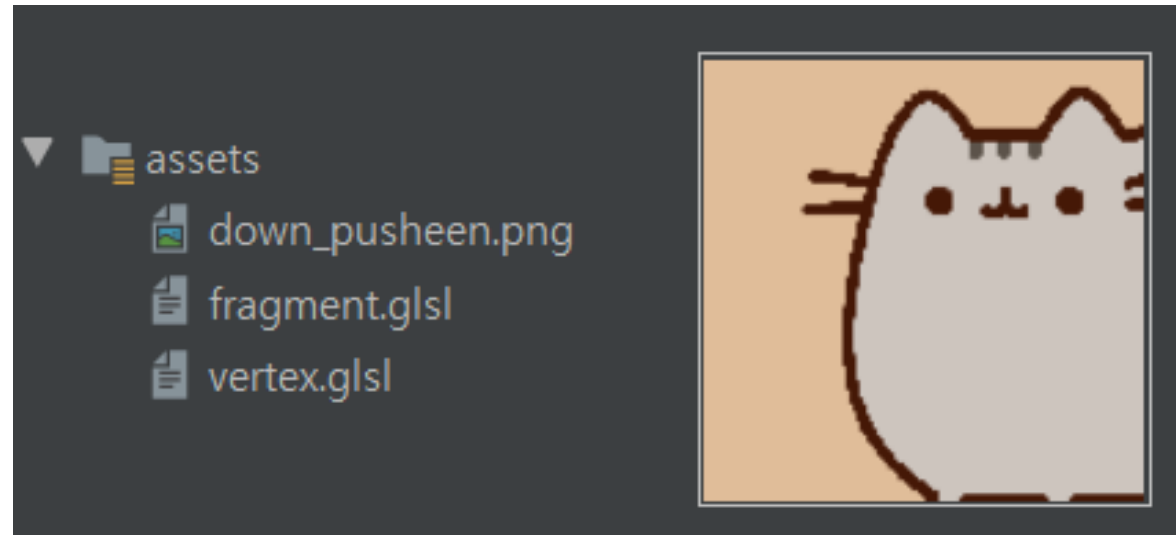
# Problem 1 (optional - advanced)



Load PNG file named `down_pusheen.png`. Then applied this texture to the teapot instead of `checkerTexel`. Then, additional points of **0.5** will be added to your score of HW2.

Hint!

- Method 1) Convert png file to binary file that we used in this system. (Texel structure)
- Method 2) Load png file using external library.

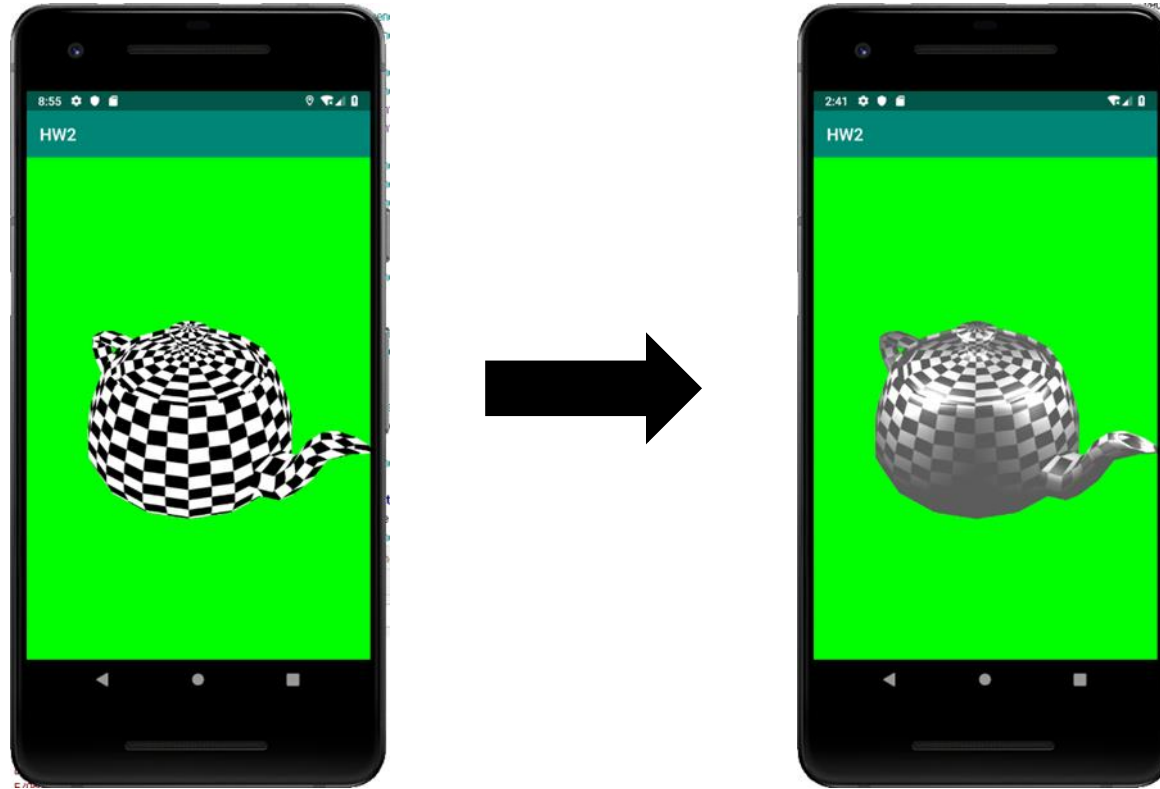


# Problem 2



Implement Phong lighting with 2 point lights.

- Note the specular lighting on the top of the teapot.



# Problem 2



In Scene.cpp, the positions of two light sources are presented.  
In light.cpp, the parameters of two light sources are presented.

```
Scene::lightL = new LeftLight(program);  
Scene::lightR = new RightLight(program);  
lightL->position = vec3(-3.0f, 9.0f, 9.0f);  
lightR->position = vec3(9.0f, 9.0f, -3.0f);
```

```
void RightLight::update() const {  
    Light::update();  
  
    GLint srcDiffLoc = glGetUniformLocation(program->get(), "srcDiffR");  
    GLint srcSpecLoc = glGetUniformLocation(program->get(), "srcSpecR");  
    GLint srcAmbiLoc = glGetUniformLocation(program->get(), "srcAmbiR");  
    GLint lightPosLoc = glGetUniformLocation(program->get(), "lightPosR");  
    GLint lightAttLoc = glGetUniformLocation(program->get(), "lightAttR");  
  
    if (srcDiffLoc >= 0) glUniform3fv(srcDiffLoc, 1, value_ptr(diffuse));  
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "srcDiffR");  
    if (srcSpecLoc >= 0) glUniform3fv(srcSpecLoc, 1, value_ptr(specular));  
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "srcSpecR");  
    if (srcAmbiLoc >= 0) glUniform3fv(srcAmbiLoc, 1, value_ptr(ambient));  
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "srcAmbiR");  
    if (lightPosLoc >= 0) glUniform3fv(lightPosLoc, 1, value_ptr(position));  
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "lightPosR");  
    if (lightAttLoc >= 0) glUniform1fv(lightAttLoc, 3, attenuation);  
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "lightAttR");
```

# Problem 2



In material.cpp, the parameters of material are presented, too.

```
void Material::update() const {
    if (textureDiff) textureDiff->update();
    if (textureDissolve) textureDissolve->update();

    GLint matSpecLoc = glGetUniformLocation(program->get(), "matSpec");
    GLint matAmbiLoc = glGetUniformLocation(program->get(), "matAmbi");
    GLint matEmitLoc = glGetUniformLocation(program->get(), "matEmit");
    GLint matShLoc = glGetUniformLocation(program->get(), "matSh");
    GLint thresholdLoc = glGetUniformLocation(program->get(), "threshold");
    GLint displacementLoc = glGetUniformLocation(program->get(), "displacement");

    if (matSpecLoc >= 0) glUniform3fv(matSpecLoc, 1, value_ptr(specular));
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "matSpec");
    if (matAmbiLoc >= 0) glUniform3fv(matAmbiLoc, 1, value_ptr(ambient));
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "matAmbi");
    if (matEmitLoc >= 0) glUniform3fv(matEmitLoc, 1, value_ptr(emissive));
    else LOG_PRINT_ERROR("Fail to find uniform location: %s", "matEmit");
}
```

# Problem 2



Fill the commented lines in the vertex shader.

```
// view vector
v_view = normalize(eyePos - worldPos);

// light vectors
// v_lightL = ;
// v_lightR = ;

// attenuations
// float distL = ;
// float distR = ;
// v_attL = ;
// v_attR = ;

// texture coordinates
v_texCoord = texCoord;

// clip-space position
gl_Position = projMat * viewMat * vec4(worldPos, 1.0);
```

# Problem 2



Fill the commented lines in the fragment shader.

```
vec3 color = texture(textureDiff, v_texCoord).rgb;

// re-normalize unit vectors (normal, view, and light vectors)
// vec3 normal = ;
// vec3 view = ;
// vec3 lightL = ;
// vec3 lightR = ;

// diffuse term
// vec3 matDiff = ;
// vec3 diffL = ;
// vec3 diffR = ;
// vec3 diff = ;

// specular term
// vec3 reflL = ;
// vec3 reflR = ;
// vec3 specL = ;
// vec3 specR = ;
// vec3 spec = ;
```

# Problem 3



In material.cpp, you can find the displacement parameter.

```
if (displacementLoc >= 0) glUniform1f(displacementLoc, displacement);  
else LOG_PRINT_ERROR("Fail to find uniform location: %s", "displacement");
```

In scene.cpp, you can update displacement and threshold over time.

- Write your code.
- $\text{displacement} = \text{threshold} = |\sin(10 \times \text{radians}(\text{time}))|$

```
void Scene::update(float deltaTime) {  
  
    Scene::program->use();  
  
    // Scene::teapot->material->threshold = ;  
    // Scene::teapot->material->displacement = ;  
  
    Scene::camera->update();  
    Scene::lightL->update();  
    Scene::lightR->update();  
  
    Scene::teapot->draw();  
}
```



# Problem 3



Translate the vertex position in the vertex shader.

- $\text{position} = \text{position} + \text{displacement} \times \text{normal}$

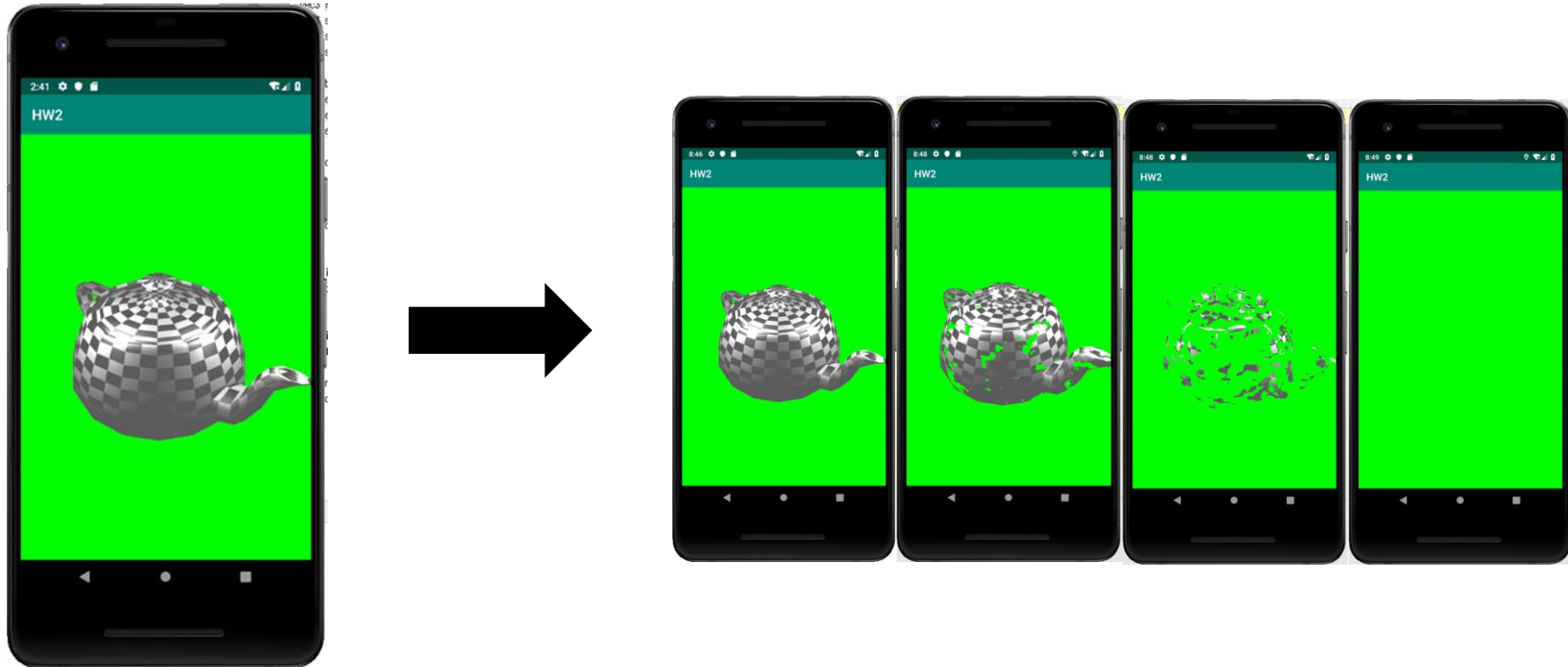
```
void main() {  
  
    // vertex displacement  
    vec3 displacedPos = position;
```

# Problem 4



Implementing alpha blending using an extra texture.

- Make a dissolve effect.



# Problem 4



The dissolving effect uses extra parameters.

- A grayscale texture
- A varying threshold

This effect has a simple logic.

- At the pixel with the texture coordinate  $(u, v)$

$$\alpha = \begin{cases} 1 & \text{threshold} \leq \text{grayscale}(u, v) \\ 0 & \text{otherwise} \end{cases}$$

# Problem 4



Use *threshold* in the material.cpp.

- Remind *threshold* was set in Problem 3.

```
if (thresholdLoc >= 0) glUniform1f(thresholdLoc, threshold);  
else LOG_PRINT_ERROR("Fail to find uniform location: %s", "threshold");
```

Use *textureDissolve* and *material* in the scene.cpp.

- Make the texture of *cloudTexels*.
- Note that this shader uses multiple textures.

```
// Scene::dissolve = ;  
Scene::material = new Material(Scene::program, diffuse, nullptr);
```

# Problem 4



Fill the commented lines in the fragment shader.

```
// dissolving
// float dissolve = ;
// if (dissolve < threshold)
//     alpha = 0.0;
```

Enable alpha blending to make the teapot transparent.

- Specify the blending function:  $\alpha_{dst} = 1 - \alpha_{src}$

```
void surfaceCreated(AAssetManager* aAssetManager) {
    glClearColor(0.0f, 1.0f, 0.0f, 1.0f);
    glEnable(GL_DEPTH_TEST);

    // Add some functions to enable alpha blending`
    // alpha_dest = 1 - alpha_src. Please find opengl blending functions.

    Scene::setup(aAssetManager);
}
```

# Submission



## Deadline

- 5. 22. 23:50 (**50% deduction if you miss the deadline.**)

## Upload followings to klas and your git repository.

- Git: Generate a .gif file with your results.  
Then upload it to your github and show your result on the main page.  
(e.g. [https://github.com/siamiz88/-GameGraphics-\\_Homework2](https://github.com/siamiz88/-GameGraphics-_Homework2))
- Klas: {student\_number}\_{name}.zip including git URL, vertex.glsl, fragment.glsl, main.cpp and scene.cpp.
  - \*\*If you use additional files (class, binary, etc.) to solve optional problem, please upload those files with detailed manual.
  - \*\* Do not submit the whole project files!! (**Penalty: 1 point of homework2 score can be deducted**)

## TA

- 송현수 (songhs@khu.ac.kr) – Please send your questions to TA first.

## Office hour

- Monday, Wednesday 2:00 PM ~ 6:00 PM
- Contact TA by email before you visit.