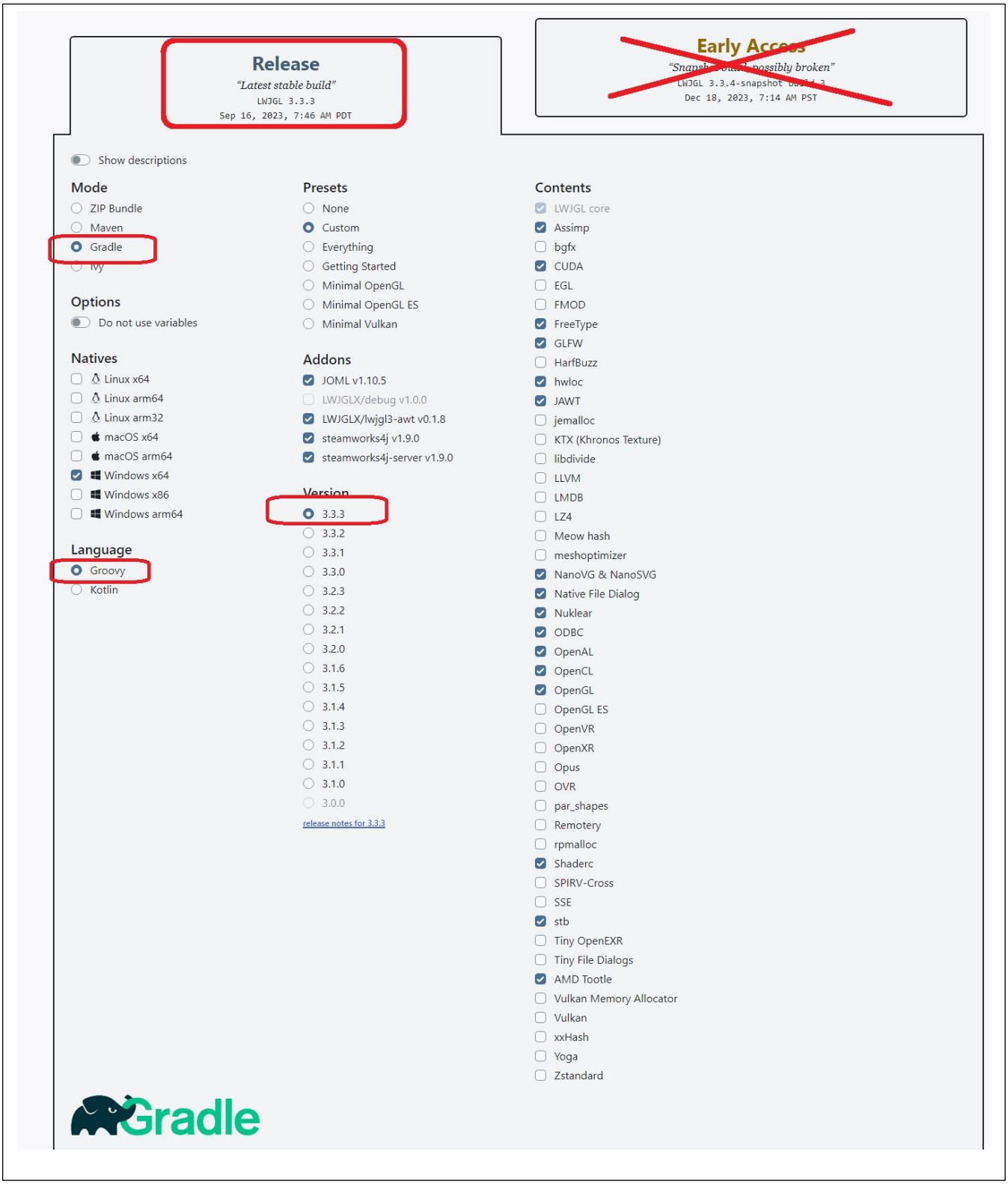
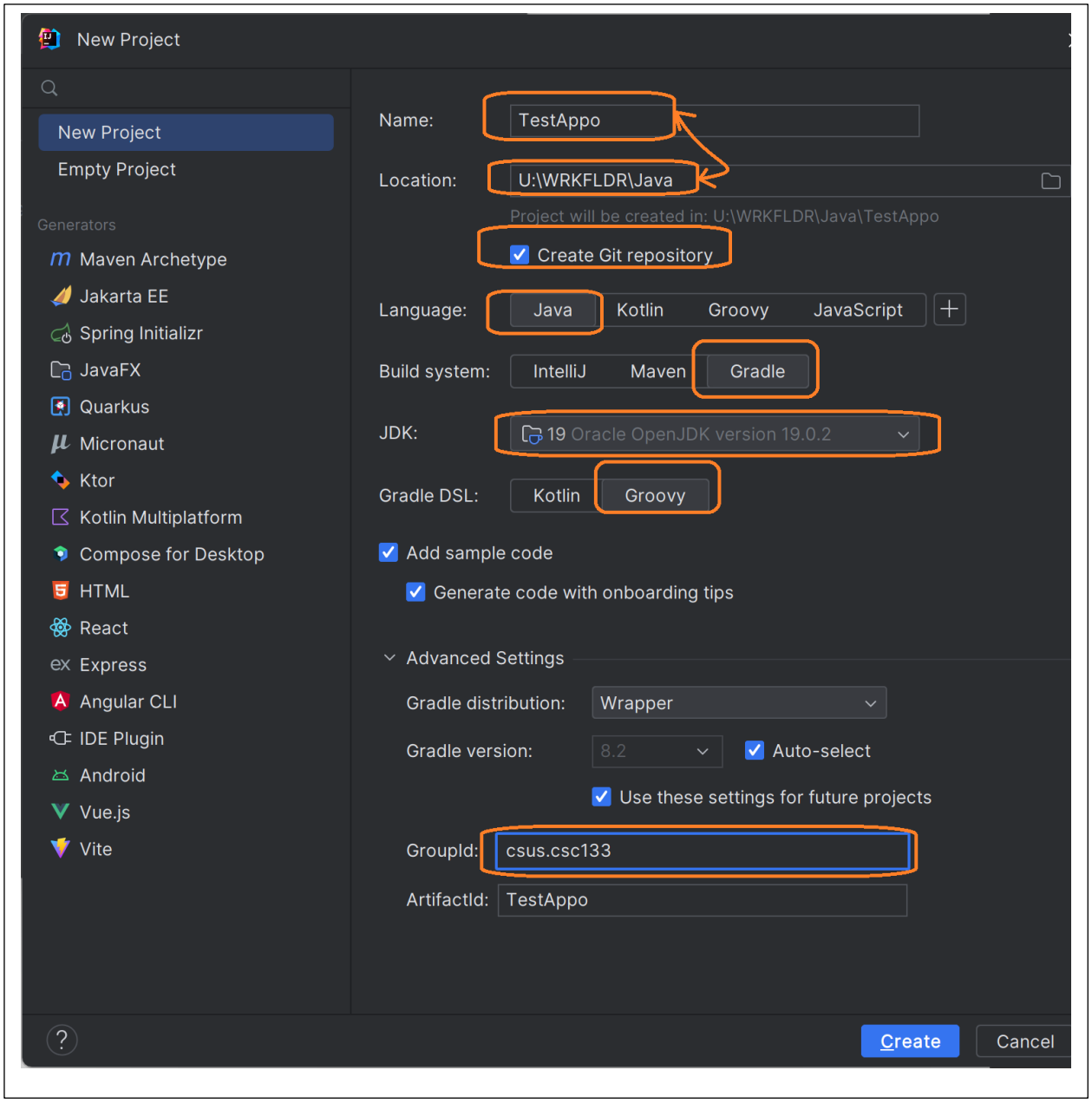
Intellij IDEA

https://www.jetbrains.com/shop/eform/students

should be available for free for CSU students.

Downloading Lightweight OpenGL for Java: lwjgl with the configuration required for the course from https://www.lwjgl.org/customize



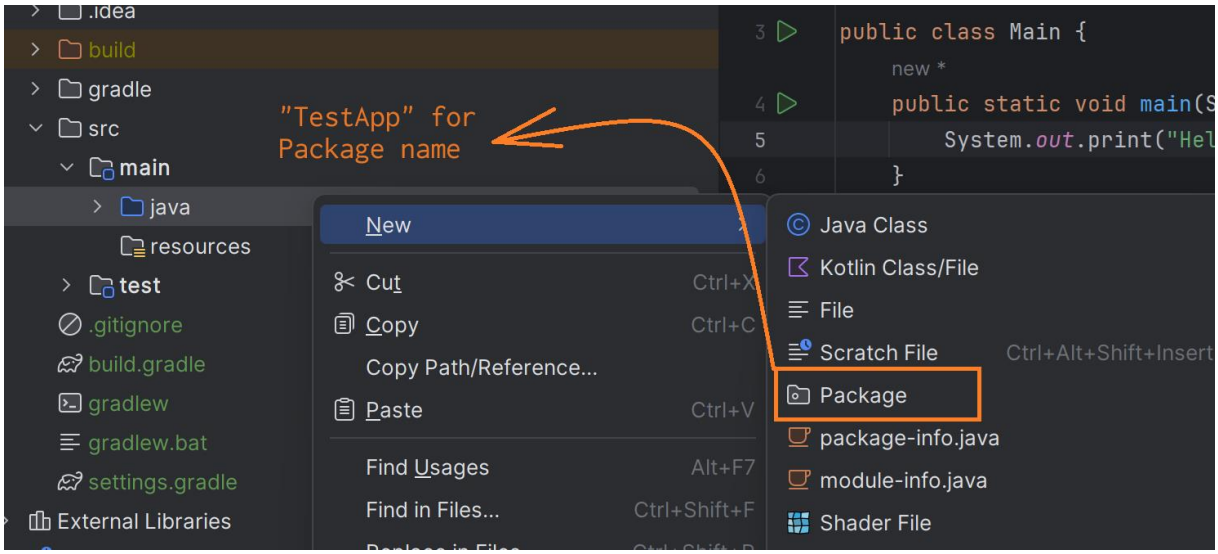


Copied libWJL-3.3.3 folder extracted from lwjgl-release-3.3.3-custom.zip to TestApp0 and inside the copied folder, deleted all *.txt files – they are not needed and just a distraction if we ever look in that folder; but they don't cause any harm though.

We get the following sample code:

```
package csus.csc133;

// Press Shift twice to open the Search Everywhere dialog and type `show
whitespaces`,
// then press Enter. You can now see whitespace characters in your code.
public class Main {
    public static void main(String[] args) {
        // Press Alt+Enter with your caret at the highlighted text to see
how
        // IntelliJ IDEA suggests fixing it.
        System.out.printf("Hello and welcome!");

        // Press Shift+F10 or click the green arrow button in the gutter to
run the code.
        for (int i = 1; i <= 5; i++) {

            // Press Shift+F9 to start debugging your code. We have set one
breakpoint
            // for you, but you can always add more by pressing Ctrl+F8.
            System.out.println("i = " + i);
        }
    }
}
```

Select a package name:



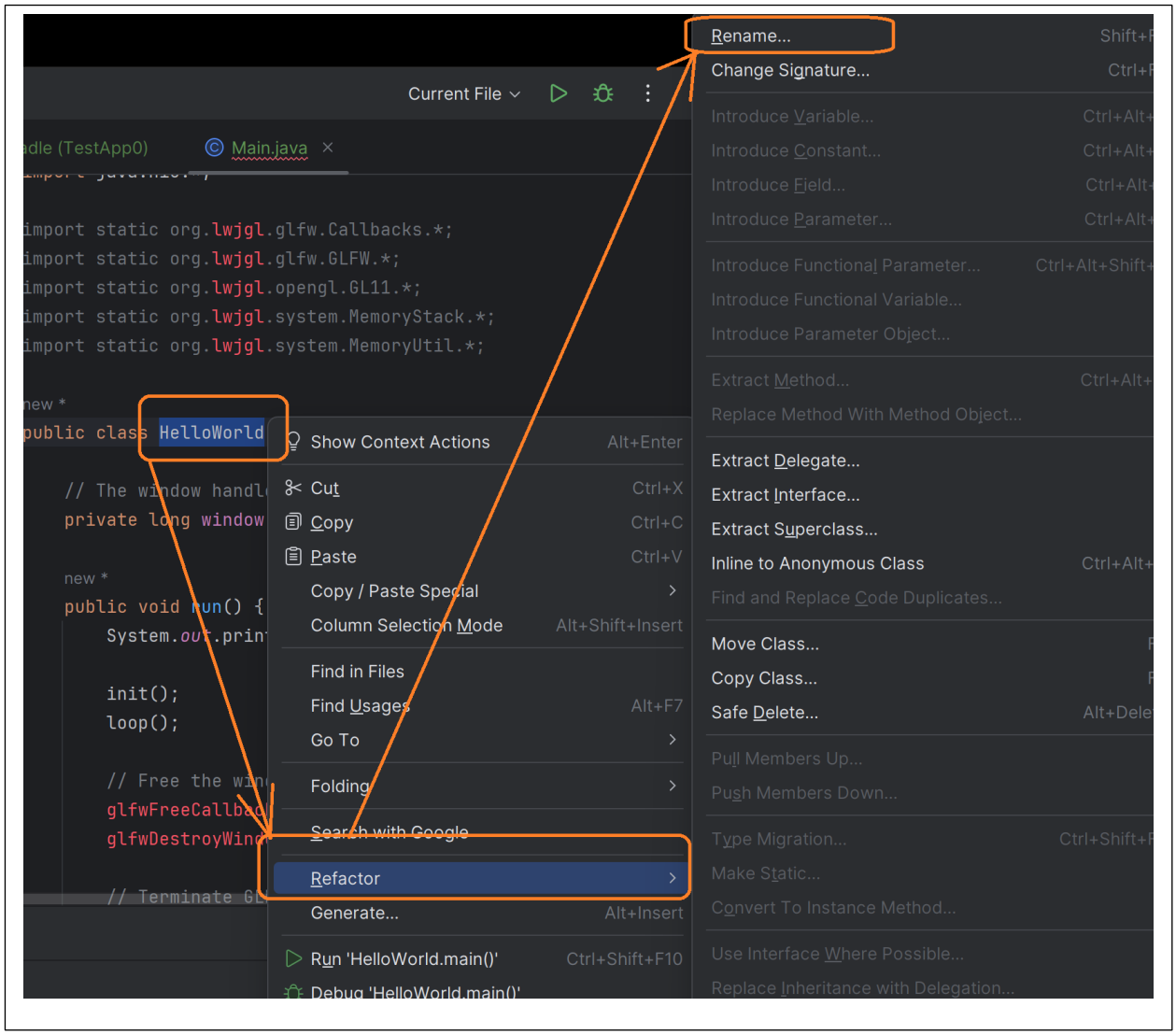
Copied the following code from https://www.lwjgl.org/guide:

```
import org.lwjgl.*;
import org.lwjgl.glfw.*;
import org.lwjgl.opengl.*;
```

```java
import org.lwjgl.system.*;

import java.nio.*;

import static org.lwjgl.glfw.Callbacks.*;
import static org.lwjgl.glfw.GLFW.*;
import static org.lwjgl.opengl.GL11.*;
import static org.lwjgl.system.MemoryStack.*;
import static org.lwjgl.system.MemoryUtil.*;

public class HelloWorld {

        // The window handle
        private long window;

        public void run() {
                System.out.println("Hello LWJGL " + Version.getVersion() + "!");

                init();
                loop();

                // Free the window callbacks and destroy the window
                glfwFreeCallbacks(window);
                glfwDestroyWindow(window);

                // Terminate GLFW and free the error callback
                glfwTerminate();
                glfwSetErrorCallback(null).free();
        }

        private void init() {
                // Setup an error callback. The default implementation
                // will print the error message in System.err.
                GLFWErrorCallback.createPrint(System.err).set();

                // Initialize GLFW. Most GLFW functions will not work before
doing this.
                if ( !glfwInit() )
                        throw new IllegalStateException("Unable to initialize
GLFW");

                // Configure GLFW
                glfwDefaultWindowHints(); // optional, the current window hints
are already the default
                glfwWindowHint(GLFW_VISIBLE, GLFW_FALSE); // the window will stay
hidden after creation
                glfwWindowHint(GLFW_RESIZABLE, GLFW_TRUE); // the window will be
resizable

                // Create the window
                window = glfwCreateWindow(300, 300, "Hello World!", NULL, NULL);
                if ( window == NULL )
                        throw new RuntimeException("Failed to create the GLFW
window");

                // Setup a key callback. It will be called every time a key is
pressed, repeated or released.
```

```
                glfwSetKeyCallback(window, (window, key, scancode, action, mods)
-> {
                        if ( key == GLFW_KEY_ESCAPE && action == GLFW_RELEASE )
                                glfwSetWindowShouldClose(window, true); // We
will detect this in the rendering loop
                });

                // Get the thread stack and push a new frame
                try ( MemoryStack stack = stackPush() ) {
                        IntBuffer pWidth = stack.mallocInt(1); // int*
                        IntBuffer pHeight = stack.mallocInt(1); // int*

                        // Get the window size passed to glfwCreateWindow
                        glfwGetWindowSize(window, pWidth, pHeight);

                        // Get the resolution of the primary monitor
                        GLFWVidMode vidmode =
glfwGetVideoMode(glfwGetPrimaryMonitor());

                        // Center the window
                        glfwSetWindowPos(
                                window,
                                (vidmode.width() - pWidth.get(0)) / 2,
                                (vidmode.height() - pHeight.get(0)) / 2
                        );
                } // the stack frame is popped automatically

                // Make the OpenGL context current
                glfwMakeContextCurrent(window);
                // Enable v-sync
                glfwSwapInterval(1);

                // Make the window visible
                glfwShowWindow(window);
        }

        private void loop() {
                // This line is critical for LWJGL's interoperation with GLFW's
                // OpenGL context, or any context that is managed externally.
                // LWJGL detects the context that is current in the current
thread,
                // creates the GLCapabilities instance and makes the OpenGL
                // bindings available for use.
                GL.createCapabilities();

                // Set the clear color
                glClearColor(1.0f, 0.0f, 0.0f, 0.0f);

                // Run the rendering loop until the user has attempted to close
                // the window or has pressed the ESCAPE key.
                while ( !glfwWindowShouldClose(window) ) {
                        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); //
clear the framebuffer

                        glfwSwapBuffers(window); // swap the color buffers
```

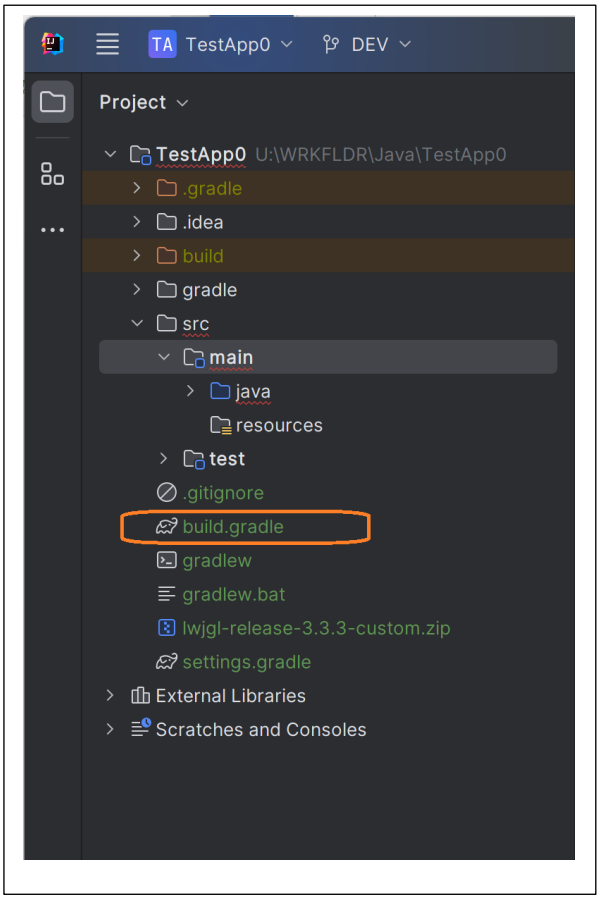
```
                        // Poll for window events. The key callback above will
only be
                        // invoked during this call.
                        glfwPollEvents();
                }
        }

        public static void main(String[] args) {
                new HelloWorld().run();
        }

}
```



We have a problem: this class is called “HelloWorld” and our default file is called “Main.java” → Right click on “HelloWorld” and “Refactor” → “Rename” and rename the class name to Main:

Adding the LWJGL library to the project:

Open build.gragle file:



Add the following text at the bottom of the file leave the commented out lines as they are: they are not needed for our projects and to make the project work with those uncommented, we need additional libraries that we won't be using:

```
project.ext.lwjglVersion = "3.3.3"
project.ext.jomlVersion = "1.10.5"
//project.ext.lwjgl3-awtVersion = "0.1.8"
project.ext.steamworks4jVersion = "1.9.0"
//project.ext.steamworks4j-serverVersion = "1.9.0"
project.ext.lwjglNatives = "natives-windows"

dependencies {
    implementation platform("org.lwjgl:lwjgl-bom:$lwjglVersion")

    implementation "org.lwjgl:lwjgl"
    implementation "org.lwjgl:lwjgl-assimp"
    implementation "org.lwjgl:lwjgl-cuda"
    implementation "org.lwjgl:lwjgl-freetype"
    implementation "org.lwjgl:lwjgl-glfw"
    implementation "org.lwjgl:lwjgl-hwloc"
    implementation "org.lwjgl:lwjgl-jawt"
    implementation "org.lwjgl:lwjgl-nanovg"
    implementation "org.lwjgl:lwjgl-nfd"
    implementation "org.lwjgl:lwjgl-nuklear"
    implementation "org.lwjgl:lwjgl-odbc"
    implementation "org.lwjgl:lwjgl-openal"
    implementation "org.lwjgl:lwjgl-opencl"
    implementation "org.lwjgl:lwjgl-opengl"
    implementation "org.lwjgl:lwjgl-shaderc"
    implementation "org.lwjgl:lwjgl-stb"
    implementation "org.lwjgl:lwjgl-tootle"
    runtimeOnly "org.lwjgl:lwjgl::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-assimp::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-freetype::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-glfw::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-hwloc::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-nanovg::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-nfd::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-nuklear::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-openal::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-opengl::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-shaderc::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-stb::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-tootle::$lwjglNatives"
    implementation "org.joml:joml:${jomlVersion}"
    //implementation "org.lwjglx:lwjgl3-awt:${lwjgl3-awtVersion}"
    //implementation "com.code-
disaster.steamworks4j:steamworks4j:${steamworks4jVersion}"
    //implementation "com.code-disaster.steamworks4j:steamworks4j-
server:${steamworks4j-serverVersion}"
}
```

If you have a Main.java modified as suggested earlier, then if you hit '**Shift F10**', then you should see some OpenGL window pop up!

My build.gradle script – in case you need to debug yours

Notice that you may have to change the "group" at line 5, to match the group you have selected for your project.

```
plugins {
    id 'java'
}

group = 'csus.csc133'
version = '1.0-SNAPSHOT'

repositories {
    mavenCentral()
}

dependencies {
    testImplementation platform('org.junit:junit-bom:5.9.1')
    testImplementation 'org.junit.jupiter:junit-jupiter'
}

test {
    useJUnitPlatform()
}

project.ext.lwjglVersion = "3.3.3"
project.ext.jomlVersion = "1.10.5"
//project.ext.lwjgl3-awtVersion = "0.1.8"
project.ext.steamworks4jVersion = "1.9.0"
//project.ext.steamworks4j-serverVersion = "1.9.0"
project.ext.lwjglNatives = "natives-windows"

dependencies {
    implementation platform("org.lwjgl:lwjgl-bom:$lwjglVersion")

    implementation "org.lwjgl:lwjgl"
    implementation "org.lwjgl:lwjgl-assimp"
    implementation "org.lwjgl:lwjgl-cuda"
    implementation "org.lwjgl:lwjgl-freetype"
    implementation "org.lwjgl:lwjgl-glfw"
    implementation "org.lwjgl:lwjgl-hwloc"
    implementation "org.lwjgl:lwjgl-jawt"
    implementation "org.lwjgl:lwjgl-nanovg"
```

```
    implementation "org.lwjgl:lwjgl-nfd"
    implementation "org.lwjgl:lwjgl-nuklear"
    implementation "org.lwjgl:lwjgl-odbc"
    implementation "org.lwjgl:lwjgl-openal"
    implementation "org.lwjgl:lwjgl-opencl"
    implementation "org.lwjgl:lwjgl-opengl"
    implementation "org.lwjgl:lwjgl-shaderc"
    implementation "org.lwjgl:lwjgl-stb"
    implementation "org.lwjgl:lwjgl-tootle"
    runtimeOnly "org.lwjgl:lwjgl::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-assimp::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-freetype::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-glfw::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-hwloc::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-nanovg::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-nfd::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-nuklear::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-openal::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-opengl::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-shaderc::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-stb::$lwjglNatives"
    runtimeOnly "org.lwjgl:lwjgl-tootle::$lwjglNatives"
    implementation "org.joml:joml:${jomlVersion}"
    //implementation "org.lwjglx:lwjgl3-awt:${lwjgl3-awtVersion}"
    //implementation "com.code-disaster.steamworks4j:steamworks4j:${steamworks4jVersion}"
    //implementation "com.code-disaster.steamworks4j:steamworks4j-server:${steamworks4j-serverVersion}"
}
```

Setting up git

At the minimum, and most of the times, you need only these commands:

- gitinit
- .gitignore - link to my .gitignore which should work for you, for starters.
- git checkout -b
- git diff
- git commit
- git restore .
- git switch -c

Some git tasks are easier with command-line and some with a GUI. I use

https://gitextensions.github.io/

for the GUI. Your mileage may vary.

Other tools I have used in developing this course

For this course, you won't need to use any of these. These are mentioned here so that you know what tools I used to tweak the assets and firm up the equations etc.

If you have longer term interest in almost any aspect of Visual Computing, learning any of these tools is an advantage - and all these are free to CSU students (Krita is a freeware):

- Krita
- Maya
- Photoshop
- MATLAB