TH COLOGNE
PROF. DR. FLORIAN NIEBLING
ADVANCED MEDIA INSTITUTE

Technology
Arts Sciences
TH Köln

APRIL 24, 2024
DELIVERY: NONE

COMPUTER GRAPHICS and Animation
LEAF - Getting STARTED

1 Kotlin and Java

The internship takes place in Kotlin. However, Java, at least version 11, is still required. Make sure that you have installed it.
If this is not yet the case, you can download openJDK 11 or higher from the following link https://jdk.java.net/archive/. It is important to install at least this version, as earlier versions may lead to problems with the execution of the programs. Please do NOT install openJDK_14, as this will lead to errors. An LTS version would presumably be advantageous (11/17). Java version 20 is entered in the Gradle file provided.
When reinstalling JAVA, bear in mind that the environment variable may not exist and must be created. Google can help you with this step.

2 Development environment

Kotlin development is generally more efficient and easier if the development takes place in a development environment designed for this purpose. Development environments help, for example, with syntax highlighting and auto-completion.
The IntelliJ development environment is recommended. Students of the TH Köln can purchase a license for the Ultimate version there using their smail address.

- IntelliJ from JetBrains (https://www.jetbrains.com/idea/)

If you want to use a different development environment, you will have to bring it into an executable state yourself. You can find help on using the necessary library on the website http://wiki.lwjgl.org/index.html.

3 Gradle

The build tool Gradle is used to bundle the project files including the necessary libraries. Depending on your operating system, you may also need to install Gradle. The following link will take you to the Gradle website, https://gradle.org/. Above all, the integration of the exact addressing of the natives of the respective operating system is very important. This is also done by Gradle. If there is a problem here, it may be due to an outdated version of Gradle or the selected DSL. The Gradle file supplied by us uses Groovy instead of Kotlin as DSL. Support for the new M1 and M2 chips from Apple is also integrated.

4 OpenGL

In this internship, you will work very closely with the hardware of your system. In some cases, you will need specific drivers for this. Make sure that you have installed the latest graphics card driver for your graphics card so that programming with OpenGL 3.3 in Kotlin/Java is possible.

5 Procedure

Unpack the zip file that comes with task sheet 1. It contains a Gradle project. If you are using IntelliJ, a double-click on `build.gradle` should suffice to open the project. The necessary libraries are then linked and indexed. Once this process has been successfully completed, start the executable file `main.kt` under `src/main/kotlin/cga/exercise/`. A window should open as shown in Figure 1. MAC users must also observe the following section before the program can start without errors.

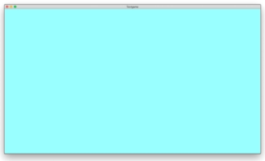


Figure 1: Start

It is possible that the installed JDK is not connected to the project from the start. If this is the case, IntelliJ will inform you and provide help to solve the problem.

6 IntelliJ under MacOS

To get the project running under MacOS, you must edit the run configuration of the main class. You can access this via the menu as shown in Figure 2. Now you get to the configuration (see Figure 3) of the executable file. Please enter the following under VM options: `-XstartOnFirstThread`. Click *`OK to`* confirm and everything should now work.

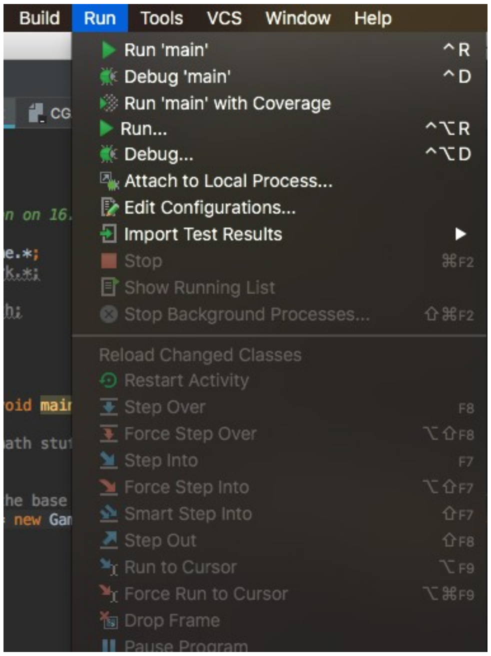
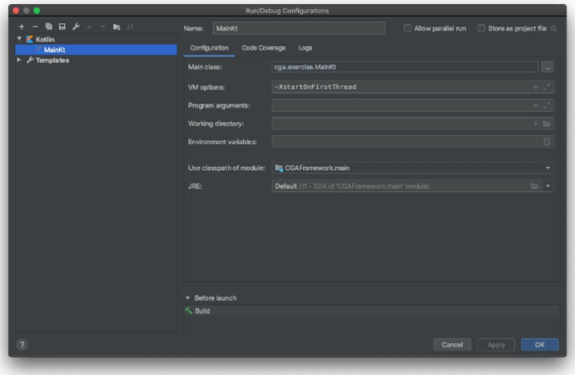


Figure 2: Run configuration



Name: MainKt

Main class: cga.exercise.MainKt

VM options: -XstartOnFirstThread

Program arguments:

Working directory:

Environment variables:

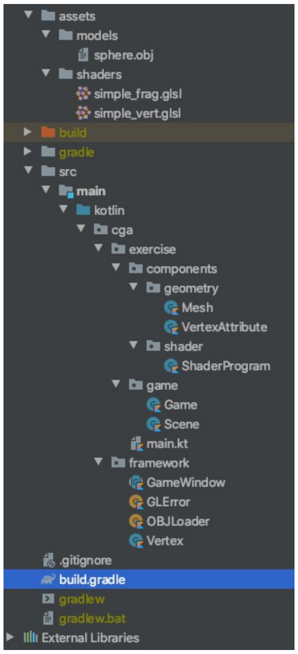
Use classpath of module: CGAFramework.main

JRE: Default (11 - SDK of 'CGAFramework.main' module)

Figure 3: VM Argument

7 Folder structure and handling

The folder structure, which is shown in Figure 4, is briefly explained below.
The `framework` contains the functionalities for the Windows, input queries and other things that you do not need to pay any attention to. These will not be changed during the internship. However, it may be interesting to take a look at these files for a better understanding.
The solution to your tasks comes in the `exercise` package. `components` contains the basic components, which you will develop in the course of the internship. All actions relating to the scene are placed in the package `game`.
Models, textures and shaders are integrated in the course of the practical course and should be saved in the `assets` folder.



```
▼ assets
  ▼ models
      sphere.obj
  ▼ shaders
      simple_frag.glsl
      simple_vert.glsl
▶ build
▶ gradle
▼ src
  ▼ main
    ▼ kotlin
      ▼ cga
        ▼ exercise
          ▼ components
            ▼ geometry
                Mesh
                VertexAttribute
            ▼ shader
                ShaderProgram
          ▼ game
              Game
              Scene
            main.kt
        ▼ framework
            GameWindow
            GLError
            OBJLoader
            Vertex
  .gitignore
  build.gradle
  gradlew
  gradlew.bat
▶ External Libraries
```

Figure 4: Folder structure

8 Conclusion

You have now finished configuring your editor and your project. You no longer need to repeat this procedure for subsequent task sheets. We generally place the files that need to be delivered later in the respective folders/packages. This makes it easier for you to place them in the project. If additional actions are required, we will indicate this on the respective task sheet.
If you have any questions, our advisors will be happy to help and advise you. On the one hand in the consultation rooms, but also by e-mail to cga-praktikum@gm.fh-koeln.de

Good luck and enjoy your internship.