

6. APRIL 2023
ABGABE:

COMPUTERGRAFIK UND ANIMATION SOSE2023 BLATT 0 - GETTING STARTED

0.1 Kotlin und Java

Das Praktikum findet in Kotlin statt. Allerdings wird hierfür dennoch Java, mindestens in der Version 11, benötigt. Stellen Sie sicher, dass Sie diese installiert haben.

Sollte dies noch nicht der Fall sein, so können Sie OpenJDK 11 oder höher unter dem folgenden Link <https://jdk.java.net/archive/> herunterladen. Es ist wichtig mindestens diese Version zu installieren, da frühere Versionen unter Umständen zu Problemen mit der Ausführung der Programme führen können. **Bitte installieren Sie NICHT das OpenJDK 14, da dies zu Fehlern führt.** Eine LTS-Version wäre vermutlich von Vorteil (11/17).

Als empfehlenswert hat sich OpenJDK 17 erwiesen. Diese Version wird auch noch von dem aktuellen Gradle Buildfile unterstützt. Sollten Sie eine JDK Version >17 verwenden, müssen Sie gegebenenfalls die Gradle Version im Buildfile anpassen.

Bedenken Sie bei einer Neuinstallation von JAVA, dass unter Umständen die Umgebungsvariable nicht vorhanden ist und angelegt werden muss. Für diesen Schritt kann Google eine Hilfe sein.

0.2 Entwicklungsumgebung

Kotlin-Entwicklung ist im Allgemeinen effizienter und gestaltet sich einfacher, wenn die Entwicklung in einer dafür ausgelegten Entwicklungsumgebung stattfindet. Entwicklungsumgebungen helfen beispielsweise durch Syntaxhighlighting und Autovervollständigung.

Es wird die Entwicklungsumgebung IntelliJ empfohlen. Studenten der TH Köln können dort mittels ihrer email-Adresse eine Lizenz für die Ultimate-Version erwerben.

- IntelliJ von JetBrains (<https://www.jetbrains.com/idea/>)

Sollten Sie eine andere Entwicklungsumgebung nutzen wollen, so sind Sie gezwungen, diese selbstständig in einen lauffähigen Zustand zu bringen. Hilfe zur Nutzung der notwendigen Bibliothek finden Sie auf der Website <http://wiki.lwjgl.org/index.html>.

0.3 Gradle

Zur Bündelung der Projektdateien inkl. der notwendigen Libraries wird das Build-Tool Gradle verwendet. Sofern Sie IntelliJ verwenden, wird die passende Gradle Version automatisch heruntergeladen, wenn Sie das Projekt das erste Mal öffnen.

Je nach Betriebssystem und Entwicklungsumgebung kann es aber notwendig sein, dass Sie Gradle händisch installieren müssen. Über den folgenden Link gelangen Sie zu der Website von Gradle, <https://gradle.org/>.

Vor allem die Integration der exakten Adressierung der Natives des jeweiligen Betriebssystems ist sehr wichtig. Dies wird auch von Gradle übernommen. Sollte es hier zu einem Problem kommen, kann es zum einen an einer veralteten Version von Gradle oder aber an der gewählten DSL liegen. Das von uns mitgelieferte Gradle-File benutzt Groovy statt Kotlin als DSL. Die Unterstützung der neuen M1- und M2-Chips von Apple ist ebenfalls integriert.

0.4 OpenGL

In diesem Praktikum arbeiten Sie sehr nah an der Hardware Ihres Systems. Hierfür benötigen Sie zum Teil spezifische Treiber. Stellen Sie sicher, dass Sie den aktuellsten Grafikkartentreiber für Ihre Grafikkarte installiert haben, sodass die Programmierung mit OpenGL 3.3 in Kotlin/Java möglich ist.

0.5 Vorgehensweise

Entpacken Sie das zip-File, welches zusammen mit dem Aufgabenblatt 1 kommt. Darin befindet sich ein Gradle-Projekt. Wenn Sie IntelliJ nutzen, sollte ein Doppelklick auf die `build.gradle` genügen, um das Projekt zu öffnen. Wenn IntelliJ nicht als Standardanwendung zum öffnen von `.gradle` Dateien eingestellt ist, sollten Sie den Root-Ordner des Projektes direkt in IntelliJ als Projekt importieren / öffnen. IntelliJ sollte nun einen Gradle-Import starten, welcher alle notwendigen Abhängigkeiten herunterlädt.

Im Anschluss werden die notwendigen Bibliotheken verknüpft und indiziert. Wenn dieser Vorgang erfolgreich erledigt ist, starten Sie die ausführbare Datei `main.kt` unter `src/main/kotlin/cga/exercise/`. Es sollte sich ein Fenster öffnen, wie in Abbildung 1 zu sehen. MAC-User müssen den nachfolgenden Abschnitt zusätzlich beachten, bevor das Projekt fehlerlos starten kann.

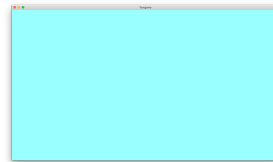


Abbildung 1: Start

0.6 IntelliJ unter MacOS

Um das Projekt unter MacOS zum Laufen zu bringen müssen Sie die Run-Configuration der Main-Klasse bearbeiten. Sie kommen über das Menü, wie in Abbildung 2 zu sehen. Nun gelangen Sie in die Konfiguration (siehe Abbildung 3) der ausführbaren Datei. Hier geben Sie unter den VM options bitte folgendes ein: `-XstartOnFirstThread`. Mit der Bestätigung auf `OK` sollte nun alles funktionieren.

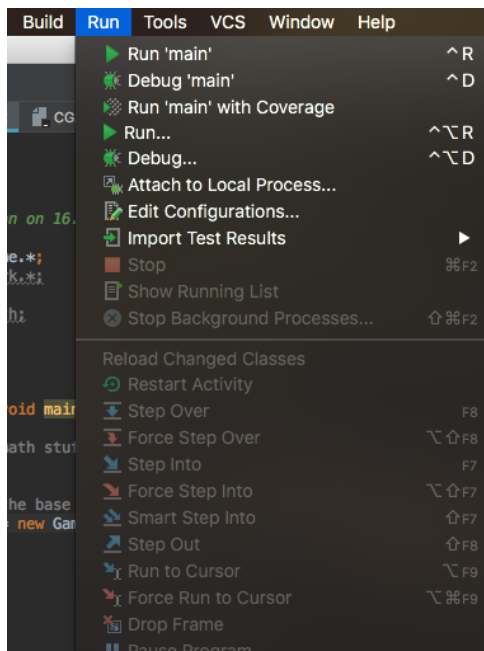


Abbildung 2: Run Konfiguration

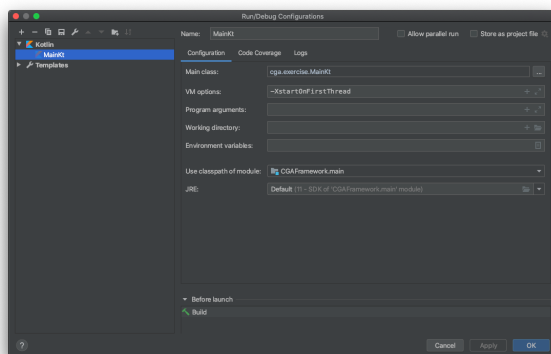


Abbildung 3: VM Argument

0.7 Ordnerstruktur und Handling

Nachfolgend wird die Ordnerstruktur kurz erklärt, welche in Abbildung 4 dargestellt ist.

Das **framework** beinhaltet die Funktionalitäten für die Fenster, Input-Abfragen und weitere Dinge, denen Sie keinerlei Beachtung schenken müssen. Diese werden während des Praktikums nicht mehr verändert. Zum erweiterten Verständnis kann es aber interessant sein, sich diese Dateien auch mal anzusehen.

Die Lösung Ihrer Aufgaben kommt in das **exercise**-Package. **components** beinhaltet die Basiskomponenten, die Sie im Laufe des Praktikums entwickeln werden. Im Package **Game** werden alle Aktionen bzgl. der Szene platziert.

Im Verlauf des Praktikums werden Modelle, Texturen und Shader integriert, welche sinnvollerweise im Ordner **assets** gespeichert werden sollten.

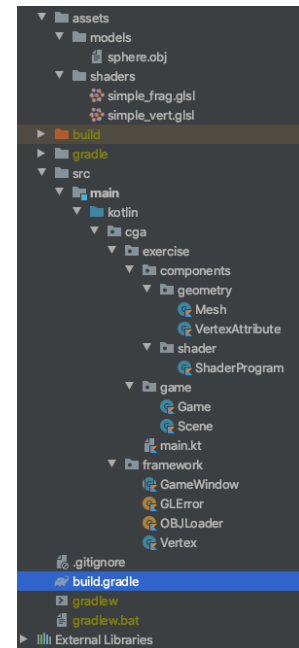


Abbildung 4: Ordnerstruktur

0.8 Abschluss

Damit haben Sie ihren Editor und ihr Projekt fertig konfiguriert. Für spätere Aufgabenblätter müssen Sie diese Prozedur nicht mehr wiederholen. Im Allgemeinen platzieren wir die Files, die nachgeliefert werden müssen, bereits in den jeweiligen Ordnern/Packages. Dies ermöglicht es Ihnen, sie einfacher im Projekt zu platzieren. Falls zusätzliche Aktionen erforderlich sind, werden wir dies auf dem jeweiligen Aufgabenblatt bekanntgeben.

Bei Fragen stehen Ihnen die Betreuer gerne mit Rat und Tat zur Seite. Zum einen in den Beratungsterminen aber auch per Mail an cga-praktikum@gm.fh-koeln.de

0.9 Was tun bei Build-Fehlern?

Wenn beim Gradle Build Step Fehler auftreten, hilft es meistens, die generierten Dateien und Ordner zu löschen und so das Projekt aus einem “sauberen” Zustand heraus zu starten. Im Falle von IntelliJ müssen folgende Dateien und Ordner (und Unterordner) gelöscht werden, sofern sie existieren:

- .idea
- .gradle
- build
- out
- gradle
- gradlew
- gradlew.bat

0.10 Versionskontrolle

Wir empfehlen dringend, die Praktikumsaufgaben in einem Versionskontrollsystem wie Git zu bearbeiten. Im ersten Aufgabenpaket liegt eine Datei `.gitignore` bei. Die sollte Ihnen bei der Zusammenarbeit helfen, indem durch die IDE generierte Dateien ignoriert werden.

Viel Erfolg und Spaß im Praktikum.