Tutorial: Criando uma Aplicação Java para Renderizar Vídeo com OpenGL usando Maven

Este tutorial irá guiá-lo passo a passo na criação de uma aplicação Java do zero para renderizar vídeo utilizando OpenGL, seguindo o código fornecido no repositório gpu. Utilizaremos o **Maven** para gerenciar as dependências do projeto, simplificando o processo de configuração.

Pré-requisitos

- **Java Development Kit (JDK)** instalado (versão 8 ou superior).
  - Download JDK ou OpenJDK
- **Apache Maven** instalado.
  - Download Maven
  - Instruções de instalação: Instalando o Maven
- **IDE** de sua preferência com suporte a Maven:
  - Eclipse
  - IntelliJ IDEA
  - NetBeans
- Conhecimento básico de Java e OpenGL.

1 Configurando o Ambiente de Desenvolvimento

1.1 1.1. Instale o JDK

- Baixe o JDK apropriado para o seu sistema operacional:
  - **Oracle JDK**: Download
  - **OpenJDK**: Download
- Siga as instruções de instalação fornecidas pelo instalador.
- Configure a variável de ambiente `JAVA_HOME` apontando para o diretório de instalação do JDK.

1.2 1.2. Instale o Apache Maven

- Baixe o Maven: Download Maven
- Siga as instruções de instalação: Instalando o Maven
- Configure a variável de ambiente `MAVEN_HOME` apontando para o diretório de instalação do Maven.
- Adicione o diretório `bin` do Maven ao `PATH` do sistema.

1.3 1.3. Configure sua IDE

- **Eclipse**:
  - Certifique-se de ter a versão do Eclipse que inclui o suporte ao Maven.
- **IntelliJ IDEA**:
  - O IntelliJ IDEA possui suporte integrado ao Maven.
- **NetBeans**:
  - O NetBeans também oferece suporte nativo ao Maven.

2 Criando o Projeto Maven

2.1 2.1. Criar um Novo Projeto Maven

Abra sua IDE e selecione a opção para criar um novo projeto Maven.

Eclipse:

1. `File` → `New` → `Maven Project`.
2. Selecione o local do projeto e clique em `Next`.
3. Selecione um **archetype** padrão, como `maven-archetype-quickstart`, e clique em `Next`.
4. Preencha os campos:
   - **Group Id**: `com.seu_nome`
   - **Artifact Id**: `gpu`
   - **Version**: `1.0-SNAPSHOT`
   - **Package**: `com.seu_nome.gpu`
5. Clique em `Finish`.

IntelliJ IDEA:

1. `File` → `New` → `Project...`.
2. Selecione `Maven` na lista e clique em `Next`.
3. Marque a opção `Create from archetype` e selecione `maven-archetype-quickstart`.
4. Preencha os campos:
   - **Group Id**: `com.seu_nome`
   - **Artifact Id**: `gpu`
   - **Version**: `1.0-SNAPSHOT`
5. Clique em `Next` e depois em `Finish`.

NetBeans:

1. `File` → `New Project...`.
2. Selecione `Maven` → `Java Application` e clique em `Next`.
3. Preencha os campos:

- **Project Name**: `gpu`
- **Group Id**: `com.seu_nome`
- **Artifact Id**: `gpu`
- **Version**: `1.0-SNAPSHOT`

4. Clique em `Finish`.

2.2 2.2. Estrutura do Projeto

O Maven irá criar a seguinte estrutura de diretórios:

```
gpu
 pom.xml
 src
     main
        java
            com
                seu_nome
                    gpu
                     dto
                     memory
                    Main.java
     test
         java
```

3 Configurando as Dependências no `pom.xml`

Abra o arquivo `pom.xml` localizado na raiz do projeto e adicione as dependências necessárias, conforme especificado no repositório gpu.

3.1 3.1. Dependências do Projeto

Adicione as seguintes dependências ao `pom.xml`:

```xml
 <properties>
        <maven.compiler.source>22</maven.compiler.source>
        <maven.compiler.target>22</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <lwjgl.version>3.3.4</lwjgl.version>
    </properties>

    <profiles>
        <profile>
            <id>lwjgl-natives-linux-amd64</id>
            <activation>
                <os>
                    <family>unix</family>
                    <name>linux</name>
                    <arch>amd64</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-linux</lwjgl.natives>
            </properties>
        </profile>
        <profile>
```

```
            <id>lwjgl-natives-linux-aarch64</id>
            <activation>
                <os>
                    <family>unix</family>
                    <name>linux</name>
                    <arch>aarch64</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-linux-arm64</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-linux-arm</id>
            <activation>
                <os>
                    <family>unix</family>
                    <name>linux</name>
                    <arch>arm</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-linux-arm32</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-linux-arm32</id>
            <activation>
                <os>
                    <family>unix</family>
                    <name>linux</name>
                    <arch>arm32</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-linux-arm32</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-linux-ppc64le</id>
            <activation>
                <os>
                    <family>unix</family>
                    <name>linux</name>
                    <arch>ppc64le</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-linux-ppc64le</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-linux-riscv64</id>
            <activation>
                <os>
                    <family>unix</family>
                    <name>linux</name>
                    <arch>riscv64</arch>
                </os>
```

```
            </activation>
            <properties>
                <lwjgl.natives>natives-linux-riscv64</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-macos-x86_64</id>
            <activation>
                <os>
                    <family>mac</family>
                    <arch>x86_64</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-macos</lwjgl.natives>
            </properties>
            <dependencies>
                <dependency>
                    <groupId>org.lwjgl</groupId>
                    <artifactId>lwjgl-vulkan</artifactId>
                    <classifier>natives-macos</classifier>
                </dependency>
            </dependencies>
        </profile>
        <profile>
            <id>lwjgl-natives-macos-aarch64</id>
            <activation>
                <os>
                    <family>mac</family>
                    <arch>aarch64</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-macos-arm64</lwjgl.natives>
            </properties>
            <dependencies>
                <dependency>
                    <groupId>org.lwjgl</groupId>
                    <artifactId>lwjgl-vulkan</artifactId>
                    <classifier>natives-macos-arm64</classifier>
                </dependency>
            </dependencies>
        </profile>
        <profile>
            <id>lwjgl-natives-windows-amd64</id>
            <activation>
                <os>
                    <family>windows</family>
                    <arch>amd64</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-windows</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-windows-x86</id>
            <activation>
                <os>
```

```
                    <family>windows</family>
                    <arch>x86</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-windows-x86</lwjgl.natives>
            </properties>
        </profile>
        <profile>
            <id>lwjgl-natives-windows-aarch64</id>
            <activation>
                <os>
                    <family>windows</family>
                    <arch>aarch64</arch>
                </os>
            </activation>
            <properties>
                <lwjgl.natives>natives-windows-arm64</lwjgl.natives>
            </properties>
        </profile>
    </profiles>

    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>org.lwjgl</groupId>
                <artifactId>lwjgl-bom</artifactId>
                <version>${lwjgl.version}</version>
                <scope>import</scope>
                <type>pom</type>
            </dependency>
        </dependencies>
    </dependencyManagement>

    <dependencies>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-assimp</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-bgfx</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-cuda</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-egl</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-fmod</artifactId>
        </dependency>
```

```
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-freetype</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-glfw</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-harfbuzz</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-hwloc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-jawt</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-jemalloc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-ktx</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-libdivide</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-llvm</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-lmdb</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-lz4</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-meow</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-meshoptimizer</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-msdfgen</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-nanovg</artifactId>
```

```
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-nfd</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-nuklear</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-odbc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-openal</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opencl</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opengl</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opengles</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-openvr</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-openxr</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opus</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-ovr</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-par</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-remotery</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-rpmalloc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
```

```
            <artifactId>lwjgl-shaderc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-spvc</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-sse</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-stb</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-tinyexr</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-tinyfd</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-tootle</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-vma</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-vulkan</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-xxhash</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-yoga</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-zstd</artifactId>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-assimp</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-bgfx</artifactId>
```

```
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-freetype</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-glfw</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-harfbuzz</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-hwloc</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-jemalloc</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-ktx</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-libdivide</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-llvm</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-lmdb</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-lz4</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-meow</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
```

```
            <artifactId>lwjgl-meshoptimizer</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-msdfgen</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-nanovg</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-nfd</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-nuklear</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-openal</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opengl</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opengles</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-openvr</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-openxr</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-opus</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-ovr</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
```

```
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-par</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-remotery</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-rpmalloc</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-shaderc</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-spvc</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-sse</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-stb</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-tinyexr</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-tinyfd</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-tootle</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-vma</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-xxhash</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
```

```
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-yoga</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.lwjgl</groupId>
            <artifactId>lwjgl-zstd</artifactId>
            <classifier>${lwjgl.natives}</classifier>
        </dependency>
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <version>1.18.32</version>
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.bytedeco</groupId>
            <artifactId>javacv</artifactId>
            <version>1.5.10</version>
        </dependency>
        <dependency>
            <groupId>org.bytedeco</groupId>
            <artifactId>javacv-platform</artifactId>
            <version>1.5.10</version>
        </dependency>
    </dependencies>
```

Listing 1: Dependências no pom.xml

3.2 3.2. Configurações do Maven Compiler Plugin

Adicione o plugin do compilador para definir a versão do Java:

```
<build>
        <resources>
            <resource>
                <directory>src/main/resources</directory>
                <includes>
                    <include>**/*</include>
                </includes>
            </resource>
        </resources>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-assembly-plugin</artifactId>
                <version>3.7.1</version>
                <configuration>
                    <descriptorRefs>
                        <descriptorRef>jar-with-dependencies</descriptorRef>
                    </descriptorRefs>
                    <archive>
                        <manifest>
                            <mainClass>com.seu_nome.Main</mainClass>
                        </manifest>
                    </archive>
                </configuration>
                <executions>
```

```
                    <execution>
                        <id>make-assembly</id>
                        <phase>package</phase>
                        <goals>
                            <goal>single</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>
        </plugins>
    </build>
```

Listing 2: Configuração do Maven Compiler Plugin

3.3 3.3. Exemplo Completo do `pom.xml`

Seu `pom.xml` deve se parecer com:

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
         http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.seu_nome</groupId>
    <artifactId>gpu</artifactId>
    <version>1.0-SNAPSHOT</version>

    <dependencies>
        <!-- Dependências conforme listado acima -->
    </dependencies>

    <build>
        <!-- Build conforme listado acima -->
    </build>
</project>
```

Listing 3: Arquivo pom.xml completo

4 Atualizar o Projeto

Após salvar o `pom.xml`, atualize o projeto Maven para baixar as dependências:

- **Eclipse**:
  - Clique com o botão direito no projeto → `Maven` → `Update Project...`
- **IntelliJ IDEA**:
  - O IntelliJ IDEA detecta alterações no `pom.xml` e atualiza automaticamente as dependências.
- **NetBeans**:
  - Clique com o botão direito no projeto → `Reload POM`

5 Escrever o Código da Aplicação

Siga o código fornecido no repositório gpu. Abaixo, detalharemos os principais componentes do código.

5.1 Implementar a Classe `RenderDataDto.java`

Crie a classe `RenderDataDto.java` no pacote dto.

```java
package com.seu_nome.dto;

import lombok.Builder;

@Builder
public record RenderDataDto(float[] vertex, float[] pixel) {}
```

Listing 4: Record RenderDataDto

5.2 Implementar a Classe `FrameBuffer.java`

Crie a classe `FrameBuffer.java` no pacote memory.

```java
package com.seu_nome.memory;

import com.seu_nome.dto.RenderDataDto;
import java.nio.ByteBuffer;
import java.nio.ByteOrder;
import java.nio.FloatBuffer;
import lombok.Getter;
import lombok.extern.java.Log;

/** A class representing a framebuffer that manages two buffers for double
    buffering. */
@Log
public class FrameBuffer {

  @Getter private static int bufferSize; // Size of each buffer

  private byte[] frontPixelBuffer; // Buffer to store pixel data

  private byte[] backPixelBuffer; // Buffer to store pixel data

  private byte[] frontVertexBuffer; // Buffer currently displayed

  private byte[] backVertexBuffer; // Buffer to write new data to

  /**
   * Constructs a FrameBuffer with specified memory addresses and buffer size.
   *
   * @param bufferSize The size of each buffer.
   */
  public FrameBuffer(final int bufferSize) {

    final int size = bufferSize * 8;
    this.frontPixelBuffer = new byte[size]; // Initialize pixel buffer
    this.backPixelBuffer = new byte[size]; // Initialize pixel buffer
    this.frontVertexBuffer = new byte[size]; // Initialize front buffer
    this.backVertexBuffer = new byte[size]; // Initialize back buffer
    FrameBuffer.bufferSize = bufferSize * 2;
  }

  /**
   * Writes float data to the pixel buffer, converting them to bytes before
       storing.
```

```
41      *
42      * @param beginAddress The starting index where data is to be written.
43      * @param data The float data to be converted and written.
44      * @throws MemoryException If the write operation exceeds buffer limits.
45      */
46     public void writeToVertexBufferFromFloats(final int beginAddress, final float[]
            data)
47             throws MemoryException {
48         this.writeToBufferFromFloats(this.backVertexBuffer, beginAddress, data);
49     }
50
51     /**
52      * Writes float data to the pixel buffer, converting them to bytes before
            storing.
53      *
54      * @param beginAddress The starting index where data is to be written.
55      * @param data The float data to be converted and written.
56      * @throws MemoryException If the write operation exceeds buffer limits.
57      */
58     public void writeToPixelBufferFromFloats(final int beginAddress, final float[]
            data)
59             throws MemoryException {
60         this.writeToBufferFromFloats(this.backPixelBuffer, beginAddress, data);
61     }
62
63     /**
64      * Writes float data to the buffer, converting them to bytes before storing.
65      *
66      * @param buffer The buffer to write data to.
67      * @param beginAddress The starting index where data is to be written.
68      * @param data The float data to be converted and written.
69      * @throws MemoryException If the write operation exceeds buffer limits.
70      */
71     public void writeToBufferFromFloats(
72             final byte[] buffer, final int beginAddress, final float[] data) throws
                MemoryException {
73
74         checkAddressRange(beginAddress, data.length, buffer);
75
76         ByteBuffer byteBuffer = ByteBuffer.allocate(data.length * 4).order(ByteOrder.
                nativeOrder());
77         FloatBuffer floatBuffer = byteBuffer.asFloatBuffer();
78         floatBuffer.put(data);
79
80         byteBuffer.rewind();
81         byteBuffer.get(buffer, beginAddress, byteBuffer.remaining());
82     }
83
84     /**
85      * Checks if the address range is valid for the given data length.
86      *
87      * @param beginAddress The starting index in the buffer.
88      * @param data The length of the data to be written.
89      * @param backBuffer The buffer to write data to.
90      * @throws MemoryException If the address range is invalid.
91      */
92     private void checkAddressRange(int beginAddress, int data, byte[] backBuffer)
93             throws MemoryException {
94         int endAddress = beginAddress + data;
```

```
95      if (beginAddress < 0 || endAddress > backBuffer.length) {
96        throw new MemoryException(
97            "Invalid data positions or data length. (beginAddress: "
98                + beginAddress
99                + ", endAddress: "
100               + endAddress
101               + ")");
102     }
103   }
104
105   /** Swaps the front and back buffers, promoting the back to front for display.
        */
106   public void swap() {
107
108     byte[] temp = frontVertexBuffer;
109     frontVertexBuffer = backVertexBuffer;
110     backVertexBuffer = temp;
111
112     temp = frontPixelBuffer;
113     frontPixelBuffer = backPixelBuffer;
114     backPixelBuffer = temp;
115   }
116
117   /**
118    * Retrieves the render data from the front buffer.
119    *
120    * @return A RenderDataDto object containing the vertex and pixel data.
121    * @throws MemoryException If invalid data positions are used.
122    */
123   public RenderDataDto getRenderData() throws MemoryException {
124     return RenderDataDto.builder()
125         .vertex(readFromVertexBufferAsFloats(0, bufferSize))
126         .pixel(readFromPixelBufferAsFloats(0, bufferSize))
127         .build();
128   }
129
130   /**
131    * Reads a segment of the front buffer as integer data.
132    *
133    * @param beginAddress The starting index in the buffer.
134    * @param endAddress The ending index in the buffer.
135    * @return An array of integers read from the buffer.
136    * @throws MemoryException If invalid data positions are used.
137    */
138   public float[] readFromPixelBufferAsFloats(final int beginAddress, final int
        endAddress)
139       throws MemoryException {
140     return this.readFromBufferAsFloats(frontPixelBuffer, beginAddress, endAddress)
          ;
141   }
142
143   /**
144    * Reads a segment of the front buffer as float data.
145    *
146    * @param beginAddress The starting index in the buffer.
147    * @param endAddress The ending index in the buffer.
148    * @return An array of floats read from the buffer.
149    * @throws MemoryException If invalid data positions are used.
150    */
```

```
151   public float[] readFromVertexBufferAsFloats(final int beginAddress, final int
          endAddress)
152       throws MemoryException {
153     return this.readFromBufferAsFloats(frontVertexBuffer, beginAddress, endAddress
            );
154   }
155
156   /**
157    * Reads a segment of the given buffer as float data.
158    *
159    * @param beginAddress The starting index in the buffer.
160    * @param endAddress The ending index in the buffer.
161    * @return An array of floats read from the buffer.
162    * @throws MemoryException If invalid data positions are used.
163    */
164   public float[] readFromBufferAsFloats(
165       final byte[] buffer, final int beginAddress, final int endAddress) throws
              MemoryException {
166
167     int length = endAddress - beginAddress;
168
169     final ByteBuffer byteBuffer = getByteBufferFromBuffer(buffer, beginAddress,
            endAddress);
170
171     FloatBuffer floatBuffer = byteBuffer.asFloatBuffer();
172     float[] floatArray = new float[length];
173     floatBuffer.get(floatArray, 0, length);
174
175     return floatArray;
176   }
177
178   /**
179    * Retrieves a ByteBuffer from the given buffer with specified start and end
           positions.
180    *
181    * @param buffer The byte array buffer.
182    * @param beginAddress The starting index in the buffer.
183    * @param endAddress The ending index in the buffer.
184    * @return A ByteBuffer positioned at the specified data range.
185    * @throws MemoryException If invalid data positions are used.
186    */
187   private ByteBuffer getByteBufferFromBuffer(
188       final byte[] buffer, final int beginAddress, final int endAddress) throws
              MemoryException {
189
190     if (beginAddress < 0 || endAddress > buffer.length || beginAddress >=
            endAddress) {
191       throw new MemoryException(
192           "Invalid data positions or data length. (beginAddress: "
193               + beginAddress
194               + ", endAddress: "
195               + endAddress
196               + ")");
197     }
198
199     ByteBuffer byteBuffer = ByteBuffer.wrap(buffer);
200     byteBuffer.order(ByteOrder.nativeOrder());
201     byteBuffer.position(beginAddress);
202     return byteBuffer;
```

```
203   }
204 }
```

Listing 5: Classe FrameBuffer

5.3 Implementar a Classe `MemoryException.java`

Crie a classe `MemoryException.java` no pacote memory.

```java
package com.seu_nome.memory;

/** Custom exception class to handle memory-related errors. */
public class MemoryException extends RuntimeException {

  /**
   * Constructs a new MemoryException with a specified detail message.
   *
   * @param message The detail message that explains the reason for the exception.
   */
  public MemoryException(String message) {

    super(message); // Call superclass constructor with the provided message
  }
}
```

Listing 6: Classe MemoryException

5.4 Implementar a Classe `Window.java`

Crie a classe `Window.java` no pacote gpu.

```java
package com.seu_nome.gpu;

import java.nio.ByteBuffer;
import java.nio.IntBuffer;
import lombok.Getter;
import lombok.RequiredArgsConstructor;
import org.lwjgl.glfw.GLFW;
import org.lwjgl.glfw.GLFWFramebufferSizeCallbackI;
import org.lwjgl.glfw.GLFWImage;
import org.lwjgl.glfw.GLFWVidMode;
import org.lwjgl.opengl.GL;
import org.lwjgl.stb.STBImage;
import org.lwjgl.system.MemoryStack;

/** Handles the creation and management of a window using GLFW. */
@Getter
@RequiredArgsConstructor
public class Window {

  @Getter private static long window; // Native handle to the GLFW window
  private final int width; // Width of the window
  private final int height; // Height of the window
  private final String title; // Title of the window

  /** Initializes and creates a window. Throws IllegalStateException if window
      creation fails. */
  public void init() {
```

```
27      // Create a new GLFW window
28      window = GLFW.glfwCreateWindow(width, height, title, 0, 0);
29      if (window == 0) {
30        throw new IllegalStateException("Failed to create window");
31      }
32
33      // Center the window on the screen
34      GLFWVidMode vidMode = GLFW.glfwGetVideoMode(GLFW.glfwGetPrimaryMonitor());
35      assert vidMode != null; // Ensure video mode is available
36      GLFW.glfwSetWindowPos(window, (vidMode.width() - width) / 2, (vidMode.height()
            - height) / 2);
37
38      // Make the OpenGL context current on this window
39      GLFW.glfwMakeContextCurrent(window);
40      // Enable v-sync
41      GLFW.glfwSwapInterval(1);
42      // Show the window
43      GLFW.glfwShowWindow(window);
44      // Create capabilities for OpenGL
45      GL.createCapabilities();
46    }
47
48    /**
49     * Sets a resize callback for the window.
50     *
51     * @param callback A callback to handle framebuffer size changes.
52     */
53    public void setResizeCallback(GLFWFramebufferSizeCallbackI callback) {
54
55      GLFW.glfwSetFramebufferSizeCallback(window, callback);
56    }
57
58    /** Sets the window icon. */
59    protected void setIcon() {
60
61      try (MemoryStack stack = MemoryStack.stackPush()) {
62        // Load the window icon image
63        IntBuffer w = stack.mallocInt(1);
64        IntBuffer h = stack.mallocInt(1);
65        IntBuffer comp = stack.mallocInt(1);
66        ByteBuffer icon = STBImage.stbi_load("src/main/resources/images/icon.png", w
              , h, comp, 4);
67        if (icon == null) {
68          return;
69        }
70        GLFWImage.Buffer icons = GLFWImage.malloc(1);
71        icons.position(0).width(w.get(0)).height(h.get(0)).pixels(icon);
72        // Set the window icon
73        GLFW.glfwSetWindowIcon(window, icons);
74        STBImage.stbi_image_free(icon);
75      }
76    }
77
78    /**
79     * Checks if the window should be closed.
80     *
81     * @return true if the window should close, false otherwise.
82     */
83    public boolean shouldClose() {
```

```
84
85      return GLFW.glfwWindowShouldClose(window);
86    }
87
88    /** Swaps the front and back buffers of the window. */
89    public void swapBuffers() {
90
91      GLFW.glfwSwapBuffers(window);
92    }
93
94    /** Processes all pending GLFW events. */
95    public void pollEvents() {
96
97      GLFW.glfwPollEvents();
98    }
99
100   /** Destroys the window and releases resources. */
101   public void cleanup() {
102
103     GLFW.glfwDestroyWindow(window);
104   }
105 }
```

Listing 7: Classe Window

5.5 Implementar a Classe `ShaderProgram.java`

Crie a classe `ShaderProgram.java` no pacote gpu.

```
1  package com.seu_nome.gpu;
2
3  import java.nio.ByteBuffer;
4  import java.nio.IntBuffer;
5  import lombok.Getter;
6  import lombok.RequiredArgsConstructor;
7  import org.lwjgl.glfw.GLFW;
8  import org.lwjgl.glfw.GLFWFramebufferSizeCallbackI;
9  import org.lwjgl.glfw.GLFWImage;
10 import org.lwjgl.glfw.GLFWVidMode;
11 import org.lwjgl.opengl.GL;
12 import org.lwjgl.stb.STBImage;
13 import org.lwjgl.system.MemoryStack;
14
15 /** Handles the creation and management of a window using GLFW. */
16 @Getter
17 @RequiredArgsConstructor
18 public class Window package com.seu_nome.gpu;
19
20 import java.nio.IntBuffer;
21 import org.lwjgl.BufferUtils;
22 import org.lwjgl.opengl.GL46;
23
24 /** Manages the compilation, linking, and usage of a shader program in OpenGL. */
25 public class ShaderProgram {
26
27   private int programId; // Identifier for the compiled shader program
28
29   /** Loads and compiles vertex and fragment shaders, links them into a program.
       */
```

```
30    public void loadShaders() {
31      // Compile the vertex shader
32      int vertexShader =
33          compileShader(
34              GL46.GL_VERTEX_SHADER,
35              """
36                     #version 460
37                       layout (location = 0) in vec2 vertexPosition;
38                       layout (location = 1) in vec4 vertexColor;
39                       layout (location = 2) in vec2 texCoord;
40                       out vec2 TexCoord;
41                       out vec4 outColor;
42                       void main() {
43                           gl_Position = vec4(vertexPosition, 0.0, 1.0);
44                           TexCoord = texCoord;
45                           outColor = vertexColor;
46                       }
47                  """);
48
49      // Compile the fragment shader
50      int fragmentShader =
51          compileShader(
52              GL46.GL_FRAGMENT_SHADER,
53              """
54                      #version 460
55                       in vec2 TexCoord;
56                       in vec4 outColor;
57                       out vec4 FragColor;
58                       uniform sampler2D ourTexture;
59                       void main() {
60                           FragColor = texture(ourTexture, TexCoord) * outColor;
61                       }
62                  """);
63
64      // Create the shader program and attach the compiled shaders
65      programId = GL46.glCreateProgram();
66      GL46.glAttachShader(programId, vertexShader);
67      GL46.glAttachShader(programId, fragmentShader);
68      GL46.glLinkProgram(programId); // Link the shaders into a usable program
69      checkLinkStatus(programId); // Check for errors in linking
70
71      // Clean up the individual shaders as they are no longer needed after linking
72      GL46.glDeleteShader(vertexShader);
73      GL46.glDeleteShader(fragmentShader);
74    }
75
76    /**
77     * Compiles a shader from source code.
78     *
79     * @param type The type of shader to compile (GL_VERTEX_SHADER or
          GL_FRAGMENT_SHADER).
80     * @param source The GLSL source code for the shader.
81     * @return The compiled shader's identifier.
82     */
83    private int compileShader(int type, String source) {
84
85      int shader = GL46.glCreateShader(type); // Create the shader
86      GL46.glShaderSource(shader, source); // Set the source code
87      GL46.glCompileShader(shader); // Compile the shader
```

```
88      checkCompileStatus(shader); // Check for compilation errors
89      return shader;
90    }
91
92    /**
93     * Checks the link status of the shader program.
94     *
95     * @param program The program identifier.
96     */
97    private void checkLinkStatus(int program) {
98
99      IntBuffer status = BufferUtils.createIntBuffer(1); // Buffer for reading
            status
100     GL46.glGetProgramiv(program, GL46.GL_LINK_STATUS, status);
101     if (status.get(0) == GL46.GL_FALSE) {
102       // If linking failed, throw an exception with the log
103       throw new RuntimeException(
104           String.format("Program link error: %s", GL46.glGetProgramInfoLog(program
                ));
105     }
106   }
107
108   /**
109    * Checks the compilation status of a shader.
110    *
111    * @param shader The shader identifier.
112    */
113   private void checkCompileStatus(int shader) {
114
115     IntBuffer status = BufferUtils.createIntBuffer(1); // Buffer for reading
            status
116     GL46.glGetShaderiv(shader, GL46.GL_COMPILE_STATUS, status);
117     if (status.get(0) == GL46.GL_FALSE) {
118       // If compilation failed, throw an exception with the log
119       throw new RuntimeException(
120           String.format("Shader compile error: %s", GL46.glGetShaderInfoLog(shader
                ));
121     }
122   }
123
124   /** Activates this shader program for use in rendering. */
125   public void use() {
126
127     GL46.glUseProgram(programId);
128   }
129
130   /** Cleans up resources associated with the shader program. */
131   public void cleanup() {
132
133     GL46.glDeleteProgram(programId);
134   }
135 }
```

Listing 8: Classe ShaderProgram

5.6 Implementar a Classe `RenderData.java`

Crie a classe `RenderData.java` no pacote gpu.

```java
package com.seu_nome.gpu;

import com.seu_nome.dto.RenderDataDto;
import com.seu_nome.memory.FrameBuffer;
import java.nio.FloatBuffer;
import lombok.extern.java.Log;
import org.lwjgl.opengl.GL46;

/** Handles the setup, updating, and drawing of render data for OpenGL. */
@Log
public abstract class RenderData extends Thread {

  protected final int width, height; // Dimensions for the texture

  protected final int bufferSize = FrameBuffer.getBufferSize(); // Size of the
      buffer

  protected final int numVertices = bufferSize / 8; // Number of vertices to draw

  protected int vao, vbo, textureId; // OpenGL object identifiers

  protected int[] pboIds; // Array of Pixel Buffer Object identifiers

  protected int nextPboIndex = 0; // Index of the next PBO to use

  /**
   * Constructs a RenderData instance with specified texture dimensions.
   *
   * @param width the width of the texture
   * @param height the height of the texture
   */
  public RenderData(final int width, final int height) {

    this.width = width;
    this.height = height;
  }

  /** Sets up OpenGL settings and initializes textures, buffers, and array objects
      . */
  protected void setup() {

    GL46.glEnable(GL46.GL_TEXTURE_2D);
    GL46.glPixelStorei(GL46.GL_UNPACK_ALIGNMENT, 4);
    setupTexture();
    setupVAOAndVBO();
    setupPBOs();
    GL46.glPointSize(4.0f);
  }

  /** Initializes the texture settings and allocates texture memory. */
  private void setupTexture() {

    textureId = GL46.glGenTextures();
    GL46.glBindTexture(GL46.GL_TEXTURE_2D, textureId);
    GL46.glTexParameteri(GL46.GL_TEXTURE_2D, GL46.GL_TEXTURE_WRAP_S, GL46.
        GL_REPEAT);
    GL46.glTexParameteri(GL46.GL_TEXTURE_2D, GL46.GL_TEXTURE_WRAP_T, GL46.
        GL_REPEAT);
```

```
55      GL46.glTexParameteri(GL46.GL_TEXTURE_2D, GL46.GL_TEXTURE_MIN_FILTER, GL46.
            GL_NEAREST);
56      GL46.glTexParameteri(GL46.GL_TEXTURE_2D, GL46.GL_TEXTURE_MAG_FILTER, GL46.
            GL_NEAREST);
57      GL46.glTexImage2D(
58          GL46.GL_TEXTURE_2D,
59          0,
60          GL46.GL_RGBA,
61          width,
62          height,
63          0,
64          GL46.GL_RGBA,
65          GL46.GL_FLOAT,
66          (FloatBuffer) null);
67    }
68
69    /** Sets up the Vertex Array Object (VAO) and Vertex Buffer Object (VBO). */
70    private void setupVAOAndVBO() {
71
72      vao = GL46.glGenVertexArrays();
73      GL46.glBindVertexArray(vao);
74      vbo = GL46.glGenBuffers();
75      GL46.glBindBuffer(GL46.GL_ARRAY_BUFFER, vbo);
76
77      int stride = 8 * Float.BYTES;
78      GL46.glVertexAttribPointer(0, 2, GL46.GL_FLOAT, false, stride, 0);
79      GL46.glEnableVertexAttribArray(0);
80      GL46.glVertexAttribPointer(1, 4, GL46.GL_FLOAT, false, stride, 2 * Float.BYTES
            );
81      GL46.glEnableVertexAttribArray(1);
82      GL46.glVertexAttribPointer(2, 2, GL46.GL_FLOAT, false, stride, 6 * Float.BYTES
            );
83      GL46.glEnableVertexAttribArray(2);
84    }
85
86    /** Sets up the Pixel Buffer Objects (PBOs) for efficient texture streaming. */
87    private void setupPBOs() {
88
89      int pboCount = 2;
90      pboIds = new int[pboCount];
91      GL46.glGenBuffers(pboIds);
92      for (int i = 0; i < pboCount; i++) {
93        GL46.glBindBuffer(GL46.GL_PIXEL_UNPACK_BUFFER, pboIds[i]);
94        GL46.glBufferData(GL46.GL_PIXEL_UNPACK_BUFFER, bufferSize, GL46.
              GL_STREAM_DRAW);
95      }
96    }
97
98    /**
99     * Draws the vertex data and updates the texture.
100    *
101    * @param dataDto the object containing the vertex and pixel data
102    */
103   protected void draw(RenderDataDto dataDto) {
104     // Update the VBO with new data
105     GL46.glBindBuffer(GL46.GL_ARRAY_BUFFER, vbo);
106     GL46.glBufferData(GL46.GL_ARRAY_BUFFER, dataDto.vertex(), GL46.GL_STREAM_DRAW)
            ;
107
```

```
108     int pboId = pboIds[nextPboIndex];
109     nextPboIndex = (nextPboIndex + 1) % pboIds.length;
110 
111     GL46.glBindBuffer(GL46.GL_PIXEL_UNPACK_BUFFER, pboId);
112     GL46.glBufferData(GL46.GL_PIXEL_UNPACK_BUFFER, dataDto.pixel(), GL46.
            GL_STREAM_DRAW);
113 
114     GL46.glUnmapBuffer(GL46.GL_PIXEL_UNPACK_BUFFER);
115 
116     GL46.glTexSubImage2D(
117         GL46.GL_TEXTURE_2D, 0, 0, 0, width, height, GL46.GL_RGBA, GL46.GL_FLOAT,
                0);
118 
119     GL46.glGenerateMipmap(GL46.GL_TEXTURE_2D);
120 
121     GL46.glDrawArrays(GL46.GL_POINTS, 0, numVertices);
122   }
123 
124   /**
125    * Cleans up resources upon shutdown, ensuring graceful termination of GLFW and
           other components.
126    */
127   protected void cleanup() {
128     GL46.glDeleteBuffers(vbo);
129     GL46.glDeleteVertexArrays(vao);
130     GL46.glDeleteTextures(textureId);
131     GL46.glDeleteBuffers(pboIds);
132   }
133 }
```

Listing 9: Classe RenderData

5.7 Implementar a Classe `GPU.java`

Crie a classe `GPU.java` no pacote gpu.

```
1  package com.seu_nome.gpu;
2 
3  import com.seu_nome.memory.FrameBuffer;
4  import com.seu_nome.memory.MemoryException;
5  import lombok.Getter;
6  import org.lwjgl.glfw.GLFW;
7  import org.lwjgl.opengl.GL46;
8 
9  /** Represents a GPU component that handles rendering operations. */
10 public class GPU extends RenderData {
11 
12   @Getter private static int width;
13 
14   @Getter private static int height;
15 
16   private final FrameBuffer frameBuffer;
17 
18   private ShaderProgram shaderProgram;
19 
20   private Window window;
21 
22   /**
23    * Constructs a new GPU instance with specified dimensions and framebuffer.
```

```
24      *
25      * @param width the width of the render window.
26      * @param height the height of the render window.
27      * @param frameBuffer the framebuffer to use for rendering.
28      */
29     public GPU(final int width, final int height, final FrameBuffer frameBuffer) {
30       super(width, height);
31
32       GPU.width = width;
33       GPU.height = height;
34       this.frameBuffer = frameBuffer;
35     }
36
37     /** The main run loop of the GPU component, handling initialization and
           rendering. */
38     @Override
39     public void run() {
40
41       init();
42       while (isRunning()) {
43         try {
44           render();
45         } catch (MemoryException e) {
46           throw new RuntimeException(e);
47         }
48       }
49       cleanup();
50     }
51
52     /**
53      * Initializes the necessary components including window, shader program, and
            other render data.
54      */
55     private void init() {
56
57       if (!GLFW.glfwInit()) {
58         throw new IllegalStateException("Failed to initialize GLFW");
59       }
60
61       window = new Window(width, height, "Emulator");
62       window.init();
63       window.setIcon();
64       GL46.glViewport(0, 0, width, height);
65       window.setResizeCallback(
66           (ignore, newWidth, newHeight) -> GL46.glViewport(0, 0, newWidth, newHeight
                ));
67
68       shaderProgram = new ShaderProgram();
69       shaderProgram.loadShaders();
70       shaderProgram.use();
71
72       setup();
73
74       GL46.glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
75     }
76
77     /**
78      * Checks if the window is still open and the rendering should continue.
79      *
```

```
80      * @return true if the window is not marked to close, false otherwise
81      */
82     private boolean isRunning() {
83
84         return !window.shouldClose();
85     }
86
87     /**
88      * Handles the rendering of each frame to the window.
89      *
90      * @throws MemoryException if there's an issue accessing frame data
91      */
92     private void render() throws MemoryException {
93
94         GL46.glClear(GL46.GL_COLOR_BUFFER_BIT | GL46.GL_DEPTH_BUFFER_BIT);
95
96         draw(frameBuffer.getRenderData());
97
98         window.swapBuffers();
99         window.pollEvents();
100    }
101
102    /**
103     * Cleans up resources upon shutdown, ensuring graceful termination of GLFW and
            other components.
104     */
105    protected void cleanup() {
106        super.cleanup();
107
108        shaderProgram.cleanup();
109        window.cleanup();
110    }
111 }
```

Listing 10: Classe GPU

5.8 Implementar a Classe `VideoFrameToVertexArray.java`

Crie a classe `VideoFrameToVertexArray.java` no pacote gpu.

```
1  package com.seu_nome.gpu;
2
3  import com.seu_nome.memory.FrameBuffer;
4  import com.seu_nome.memory.MemoryException;
5  import java.awt.*;
6  import java.awt.image.BufferedImage;
7  import lombok.RequiredArgsConstructor;
8  import lombok.extern.java.Log;
9  import org.bytedeco.javacv.FFmpegFrameGrabber;
10 import org.bytedeco.javacv.Frame;
11 import org.bytedeco.javacv.Java2DFrameConverter;
12
13 /** This thread processes a video file, converting frames to vertex arrays for
       rendering. */
14 @Log // Lombok annotation for logging
15 @RequiredArgsConstructor // Lombok generates a constructor for all final fields
16 public class VideoFrameToVertexArray extends Thread {
17
18     private final String videoFilePath; // Path to the video file
```

```
19
20    private final Java2DFrameConverter converter =
21        new Java2DFrameConverter(); // Converter for frames to images
22
23    private final int width; // Width of the target rendering
24
25    private final int height; // Height of the target rendering
26
27    private final FrameBuffer frameBuffer; // Frame buffer to write the converted
          frames
28
29    /**
30     * Resizes a BufferedImage to the specified dimensions.
31     *
32     * @param originalImage The original BufferedImage.
33     * @param targetWidth The desired width.
34     * @param targetHeight The desired height.
35     * @return A new resized BufferedImage.
36     */
37    private static BufferedImage resizeImage(
38        BufferedImage originalImage, int targetWidth, int targetHeight) {
39
40      BufferedImage resizedImage =
41          new BufferedImage(targetWidth, targetHeight, BufferedImage.TYPE_INT_ARGB);
42      Graphics2D g2d = resizedImage.createGraphics();
43      g2d.drawImage(originalImage, 0, 0, targetWidth, targetHeight, null);
44      g2d.dispose();
45      return resizedImage;
46    }
47
48    /** Entry point for the thread; begins the video processing. */
49    @Override
50    public void run() {
51
52      this.processVideo();
53    }
54
55    /** Processes each frame of the video, converting and writing to the frame
          buffer. */
56    private void processVideo() {
57
58      try (FFmpegFrameGrabber grabber = new FFmpegFrameGrabber(videoFilePath)) {
59        grabber.start();
60        Frame frame;
61
62        while ((frame = grabber.grabImage()) != null) {
63          long time = System.currentTimeMillis();
64
65          processFrameAndWriteInBuffer(frame);
66
67          time = System.currentTimeMillis() - time;
68          long sleepTime =
69              Math.max(0, 1000 / 60 - time); // Calculate time to delay to maintain
                  frame rate
70          try {
71            Thread.sleep(sleepTime);
72          } catch (InterruptedException e) {
73            log.severe(e.getMessage());
74            return;
```

```
        }
      }
      grabber.stop();
      this.processVideo(); // Restart video processing to loop continuously
    } catch (Exception e) {
      throw new RuntimeException(String.format("Error processing video: %s", e.
          getMessage()));
    }
  }

  /**
   * Processes a single frame, resizing and mapping it into the frame buffer.
   *
   * @param frame The frame to be processed.
   * @throws MemoryException If there's an issue writing to the frame buffer.
   */
  private void processFrameAndWriteInBuffer(Frame frame) throws MemoryException {

    BufferedImage originalImage = converter.getBufferedImage(frame);
    BufferedImage resizedImage = resizeImage(originalImage, width, height);
    int address = 0;
    for (int y = 0; y < height; y++) {
      for (int x = 0; x < width; x++) {
        float normX = (x / (float) width) * 2 - 1;
        float normY = ((height - y) / (float) height) * 2 - 1;

        int color = resizedImage.getRGB(x, y);

        float r = ((color >> 16) & 0xFF) / 255.0f;
        float g = ((color >> 8) & 0xFF) / 255.0f;
        float b = (color & 0xFF) / 255.0f;
        float u = x / (float) width;
        float v = y / (float) height;

        frameBuffer.writeToPixelBufferFromFloats(address * 4, new float[] {r, g, b
            , 1});

        frameBuffer.writeToVertexBufferFromFloats(
            (y * width + x) * 32, new float[] {normX, normY, r, g, b, 1, u, v});

        address += 4;
      }
    }
    frameBuffer.swap();
  }
}
```

Listing 11: Classe VideoFrameToVertexArray

5.9 Criar a Classe Principal `Main.java`

Dentro do pacote `com.seu_nome.gpu`, crie a classe `Main.java`:

```
package com.seu_nome;

import com.seu_nome.gpu.GPU;
import com.seu_nome.gpu.VideoFrameToVertexArray;
import com.seu_nome.memory.FrameBuffer;
import lombok.extern.java.Log;
```

```java

@Log
public class Main {

  private static final int WIDTH = 1080;

  private static final int HEIGHT = 720;

  private static final int FRAME_BUFFER_SIZE = WIDTH * HEIGHT * 4; // 4 bytes per
      pixel

  public static void main(String[] args) {

    final FrameBuffer frameBuffer = new FrameBuffer(FRAME_BUFFER_SIZE);

    if (args.length < 1) {
      throw new IllegalArgumentException("Video file path not provided.");
    }

    final VideoFrameToVertexArray videoFrameToVertexArray =
        new VideoFrameToVertexArray(args[0], WIDTH, HEIGHT, frameBuffer);
    GPU gpu = new GPU(WIDTH, HEIGHT, frameBuffer);

    videoFrameToVertexArray.start();
    gpu.start();

    while (gpu.isAlive()) {
      if (gpu.getState() == Thread.State.TERMINATED) {
        gpu.interrupt();
        videoFrameToVertexArray.interrupt();
      }
    }
  }
}
```

Listing 12: Classe Main.java

5.10 5.3. Ajustes Finais

- Certifique-se de que o caminho para o arquivo de vídeo (`videoPath`) está correto.
- Verifique se todas as dependências estão sendo corretamente baixadas pelo Maven.
- Certifique-se de que todos os pacotes estão com os nomes corretos (substitua "seu_nome").

6 Executar a Aplicação

6.1 6.1. Compilar e Executar o Projeto

- **Eclipse**:
  - Clique com o botão direito na classe `Main` → `Run As` → `Java Application`.
- **IntelliJ IDEA**:
  - Abra a classe `Main` e clique no ícone de execução ao lado do método `main`.
- **NetBeans**:

– Clique com o botão direito na classe `Main` → `Run File`.

A aplicação será compilada e executada, e a janela OpenGL irá reproduzir o vídeo especificado.

Referências

- Repositório gpu
- LWJGL - Site Oficial
- JavaCPP Presets - FFmpeg
- Apache Maven - Site Oficial

Conclusão

Neste tutorial, você aprendeu a criar uma aplicação Java que renderiza vídeos utilizando OpenGL e FFmpeg, seguindo o código do repositório fornecido. Utilizamos o Maven para gerenciar as dependências, o que simplifica o processo de configuração do projeto. Com este conhecimento, você pode expandir a aplicação, adicionando funcionalidades como controle de reprodução, suporte a diferentes formatos de vídeo e efeitos visuais.

Explore o repositório gpu para entender melhor o código e as possíveis extensões que você pode implementar.