Die Hausaufgaben müssen von jedem Studierenden einzeln bearbeitet und abgegeben werden. Für die Hausaufgabe sind die aktuellen Informationen vom Blog `https://seblog.cs.uni-kassel.de/ws2122/programming-and-modelling/` zu berücksichtigen.

Abgabefrist ist der 02.12.2021 - 23:59 Uhr

Abgabe

Wir benutzen für die Abgabe der Hausaufgaben Git. Jedes Repository ist nur für den Studierenden selbst sowie für die Betreuer und Korrektoren sichtbar.

Für die Hausaufgabe benötigt ihr ein **neues Repository**, welches über folgenden Link angelegt werden kann, falls nicht bereits geschehen:

`https://classroom.github.com/a/SlVERCvA`

Nicht oder zu spät gepushte (Teil-)Abgaben werden mit 0 Punkten bewertet.

Vorbereitung

Zur Bearbeitung der Hausaufgabe sollte eine Entwicklungsumgebung verwendet werden. Wir empfehlen aufgrund der Nachvollziehbarkeit die Verwendung von VSCode (siehe Aufgabenblatt 3). Die Abgabe muss als **lauffähiges** Projekt abgegeben werden.

Abgaben, die nicht lauffähig sind, werden mit 0 Punkten bewertet!

Zukünftige Abgaben

Das oben genannte Repository ist der Startpunkt für die Anwendung, die im weiteren Verlauf dieser Veranstaltung entwickelt werden soll. Das Repository wird also fortan nicht mehr für jede Hausaufgabe gewechselt, sondern für kommende Abgaben weiterverwendet.

Commit Message-Vorgaben

In dieser und für alle zukünftigen Hausaufgaben führen wir eine Vorgabe für die Commit Messages ein.

Der letzte Commit zu jeder Teilaufgabe soll mit der Commit Message „HA<Number> finished Task <TaskNumber>" versehen sein.

Falls nach diesen Commits eine Aufgabe erneut bearbeitet wird, z. B. um einen Fehler zu korrigieren, soll die jeweilige Commit Message einfach noch einmal als letzte Commit Message verwendet werden.

Das Ignorieren der Vorgaben wird mit 0 Punkten bewertet!

Project Setup

In dieser Aufgabe nutzen wir die Funktionalität der fulib.org-Plattform. Diese ermöglicht das einfache Erstellen eines Gradle-Projekts. Es stellt somit ein Tool dar, das alle Schritte, die ein herkömmliches Projektsetup (wie in Hausaufgabe 3) mit sich zieht.

`https://fulib.org/`

1. Lösche aus dem Textfeld `Scenario` alles außer `# My Frist Scenario`.
2. Ändere den Text `# My First Scenario` zu `# Nine Men Morris`.

Abbildung 1: Buttons auf fulib.org

3. Lade nun über den in Abbildung 1 gezeigten „Configure or Export"-Button das Projekt herunter. In dem sich öffnenden Fenster trägst du folgende Dinge ein:

Package/Group Name:	de.uniks.pmws2122.model
Project Name:	PMWS2122_NineMen_<GitHubName>
Version:	1.0.0
Scenario File Name:	Scenario.md
Decorator Class Name:	GenModel
Export Project Format:	Gradle Project

   Anschließend kann das Projekt mit dem grünen „Export"-Button als Gradle-Projekt in einer .zip-Datei heruntergeladen und entpackt werden.

4. Öffne das Projekt mit VSCode und führe `gradle check` aus. Der Task kann in VSCode im Gradle-Seitenmenü unter Tasks > verification gefunden werden.

Committe und pushe die Änderungen an deinem Gradle-Projekt abschließend auf den main-Branch.

Achte darauf, das Repository der aktuellen Hausaufgabe zu verwenden.

Aufgabe 1 - Fulib (22P)

In dieser Aufgabe wird das vollständige Datenmodell des Spiels "Nine men's morris" mit der Hilfe von Fulib generiert. Wie der Name bereits vermuten lässt, ist Fulib die treibende Technologie hinter Fulib.org. Diese kann in bestehenden Projekten genutzt werden, um Definitionen für das Datenmodell zu tätigen.

Wir nutzen die bereits generierte Klasse GenModel, um das Klassendiagramm aus Abbildung 2 zu definieren sowie zu generieren.

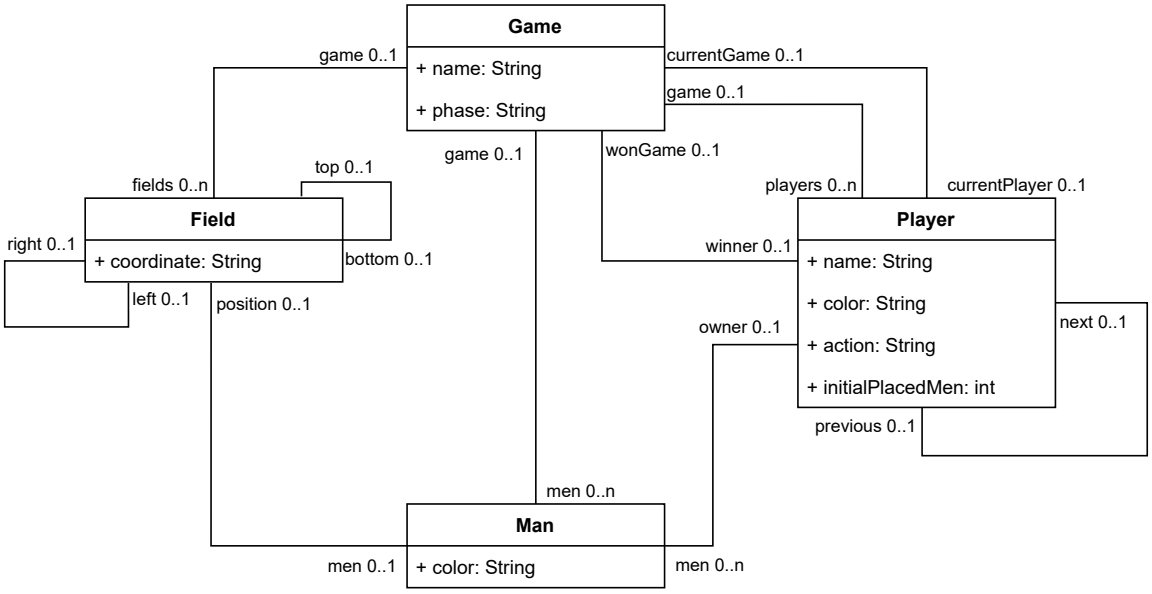

Abbildung 2: „Nine men's morris"-Klassendiagramm

Hinweis:

Für die Variablen action und phase werden in einer zukünftigen Aufgabe Konstanten definiert. Für Hausaufgabe 4 definieren wir diese vorläufig als String-Attribut.

Committe und pushe die Änderungen an deinem Gradle-Projekt abschließend auf den main-Branch.

Bei der Bewertung wird vor allem auf die Erweiterung des Modells geachtet.

Achte darauf, das Repository der aktuellen Hausaufgabe zu verwenden.

Aufgabe 2 - Initialisierung (25P)

In dieser Aufgabe soll die buildGame()-Methode implementiert werden:

- Erstelle eine Klasse ModelService unter src/main/java im Package de.uniks.pmws2122.model. Diese ist der Ausgangspunkt unserer Logik.
- Erstelle nun die Methode public void buildGame(String playerNameOne, String playerNameTwo) im ModelService. Nur innerhalb dieser Methode dürfen Instanzen einer Klasse initialisiert werden.
- Die Methode muss folgende Objekte anlegen und verknüpfen:
  - Ein Game-Objekt mit dem Namen "Epic battle". Das Game-Objekt muss als private Variable im ModelService gespeichert werden.
  - Zwei Player-Objekte mit den übergebenen Namen, den jeweiligen Farben Schwarz und Weiß, einer leeren Aktion, sowie Verbindungen zum Game-Objekt und Verbindungen untereinander durch die Assoziationen next und previous.
  - Alle Felder mit Koordinaten und korrekter Verknüpfung durch die Assoziationen top, left, bottom und right. (Siehe Wikipedia[1])
- Alle Objekte müssen mit dem Game-Objekt verbunden sein.
- Erstelle eine Test-Methode testBuildGame in der bereits generierten Klasse ScenarioTest.java unter src/test/java im Package de.uniks.pmws2122.model. Annotiere die Methode mit @Test.
- Implementiere den Test. Der Test soll prüfen, ob alle Objekte korrekt erstellt und verknüpft worden sind. Nutze dafür das Game-Objekt aus dem ModelService. Dieses Objekt sollte eine Verbindung zu allen anderen Objekten haben. Nutze sinnvolle asserts um die korrekte Initialisierung zu testen.

Committe und pushe die Änderungen an deinem Gradle-Projekt abschließend auf den main-Branch.

Bei der Bewertung wird vor allem auf die vollständige Umsetzung der Spielsituation geachtet.

Achte darauf, das Repository der aktuellen Hausaufgabe zu verwenden.

[1] `https://commons.wikimedia.org/wiki/File:Nine_Men%27s_Morris_board_with_coordinates.svg#/media/File:Nine_Men's_Morris_board_with_coordinates.svg`

Anhang

Es folgt eine Auflistung hilfreicher Webseiten und weiterer Erklärungen zu den Themen dieser Hausaufgabe. Die Links sind als Startpunkt zur selbstständigen Recherche angedacht. Das Durcharbeiten der folgenden Quellen ist kein bewerteter Anteil der Hausaufgaben.

VS Code

- Download: `https://code.visualstudio.com/`
- Erweiterungen: *Extension Pack for Java* und *Gradle Extension Pack*

fulib

- Fulib.org: `https://fulib.org/`
- Fulib-Dokumentation: `https://fulib.org/docs/fulib/README.md`
- Fulib-InnerClasses-Beispiel: `https://github.com/fujaba/fulib/blob/master/test/src/gen/java/de/uniks/studyright/GenModel.java`