Das Projekt muss von jedem Studierenden einzeln bearbeitet und abgegeben werden. Für das Projekt sind die aktuellen Informationen vom Blog https://seblog.cs.uni-kassel.de/ws2223/programming-and-modelling/ zu berücksichtigen.

Abgabefrist ist der 02.02.2023 - 23:59 Uhr

Abgabe

Für das Projekt benötigst du ein **neues** Repository. Dieses kann über folgenden Link eingesehen oder auch erstellt werden, falls nicht bereits geschehen:

https://classroom.github.com/a/2OeNcIrG

Vorbereitung

Mach dich mit dem Spiel „Tic-Tac-Toe" vertraut. Es gelten folgende Eigenschaften[1]:

> Tic-tac-toe is a paper-and-pencil game for two players who take turns marking the spaces in a three-by-three grid with X or O. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner. It is a solved game, with a forced draw assuming best play from both players.

Auf den folgenden Seiten werden die zu bearbeitenden Aufgaben beschrieben. Das Programm darf in soweit frei gestaltet, mit zusätzlicher Funktionalität versehen und zusätzlich getestet werden, sodass die dargelegten Anforderungen erfüllt sind. Durch zusätzliche Funktionalität können allerdings keine Bonuspunkte erzielt werden.

Wir empfehlen, regelmäßig zu committen und zu pushen. Es gibt keine Vorgaben für Commit-Messages und zur Bearbeitungsreihenfolge. Dein Projekt muss vor der Deadline auf den main-Branch gepusht worden sein.

[1] Tic-Tac-Toe - Wikipedia: https://en.wikipedia.org/wiki/Tic-tac-toe

Aufgabe 1. Modellierung

Im Laufe des Projekts soll das Spiel „Tic-Tac-Toe" modelliert und programmiert werden, sodass als Endprodukt ein Programm ensteht, das es zwei Spielern ermöglicht, gegeneinander „Tic-Tac-Toe" über eine grafische Oberfläche zu spielen.

Hierzu sollen zunächst die kennengelernten Methoden zur Modellierung angewendet werden.

Achte auf die vorgegebenen Dateiformate. Von Hand gezeichnete Diagramme oder Wireframes werden nicht akzeptiert. Scenebuilder-Screenshots werden als Wireframes nicht akzeptiert.

Hinweis: Das Spielfeld darf **nicht** als multidimensionales Array implementiert werden, sondern muss aus Objekten bestehen, die die benötigten Beziehungen zueinander haben.

1.1 Szenarien

Schreibe drei Szenarien zu „Tic-Tac-Toe". Die Szenarien müssen in Englisch verfasst sein.

Die Szenarien sollen in deinem Repository im Ordner `modelling` in der Datei `scenarios.md` zu finden sein.

1.2 Objektdiagramme ableiten

Leite aus deinen Szenarien Objektdiagramme für Start- und Endsituation ab. Lege die sechs erstellten pdf-Dateien wie folgt in deinem Repository ab:

`modelling/diagrams/<Szenario-Titel>_<Start bzw. Result>.pdf`

1.3 Klassendiagramm ableiten

Leite aus deinen Objektdiagrammen ein Klassendiagramm ab. Lege die erstellte pdf-Datei wie folgt in deinem Repository ab:

`modelling/diagrams/classdiagram.pdf`

1.4 Wireframes

Erstelle Wireframes für einen SetupScreen und einen IngameScreen deines Spiels. Lege die pdf-Dateien wie folgt in deinem Repository ab:

`modelling/wireframes/Setup.pdf` `modelling/wireframes/Ingame.pdf`

Aufgabe 2. Programmierung

In dieser Aufgabe wird das „Tic-Tac-Toe"-Spiel vervollständigt. Dazu wird die Oberfläche erstellt und die Spiellogik implementiert. Entsprechend des MVC-Patterns sollen sie durch Controller miteinander verknüpft werden. Außerdem soll durch Tests sichergestellt werden, dass dein Programm funktioniert.

Die Anwendung muss unter Verwendung von JavaFX umgesetzt werden.

2.1 Datenmodell generieren

Verwende die bereits in deinem Projekt existierende Klasse GenModel, um dein Klassendiagramm aus Aufgabe 1.3 zu definieren sowie zu generieren.

2.2 FXML-Dateien

Erstelle anhand der in Aufgabe 1 erstellten Wireframes die entsprechenden FXML-Dateien mit dem Scenebuilder. Lege diese unter src/main/resources im Ordner de/uniks/pmws2223/tictactoe/view ab.

2.3 MVC

Implementiere folgende Klassen:

- de.uniks.pmws2223.tictactoe.service.GameService
  enthält benötigte Logik-Methoden für das Spiel
- de.uniks.pmws2223.tictactoe.Constants
  als zentraler Ort für Konstanten
- de.uniks.pmws2223.tictactoe.controller.SetupController
  als Controller für den Setup-Bildschirm
- de.uniks.pmws2223.tictactoe.controller.IngameController
  als Controller für den Ingame-Bildschirm
- de.uniks.pmws2223.tictactoe.controller.GameOverController
  als Controller für den GameOver-Bildschirm
- de.uniks.pmws2223.tictactoe.controller.<THING>Controller
  als benötigte(r) Sub-Controller

Danach sollte deine Anwendung **vollständig** ausführbar und spielbar sein. Starte sie mit der Klasse Main, um das Spiel auszuprobieren und zu spielen, bis ein Sieger feststeht.

2.4 Tests

Lege für drei selbstgewählte Logik-Methoden [2] des `GameServices` je eine eigene Test-Methode im `GameServiceTest` an. Teste in den Test-Methoden die Funktionalität der Logik-Methode durch sinnvolles Testen. Erkläre in jeder Methode in Form von Kommentaren, welche

[2] Hierbei dürfen keine Methoden gewählt werden, die ausschließlich dazu dienen, weitere Methoden aufzurufen. Getter und Setter sind ebenfalls ausgeschlossen, da sie keine Spiellogik beinhalten.

Anforderung an das Verhalten der Logik-Methode geprüft wird. Lege den `GameServiceTest` im Modul `src/test/java` im package de.uniks.pmws2223.tictactoe.service ab.

Implementiere außerdem einen GUI-Test. Lege hierfür die Klasse `FullGameTest` im Modul `src/test/java` im package de.uniks.pmws2223.tictactoe an. Implementiere darin einen Test, der ein Spiel mit zwei Spielern startet und diese so gegeneinander spielen lässt, dass es einen Gewinner (ohne Aufgeben) gibt.

Es dürfen zusätzliche Tests implementiert werden.

Informationen zur Abgabe und Präsentation

- Dein Projekt muss pünktlich abgegeben werden (siehe Deadline).
- Im Laufe der zweiten Bearbeitungswoche wird ein Link gepostet (Ankündigung über Discord), über das sich die Studierenden einen Zeitslot auswählen müssen, in dem sie ihr Projekt präsentieren. Diese Präsentationen werden erst nach der Abgabe stattfinden.
- Die Präsentation wird über Discord stattfinden. Hierfür ist eine Webcam erforderlich, da wir deine Identität prüfen müssen. Halte hierfür deinen Personalausweis und deinen Studierendenausweis bereit.
- Es soll **keine** PowerPoint-Präsentation vorbereitet werden. Die Ergebnisse des Projektes werden am laufenden Programm, am Programmcode und anderen Projektdateien erklärt.
- Innerhalb von ca. 5 Minuten sollen die Ergebnisse von Aufgabe 2 als Demo präsentiert werden.
- Innerhalb von ca. 10 Minuten soll der Programmcode präsentiert werden.
- Die Prüfenden können jederzeit Fragen dabei stellen.
- Die Auswertung wird einige Zeit dauern, daher findet die Notenvergabe erst entsprechend später statt.