Praktikumsaufgaben für VW3

Programmierung interaktiver Systeme im Wintersemester 2023/24
Prof. Dr. Martin Weigel

GUI für Tic-Tac-Toe · **Mit dem Zweiten spielt man besser**

Für Ihre Projektarbeit wird es wesentlich sein, dass Sie im Code eine Trennung vornehmen: Die Anwendungslogik ist der Kern ihrer Anwendung. Sie beinhaltet die Daten und Logik ihres Programms und kann auch eigenständig z.B. in der JShell ausgeführt werden. Die Visualisierung ist nur ein Aufsatz auf diese Anwendungslogik. Dadurch wird es einfacher zusätzliche Benutzerschnittstellen für die Anwendung bereitzustellen (z.B. Webseiten oder Konsolenprogramme). Es vermeidet auch eine zu starke Abhängigkeit von einem GUI-Framework (z.B. Processing), falls Sie dieses irgendwann austauschen möchten.

Das MVC Pattern trennt die Aufgabenbereiche wie folgt:

Model: Enthält die Daten und essentielle Business-Logik der Anwendung. Das Model kennt weder den View noch den Controller und darf auch keine Processing-Funktionen verwenden.

View: Zeichnet die grafische Benutzerschnittstelle (GUI). Leitet Benutzereingaben an den Controller weiter, den der View per Interface kennt. Nur der View darf die Objekte und Funktionen des GUI-Frameworks (also Processing) verwenden. Der View besitzt eine Referenz auf das zu verwendenden *Controller*-Objekt.

Controller: Verarbeitet die Benutzereingaben. Ruft Methoden des Models auf, um Daten zu holen und zu verändern. Entscheidet *was* im View gezeichnet werden soll, aber nicht *wie*. Dadurch ist der Controller das Bindeglied zwischen Daten und Anwendungslogik (=*Model*) und der Benutzerschnittstelle (=*View*). Der Controller besitzt eine Referenz auf das zu verwendende *View- und Model*-Objekt.

1 GUI für Tic-Tac-Toe

Sie finden im Moodle ein IntelliJ IDEA Projekt, welches das Model eines Tic-Tac-Toe Spieles enthält. Versuchen Sie den Code in `TicTacToeModel.java` und die typische Benutzung der Funktionen zu verstehen. Hierfür können Sie die Datei zum Beispiel mit der JShell öffnen (`/open TicTacToeModel.java`) und die Funktionen ausprobieren (siehe Klassen-Kommentar).

Sobald Sie mit dem Code vertraut sind ist es ihre Aufgabe einen passenden *View* und *Controller* für das Spiel zu schreiben. Nur der *View* sollte die Bibliothek von Processing benutzen und darf die Model-Klasse nicht kennen. D.h. weder als Variable noch als Parameter darf das Model (bzw. dessen Objekte) referenziert sein. Der *Controller* darf sowohl den *View* als auch das *Model* kennen. Schauen Sie am Besten in die *Main.java* der aktuellen Übung, um eine Idee zu bekommen wie Sie die MVC-Klassen initialisieren können.

Ihre Anwendung soll es ermöglichen mit der Maus ein Zeichen in eines der neun Felder zu setzen. Es soll angezeigt werden wenn das Spiel beendet ist und ob ein/e Spieler/in gewonnen hat. Geben Sie den Anwender*innen auch eine Möglichkeit ein neues Spiel zu starten.

Aufgaben:

1. Erstellen Sie ein UML-Klassendiagramm aus dem gegebenen TicTacToe-Code. Sie können dieses per Hand zeichnen oder ein Programm wie PlantUML verwenden.
2. Zeichnen Sie die zwei benötigten Interfaces (für View und Controller), die View-Klasse und die Controller-Klasse in das UML-Diagramm. Welche Methoden werden benötigt?
3. Implementieren Sie View und Controller für die Anwendung

> ⓘ Wenn Sie möchten können Sie statt den klassischen 'X' und 'O'-Zeichen auch gerne ihre zwei Lieblingspokemons verwenden. Falls Sie sich nicht entscheiden können befindet sich im `images`-Ordner ein Piepi und ein Tangela Bild. Da der `images`-Ordner als Ressourcen-Verzeichnis markiert wurde, werden die Bilder beim Kompilieren zu ihrer Anwendung gelegt. Sie können die Bilder daher direkt per `loadImage("035.png");` laden.

2 Mit dem Zweiten spielt man besser

Ein Großer Vorteil des MVC-Konzeptes ist die Austauschbarkeit der Benutzerschnittstelle. Implementieren Sie einen zweiten View für das Spiel mit Processing. Der View soll eine andere Darstellung auf das Spiel ermöglichen.

Ideen für ihren zweiten View:

- Unterstützen Sie ein anderes *Farbschema*, z.B. ein Dark-Theme
- Ändern Sie die Sprache (z.B. englischer Text)
- Zeichnen Sie Digimons statt Pokemons
- ...

Ihre beiden Views sollen separat voneinander ausführbar sein, also eigene Main-Methoden besitzen, aber den gleichen Controller und das gleiche Model verwenden.