Hochschule Ravensburg-Weingarten
Technik | Wirtschaft | Sozialwesen

Studiengang Angewandte Informatik
Grafische Bedienoberflächen
Prof. Dr. S. Keller

10/03/2016

ÜBUNGSBLATT 2

Teil 1: Verbinden Sie die Oberfläche aus Aufgabenblatt 2 mit der Spielelogik und befolgen Sie dabei das Model-View-Controller Programmiermodell.

Die Spielelogik wird in der Klasse **DionaRapModel** gekapselt. Die Verwendung erarbeiten Sie sich bitte aus den Javadoc-Helpfiles in Moodle und den Anmerkungen unten.

Die Klassen der Spielelogik sind in einem JAR-Archiv „DionaRap_Model_MT“ enthalten (befindet sich ebenfalls in Moodle), welches Sie in Ihr Projekt einbinden müssen. Benutzen Sie das Kontextmenü Ihres Projektes und öffnen Sie den „*Properties*“ Dialog. Unter dem Reiter „*Libraries*“ fügen Sie die Datei `DionaRap_Model_MT.jar` über den Dialog „*Add External JARs...*“ hinzu.

Erzeugen Sie eine Instanz der Klasse DionaRapModel zum Initialisieren eines Spieles und zeigen Sie auf Ihrem Spielfeld die Position des Spielers, der Hindernisse, der Gegner usw. an.
Die Anzahl der Hindernisse und Gegner können Sie frei wählen. Das Spielfeld soll 10 x 10 groß sein.

Das Panel, in dem das Spielfeld realisiert wird, muss dazu Methoden bereitstellen, die den Spieler, die Gegner, Hindernisse und die durch die Waffe zerstörten Felder zeichnet bzw. wieder löscht. In der ersten Variante soll noch keine Grafik verwendet werden. Ein Spieler wird auf dem Spielfeld durch den Text „S“ dargestellt. Analog verfahren Sie mit Gegnern und Hindernissen (Gegner: „G“, Hindernis: „H“). Nach einem Schuss durch den Spieler werden alle Spielfelder, die von dem Schuss getroffen wurden, mit einer Spielfigur der Klasse **Destruction** belegt. Diese Felder markieren Sie mit einem „*****“. Der Text soll die **inverse Farbe des Hintergrundes** haben und ist damit unabhängig von der Wahl der Hintergrundfarbe stets sichtbar. *(Hinweis: Hilfestellung dazu erhalten Sie im Moodle-Kurs, Wiki)*

Die Vortexe liegen in den vier Eckfeldern sollen farbig markiert werden.

Alle Spielfiguren erben von der Klasse AbstractPawn und können mit der Methode getAllPawns() vom DionaRapModel abgefragt werden. Die Methode liefert ein Array mit allen Spielfiguren zurück. Durchlaufen Sie also das Array und setzen Sie die Figuren auf das Spielfeld. Jede Spielfigur gibt über eine get-Methoden getX() und getY() seine Koordinaten zurück.

Teil 2: Geben Sie Ihrem Spiel Leben, indem Sie auf Interaktionen reagieren

2. Reagieren Sie auf eine Bewegungseingabe über die Spieletastatur im Navigationfenster. Dazu ist ein Listener in einer eigenen Klasse zu realisieren, der auf Events der Buttons 1 – 4 und 6 - 9 reagiert.

3. Schreiben Sie eine Listenerklasse, die auf die Taste 5, das Schießen des Spielers, reagiert.

4. Implementieren Sie einen KeyListener, damit Ihr Spiel auch mit der Computertastatur spielbar ist.

5. Implementieren Sie die Funktionalität, dass das seitliche Navigationsfenster beim Verschieben des Hauptfensters diesem folgt. Dazu müssen Sie einen ComponentListener beim Hauptfenster registrieren, welcher auf verschiedene Ereignisse reagiert und die Position des seitlichen Fensters entsprechend angleicht. Informationen hierzu finden Sie in Moodle:
https://www.elearning.hs-weingarten.de/mod/wiki/view.php?id=309

Anmerkungen:

- Aktivitäten des Spielers werden der Spielelogik über den **DionaRapController** mitgeteilt. Der Controller bietet dazu die Methoden shoot() und movePlayer(), die für die Aktionen des Spielers aufzurufen sind.

- DionaRapModel erzeugt einen Event, sobald sich der Zustand in der Spielelogik geändert hat. Dabei gibt es zwei Arten von Events:

  - `DionaRapChangedEvent`

    Ein solcher Event wird ausgelöst, wenn sich die Position einer Spielfigur geändert hat und folglich das Spielfeld neu gezeichnet werden muss.

  - `GameStatusEvent`

    Ein solcher Event wird ausgelöst, wenn das Spiel gewonnen oder verloren wurde. Als Reaktion auf diesen Event implementieren Sie vorerst eine Ausgabe auf der Konsole. Im **nächsten Übungsblatt** wird als Reaktion auf diesen Event eine Dialogbox erzeugt.

  Sie benötigen eine Listenerklasse, die das Interface DionaRapListener implementiert.
  Diese Listenerklasse muss dann beim DionaRapModel registriert werden: hierzu dient die Methode addModelChangedEventListener(...).

- Wenn Sie einen bestimmten Typ prüfen müssen (z.B. ob es sich bei der AbstractPawn um einen Vortex handelt), dann geht das mit dem Schlüsselwort **instanceof**.

- Die Listener für die Buttons sollen keine direkte Referenz auf den DionaRapController besitzen und müssen daher ausgehend vom jeweiligen Button, der im ActionEvent abgefragt werden kann, zu einem Objekt finden, welches den Controller referenziert und zurückgeben kann (z.B. mithilfe einer „get“-Methode).

Diagramm:

Bild 1 zeigt das zugehörige Klassendiagramm.

Das Hauptfenster gehört zum Bereich der „View“ des MVC Musters und kennt sowohl den Controller als auch das Model.

Klassen DionaRapModel, DionaRapController(dicker Rand) und die Schnittstelle DionaRapListener sind in dem zur Verfügung gestellten JAR-Archiv enthalten. Die grau hinterlegten Klassen sind für diese Übung neu zu erstellen.

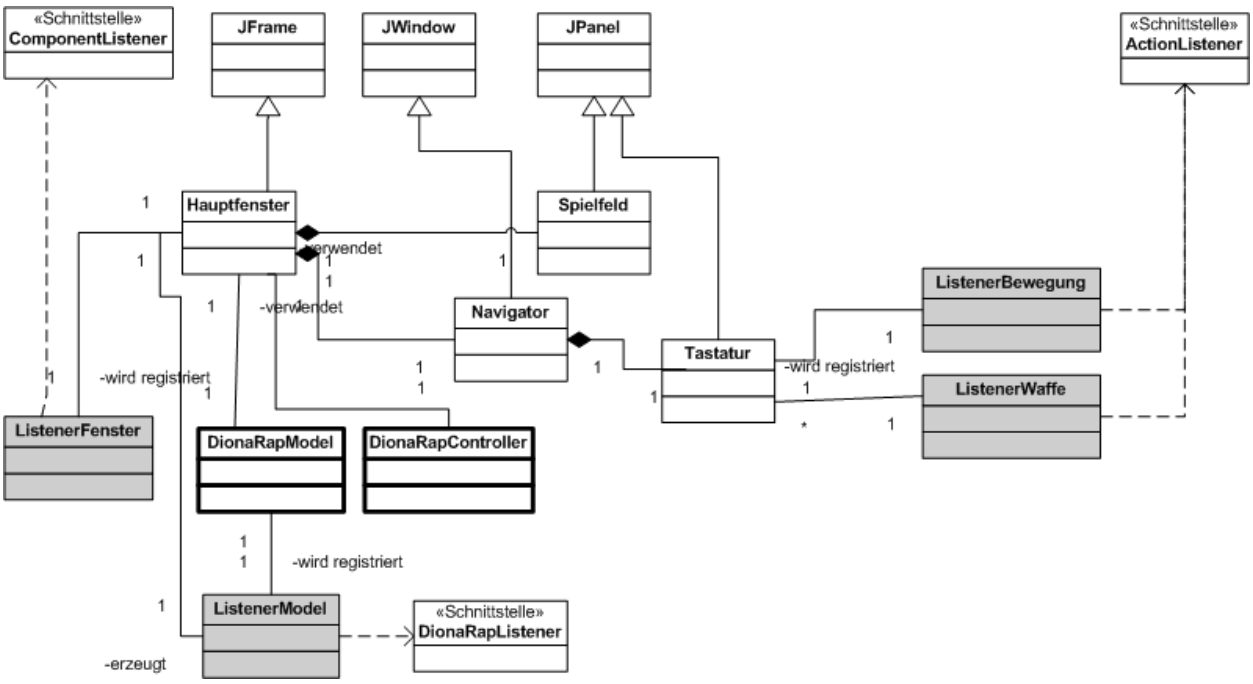

Bild 1: Klassendiagramm