(Simulation) Dokumentation der Praktischen Arbeit

Zur Prüfung zum mathematisch-technischen Softwareentwickler

Markus Faßbender
11.04.2014

1 Inhalt

2 Aufgabenanalyse

2.1 Analyse

Es soll die Simulation eines Zwei-Personen-Spiels mit dem Namen Nim umgesetzt werden. Dabei werden aus mehreren Reihen, aber mindestens 1 und maximal 9, Streichhölzer gezogen. Ein Zug darf immer nur 1-n Hölzer aus einer Reihe ziehen, die n Hölzer hat. Die Reihenfolge der vertikal angeordneten Reihen ist nicht relevant, da ja eine beliebige ausgewählt wird.

Der Spieler, welcher die letzten Hölzchen zieht, gewinnt. Daher muss eine Strategie für Spieler 1 entwickelt werden, die eine möglichst gute Ausgangslage schafft. Spieler 2 ist durch den Zufall gesteuert, solange er nicht sofort gewinnen kann. Berücksichtigt werden müssen auch Spielfehler, dass kein Hölzchen genommen wurde oder aus mehreren Reihen.

Wie eine möglichst gute Ausgangslage aussieht ist nicht näher definiert. Daher muss eine Strategie entworfen werden, die diese erzeugt bzw. findet. Ein Spiel wird 10mal durchgeführt und es gibt in jeder Runde einen Sieger. Am Ende wird eine kurze Statistik ausgeben.

2.2 Eingabeformat

Das Eingabeformat besagt, dass aus einer Eingabedatei gelesen wird. Diese muss die Dateiendung „.in“ haben. Die Datei beginnt mit einer oder mehreren Kommentarzeilen. Diese haben als erstes Zeichen ein Doppelkreuz („#“), gefolgt von einer kurzen Beschreibung. Ich gehe davon aus, dass mindestens ein Zeichen folgen muss.

Nach den Kommentarzeilen ist genau eine Zeile mit den Startreihen des Spiels. Dabei sind ganze Zahlen von 1 bis 9 gültig, von denen mindestens eine und maximal 9 durch Leerzeichen getrennt sind. Beispiel:

```
# IHK Beispiel 1
3 4 5
```

2.3 Ausgabeformat

Die Ausgabe erfolgt in eine zur Eingabedatei gleichnamige Ausgabedatei mit der Dateiendung „.out“. Hinein werden zu Beginn alle Kommentarzeilen geschrieben, gefolgt von einer Zeile mit der Startverteilung, zwei Zeilen Gewonnen Spiele von Spieler 1 und Spieler 2. Dann folgenden über mehrere Zeilen zwei Beispiele von gewonnenen Spieler. Das Format ist durch die Beispiele in der Aufgabenstellung vorgegeben.

```
# IHK Beispiel 1
Startverteilung: (3,4,5)
Gewonnene Spiele Spieler 1: 100 %
Gewonnene Spiele Spieler 2: 0 %
Beispiel eines von Spieler 1 gewonnenen Spiels:
Zug  0, Spieler 1 : (3,4,5) -> (0,4,5)
Zug  1, Spieler 2 : (0,4,5) -> (0,4,2)
Zug  2, Spieler 1 : (0,4,2) -> (0,1,2)
Zug  3, Spieler 2 : (0,1,2) -> (0,0,2)
Zug  4, Spieler 1 : (0,0,2) -> (0,0,0)
```

```
Beispiel eines von Spieler 2 gewonnenen Spiels:
Nicht vorhanden.
```

2.4 Anforderung an das Gesamtsystem

Das Programm kann in ein Model, eine View und einen Controller unterteilt werden. Der Controller liest über die View die Daten ein, speichert diese im Model und führt die Spiele durch. Anschließend wird die Statistik wieder über die View ausgegeben.

Dabei ist es wichtig mögliche Fehler zu behandeln, um Abstürze und unerwartetes Verhalten zu vermeiden. Außerdem wird eine aussagekräftige Meldung ausgegeben, welcher Fehler aufgetreten ist. Die Robustheit des Programms wird mit Testfällen überprüft.

2.5 Sonderfälle

Durch die Aufgabenstellung und das Eingabeformat ergeben sich folgende Sonderfälle:

- Die Startverteilung ist eine Reihe mit einer Eins (Minimalfall).
- Die Startverteilung besteht aus neun Reihen mit Neuen (Maximalfall).

Die Behandlung von Sonderfällen ist in der Verfahrensbeschreibung enthalten.

2.6 Fehlerfälle

Die auftretenden Fehler können in grob in drei Kategorien aufgeteilt werden: technische, syntaktische und semantische Fehler. Technische Fehler liegen vor, wenn die angegebene Datei nicht vorhanden ist oder keine Zugriffsrechte vorhanden sind. Syntaktische Fehler treten auf, wenn die Eingabedatei die Formatvorgaben nicht korrekt einhält. Bei semantischen Fehlern sind fehlerhafte Werte (z.B. 0 als Reihe) enthalten.

Durch die Analyse der Aufgabenstellung und Eingabeanforderung ergeben sich folgende syntaktische Fehlerfälle:

- Die Eingabedatei enthält keine Kommentarzeile
- In der Eingabedatei werden falsche Kommentarzeichen verwendet
- Die Eingabedatei enthält falsche Trennzeichen von Reihenwerten
- Die Eingabedatei enthält mehrere Zeilen mit Reihenwerten

Durch die Analyse der Aufgabenstellung und Eingabeanforderung ergeben sich folgende semantische Fehlerfälle:

- Die Reihen sind nicht als ganze Zahlen gegeben
- Es sind keine oder mehr als 9 Reihen angegeben
- Der Wert einer Reihe ist kleiner 1 oder größer 9
- Es befinden sich doppelte oder überflüssige Leerzeichen in der Zeile mit Reihenwerten

3 Verfahrensbeschreibung

3.1 Strategie von Spieler 1

Es ist sehr wichtig, die gute Ausgangslage zu finden. Daher ist meine Strategie in drei Schritte aufgeteilt. Zuerst werden alle möglichen Schritte erzeugt, wobei einige Einschränkungen gelten. Bei einer 1 in einer Reihenfolge kann nur 0 oder 1 rauskommen. Bei einer 2 kann nur 0 oder 1 rauskommen. Sonst kann 0, 1 oder 2 rauskommen. So generiere ich für jede Reihe, die ungleich 0 ist, 1-3 mögliche Entscheidungen.

Im zweiten Schritt werden die Möglicheiten nach folgendem Schema bewertet:

- a entspricht der Anzahl aller Reihen
- b entspricht der Anzahl der Reihen ungleich 0
- e entspricht der Anzahl der Reihen mit Einsen
- s ist eine Summe und zu beginn 0

wenn b gerade ist

> s wird um a-b erhöht
> s wird um e erhöht

sonst

> s wird um (a-b-3) erhöht
> s wird um e verringert

Dadurch erhalte ich eine Punktzahl, die bestimmt wie gut eine Möglichkeit ist. Eine ungerade Anzahl an Reihen ungleich 0 ist also sehr gut, insbesondere wenn darin Einsen sind. Eine unge Anzahl an Reihen ungleich 0 ist schlecht und noch schlimmer, wenn Einsen darin vorkommen.

Die beste Entscheidung wird im dritten Schritt ermittelt und ausgeführt. Bei mehreren gleichguten kann eine beliebige (ggf. die erste) ausgewählt werden.

3.2 Sonderfälle

Die oben genannten Sonderfälle werden wie folgt behandelt:

Die Startverteilung ist eine Reihe mit einer Eins
Der Algorithmus wird normal ausgeführt.

Die Startverteilung besteht aus neun Reihen mit Neuen
Der Algorithmus wird normal ausgeführt.

3.3 Fehlerfälle

Die oben genannten Fehlerfälle werden wie folgt behandelt:

Die Eingabedatei enthält keine Kommentarzeile
Das Programm schreibt eine Fehlermeldung in die Ausgabedatei und wird abgebrochen.

In der Eingabedatei werden falsche Kommentarzeichen verwendet
Das Programm schreibt eine Fehlermeldung in die Ausgabedatei und wird abgebrochen.

Die Eingabedatei enthält falsche Trennzeichen von Reihenwerten
Das Programm schreibt eine Fehlermeldung in die Ausgabedatei und wird abgebrochen.

Die Eingabedatei enthält mehrere Zeilen mit Reihenwerten
Das Programm nimmt die erste gefundene Zeile mit Reihenwerten und führt den Algorithmus mit diesen durch. Alle weiteren Zeilen werden ignoriert, es wird aber eine Warnung an die Ausgabedatei angehängt.

Die Reihen sind nicht als ganze Zahlen gegeben
Das Programm schreibt eine Fehlermeldung in die Ausgabedatei und wird abgebrochen.

Es sind keine oder mehr als 9 Reihen angegeben
Das Programm schreibt eine Fehlermeldung in die Ausgabedatei und wird abgebrochen.

Der Wert einer Reihe ist kleiner 1 oder größer 9
Der Wert wird auf 1 oder 9 korrigiert und der Algorithmus normal ausgeführt. Es wird eine Warnung in der Ausgabedatei angehängt.

Es befinden sich doppelte oder überflüssige Leerzeichen in der Zeile mit Reihenwerten
Die Leerzeichen werden ignoriert und der Algorithmus normal ausgeführt.

In Fällen wo nach einem Fehler weitergearbeitet wird, wird weiterhin auf andere Fehler geprüft.

3.4 Gesamtsystem

3.4.1 Main-Funktion

Die Mainfunktion liest die benötigten Pfade aus den Übergabeparametern aus und initialisiert damit den Controller. Anschließend startet sie den Controller zum Ausführen der Spiele.

3.4.2 Model

Die Daten werden im Model verwaltet. Dabei kann nicht direkt auf die Daten, sondern nur über die Schnittstellen-Methoden auf diese Zugegriffen werden. Dies erlaubt eine leichte Handhabung für Erweiterungen und hält die Kommunikation übersichtlich.

3.4.3 View

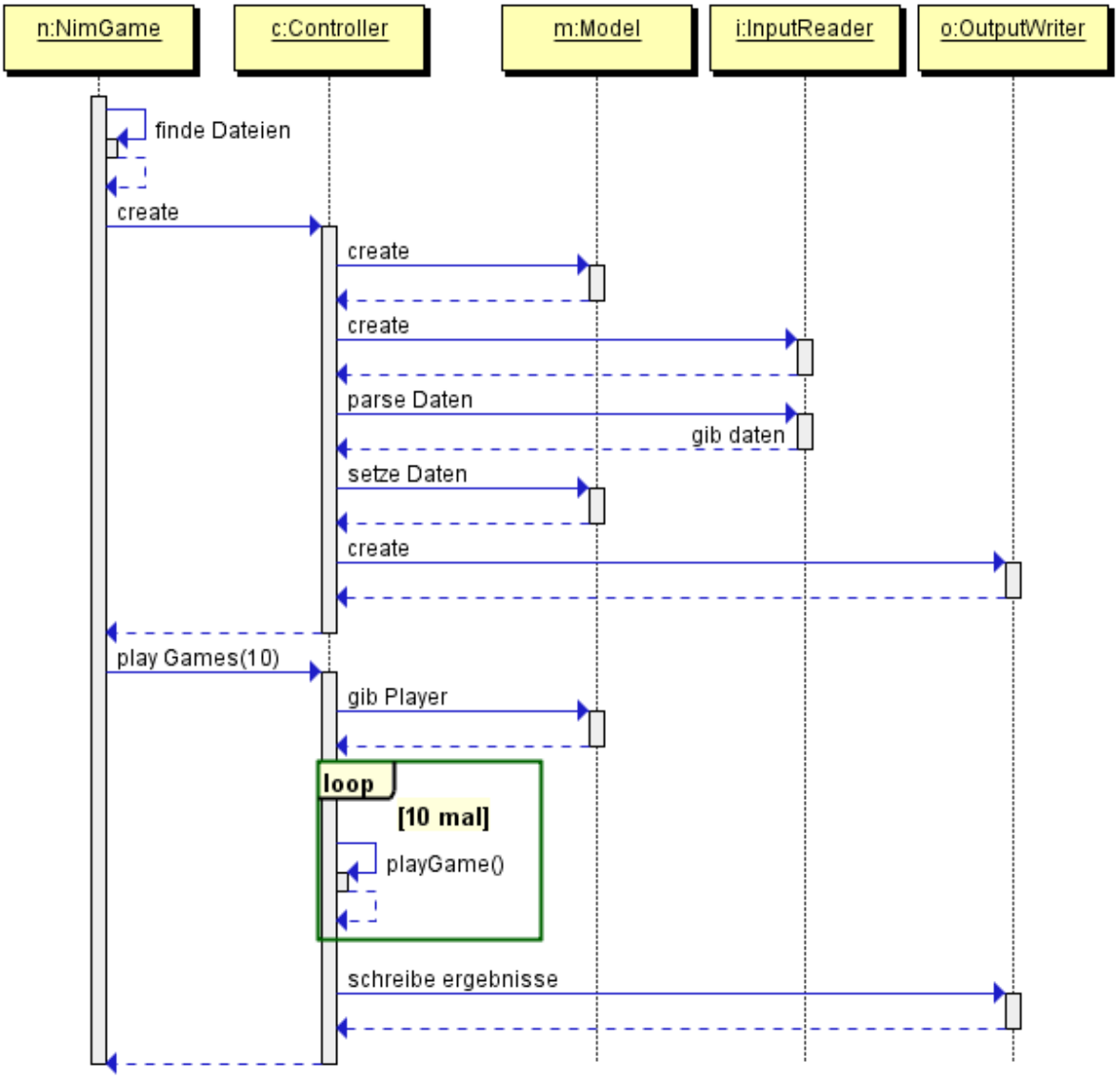
Zur View gehören sowohl Eingabe als auch Ausgabe. Hier werden also Methoden zum Einlesen von Daten und zur Ausgabe von der Statistik bereitgestellt. Auch Warnungen und Fehler können ausgegeben werden.

3.4.4 Controller

Der Controller beinhaltet die eigentliche Logik des Programms. Dieser startet das Einlesen aus der Datei, schreibt die Daten in das Model und führt alle Berechnungen und Durchläufe aus. Abschließend wird das Ergebnis in die Ausgabe geschrieben.

Die Strategien werden hier verwaltet und ausgeführt, sodass der gesamte Prozess abgekapselt ist.

3.4.5 Gesamtablauf

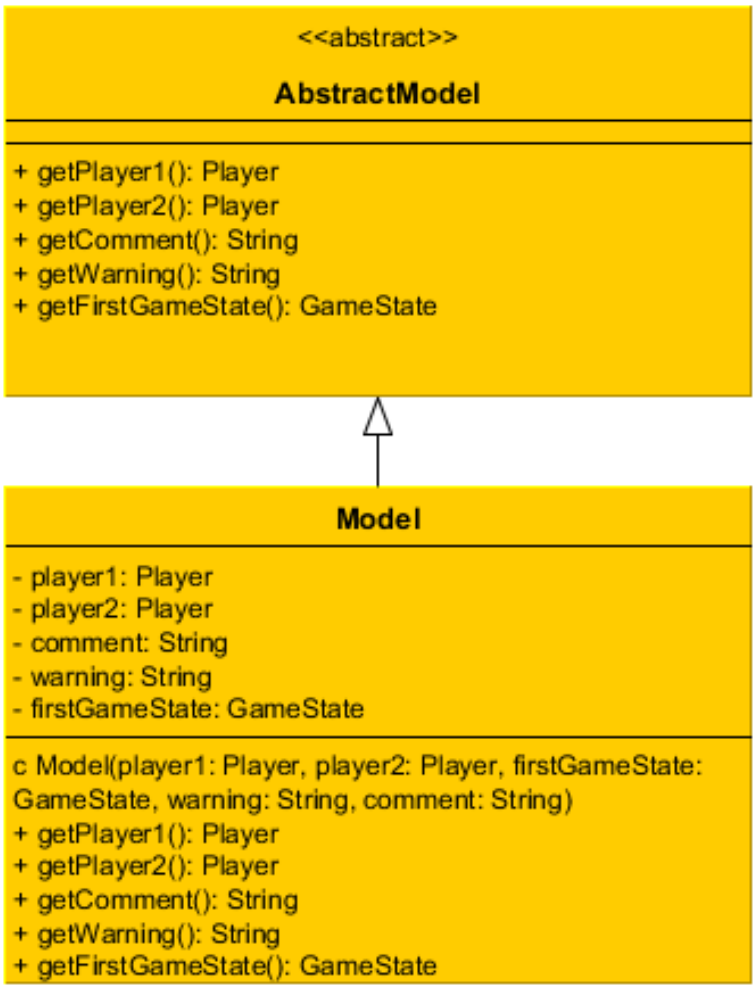


4 Programmbeschreibung

4.1 Pakete

Das Programm wird in drei Pakete unterteilt: Model, View und Controller. Dabei sind Teile des Controllers, nämlich die Strategien, als eigenes Unterpaket zusammengefasst. Für die Model und die View wurden abstrakte Klassen als Schnittstelle definiert. Diese bieten alle wichtigen Funktionen und werden jeweils in einer Unterklasse konkrete implementiert. Dadurch bleibt das Programm leicht erweiterbar.

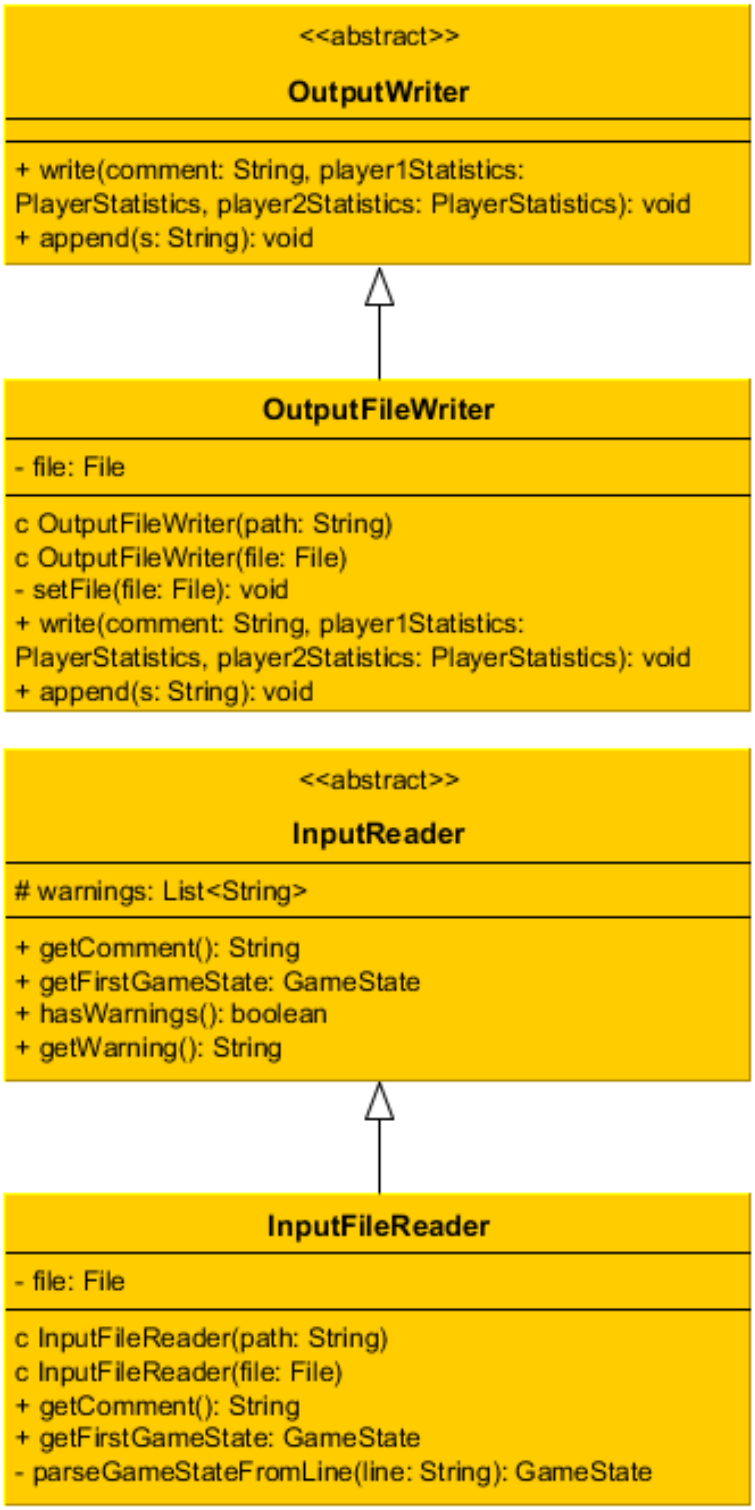
4.1.1 Model-Klassen



AbtractModel: Kapselt alle nach außen benötigten Funktionen für die beiden Spieler, den Kommentar und den ersten Spielstatus.

Model: Implementiert konkrete die vorgebenen Funktionen und speichert die Daten als Java-Objekte.

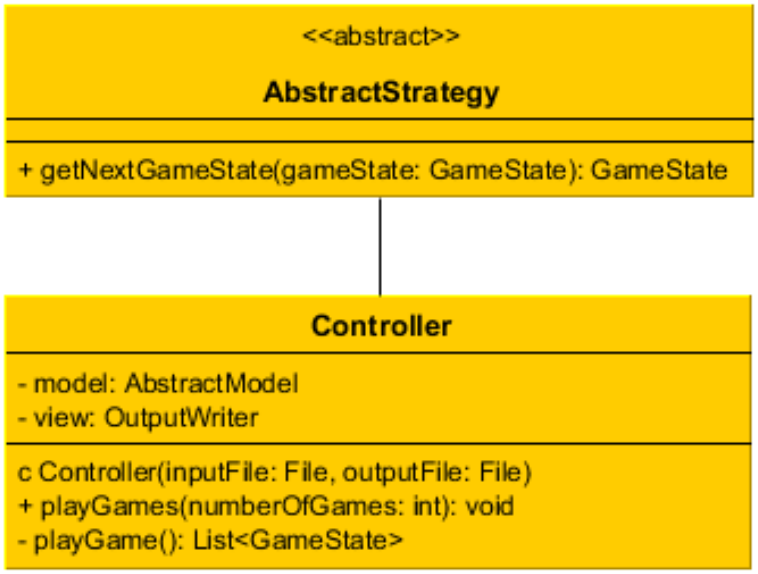
4.1.2 View-Klassen



InputReader: Gibt funktionen zum Einlesen an.
InputFileReader: Implementiert das Einlesen aus einer Datei.

OutputWriter: Gibt Funktionen zur Ausgabe an.
OutputFileWriter: Implementiert die Ausgabe in eine Datei.

4.1.3 Controller-Klassen

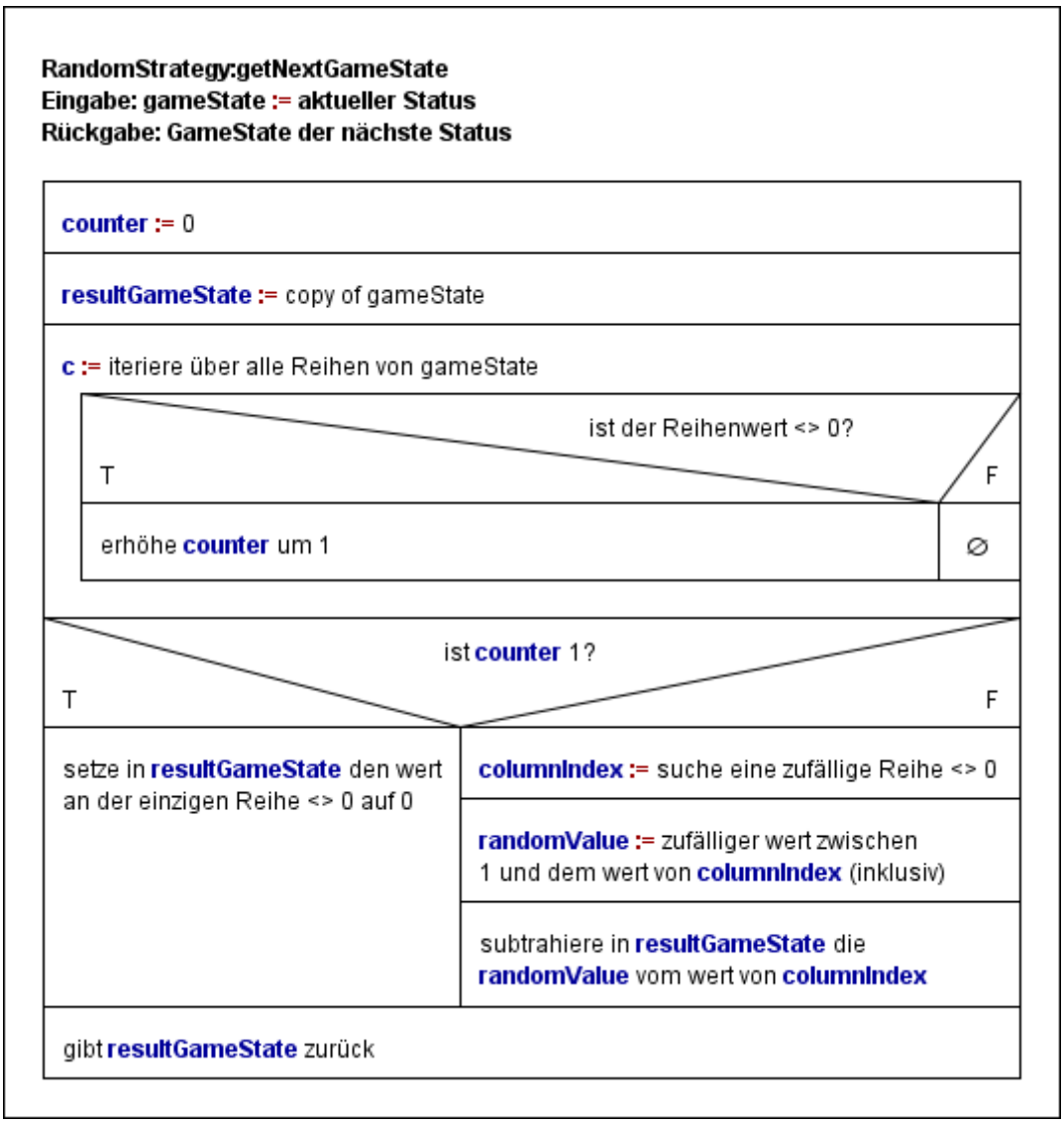


Controller: Stellt eine Funktion zur Ausführen von mehreren Spielen bereit.

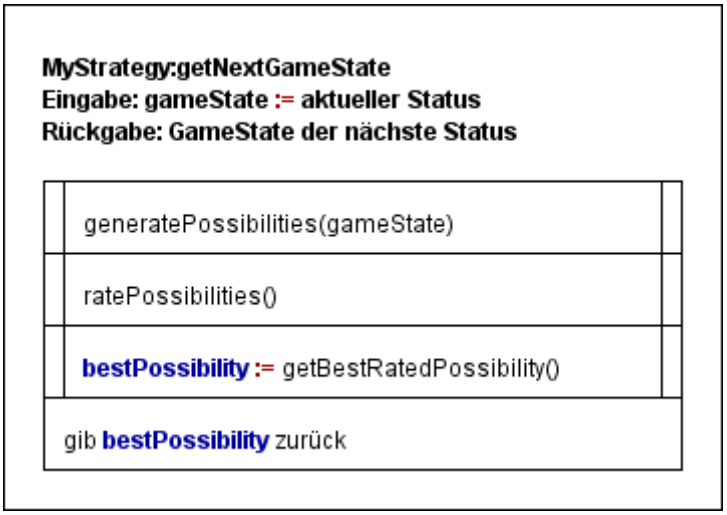
AbstractStrategy: Funktion, um den nächsten Spielstatus zu ermitteln.

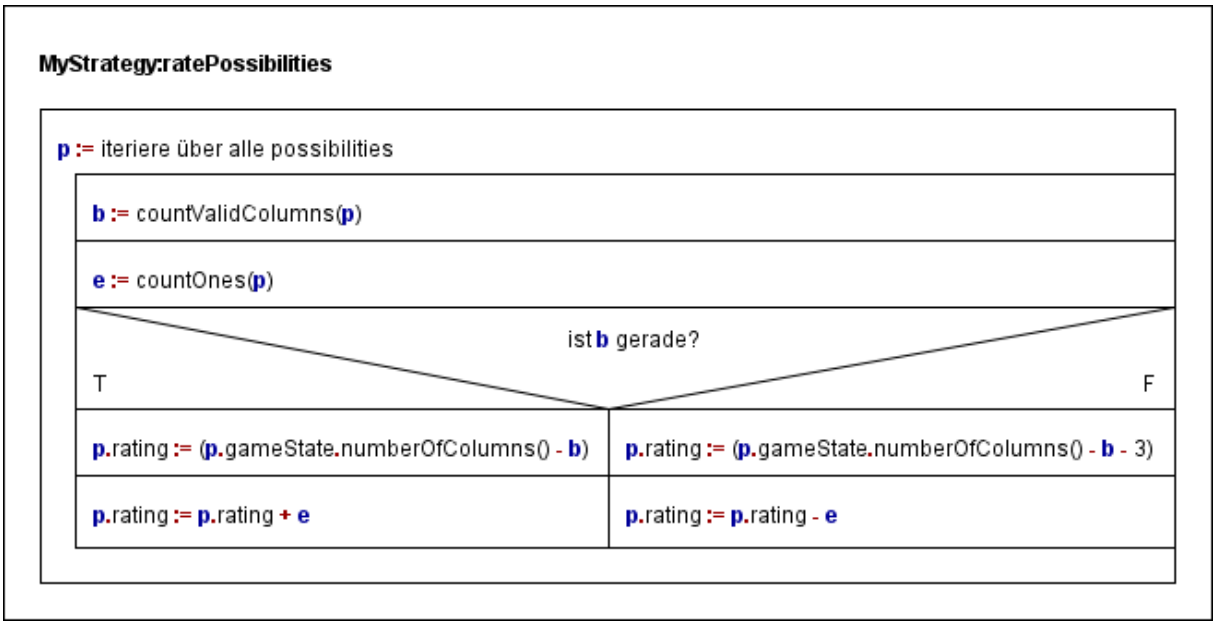
4.2 Präzisierung

4.2.1 Präzisierung RandomStrategy

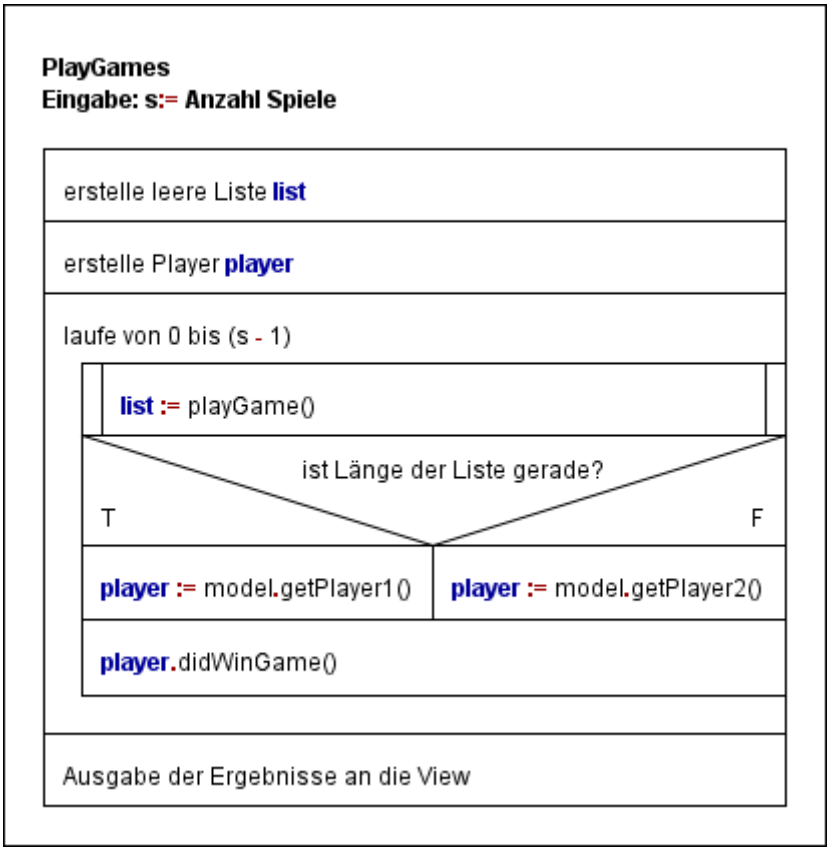


4.2.2 Präzisierung MyStrategy

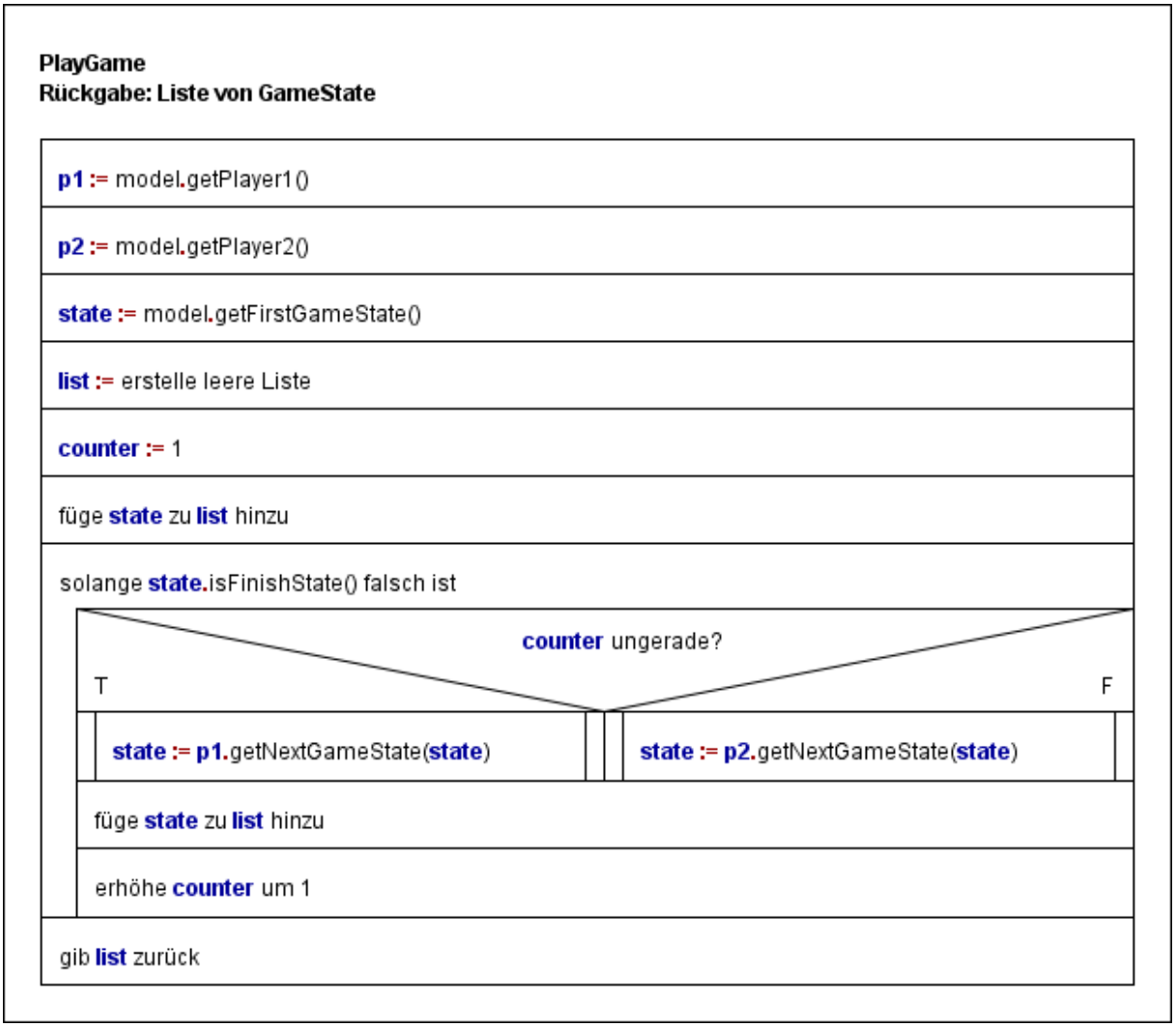




4.2.3 Präzisierung PlayGames



4.2.4 Präzisierung PlayGame



5 Testdokumentation

Parallel zu der Entwicklung des Softwaresystems wurden Test geschrieben, die die Funktionalität überprüfen. Bei jeder Änderung wurden die Tests durchgeführt, um mögliche Fehler frühzeitig zu aufzudecken. Die Tests sind in Form von Eingabedateien auf der CD im Ordner „tests“ abgelegt.

5.1 Normalfall

Ein Normalfall liegt vor, wenn keine Sonderfälle oder Fehlerfälle auftreten. Die in der Aufgabenstellung abgedrucken Beispiele wurden alle erfolgreich durchgeführt. Das Beispiel 4 ist ebenfalls in den vordefinierten Tests enthalten als *ihk_example_4.in* enthalten.

5.2 Sonderfall

Die festgelegten Sonderfälle einer minimalen und maximalen Eingabedatei sind ebenfalls in den vordefinierten Tests enthalten.

5.3 Fehlerfall

Es sind neun Tests für Fehlerfälle enthalten, weil in der Beschreibung mehrere Fälle zusammengefasst worden sind. In allen Fällen wird je nach Fehlerart eine Fehleranalyse oder eine Warnung an die Ausgabe angehängt. Danach wird das Programm korrekt beendet.

Zusätzlich wurde noch getestet, wie das Programm auf nicht vorhandene Eingabedateien reagiert. In dem Fall wird in dem Runscript „run.bat“ eine Fehlermeldung ausgegeben.

5.4 Ausführliches Beispiel

Als Beispiel verwende ich das IHK-Beispiel 1 mit den Reihen (3,4,5).

1. Spieler 1 ist am Zug und bewertet die Möglichkeiten. Aus der Bewertung wird die erste Möglichkeit als Beste genommen.

   ```
   Zug  1, Spieler 1 : (3,4,5) -> (0,4,5)
   ```

(0,4,5) =1+0=1	(1,4,5) =-3-1=-4	(2,4,5) =-3-0=-3
(3,0,5) =1+0=1	(3,1,5) =-3-1=-4	(3,2,5) =-3-0=-3
(3,4,0) =1+0=1	(3,4,1) =-3-1=-4	(3,4,2) =-3-0=-3

2. Spieler 2 ist am Zug und führt eine zufällige Wahl aus:

   ```
   Zug  2, Spieler 2 : (0,4,5) -> (0,1,5)
   ```

3. Spieler 1 ist am Zug und bewertet die Möglichkeiten. Aus der Bewertung wird die vierte Möglichkeit als Beste genommen.

```
Zug  3, Spieler 1 : (0,1,5) -> (0,1,1)
```

(0,0,5) =1-0=1	-	-
(0,1,0) =-1-1=-2	(0,1,1) =1+1=2	(0,1,2) =1+1=2

4. Spieler 2 ist am Zug und führt eine zufällige Wahl aus:

```
Zug  4, Spieler 2 : (0,1,1) -> (0,1,0)
```

5. Spieler 1 ist am Zug und bewertet die Möglichkeiten. Aus der Bewertung wird die einzige Möglichkeit als Beste genommen.

```
Zug  5, Spieler 1 : (0,1,0) -> (0,0,0)
```

(0,0,0) =3+0=3

6. Der Spielstatus ist auf (0,0,0) und das Spiel ist fertig. Spieler 1 hat gewonnen, weil er das letzte Streichholz gezogen hat. Die Gesamtausgabe ist:

```
# IHK Example 1
Startverteilung: (3,4,5)
Gewonnene Spiele Spieler 1: 90 %
Gewonnene Spiele Spieler 2: 10 %
Beispiel eines von Spieler 1 gewonnenen Spiels:
Zug  1, Spieler 1 : (3,4,5) -> (0,4,5)
Zug  2, Spieler 2 : (0,4,5) -> (0,1,5)
Zug  3, Spieler 1 : (0,1,5) -> (0,1,1)
Zug  4, Spieler 2 : (0,1,1) -> (0,1,0)
Zug  5, Spieler 1 : (0,1,0) -> (0,0,0)
Beispiel eines von Spieler 2 gewonnenen Spiels:
Zug  1, Spieler 1 : (3,4,5) -> (0,4,5)
Zug  2, Spieler 2 : (0,4,5) -> (0,4,2)
Zug  3, Spieler 1 : (0,4,2) -> (0,1,2)
Zug  4, Spieler 2 : (0,1,2) -> (0,1,1)
Zug  5, Spieler 1 : (0,1,1) -> (0,0,1)
Zug  6, Spieler 2 : (0,0,1) -> (0,0,0)
```

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung

Aufgrund der Konzeption nach dem MVC-Muster können Anforderungsänderung mit geringem Aufwand umgesetzt werden. Zum Beispiel könnte statt aus Dateien von der Kommandozeile oder einer GUI eingelesen werden. Auch die Ausgabe könnte leicht auf die Kommandozeile oder eine GUI umgelenkt werden. Weiterhin ist der Algorithmus so implementiert, dass eine Erweiterung der Strategien leicht durchzuführen ist. Dazu muss nur eine weitere Strategie entworfen und hinzugefügt werden. Das Gesamtsystem ist also offen gegenüber Erweiterungen und geschlossen für Änderungen.

6.2 Ausblick

Meine Lösungsstrategie kann noch verbessert werden, da nicht immer die optimale Lösung gefunden bzw. am besten bewertet wird. So könnten beispielweise nicht nur die Reihen auf eine gerade/ungerade Anzahl geprüft werden, sondern auch die Summe der Spalten.

Als mögliche Erweiterung des Programms wäre denkbar:

- Implementierung einer graphischen Oberfläche
    - Maske um die Daten leicht und Fehlerfrei eingeben zu können.
    - Anschauliche Darstellung der Gewinnverteilung (z.B. über ein Kuchendiagramm).
    - Abbildung einzelner Spielzüge, jedoch muss dann nicht nur am Ende die View ausgegeben werden.
- Verbesserte Strategie auch für Spieler 2.
- Es könnten mehr Runden pro Gesamtspiel durchgeführt werden (also mehr als 10).
- Man könnte auch mehr Spieler an dem Spiel beteiligen – die Strategien müssten dann komplett neu erarbeitet werden.
- Ein gegenläufiges Spielprinzip hinzufügen: Wer das letzte Streichholz zieht verliert.

7 Änderungen

7.1 Abweichungen

- Die Methode *playGame()* in der Klasse *Controller* soll ein einziges Spiel kapseln. Da dies nach außen nicht sichtbar sein sollte, wurde die Sichtbarkeit auf private eingeschränkt.
- Der Konstruktor von *AbstractModel* wurde entfernt, da dieser dann direkt implementiert werden müsste und dies nach meinem Entwurf nicht möglich ist.

7.2 Ergänzungen

- Es wurde ein Konstruktur zur Klasse *Possibility* hinzugefügt, die ein Objekt direkt mit einer Liste von Gamestates initialisiert. Dadurch wird sichergestellt, dass nur gültige Objekte erzeugt werden können.
- Es wurde ein Copy-Konstruktor zu GameState hinzugefügt, um Objekte korrekt kopieren zu können.
- Es wurde die Methode *toString()* wird für eine saubere Formatierung in der Klasse *GameState* überschrieben.
- Die abstrakte Klasse *OutputWriter* und die implementiere Klasse *OutputFileWriter* wurden um die Methode *append(String s)* erweitert, um Fehler und Warnings ausgeben zu können, ohne das vorher ausgegebene zu überschreiben.
- Die Methode *OutputFileWriter* wurde um den Setter *setFile(File file)* erweitert, um doppelten Code in den beiden Konstruktoren zu vermeiden.
- Konstanten für das Einlesen von Dateien zu der Klasse *InputReader* hinzugefügt.
- Für das saubere Trennen von Aufgaben wurde der Klasse *InputFileParser* die private Methode *parseGameStateFromLine(String s)* hinzugefügt.

8 Benutzeranleitung

8.1 Verzeichnisstruktur der CD

Auf der CD sind folgende Verzeichnisse angelegt:

Wurzelverzeichnis

- Vorkompilierte ausführbares Programm.
- Die Beschreibung als Master-Datei im Wordformat und als PDF.
- „bin“ um die *.class-Dateien und das Manifest abzulegen.
- „diagrams“ enthält die verschiedenen Diagrammtypen jeweils als Orginaldateien und Bilddateien.
- „documentation“ enthält die JavaDoc.
- „scripts“ enthält Batch-Skripte für das
    - Kompilieren und erzeugen des ausführbaren Programms.
    - Automatische Durchführen aller Tests.
    - Aufräumen bzw. Löschen der erzeugten Dateien.
- „src“ enthält den Quellcode.
- „tests“ enthält die Eingabe- und Ausgabedateien der Tests.

8.2 Systemvoraussetzungen

Um das Programm auszuführen zu können, muss das Java Software-Development-Kit (SDK) in der Version 1.7 oder höher installiert sein. Die aktuelle Java-Version kann unter folgendem Link heruntergeladen werden:

http://java.com/de/download/index.jsp

Von Oracle angegebene Systemvoraussetzungen sind:[1]

- Windows 8 Desktop
- Windows 7
- Windows Vista SP2
- Windows XP SP3 (32-Bit); Windows XP SP2 (64-Bit)
- Windows Server 2008

Ein Pentium 2 266 MHz oder schnellerer Prozessor mit einem physikalischen RAM von mindestens 128 MB wird empfohlen. Außerdem benötigen Sie mindestens 124 MB freien Speicherplatz auf dem Datenträger.

8.3 Installation

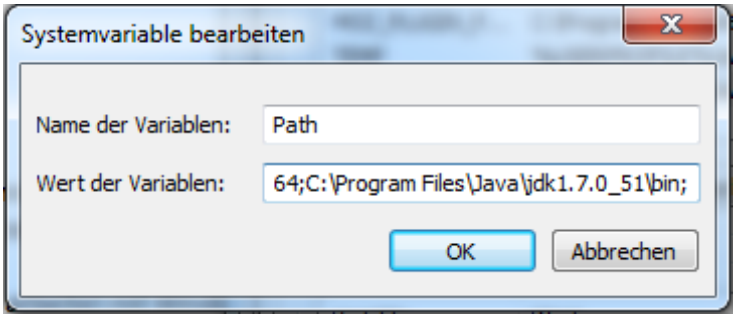
Der gesamte Inhalt der CD kann in ein beliebiges Verzeichnis kopiert werden. Wichtig ist, dass das Programm das Verzeichnis beschreiben darf.

[1] Siehe http://java.com/de/download/win_sysreq-sm.jsp (Stand 16:22 Uhr - 09.04.2014)

Um das Programm unter Windows ausführen zu können muss in der Systemvariable „Path“ der Pfad in das „bin“-Verzeichnis der Java-SDK-Installation gesetzt sein.

Die folgende Beschreibung zum Setzen der Variable arbeitet mit Windows 7 Professional:

1. Öffnen des Startmenüs
2. Rechtsklick auf „Computer“, dann Linksklick auf „Eigenschaften“
3. Über das linke Menü „Erweiterte Systemeinstellungen“ öffnen
4. In dem neuen Fenster auf „Umgebungsvariablen“ klicken
5. Bei „Systemvariablen“ die Variable „Path“ auswählen und auf „Bearbeiten…“ klicken
6. Bei „Wert der Variablen“ den Pfad zur Java-Installation mit dem Unterordner „bin“ angeben. In meinem Beispiel wäre das: *C:\Program Files\Java\jdk1.7.0_51\bin;*
   Dabei muss auf die Semikolon vor und nach dem Pfad geachtet werden, da diese als Trennzeichen dienen.
7. Mit Klick auf „OK“ bestätigen und ggf. die Konsole erneut öffnen.



8.4 Ausführen der Skripte

Nachdem die Systemvariable korrekt gesetzt wurde, können die Batch-Skripte aus dem Unterordner „scripts“ direkt ausführt werden. Dies kann entweder über die Konsole geschehen, indem einfach nur <dateiname>.bat aufgerufen wird oder im Dateisystem ein Doppelklick auf die Datei ausgeführt wird.

9 Entwicklerdokumentation

Die Entwicklerdokumentation befindet sich in Form einer vollständigen JavaDoc in dem Unterordner „documentation\javadoc“. Durch öffnen der Datei „index.html“ wird die Gesamtübersicht geladen.

10 Entwicklungsumgebung

Entwickelt und getestet wurde unter folgendem System:

Programmiersprache	Java 1.7 Update 51
Rechner	Intel® Pentium® CPU P6200 2.13 GHz 2.13 GHz 4 GB Arbeitsspeicher
Betriebssystem	Windows 7 Professional 64 Bit-Betriebssystem

11 Erklärung

Ich versichere durch meine Unterschrift, dass ich das Prüfungsprodukt selbstständig und ohne fremde Hilfe angefertigt und alle Stellen, die ich wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen habe, als solche kenntlich gemacht habe. Die Arbeit hat in dieser Form keiner anderen Prüfungsinstitution vorgelegen.

Das auf den beiden identischen CDs abgelegte Prüfungsprodukt entspricht der gedruckten Version.

Des Weiteren befinde ich mich gesundheitlich in der Lage die Prüfung abzulegen.

Ort und Datum	Unterschrift des Prüfungsteilnehmers

12 Verwendete Hilfsmittel

- Eclipse Standard/SDK (32 Bit)
  Version: Kepler Service Release 1
  Entwicklungsumgebung für Java und andere Programmiersprachen.
  http://eclipse.org/
- Notepad++
  Open-Source-Texteditor für Windows
  http://notepad-plus-plus.org
- yEd
  Plattform unabhängiger Graph-Editor. Unter anderem Fähig UML-Diagramme zu erstellen.
  http://www.yworks.com/de/products_yed_about.html
- Structorizer
  Plattform unabhängiges Programm zur Erzeugung von Nassi-Shneidermann-Diagrammen.
  http://structorizer.fisch.lu/
- Quick Sequenz Diagram Editor
  Plattform unabhängiges Programm zur Erzeugung von Sequenzdiagrammen.
  http://sdedit.sourceforge.net/index.html
- GIMP
  Plattform unabhängiges Programm zur Grafikbearbeitung.
  http://www.gimp.org/