Handbuch Java Programm

4-Gewinnt

Marcel Gertsch
Funkstrasse 75
3084 Wabern

marcel@gertsch.be
079 213 19 99

Inhaltsverzeichnis

Projekt 4-Gewinnt

Ausgangslage

Sie arbeiten als Praktikant bei der Firma TOPOMEDICS und erhalten für die Aufgabe ein "Brettspiel" zu entwerfen. Damit sollen Kinder verschiedener Altersstufen im Wartezimmer von Ärzten an einem Bildschirm mit Tastatur beschäfrigt werden können. Es soll jedoch keine Hektik aufkommen und andere Patienten stören. Es spielt meist ein Kind gegen den Computer. Egal wie einfach der Computer gewinnen könnte, er soll dem Spieler immer die Möglichkeit geben, zu gewinnen. Bei etwas komplexeren Spielen kann der Spieler den Schwierigkeitsgrad vor dem Spielen aussuchen. Die Spiele verfügen über eine abrufbare Spielanleitung. Die Spiele dürfen keine Urheberrechte verletzen, daher ist es wünschenswert, abgewandelte Versionen des Originalspiels zu implementieren. Auch geht es in erster Linie um den Zeitvertrieb und nicht um das perfekte Spiel.

Projektziel

Ein Funktionierendes Java Programm, bei welchem man aussuchen kann, ob man gegen den Computer oder gegen einen zweiten Spieler spielen möchte.

Spielanleitung

Bei 4-Gewinnt versucht jeder Spieler vier Steine seiner Farbe in einer Reihe zu platzieren - vertikal, horizontal oder diagonal.
Die beiden Spieler platzieren die Steine abwechslungsweise im 7x6 Felder grossen Spielfeld. Der erste, welcher 4 Steine seiner Farbe in einer Reihe platziert, hat das Spiel gewonnen.

Zielgruppe

Die Zielgruppe meines Java Programmes sind Kinder im Wartezimmer eines Spitals. Das Spiel soll simpel sein, und wird für Kinder im Wartezimmer eines Arztes entwickelt. Daher soll es kein aufbrausendes Spiel sein, sondern lediglich zur Ablenkung und Zeitvertreibung dienen.

Termine

Start des Projektes ist der 7. November 2018. Bis zur grossen Pause dieses Morgens, musste der Entscheid bezüglich des Themas vom Projekt festgelegt sein. Am 9. November um. 18:00 Uhr ist die Deadline für die Abgabe des Konzepts. Abgabezeitpunk ist spätestens der **21. November 2018 um 16:30 Uhr** per E-Mail an markus@ruggiero.ch.

Projekterklärung

VierGewinnt ist die Hauptklasse. Darin findet der gesamte Ablauf statt. Je nach dem ob man gegen den Computer oder gegen einen anderen Spieler spielt, wird eine andere while-Schlaufe aufgerufen. Die erste ist für Spieler gegen Spieler, die zweite gegen den Computer.

Die Klasse **Spielfeld** hat eine Methode *spielStart*. Mit dieser startet das Spiel und enthält ein paar Informationen mehr, als beim normalen Ablauf. *PrintSpielfeld* überprüft immer, welches Feld noch leer ist und fügt am richtigen Ort "X" oder "O" ein.

SpielerVsSpieler wird dann aufgerufen, wenn nicht gegen den Computer gespielt werden will. *werIstDran* verlangt eine Eingabe für die Reihe und bestimmt, welcher Spieler gerade am Zug ist. *maxHoehe* überprüft, ob das oberste Feld der gewählten Reihe bereits belegt ist. Falls ja, dann wird eine neue Benutzereingabe verlangt.
getReihe und *getFigur* geben den Wert der aktuellen Reihe (1-7) und Figur (X oder O) zurück.

SpielerVsComputer wird dann aufgerufen, wenn gegen den Computer gespielt werden will. *spielerTurn* verlangt logischerweise die Eingabe des Benutzers und *computerTurn* generiert eine Zahl zwischen 1-7, welche maximal 2 mehr oder weniger sein darf, als die letzte Eingabe des Benutzers. *getReihe* gibt den Wert der aktuellen Reihe (1-7) aus.

Die Klasse **HatWerGewonnen** überprüft nach jedem Spielzug, ob jemand gewonnen hat. Es überprüft jede vertikale, horizontale und diagonale Möglichkeit für einen Sieg.

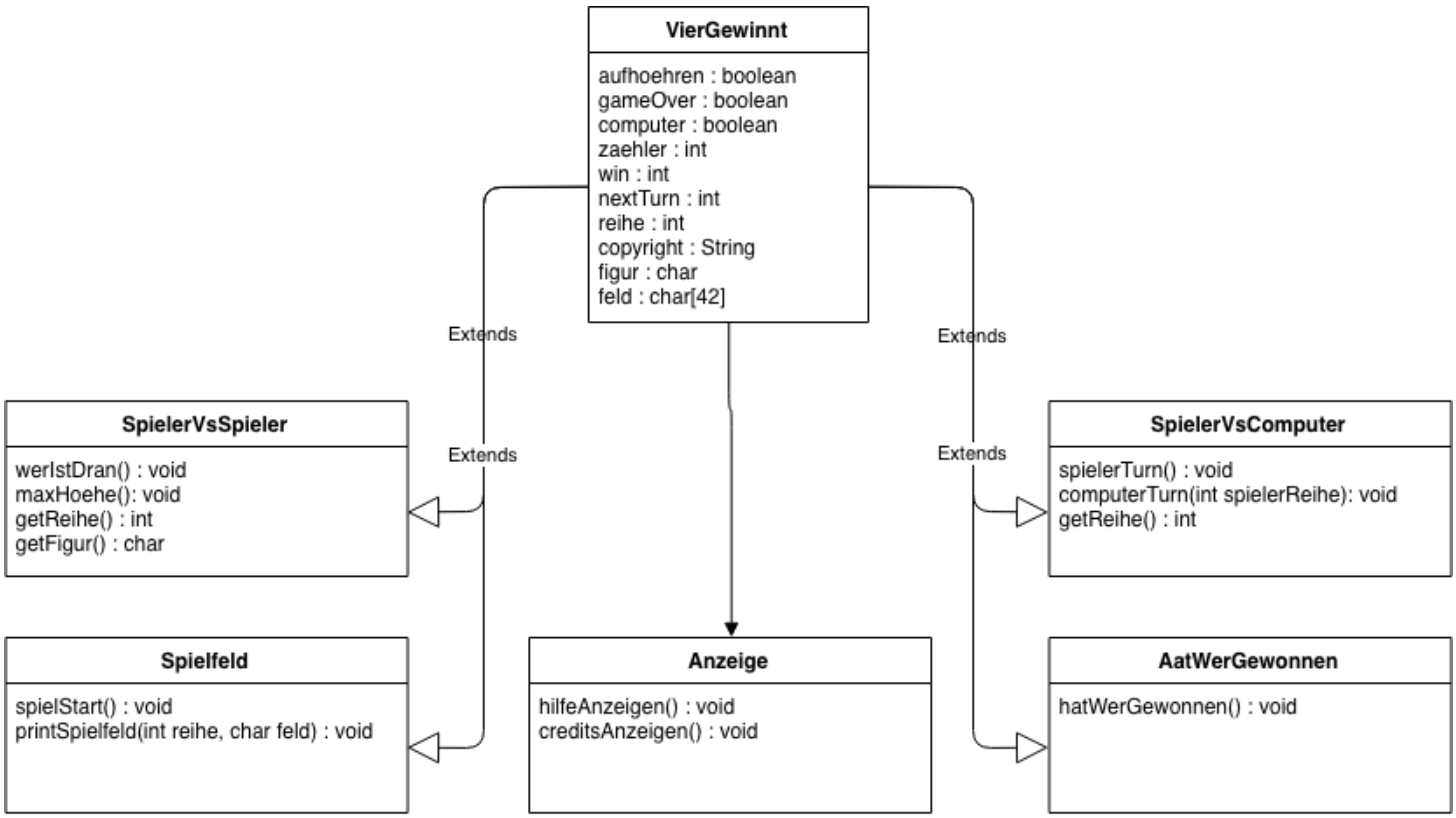
Anzeige hat wenig Fähigkeiten. Sie beinhaltet zwei Methoden. *hilfeAnzeigen* gibt die Spielregen aus. Es erklärt, wie das Spiel genau funktioniert. *creditsAnzeigen* gibt Informationen über Author, Version, Firma und wann es veröffentlicht wurde.

Skizze

Folgendermassen wird das Programm aussehen.

```
----------------------------
   |   |   |   |   |   |   |
----------------------------
   |   |   |   |   |   |   |
----------------------------
   |   |   |   |   |   |   |
----------------------------
   |   |   |   | X |   |   |
----------------------------
   | O |   | O | O |   |   |
----------------------------
   | X | X | X | O |   |   |
----------------------------
 1   2   3   4   5   6   7
Bitte geben Sie eine Zahl zwischen 1-7 ein:
```

UML



Anforderungskatalog

Funktionale Anforderungen	**Version**
- Spielfeld wird korrekt ausgegeben	v0.1
- Spieler können abwechslungsweise Eingaben machen	v0.2
- Das Spiel erkennt, ob ein Feld bereits besetzt ist & setzt es darüber - Bei maximaler Höhe erfolgt eine neue Eingabe des Benutzers	v0.3
- Erkennt, wenn 4 in einer Reihe (vertikal, horizontal oder diagonal) sind, dass der Spieler dann gewonnen hat	v0.4
- Spielen gegen Computer oder Spieler	v1.0
- JUnit Tests implementiert	v1.1

Nicht Funktionale Anforderungen
Keine Abstürze
Keine Endlosschleifen
Schnelle Reaktion nach Eingabe
Ungültige Eingaben abfangen

Tests

Test	**Erfüllt?**	**Massnahmen**
Tastatursteuerbar	Ja	
Neues Spiel nach Beenden des Vorherigen	Ja	
Aussuchen ob Computer oder Spieler als Gegner	Ja	
Schwierigkeitsgrad auswählen	Nein	nicht nötig
Falsche Eingaben werden abgefangen		
/ Buchstaben	Ja	
/ Zahlen über 7 & unter 1	Ja	
/ Reihen, welche bereits max Höhre erreicht haben	Ja	
/ 99 - erfolgt die Spielanleitung	Ja	
/ 98 - erfolgen die Credits	Ja	

Schlussreflektion

Auch wenn das ganze nicht überall korrekt durchgeführt wurde (z.B. mit den Vererbungen), hat mir das Projekt sehr viel Spass gemacht. Zu Beginn hatte ich Mühe, doch je länger ich daran gearbeitet habe, desto besser lief es. Ich konnte im Vorhinein nicht genau planen, was genau wie aussehen soll und wie es aufgebaut wird.
Erst als das Projekt gestartet hat, ich erstmals mit Java ein wenig warm wurde, lief alles besser. Mit dieser Arbeit konnte ich viel von Java profitieren. (Auch wenn es definitiv nicht mein Spezialgebiet sein wird - gehe Richtung Webapplikation).

Ich verstehe nun besser, wie die ganzen Zusammenhänge in Java funktionieren.

Quellen

Dies sind alle Quellen, die ich benötigt habe. Die Idee der Realisierung und die Umsetzung an sich habe ich komplett alleine gemacht.

char array
https://www.dotnetperls.com/char-array-java

tilde in Java
https://stackoverflow.com/questions/1483504/java-what-does-mean

equeals ignore case
https://www.tutorialspoint.com/java/java_string_equalsignorecase.htm

random number generator
https://www.mkyong.com/java/java-generate-random-integers-in-a-range/

junit assertion
https://www.tutorialspoint.com/junit/junit_using_assertion.htm

clear console
https://stackoverflow.com/questions/2979383/java-clear-the-console