Hochschule Hamm-Lippstadt

Soziale Medien und Kommunikationsinformatik

Sommersemester 2017

Programmieren II

Dozent: Stefan Henkler

**„Vier Gewinnt“**

|  |  |  |  |
| --- | --- | --- | --- |
| Kevin-Karl Klages  Matr.-Nr.: 2150121 | Jens Michalke  Matr.-Nr.: 2152013 | Lucille Reeve  Matr.-Nr.: 2150229 | Hannah Kuhn Matr.-Nr.: 2150668 |

Lippstadt, 13.07.2017**Inhaltsverzeichnis**

1. Einführung 3

1.1 Zweck und Zielgruppe 3

1.2 Ziele des Projektes 3

1.3 Referenzen 3

2. Funktionale Anforderungen 3

2.1 UML-Diagramm 3

2.2 Muss-Funktionen 3

**2.2.1 GUI 4**

**2.2.2 Server 4**

**2.2.3 Client 4**

**2.2.4 Gewinnabfrage 4**

2.3 Benutzercharakteristika 3

2.4 Soll-Kriterien 3

2.5 Erweiterungen 3

2.6 3

3. Management Summary 4

3.1 Ausgangslage 4

3.2 Arbeitspakete 4

3.3 Probleme 4

3.4. Lösung 4

3.5. Lessons Learned 4

3.6. Nicht implementiert ? 4

# 1. Einführung

Kurzzusammenfassung, was in Kapitel 1 folgt hier

## 1.1 Zweck und Zielgruppe

Zweck des „Vier Gewinnt“ Spiels ist die Unterhaltung. Das Spiel ist für Menschen jeder Altersklasse vorgesehen.

## 1.2 Ziele des Projektes

* Programmiersprache Java
* Mehrpersonenspiel umsetzen
* Mit Grafischer Oberfläche
* Verteiltes Spiel
* Spieler können an unterschiedlichen Rechnern gegeneinander spielen
* Computergegner nicht Teil der Aufgabe

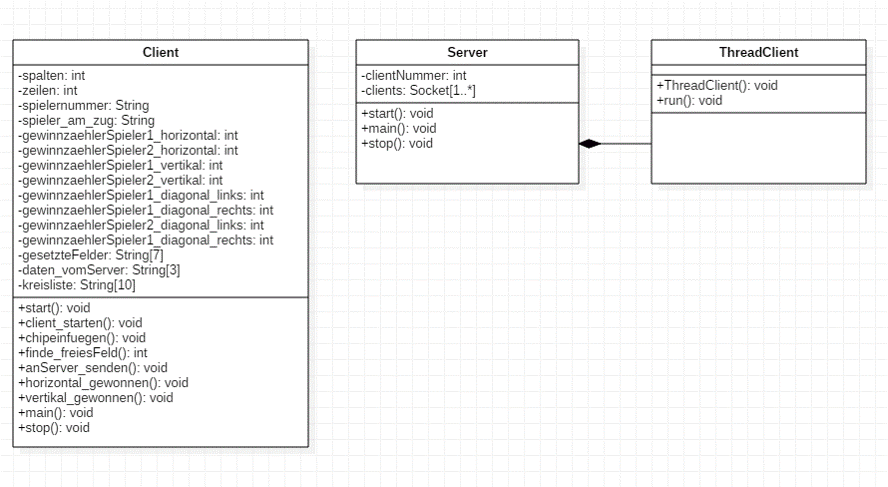
## 1.3 Referenzen

* Oracle Docs: https://docs.oracle.com/javase/7/docs/api/java/net/ServerSocket.html
* Java ist auch eine Insel: http://openbook.rheinwerk-verlag.de/javainsel/

# 2. Funktionale AnforderungenC:\Users\Hannah\Documents\Studium\4. Semester\2. Semester - Programmieren2\59fcf5e0-662c-4a76-8ccf-0bde09b7f5df.jpg

Zu Beginn haben wir erst ein Prototyp Paper angefertigt, um uns einen groben Überblick über den zu entwickelnden Code zu verschaffen. Der Fokus lag dort zum größten Teil auf der GUI mit dem Spielfeld und der Gewinnabfrage. Außerdem wurden zuerst die Spielregeln, abgeleitet vom „Originalen“ Brettspiel, zusammengetragen. Daraufhin haben wir uns für das klassische „Vier Gewinnt“ entschieden, bei dem das Spielfeld aus 7 Spalten und 6 Zeilen besteht. Jeder Spieler besitzt 21 Steine und setzt abwechselnd jeweils einen Stein in das Spielfeld. Dieser fällt immer bis zum untersten freien Platz durch. In dem Zusammenhang kam die Idee ein zweidimensionales Array zu verwenden. Die Spalten und Zeilen bilden die Indizes und verweisen auf den jeweiligen Spieler beziehungsweise auf ein leeres Feld. Gewonnen hat derjenige, der vier Steine einer Farbe entweder horizontal, vertikal oder diagonal angeordnet in einer Reihe hat. Tritt einer dieser Fälle nicht ein und das komplette Spielfeld ist belegt, liegt ein Unentschieden vor.

## 2.1 UML-Diagramm



## 2.2 Muss-Funktionen

Zu den Muss-Funktionen des Programms zählen eine GUI für Eingaben der Spieler und Anzeigen des Spiels. Zusätzlich muss ein Server vorhanden sein, um das Spielen über ein Netzwerk zu ermöglichen. Ein Client muss in der Lage sein sich mit dem Server zu verbinden, mit ihm zu kommunizieren und ist unter anderem für die Gewinnabfrage zuständig.

**2.2.1 GUI**

* Benutzung von Java FX Bibliothek
* Grid Pane Layout zur Realisierung des Spielfelds
* Spielfeld: 7 Spalten, 6 Zeilen
* Einfügen der Kreise mit Hilfe einer for-Schleife

**2.2.2 Server**

**2.2.3 Client**

Der Client beinhaltet das User Interface und die Logik des Spiels. Bei Start des Clients wird zuerst die Benutzeroberfläche generiert. Verbindet man sich mit einem Server, so wird dem Benutzer eine Spielernummer/ Zuschauernummer zugeordnet und ein neuer Thread gestartet um Nachrichten vom Server abzufangen. In den Client sind neben den im folgenden Teil erklärten Gewinnabfragen noch weitere Methoden implementiert. Darunter zählen einen neuen Chip in das Spielfeld einzufügen, es an die richtige Stelle einzufügen und den Spielzug an den Server zu kommunizieren, damit auch der Mitspieler und die Zuschauer den aktuellsten Spielstand angezeigt bekommen.

**2.2.4 Gewinnabfrage**

* **Vertikale Gewinnabfrage**

Für die vertikale Gewinnabfrage wird der Spieler der zuletzt einen Stein gesetzt hat in einem zweidimensionalen Array abgespeichert. Verweisen die Indizes des Arrays, die in diesem Fall die Spalte und Zeile darstellen in der der Stein gesetzt wurde, auf Spieler 1 so wird die komplette Methode für den jeweiligen Spieler durchlaufen. Die Methode vergleicht dann jedes einzelne Feld in der festgelegten Spalten und überprüft ob vier Steine einer Farbe vertikal in einer Reihe sind und speichert die Anzahl der Steine einer Farbe in der Gewinnzählervariablen. Wenn die Gewinnzählervariable dann 4 beträgt, wird die Schleife beendet und ein Text ausgegeben.

* **Horizontale Gewinnabfrage**

Ähnlich wie bei der vertikalen Abfrage wird auch bei der horizontalen Abfrage der Spieler der als letztes einen Stein gesetzt hat in einem Array abgespeichert. Allerdings wird hier lediglich die Zeile in der der Stein gesetzt wurde überprüft und die Spalten bis zum letzten Feld abgegangen. Insgesamt läuft die Schleife der Methode mindestens vier Mal und maximal sieben Mal durch. Wenn ein Stein der anderen Farbe in der Reihe enthalten ist, dann wird die Gewinnzählervariable auf Null gesetzt und die gesetzte Zeile wird für den jeweiligen Spieler bis zur letzten Spalte zu Ende überprüft. Auch bei der horizontalen Abfrage wird ein Gewinntext ausgegeben, wenn die Variable vier beträgt.

* **Diagonale Gewinnabfrage**

Für die diagonale Gewinnabfrage existieren zwei verschiedene Funktionen. Eine überprüft die diagonale nach links oben und die andere nach rechts oben. Hierbei wird die Gewinnzählervariable auf 1 gesetzt, da wir davon ausgehen, dass ein Stein bereits gesetzt wurde. Beide Methoden besitzen zwei for-Schleifen, eine die die diagonale Reihe nach rechts unten testet und eine die nach links oben testet.

Dazu wird eine zweite Zählervariable i immer 1 aufaddiert und auf die aktuelle Zeile gerechnet bzw. von der aktuellen Zeile subtrahiert. So gelangt man dann zu jedem neuen Feld der diagonale nach links und kann dann überprüfen, ob Steine einer Farbe nebeneinander liegen.

## 2.3 Benutzercharakteristika

Grundsätzlich kann das Spiel von jedem verwendet werden, der einen Computer besitzt. Die einzige Schwierigkeit könnte bei der Anmeldung an den Server liegen.

## 2.4 Soll-Kriterien

Es ist nicht möglich ein Unentschieden zu erzielen, da die passende Gewinnabfrage nicht implementiert wurde.

**2.5 Erweiterungen**

Da es in der heutigen Zeit immer beliebter wird anderen beim Spielen zu zusehen, haben wir auch einen Zuschauermodus eingebaut. Die ersten beiden Spieler die sich beim Server einloggen sind aktive Spieler und alle weiteren haben zwar die Einsicht auf das Feld, können aber keine Steine setzten oder andere Aktionen ausführen.

**3. Management Summary**

**3.1 Ausgangsanalyse**

**3.2 Arbeitspakete**

**3.3 Probleme**

Zu Beginn hatten wir beim Erstellen der GUI Probleme die weißen Kreise in das Layout einzufügen. Deshalb sind wir von der SWIG Bibliothek auf Java FX umgestiegen.

**3.4 Lösungen**

**3.5 Lessons Learned**