Programmierprojekt SS2015: Netzwerk Racetrack

Teilnehmer: Denis Heid, Sotirios Pavlidis, Stefan Feil, Tobias Kaulich
Betreuer: Till Bruckdorfer, Michael Bekos
Arbeitsbereich Algorithmik

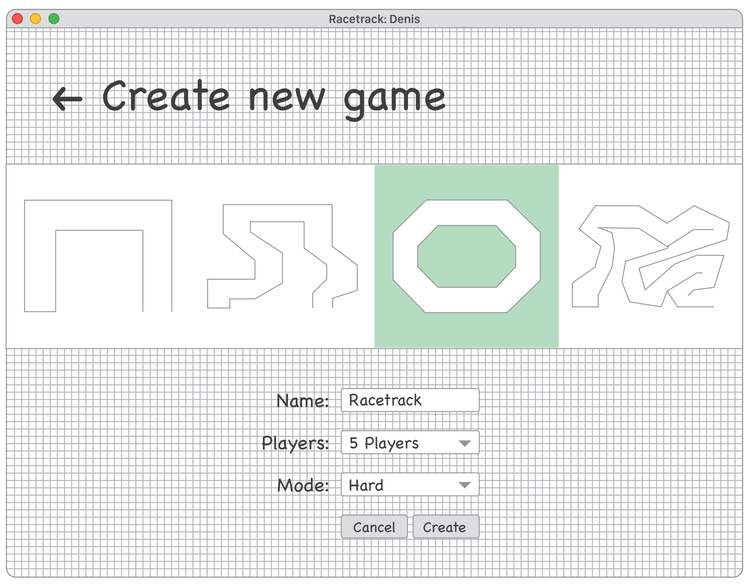
1 Einführung

Netzwerk Racetrack ist ein rundenbasiertes Rennspiel, abgeleitet vom Original, welches man auf Papier spielt und nun als Netzwerkspiel implementiert wurde. Spieler können sich einen Namen geben und bekommen unterschiedliche Farben zugewiesen mit denen sie sich wiedererkennen können. Die Herausforderung bei diesem Spiel ist es, vom Startpunkt mit weniger Zügen als die Mitspieler ins Ziel zu gelangen. Erschwert wird das Abbiegen durch die Trägheit der Autos und einer begrenzten Reaktionszeit.

2 Das Spiel

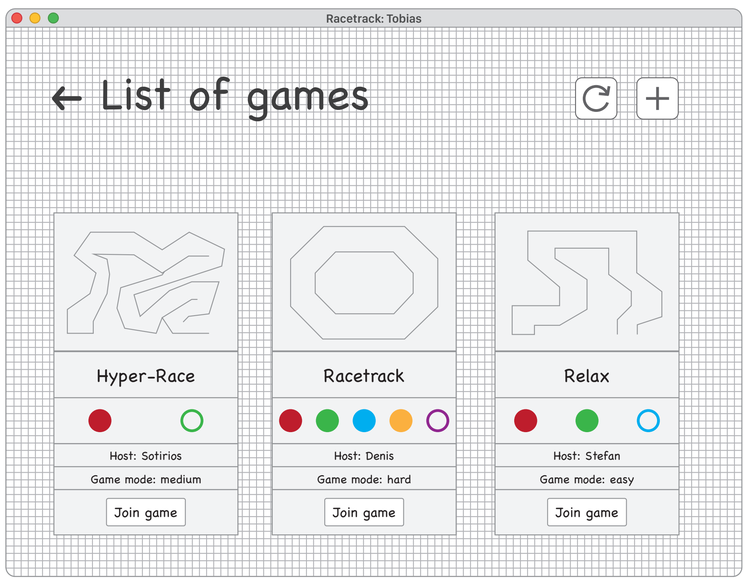
2.1 Spiel erstellen

Sobald man auf dem Server angemeldet ist und auf Startbutton klickt, gelangt man in die Spieleliste. Hier kann man entweder einem bestehenden Spiel beitreten oder ein Eigenes erstellen. In der oberen rechten Ecke ist der Aktualisierungsbutton, mit dem vom Server aktuellere Informationen angefordert werden können. Das Erstellen eines neuen Spiels erfolgt über das "+" Symbol neben dem Aktualisierungsbutton: Man hat die Möglichkeit die Spieleranzahl, den Schwierigkeitsgrad sowie die Strecke auszuwählen. Es gibt 5 verschiedene Strecken und 3 Schwierigkeitsgrade. Die Spieleranzahl reicht von 2 bis 5. Der Schwierigkeitsgrad beeinflusst die Zeit die man zur Auswahl seines Zuges hat (leicht: 30s, mittel: 15s, schwer: 5s).



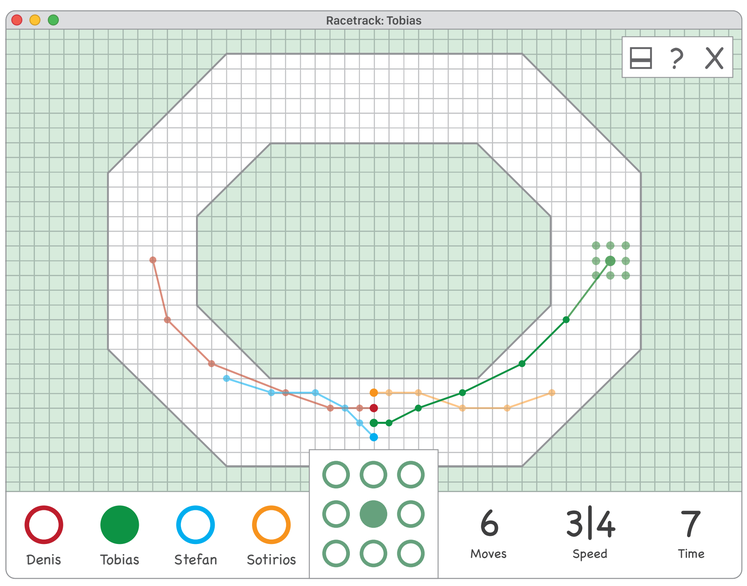
2.2 Spielanleitung

Am Anfang eines Spiels wählt sich jeder Spieler in umgekehrter Reihenfolge zur Spiel-Beitritts-Reihenfolge einen Startpunkt von einer Auswahl an Startpositionen aus. Im darauf folgenden Spielverlauf ist diese Reihenfolge umgedreht und der erste Spieler kann seine Position ändern. Dabei darf die Geschwindigkeit immer um +/-1 oder 0 in jeweils X- und Y-Richtung verändert werden. Der Sieger steht fest, sobald alle anderen Spieler gegen die Streckenbegrenzung gefahren sind oder er über die Ziellinie gefahren ist. Wenn man innerhalb des Zeitlimits keinen Zug macht wird die vorhandene Geschwindigkeit beibehalten.



3 Grafische Oberfläche

Am unteren Rand des Spielfeldes sind links die teilnehmenden Spieler aufgereiht, dabei ist der sich an der Reihe befindende Spieler mit einem gefüllten Kreis visuell hervorgehoben. Spieler, die nicht mehr verbunden sind oder durch einen Crash nicht mehr mitspielen können, sind ausgegraut. Sobald der jeweilige Spieler an der Reihe ist, kann er entweder über das Auswahlfeld unten mittig seine Geschwindigkeit verändern (jeder Punkt repräsentiert eine mögliche Kombination von X-/Y- Geschwindigkeitsveränderung), oder er drückt auf dem Spielfeld auf eine der erreichbaren Positionen. Der Timer in der unteren rechten Ecke zeigt an wie viel Zeit für den Spielzug verbleibt. Daneben lässt sich die momentane Geschwindigkeit erkennen, sowie die Anzahl der gespielten Züge.



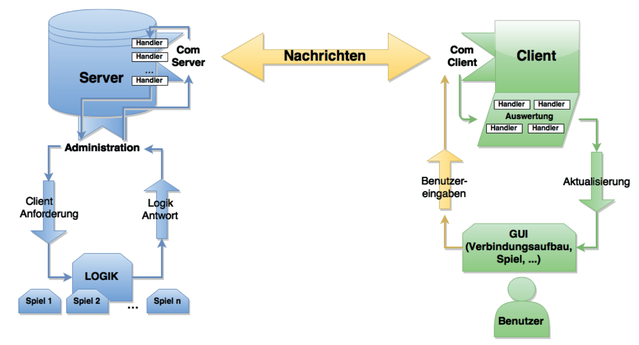
4 Architektur

4.1 Netzwerk

Die Kommunikation zwischen Server und Client ruht auf einer TCP-Verbindung (TCP eignet sich am besten für die sichere Übertragung von minimalem Informationsgehalt). Sie wird einmalig bei der ersten Verbindung aufgebaut und besteht solange, bis der Nutzer sich ausloggt.

4.2 Client

Auf der Seite des Clients werden Informationen eingehender Nachrichten in einer Datenstruktur gespeichert, aus der die grafische Oberfläche ihre anzuzeigenden Informationen nimmt. Für jeden Nachrichtentyp gibt es einen Handler. Dieser wertet die Nachricht aus und benachrichtigt gegebenenfalls andere Teile des Programms, wie z.B. die GUI. Der Client besteht aus GUI, Datenstruktur und Controller. Er ist dafür zuständig Operationen des Users entgegenzunehmen und weiterzuleiten - hier wird nichts berechnet um eventuelles Schummeln zu unterbinden.



4.3 Server

Der Server ist das Herzstück, er übernimmt die Verwaltung von Spielern und Spielen sowie die Validierung und Verarbeitung der Eingaben aller User. Eine Administrationsebene ist für die Zuordung eingehender Nachrichten an die jeweiligen Spiele zuständig und die Verbreitung von validierten Informationen. Jedes Spiel ist ein vom Rest abgetrenntes Objekt, mit dazugehöriger Logik.

5 Team

Um mit den anfangs acht Teilnehmern an einem Projekt zusammen arbeiten zu können, wurde es in vier Teilbereiche eingeteilt mit gleichmäßig verteilter Gruppengröße von zwei pro Bereich: Netzwerk, Administration, Logik und GUI. Da sehr früh im Semester vier Teilnehmer das Team verließen, teilten wir uns in zwei Paare, mit Verantwortlichkeit für je zwei Bereiche auf, die zum Teil einander aushalfen. In wöchentlichen Sitzungen wurde der Fortschritt des Programms diskutiert, Ziele gesetzt und bisher Erreichtes vorgestellt. Aufgrund der stark geschrumpften Gruppengröße mussten mehrere Projektziele gestrichen werden, z.b. eine künstliche Intelligenz.

6 Methoden und Erfahrungen

Als Programmiersprache haben wir Java verwendet, als Editor Eclipse, zum Kommunizieren nutzten wir die Plattform Trello und das Projekt Repository verwalteten wir mit GIT. Aus diesem Projekt konnten wir mitnehmen: eine genaue Projektplanung zu frühem Zeitpunkt führt zu stark reduziertem Programmieraufwand. Desweiteren haben wir festgestellt, dass man Vorstellungen sehr präzise formulieren muss, damit jeder im Team das gleiche Bild vor Augen hat. Darunter fällt die Einhaltung von Deadlines, Designabsprachen und Schnittstellenverträgen.

7 Fakten

Anzahl Codezeilen: 14076
Anzahl Commits: 744
Anzahl Klassendateien: 102
Beliebteste Programmieruhrzeit: 13 Uhr
Durchschnittliche Klassengröße: 138 Zeilen