Programmierprojekt SS2015: Racetrack

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

1 Einführung

Racetrack ist ein rundenbasiertes Online Rennspiel, welches man auf einem zentralen Server spielt. Spieler können sich einen Namen geben und bekommen unterschiedliche Farben zugewiesen, mit denen sie sich wiedererkennen können. Das Spielfeld ähnelt stark einem Koordinatensystem mit streckenmarkierenden Linien, einem Start und einem Ziel. Die Herausforderung bei diesem Spiel ist es, vom Startpunkt aus mit weniger Runden als die Mitspieler ins Ziel zu gelangen.

1.1 Spielanleitung

Am Anfang eines jeden Spiels wählt sich jeder Spieler, angefangen beim letzten Teilnehmer, einen Startpunkt von einer Auswahl an Positionen aus. Im nun darauf folgenden Spielverlauf ist die Reihenfolge umgedreht und der erste Spieler kann seine Position ändern. Dabei darf immer die Geschwindigkeit um +/-1 und 0 in jeweils X- und Y-Richtung verändert werden. Somit kann man über die Runden hinweg sehr schnell werden, man muss aber auf Kurven aufpassen, da man durch nicht rechtzeitiges Abbremsen sehr leicht aus der Strecke fliegen kann und man dadurch verliert. Der Sieger steht fest, sobald entweder nur noch ein Spieler auf der Strecke verbleibt oder der Erste das Ziel erreicht.

1.2 Grafische Oberfläche

Spiel

Sobald man auf dem Server angemeldet ist und auf Start drückt, gelangt man in den Spielbrowser. Hier kann man entweder einem bestehenden Spiel beitreten oder ein eigenes erstellen. Gelistet sind alle Spiele, die noch nicht voll oder gestartet sind. Das Erstellen erfolgt über das (+) Symbol in der oberen rechten Ecke. Links daneben ist der Nachladebutton, mit dem vom Server aktuellere Informationen angefordert werden können. Erstellt man ein neues Spiel, hat man die Möglichkeit die Spieleranzahl, den Schwierigkeitsgrad sowie die Strecke auszuwählen. Es gibt bei Release 5 verschiedene Strecken, 3 Schwierigkeitsgrade und die Spieleranzahl reicht von 2 bis 5. Beim einfachsten Schwierigkeitsgrad hat man 30 Sekunden Zeit für seine Runde, beim mittleren 15 Sekunden und beim schweren 5. Somit kann man sehr einfach kontrollieren, wie chaotisch das Spiel werden soll - Für das absolute Durcheinander empfehlen wir den schweren Modus mit 5 Spielern, für das angeheneme Zusammensein eine leichte Runde zu zweit.

User Interface

Der größte Teil des Bildschirmes wird durch die Strecke eingenommen, für taktisches Vorausberechnen ist sie komplett zu sehen. Am unteren Rand des Spielfeldes sind die teilnehmenden Spieler aufgereiht, dabei ist der sich ander Reihe befindende Spieler visuell hervorgehoben, Spieler die disconnected sind oder durch einen Crash nicht mehr mitspielen können sind ausgegraut. Sobald man an der Reihe ist, kann man entweder über das sich in der unteren Mitte befindende Auswahlfeld seine Geschwindigkeit verändern (jeder Punkt repräsentiert eine mögliche Kombination von X-/Y- Gewschwindigkeitserhöhung/-Verringerung), oder man drückt auf dem Spielfeld auf eine der für einen erreichbaren Positionen. Rechts neben dem Auswahlfeld sieht man die zurzeitige Rundenanzahl, seine Geschwindigkeit und eventuell, falls man an der Reihe ist, seine zur Auswahl der Position verbleibende Zeit. Am oberen Bildschirmrand hat man eine textliche Repräsenation des zurzeitigen Spielzuges, rechts daneben die Option diese aus/ein-zublenden, eine Hilfefunktion und die Möglichkeit das Spiel zu verlassen.

2 Architektur

2.1 Netzwerk

Die Kommunikation zwischen Server und Client ruht auf einer TCP-Verbindung (TCP deshalb, da das Spiel rundenbasiert ist, und sehr wenig Information über den Spielverlauf gesendet wird, die dafür aber sehr relevant ist und auf jeden Fall ankommen muss). Sie wird einmal beim erstigen Verbinden aufgebaut und besteht die ganze Zeit über, solange der Nutzer sich nicht wieder ausloggt. Der Server wartet nach einmaligem starten auf Port 55570 auf Clients. Beim Verbindungsaufbau bekommt jeder Client eine eindeutige Verbindungsnummer zugewiesen, die diese Verbindung repräsentiert, für den Nutzer selber sichtbar ist aber nur sein (eindeutiger) Name.

2.2 Client

Auf dem Client werden Informationen eingehender Nachrichten in eine Art Austauschobjekt eingespeichert, aus der die grafische Oberfläche ihre anzuzeigenden Informationen nimmt. Über Handler, welche die Nachrichten verarbeiten, wird dieser Prozess des Anzeigens neuer Informationen eingeleitet. Der Client ist nur eine visuelle Repräsentation der Serverinformationen, dafür zuständige Operationen des Users entgegenzunehmen und weiterzuleiten - hier wird nichts berechnet um eventuelles Cheaten zu unterbinden, alles läuft über den zentralen Server.

2.3 Server

Ohne den Server läuft nichts, er verwaltet die Clients, ordnet sie Spielen zu, übernimmt die Berechnung und Validierung von eingehenden Eingaben der User. Eine Administrationsebene ist für die Zuordung eingehender Nachrichten zuständig und die verbreitung von validierten Informationen, jedes Spiel ist ein vom Rest abgetrenntes Objekt, mit dazugehöriger Logik.

3 Team

Um mit den anfangs 8 Teilnehmern an einem Projekt gut arbeiten zu können, wurde es in 4 Teilbereiche eingeteilt mit gleichmäßig verteilter Gruppengröße von 2 pro Bereich. Netzwerk, Administration, Logik und GUI. Da im Verlaufe des Semesters die Hälfte des Teams dieses verließ, wurden aus den 4 2er-Teams kleinere Einheiten, die zum Teil einander aushelfen mussten. Unter hilfreicher Anleitung der 2 Tutoren wurde das Projekt seiner Fertigstellung entegegengeleitet, sie unterstützen die Koordination in der Gruppe, so dass Bereiche, die hinterherhinkten die nötige Aufmerksamkeit bekamen. In wöchentlichen Sitzungen wurde der Fortschritt des Programms diskutiert, Ziele gesetzt und bisher erreichtes vorgestellt. Dadurch wurde das Projekt kontinuierlich der Fertigstellung entgegengebracht.

4 Methoden und Erfahrungen

Als Programmiersprache haben wir Java verwendet, es bietet alles was man für ein solches Projekt braucht und ist in Kombination mit Eclipse somit hervorragend geeignet. Zu Beginn des Projekts haben wir uns mit der Programmierumgebung vertraut gemacht, indem wir verschiedene Mini-Projects zu bearbeiten bekommen haben welche simplistische Repräsenationen der eigentlichen Aufgaben darstellten, um mit der den Problemenstellungen klarzukommen. Außerdem haben wir uns mit Debugging in Java, JUnit, Trello und GIT vertraut gemacht. Diese Hiflsmittel verwendeten wir um gut als große Gruppe an einem Programm zu arbeiten und uns zu koordinieren.
Wir lernten in diesem Projekt sehr viel über das Arbeiten im Team, daraus resultierende Vorzüge und Probleme. Darüber hinaus haben wir festgestellt, dass eine genaue Projektplanung zu einem später stark reduzierten Programmieraufwand führt und man sich als Team präzise abstimmen muss um einen reibungslosen Ablauf zu garantieren.

5 Fakten

Anzahl Codezeilen: 51180
Anzahl Commits: 644
Anzahl Klassendateien: 95
Beliebteste Programmieruhrzeit: Mittwochs 0 Uhr und 11 Uhr
Durchschnittliche Klassengröße: 141 Zeilen