Moorhuhn als Web-Multiplayer HTML5 Spiel

Projektziel

Das Ziel des Projektes ist es ein Moorhuhn-ähnliches Multiplayer Spiel als Web-Applikation zu programmieren. Dabei gibt es eine festgelegte Anzahl an Hühnern, die sich mit Pfeiltasten über das Spielfeld bewegen und einen Schützen, der versucht diese abzuschießen.

Projektaufbau

Für das Projekt wird mit einem PEAN-Stack (PostgreSQL, Express, AngularJS, NodeJS) aufgebaut.

Wir haben uns für NodeJS entschieden, da sich Node zum einen aufgrund der Asynchronität und zum anderen, dass es sich für die Spieleentwicklung aufgrund der unterstützten Bibliotheken, wie zum Beispiel Socket.io eignet. Außerdem können so Teile des Spiels vom Server auf den Client bedenkenlos kopiert werden, da beides in JavaScript geschrieben wird.

Für Express haben wir uns entschieden, da es einen einfachen und klar strukturierten Aufbau hat. Alternativ dazu haben wir uns noch Sails.js, sowie Meteor.js angeschaut, allerdings waren diese deutlich komplexer und unübersichtlicher.

AngularJs haben wir als Frontend gewählt, da unsere Applikation recht simpel aufgebaut ist und wir nur wenige „Ansichten“ der Webseite haben. Außerdem empfanden wir es als sehr angenehm für das Benutzererlebnis, ein One-Page-Frontend zu haben.

Die Datenbank haben wir mit PostgreSQL realisiert. Dies hat den einfachen Grund, dass wir dieses in einem anderen Modul bereits tiefgründig behandelt haben und uns somit nicht an Kleinigkeiten aufhalten, sondern uns auf die wichtigen Sachen des Projekts konzentrieren können.

Challenge: HTML5 Darstellung

Zur Darstellung des Spiels wurde im HTML das Canvas Element benutzt. Dabei wird für jedes darzustellende Bild das Spielfeld neugezeichnet. Die über das Spielfeld fliegenden Hühner sind einfache png-Dateien, die auf das Canvas gezeichnet werden. Das Spielfeld wird mit Dimensionen, die deutlich höher sind, Dimensionen auf dem Bildschirm sind, berechnet. Somit ist es möglich, dsas Spielfeld auf voller Größe mit verschiedenen Bildschirmauflösungen darzustellen. Die Größe des Canvas Elements kann somit auch geändert werden. Auch während des Spiels ist dies möglich, ohne dass irgendwelche Darstellungsfehler oder Logikfehler auftreten. Um die Animationen möglichst flüssig zu gestalten, wird das Spielfeld pro Sekunde circa 30 mal neu berechnet. Zum Darstellen wird dann die vom Browser bereitgestellte Funktion window.requestAnimationFrame verwendet.

Challenge: Echtzeitkommunikation/Synchronisation

Um einen flüssigen Spielverlauf zu garantieren, wird zur Server-Client Kommunikation Socket.io verwendet. Socket.io baut eine Websocket Verbindung auf, mit der in Echtzeit Nachrichten zwischen Client und Server ausgetauscht werden können.

Zur Steuerung der Hühner werden die Pfeiltasten verwendet. Diese können sich somit nur horizontal oder vertikal bewegen. Diagonale Bewegung, sowie Stehenbleiben ist nicht möglich.

Damit jedoch nicht nach jeder Neuberechnung auf dem Server die Positionen an den Client geschickt werden. Berechnen Client und Server parallel die Positionen der Hühner. Der Server synchronisiert dann die Positionen, die er berechnet hat, mit denen der Clients in regelmäßigen Abständen.

Resolution