Datenbiz

Ausarbeitung für die Nebenperspektive Visual Computing

Entwicklungsprojekt 2019/2020

Adrian Bertram & Janina Sperling

Hauptperspektive Social Computing

Inhaltsverzeichnis

1 Einleitung 2
2 Wie funktioniert Datenbiz? 3
3 Konzeption 4
4 Programmierung 5
4.1 Objekte in Unity 5
4.2 Felder 6
4.3 Spielkarten 6
4.3.1 Ereigniskarten 7
4.4 Würfel 7
4.5 PlayerMovement 7
4.6 Spiel beenden 8
5 Designaspekte 8
6 Zukunftsvision 10
7 Anhang 11
7.1 Use Case Player Turn 11
7.2 Abwägung der Entwicklungsumgebung 12

1 Einleitung

Vielen Studierenden fällt es meist schwer für Prüfungen zu lernen und dies auch regelmäßig zu tun. Oft steht dies damit im Zusammenhang, dass das Lernen als etwas Unangenehmes und Anstrengendes empfunden wird und nicht wirklich abwechslungsreich gestaltet werden kann.

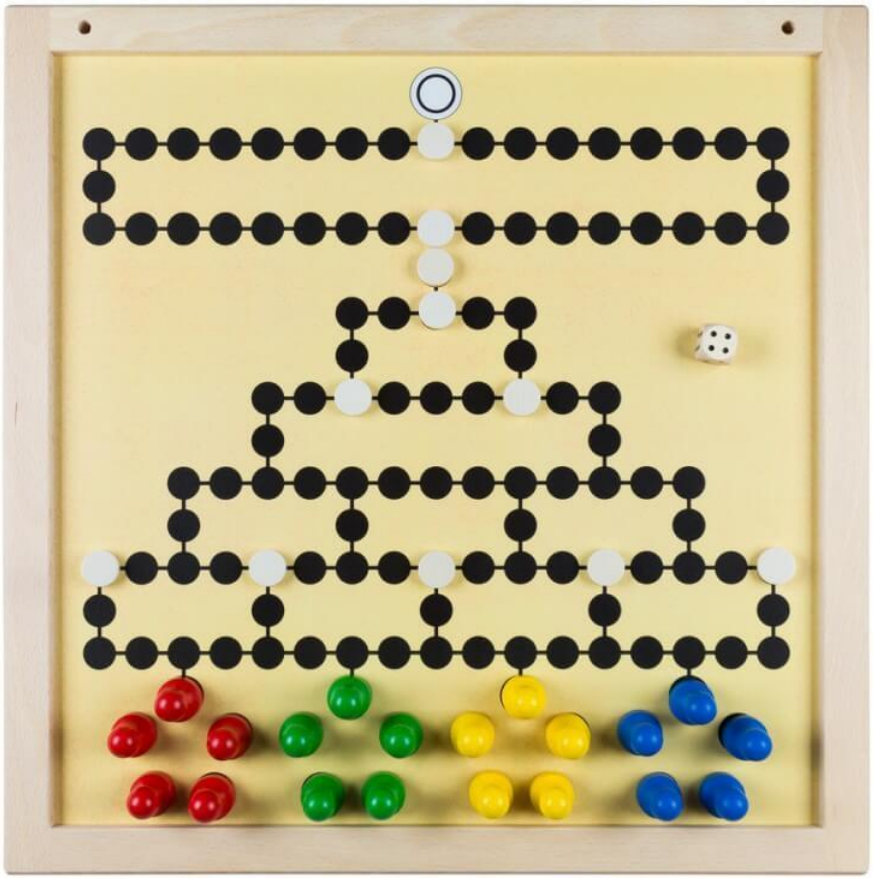
Dieser Prozess des Lernens soll zum Teil vereinfacht werden, in dem der Lernstoff auf eine spielerische Weise vertieft wird. Dadurch soll langfristig Spaß sowie eine gewisse Leichtigkeit beim Lernen entstehen. Konkreter soll durch spielerische Mittel das Lernen angenehmer gestaltet werden. Außerdem soll eine Abwechslung durch das mögliche Abfragen unter Freunden entstehen, da solche soziale Elemente den langwierigen Lernprozess entschlacken und motivieren können.
Es wird erhofft, dass daraus bessere Prüfungsergebnisse resultieren, welche wiederum motivierend für die Studierenden sind.

Als Themengebiet zur Umsetzung des Projekts wurde sich für das Modul Datenbanken entschieden. Dies liegt daran, dass das Modul sehr umfangreichen Lernstoff beinhaltet, welcher sich gut abfragen lässt und somit von den Studierenden besser verinnerlicht werden kann.

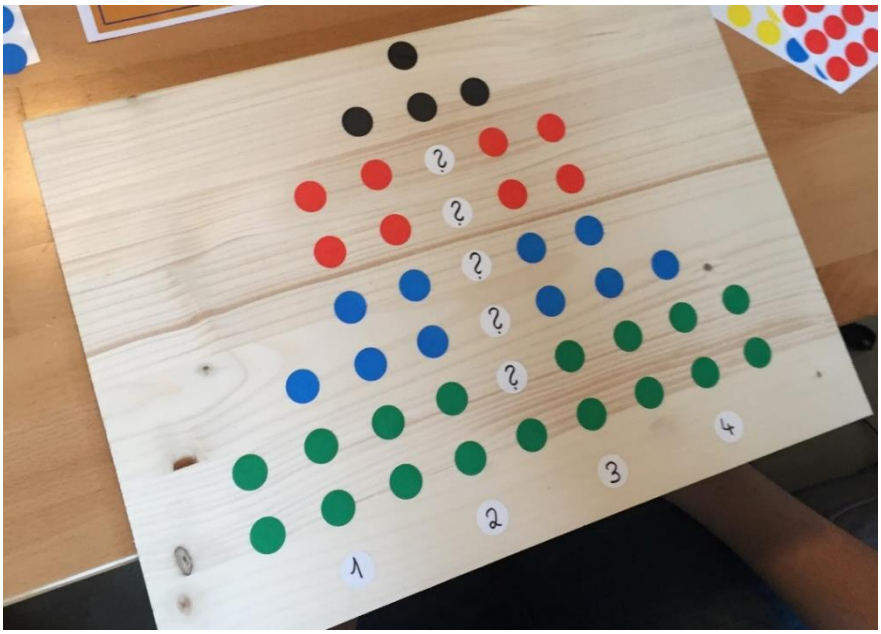
Als Hauptperspektive für dieses Projekt wurde Social Computing gewählt, da dieses Projekt ein Serious Game ist und dieser Themenbereich ein wichtiger Baustein in diesem Modul ist. Zudem wurde dieses Projekt zuvor bereits in dem Modul Social Computing analog umgesetzt.
Die Nebenperspektive ist Visual Computing, da wir visuelle Daten erzeugen und wir diverse Bibliotheken und Frameworks für dieses Projekt benötigen. Außerdem umfasst Visual Computing viele Aspekte, welche für die digitale Spielentwicklung von großer Relevanz sind.

2 Wie funktioniert Datenbiz?

Die Inspiration für unser Spielbrett haben wir von dem Spiel Malefiz.



Datenbiz lässt sich ebenfalls mit bis zu 4 Spielern spielen.
Zu Beginn wird ausgewählt, wie viele Spieler am Spiel teilnehmen möchten.
Nachdem man die Anzahl ausgewählt hat, beginnt das Spiel.



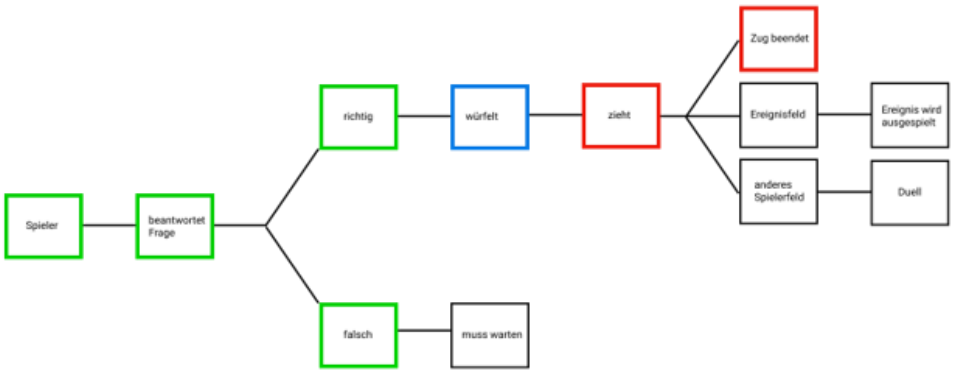
Als erstes beginnt Spieler 1 und zieht eine Karte. Es erscheint eine Frage. Wird die richtige Antwort ausgewählt, darf der Spieler würfeln und sich auf dem Spielbrett fortbewegen. Dann ist der nächste Spieler am Zug. Welche Frage gezogen wird, bestimmt die farbliche Kennung der Spielfelder, auf denen der Spieler sich befindet. Die Schwierigkeit wird aufsteigend durch die Farben Grün, Blau, Rot und Schwarz bestimmt. Das Startfeld bestimmt eine grüne Frage.

Die Felder mit den Fragezeichen in der Mitte sind Ereignisfelder. Dort bekommt man Karten, wie zum Beispiel: "Du darfst noch einmal Würfeln" oder "gehe 3 Felder in eine Richtung deiner Wahl".

Wenn ein Spieler das Feld eines anderen Spielers betritt, kommt es zum Duell. Beide Spieler beantworten solange Fragen, bis einer der beiden Spieler eine Frage falsch beantwortet hat und der andere richtig. Der verlierende Spieler muss nun auf das alte Spielfeld des Duell-Anforderers gehen.

Wer zuerst das oberste schwarze Feld erreicht und dort eine Frage richtig beantwortet, hat gewonnen.

3 Konzeption



Das obige Modell beschreibt einen Use Case eines Spielerzugs in unserem System. Die farbliche Kennung markiert die Unterteilung dieses Spielerzugs in die einzelnen Proof of Concepts.

Grün beschreibt die Abfrage, zu diesem Prozess gehört die korrekte Abfrage der passenden Frage zu der Farbe des Feldes, die Auswahl der Antwortmöglichkeiten und das Schließen der Frage- bzw. Antwortfelder. Außerdem muss die richtige Lösung angezeigt werden, falls die Frage falsch beantwortet wird, und im Falle einer richtigen Auswahl diese auch wiedergegeben werden als solche.

Blau beschreibt das Würfeln, der Würfelwurf soll beim Klicken auf den Würfel dargestellt werden und die resultierende Würfelzahl korrekt übergeben werden.

Rot beschreibt das PlayerMovement, welches besagt, dass die Figur bewegt werden können soll und der aktive Block/Position des Spielers gespeichert werden soll.

Diese 3 Proof of Concepts umfassen dann einen großen Proof of Concept, wo alle diese kleinen Proof of Concepts erfüllt sein müssen.

Die komplette Beschreibung des Use Cases findet sich detailliert im Anhang.

Bei der Konzeption war es wichtig, dass wir zu Beginn eine klare Definition eines Spielerzugs machen, um die zeitliche Abfolge unserer Programmierung strukturieren zu können.

Es ist noch wichtig zu erwähnen, dass wir die Duell-Funktion aus zeitlichen Gründen nicht mehr implementieren konnten.

Der Spieler beginnt seinen Zug mit dem Drücken auf den "Frage ziehen" - Button.

Dieser ist nur einmal klickbar, um Betrügen zu vermeiden. Sonst könnte ein Spieler Karten ziehen, bis er eine Frage beantworten kann.

Bevor wir die Karten in Unity eingebunden haben musste erstmal etwas wichtiges für uns geklärt werden: möchten wir Multiple Choice Karten oder soll der User seine Antwort eingeben können?
In der analogen Version von Datenbiz kann man seine Antwort einfach sagen und die Mitspieler beurteilen, ob die gegebene Antwort der vom Spiel vorgesehenen Antwort entspricht.
Diese Variante wollten wir gerne beibehalten, jedoch haben wir viele negative Aspekte bei der Entwicklung eines eigenen Algorithmus für die Überprüfung einer eingegebenen Antwort gesehen. Nicht nur, dass dabei auf Groß- und Kleinschreibung sowie auf Rechtschreibfehler geachtet werden muss, sondern auch, dass es viele sinngemäß richtige Antworten gibt, die nicht alle berücksichtigt werden können. Aufgrund dessen und dem relativ kleinen Zeitrahmen, welcher uns für dieses Projekt zur Verfügung steht, haben wir uns gegen einen eigenen Algorithmus und somit für Multiple Choice Karten mit jeweils drei Antwortmöglichkeiten entschieden.

Des Weiteren haben wir ebenfalls drauf geachtet, dass der Würfel nur einmal klickbar ist, um weiteres Betrügen zu vermeiden.

4 Programmierung

Bei der Programmierung haben wir zuerst die verschiedenen Elemente eingebunden, um diese anschließend für unseren Prototypen zu verknüpfen. Dabei war uns stehts wichtig, dass unser Spiel mit verschiedenen Endgeräten kompatibel ist.
Die grundlegenden Elemente lassen sich aus dem oben genannten Use Case schließen. Diese und weitere Elemente sind anschließend aufgelistet und mit einer Erläuterung der Programmierung versehen.

4.1 Objekte in Unity

Als Engine haben wir Unity benutzt. Eine Erläuterung zu unseren Beweggründen diesbezüglich findet sich im Anhang.
Um das Spielbrett darzustellen wurde mit Cubes gearbeitet. Jeder Cube entspricht einem Feld auf dem Brett und ist versehen mit einem Tag, welcher aussagt, welche Farbe und somit auch welchen Schwierigkeitsgrad dieses Feld hat.
Außerdem haben wir 4 Spielobjekte für die einzelnen Spielfiguren erstellt.
Die Spielkarten als auch der Würfel werden über ein Image visualisiert und die Interaktionen durch den User werden durch Buttons ermöglicht.

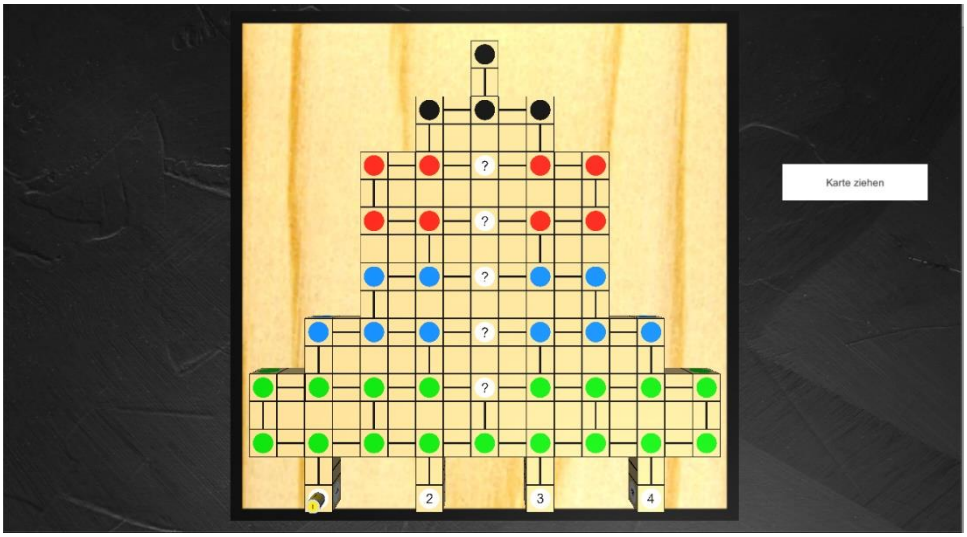
Zudem haben wir zwei Szenen erstellt, damit die gewünschte Spieleranzahl vor Beginn des Spiels angegeben werden kann.
Die ausgewählte Anzahl von Spielern in der ersten Szene wird über die Player Preferences an die zweite Szene übergeben (Skript Startbildschirm). Die erste Szene für den Startbildschirm ist im Folgenden dargestellt.

Abhängig von der gewünschten Spieleranzahl werden dann beim Laden der zweiten Szene im Skript SpielerCount die nicht benötigten Spielfiguren auf inaktiv gesetzt.

4.2 Felder

Die Eigenschaften der Felder sind im Tile Skript deklariert. Das für uns wichtigste Element eines Feldes ist das NextTile Array, welches aussagt, welche die angrenzenden Felder sind. Dieses Array findet Gebrauch, sobald ein Spieler seine Figur auf dem Feld bewegen darf.
Das Array enthält ausschließlich die angrenzenden Felder des aktuell ausgewählten Feldes. Dadurch kann nicht der komplette Weg angezeigt werden, sondern es müssen immer die entsprechenden Felder angesprochen werde.

Das Skript enthält zudem die Funktionen, um die Collider der angrenzenden Felder an- oder auszuschalten, damit diese für den User klickbar sind. Zudem wird hier auch der Highlight-Effekt für die begehbaren Felder eingebunden, indem wir verschiedene Materials für die Cubes verwendet haben, die hier ausgewählt werden. Das Spielbrett inklusive aller Felder im Startzustand sieht wie folgt aus:



4.3 Spielkarten

Für die Einbindung der Spielkarten wurden zwei Skripte erstellt: Das Erste zum Ziehen einer Karte und das Zweite, um die getippte Antwort auf Richtigkeit zu überprüfen und die entsprechende Lösung anzuzeigen.

Im Skript KartenZiehen wird durch den Tag des Feldes, auf welchem der aktuelle Spieler sich befinden, das entsprechende Kartenarray angesprochen und anschließend eine wahllose Karte aus diesem Set dem Spieler angezeigt. Ist die Karte keine Ereigniskarte, so werden zudem die Buttons zur Auswahl der Antwortmöglichkeiten aktiviert.

Im Skript AntwortAnzeigen wird erneut der Tag des Feldes herausgefunden und zudem wird ein Lösungsarray angegeben. Dieses Array sagt aus, bei welcher Karten welche Antwortmöglichkeit die Richtige ist. Dies ist aktuell keine optimale Lösung, jedoch für den Prototypen mit ausschließlich vier Karten pro Set vollkommen ausreichend und funktionsfähig.
Hat der User eine Antwortmöglichkeit ausgewählt, so wird diese Antwort mit dem Lösungsarray vergleichen und dem Ergebnis entsprechend eine Antwortkarte ausgegeben. Da es pro Frage zwei Antwortkarten gibt (1. Richtige Antwort gewählt 2. Falsche Antwort gewählt), wird die gewünschte Karte hier nicht über ein Array, sondern über den Dateipfad angesprochen.
Ist die gewählte Antwort richtig, so wird der Würfel aktiviert und zusätzlich wird abgefragt, ob sich der Spieler auf dem letzten Feld befindet, womit er das Spiel gewonnen hätte.
Ist die gewählte Antwort falsch, so ist automatisch der nächste Spieler am Zug.
Der nächste Spieler wird im Skript SpielerCount angegeben.

4.3.1 Ereigniskarten

Aktuell sind drei Ereigniskarten in das Spiel eingebunden

1. Der Spieler darf erneut würfeln. Dies wurde einfach durch Aktivierung des Würfels ermöglicht.
2. Der Spieler darf drei Felder in die Richtung seiner Wahl gehen. Dafür wird die entsprechende Funktion aus dem PlayerMovement Skript mit dem Übergabeparameter 3 aufgerufen.
3. Jeder Spieler muss an seine Startposition zurückkehren. Dabei wird die Position, der currentTile und auch das tileArray wieder auf den Anfang gesetzt.

4.4 Würfel

Das Würfelrollen wird durch das Anzeigen von 20 Würfelseiten nacheinander simuliert.
Die endgültige Würfelanzahl wird in einer Variable gespeichert und an das PlayerMovement Skript des aktuellen Spielers übergeben.

4.5 PlayerMovement

Im PlayerMovement Skript soll die Spielfigur auf ihre neue Position bewegt werden.
Wir haben uns bewusst dafür entschieden, den User jedes Feld (entsprechend der Würfelanzahl) separat auswählen zu lassen, anstatt alle Schritte mit einem Klick zu gehen, um ein angenehmes Spielgefühl zu erzeugen.

Die wichtigsten Elemente in diesem Skript sind der currentTile und das tileArray.
Der currentTile sagt aus, auf welchem Feld der Spieler sich aktuell befindet, und wird beim Starten des Spiels gleich dem startingTile gesetzt, welcher für jeden Spieler im Unity Editor festgelegt ist.
Das tileArray enthält die angrenzenden Felder des currentTiles.

In einer For-Schleife wird Folgendes für jeden zu gehenden Schritt ausgeführt:
Zuerst erfolgt eine Abfrage, ob der currentTile dem letzten Feld entspricht. Ist die der Fall, so wird auf das SpielBeenden Skript verwiesen.
Trifft dies nicht zu, werden die Collider für die angrenzenden Felder aktiviert.
Anschließend wird eine While-Schleife durchlaufen, in der auf einen Klicks des Users gewartet wird.

Innerhalb dieser Schleife wird mit Hilfe einer weiteren For-Schleife gecheckt, ob eines der angrenzenden Felder angeklickt wurde. Sobald das passiert, werden die Collider deaktiviert und der currentTile, das tileArray sowie die Spielerposition werden überschrieben.

Zuletzt wird der Würfel wieder deaktiviert und die Variable für den aktuellen Spieler wird um 1 erhöht.

4.6 Spiel beenden

Im gleichnamigen Skript wird abhängig von dem Spieler, welcher gewonnen hat, eine „Du hast gewonnen“-Karte ausgegeben. Durch Klicken des Users wird anschließend erneut die Start-Szene geladen. Da wir keine Variablen außerhalb der Szene speichern wird das Spiel wieder in den Anfangsmodus zurückgesetzt.

5 Designaspekte

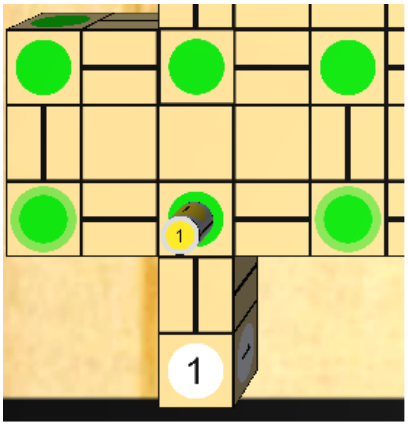
Die visuelle Darbietung hat bei uns einen großen Wandel vollzogen. Zu Beginn hatten wir erst ein "Work in Progress" Spielfeld, welches noch einige visuelle Verbesserungen dringend nötig hatte. Hier ist das erste Spielfeld zu sehen, mit welchem wir zu Beginn gearbeitet haben. Darunter zum direkten Vergleich noch einmal unser Aktuelles.

Wie sie unschwer erkennen können, ist dieses Bild lediglich ein Auszug aus unserem vorigen Audit. Dies liegt daran, dass wir in Unity leider nicht mehr auf den Zeitpunkt zurückgehen können, wo wir die alten Materials benutzen, da wir diese mittlerweile von unseren Rechnern gelöscht haben. Sonst hätten wir die beiden Screenshots einzeln herausgenommen und benutzt.

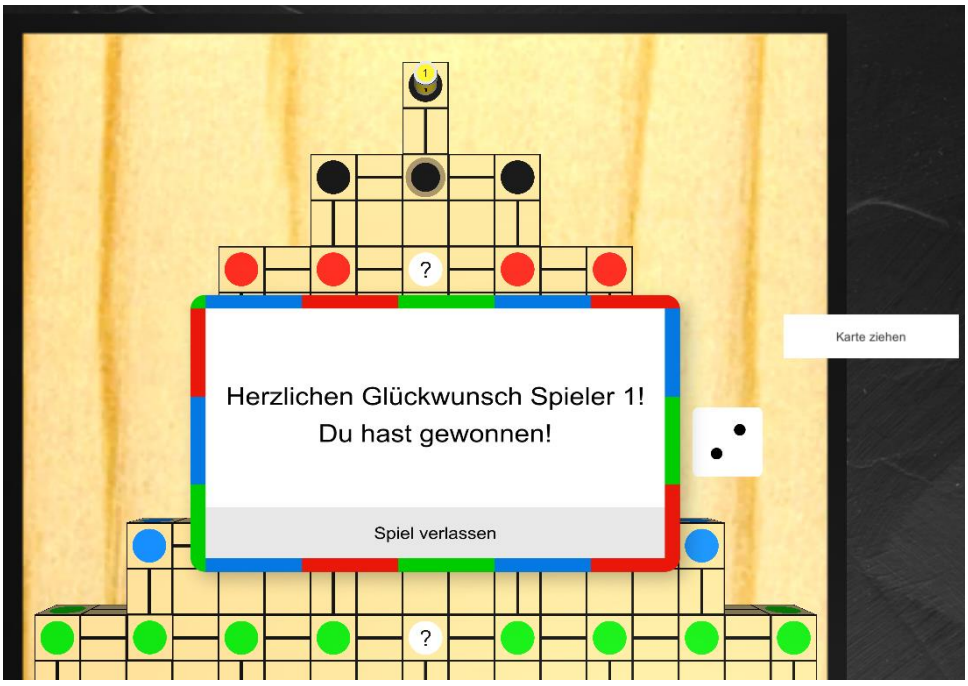
Der Spielaufbau des Bretts besteht aus Cubes, welche wir auf eine Plane gestellt haben. Der schwarze Hintergrund ist ebenfalls eine Plane. Wir haben uns für diese Farbe entschieden, weil wir einen unauffälligen, aber schönen Hintergrund haben wollten, der den Spieler jedoch nicht vom Spiel ablenken soll. Außerdem passt das Holz sehr gut zu dem dunklen Hintergrund.

Ein weiterer Punkt, der für unser neues Design spricht, ist die Orientierung an der analogen Version, welche wir in Social Computing bereits gefertigt haben.

Natürlich mussten wir viele Dinge in der digitalen Version anpassen und verändern, von Spielmechaniken bis hin zu Animationen. Wir haben nämlich außerdem bei einem Spielerzug eine Highlight-Mechanik hinzugefügt, welche dem Spieler bei jedem Move anzeigt, auf welches Spielfeld er gehen kann. Hier sehen sie eine Demonstration dieser Mechanik.



Außerdem haben wir im Zuge unseres Polishings noch ein wichtiges Element hinzugefügt. Eine Ankündigung, wenn ein Spieler das Spiel gewonnen hat.



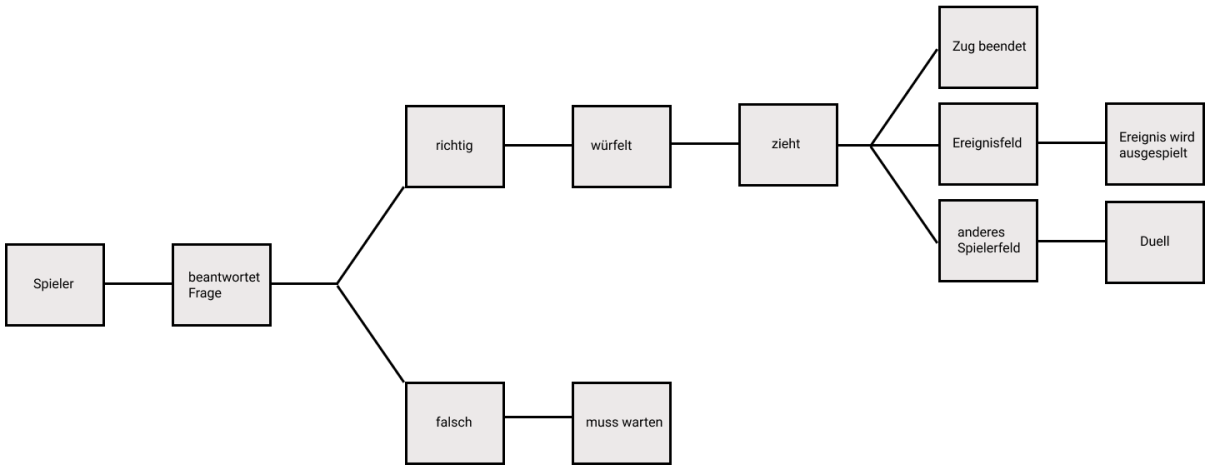
6 Zukunftsvision

Um nun kritisch auf unser Projekt zurückzublicken, haben wir alle wichtigen PoC's erfüllt. Jedoch gibt es doch ein paar Elemente, die wir gerne noch implementiert hätten, wenn wir noch mehr Zeit gehabt hätten. Als erstes die Duell - Funktion. Ursprünglich sollte, wenn ein Spieler das Spielfeld eines anderen Spielers betritt, ein Duell stattfinden, indem beide Spieler Fragen beantworten, bis einer der beiden Spieler eine Frage richtig beantwortet hat und der andere falsch. In dem Fall wäre dann der Spieler mit der falschen Antwort auf die Position des am Zug befindlichen Spielers zurückgefallen, bevor er den Würfel geworfen hat.

Ein weiteres Beispiel wäre die Fortbewegung, beziehungsweise die Positionsermittlung der Spieler. Aktuell geschieht dies über die x, y, und z Koordinaten, welches leider dazu geführt hat, dass man keine Bewegungsanimation für die Spielfiguren implementieren konnte. Dies müsste man mit einer Überarbeitung des Codes in Vektoren verändern, damit die zurückgelegte Strecke ermittelt und somit auch eine Animation erstellt werden kann. Aktuell werden nur die Koordinaten der Spielfigur transformiert. Des Weiteren hätten wir gerne eine Statistik eingeführt, welche einem Spieler beispielsweise die Quote der richtig beantworteten Frage für jede Schwierigkeitsstufe anzeigt. Auch die Möglichkeit der Erstellung eigener Karten für die Spieler wäre ebenfalls eine sinnvolle Erweiterung unseres Spieles.

7 Anhang

7.1 Use Case Player Turn



use case Spielerzug

actors Spieler

precondition true

main flow:

1. Der Spieler beginnt das Spiel
2. Dem Spieler wird eine Frage gestellt
3. Der Spieler beantwortet die Frage korrekt
4. Der Spieler würfelt
5. Der Spieler darf die gewürfelte Anzahl voran ziehen
6. Zug ist beendet

alternative flow Frage nicht korrekt beantwortet

3a. Wird die Frage nicht korrekt beantwortet, darf nicht gewürfelt werden und der Zug ist damit beendet.

alternative flow Ereignisfeld

6a.1. Betritt der Spieler nach dem Bewegen der Spielfigur ein Ereignisfeld, wird eine Ereigniskarte gezogen und ausgespielt.

6a.2 Der Zug ist beendet

alternative flow anderes Spielerfeld

6b.1. Betritt der Spieler nach dem Bewegen der Spielfigur ein Feld auf dem bereits ein anderer Spieler steht, duellieren sich beide

6b.2. Beiden wird eine unterschiedliche Frage gestellt, wer die Frage richtig beantwortet darf auf dem Feld bleiben, der andere muss auf die vorige Position des herausfordernden Spielers. Sollten beide die Frage richtig beantworten wird die Zeit gemessen und der schnellere Spieler gewinnt.

7.2 Abwägung der Entwicklungsumgebung

Die möglichen Entwicklungsumgebungen lassen sich auf Grund der Art unseres Projektes bereits einschränken. Mögliche Umgebungen zur Spielentwicklung sind beispielsweise Unreal Engine, 3D Gamemaker und Unity.
Wir haben uns für Unity entschieden, da wir ein 2D Brettspiel entwickeln möchten und diese Umgebung uns dafür viele Möglichkeiten bietet. Unity bietet den gewünschten Support zur Spielentwicklung und es gibt zahlreiche Guides, um sich in die Entwicklungsumgebung herein zu arbeiten. Weiterhin ist die Collaborate Funktion, welche Unity bietet, ein wichtiger Faktor für unsere gemeinsame Entwicklung des Projekts. Zudem ermöglicht Unity das Zurückspringen zu vergangenen Speicherzeitpunkten, was gerade in Kombination mit der Collaborate Funktion sehr hilfreich ist.