

Dimension 730-B (FPS)

IT-Projekt 2022

Jan Grießer, Joshua Holder, Jannik Brack

Fachlehrer:

Manuel Jilg

Markus Gegg

Frederique Chauffer

Freitag, 24. Juni 2022

Inhalt

[Warum ein First Person Shooter? 2](#_Toc106817665)

[Vorstellungen, Ideen und Arbeitsablauf 2](#_Toc106817666)

[Ziel des Spiels 2](#_Toc106817667)

[Gesetzte Meilensteine 2](#_Toc106817668)

[Erklärung des Programms 2](#_Toc106817669)

[Programmiersprache und -umgebung 2](#_Toc106817670)

[Ablauf des Spiels 3](#_Toc106817671)

[Wichtiger Code 5](#_Toc106817672)

[Betrachtung 5](#_Toc106817673)

[Erreichte Meilensteine 5](#_Toc106817674)

[Fazit 6](#_Toc106817675)

# Warum ein First Person Shooter?

Joshua und ich hatten schon vor Bekanntgabe des Projekts die Idee ein Spiel zu entwickeln, was sie auch Anfang dieses Jahres in die Tat umsetzten. Als wir bei der Bekanntgabe des Projekts herausfanden, dass wir in dreier Teams arbeiten konnten, haben wir Jan als drittes Mitglied in unsere Gruppe aufgenommen. Wir entschieden uns für einen First Person Shooter, weil sie heutzutage in der Gamingwelt oft vertreten sind und es anspruchsvoll werden würde einen selber zu Programmieren.

# Vorstellungen, Ideen und Arbeitsablauf

## Ziel des Spiels

Das Ziel unseres Spiels ist es mit der Hilfe von verschiedenen Waffen der drei Waffentypen (primäre Waffen, sekundäre Waffen, Nahkampfwaffen) und erworbenen Skills verschiedene Wellen an Zombies zu besiegen und sich bis zum Boss durchzukämpfen.

## Gesetzte Meilensteine

Bis zum Abgabetermin hatten wir uns vorgenommen diese Ziele zu erreichen:

* Der Spieler kann sich bewegen
* Das Spiel sollte die Bunnyhopp Funktion haben
* Der Spieler sollte mit Items und Waffen interagieren können

(Waffen sollten durch Munition beschränkt sein)

* Der Spieler sollte die Möglichkeit haben Waffen zu benutzen
* Der Spieler sollte verschiedene Skills erwerben können
* Der Spieler sollte die Möglichkeit haben sich zu heilen
* Gegner sollen Bewegungen anhand einer KI ausführen können
* Es sollte einen Bosskampf am Ende des Spiels geben

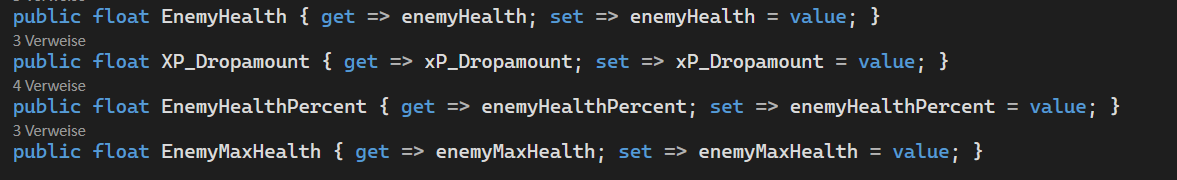
# Erklärung des Programms

## Programmiersprache und -umgebung

Als Programmierumgebung nutzen wir Unity und die dazugehörige Programmiersprache C#. Unity ist eine hilfreiche Programmierumgebung für die Entwicklung von Spielen und wird von vielen verschiedenen Usern und auch einigen kleinen Spiele Entwicklern benutzt. Das liegt daran, dass Unity extra auf das Programmieren von Spielen angepasst wurde. Auch einige Hilfestellungen, wie zum Beispiel das Erzeugung von Objekten, kreieren eines Layouts und vor allem das Testen des Programms werden zur Verfügung gestellt. Zudem ist Unity eine Oberfläche, die Visuelles und Gecodetes leicht miteinander Verknüpft und die Interaktionen untereinander vereinfacht.

Wichtig zu erwähnen ist, dass trotz der Vereinfachung durch ein vorgecodetes KI System, Unity nur die Benutzeroberfläche ist. Der/Die Entwickler muss Codes und 3D Modelle immer noch selbst erstellen.

Trotz der ganzen hilfreichen Tools ist das Erstellen eines Spiels in Unity aber lange noch nicht aufwandslos und mit viel Arbeit verbunden. Durch die uns noch unbekannte Programmiersprache C# gab es Probleme bei Gettern und Settern, da diese anders programmiert werden wie in Java.



*So sehen die Getter und Setter in C# aus*

## Ablauf des Spiels

Bei Spielstart wird der Spieler in der Egoperspektive seines Charakters auf der Welt spawnen. Er kann sich nun mit den Tasten „W A S D“ bewegen und mit der Leertaste springen. Im Spiel kann der Spieler nun oben links seine Lebensleiste, unten rechts die Anzahl an Munition für die aktuelle Waffe und unten in der Mitte seine Waffenslots sehen. Der Spieler Startet anfangs mit 2000 HP, was der dieser dann auch an der Lebensleiste erkennen kann.

Um sich gegen die Zombies zu wehren hat der Spieler entweder die Möglichkeit mit linksklick Anzugreifen. Er kann entweder mit der Faust zu schlagen oder eine der 3 Waffen aus seinen Slots ausrüsten indem er die Tasten 1, 2 und 3 benutzt. Jeder Slot ist für einen bestimmten Waffentypen gemacht (Slot 1 primäre Waffen, Slot 2 sekundäre Waffen, Slot 3 Nahkampfwaffen) um den Spieler in seinen Waffen einzuschränken. Der Spieler beginnt am Anfang mit einem „Dolch der Tausend Monden“ (Nahkampf) um sich gegen die Zombies wehren zu können und dadurch mehr Waffen freizuschalten.

Durch das drücken von „TAB“ kann der Spieler das Inventar öffnen um seine Items zu sehen und sie auch zu benutzen. Im Fenster für das Inventar werden rechts die drei Waffenslots angezeigt. In dem linken Feld werden nun die Items angezeigt, die man im verlauf des Spiels gesammelt hat. Durch einen Linksklick können Items verbraucht werden.

Links unten im Inventar sieht man das aktuelle Level des Spielers. Durch das Töten von Zombies speichert der Spieler Erfahrungspunkte, die er für einen Levelaufstieg benötigt. Um die Erfahrungspunkte jedoch auch zu erhalten muss der Spieler sterben und respawnen. Wenn der Spieler schließlich gestorben ist und eine gewisse Grenze von Erfahrungspunkten erreicht hat, erhöht sich sein Level und damit auch gleichzeitig seine maximalen Leben und der Schaden der Waffen. Damit das Spiel mit der Zeit nicht zu leicht wird, werden die Zombies parallel zum Spieler auch verstärkt.

In der oberen rechten Ecke befinden sich die Anzahl der aktuellen Skillpunkte des Spielers, die er zum Erwerben von Skills nutzen kann. Alle 10 Level bis einschließlich Level 50 bekommt der Spieler einen Skillpunkt, der in einen von fünf beliebigen Skills investiert werden kann.

In der oberen linken Ecke des Inventars hat der Spieler zwei Buttons zu Auswahl, mit denen man durch die beiden Seiten navigiert. Eine Seite für das normale Inventar in dem man sich standartmäßig befindet und eine zweite Seite für die Skills die der Spieler erwerben kann.

Wenn der Spieler ein oder mehr Skillpunkte gesammelt hat, kann er im Inventar zu den Skills wechseln und durch einen Linksklick auf das jeweilige Icon diesen dann auch freischalten. In der Aktuellen Version hat der Spieler die Möglichkeit insgesamt fünf Skills zu erwerben. Diese sind:

*Close Call*

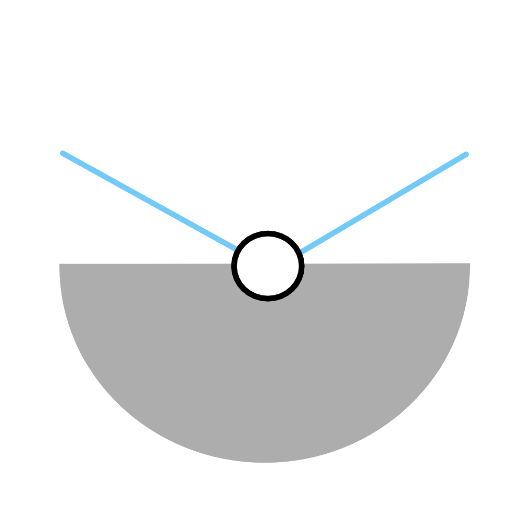
Wenn der Spieler durch schaden auf 0HP fallen sollte, hat er eine 20% Chance anstatt zu sterben, 10% seiner Lebenspunkte zurückzugewinnen.

*Deadly Precision*

Der Spieler hat eine 20% Chance beim angreifen doppelten Schaden zu verursachen.

*Fast Hands*

Die Nachladezeit aller Waffen wird um die Hälfte reduziert.



*Side Step*

Der Spieler kann durch das drücken von „F + A oder S oder D“ einen Dash nach Hinten in den hier gezeigten Bereich ausführen.

Sichtkegel von dem Spieler

S

*Fast Chicken*

Bereich für „Side Step“

Sobald die Leben des Spielers auf oder unter 10% fallen, erhöht sich die Laufgeschwindigkeit um 50%.

Der Spieler hat die Aufgabe die verschiedenen Wellen aus Zombies, von denen er angegriffen wird, zu besiegen um zur nächsten Welle vorzudringen. Wichtig zu erwähnen ist, dass die Wellen nach jedem Mal schwieriger werden indem mehr und auch verschiedene Arten von Zombies mit unterschiedlichen Angriffen gespawnt werden. Um das zu schaffen ist es Hilfreich sich weitere Waffen zu suchen, welche er auf der Welt finden kann. Die Munition für die Waffen wird von Zombies gedroppt und kann durch annähern des Spielers aufgehoben werden. Um das Magazin der Aktuellen Waffe nachzuladen muss der Spieler die Taste „R“ drücken. Zudem können die Zombies auch Items zur Heilung droppen, welche dann im Inventar genutzt werden können. Es gibt ein Item das 25% der Leben und eines, dass 50% der Leben des Spielers wiederauffüllen kann. Diese werden im Verlauf des Spiels noch sehr wichtig sein, da es bis auf „*Close Call*“ keine weitere Möglichkeit gibt sich zu heilen.

Nach einer gewissen Anzahl an Wellen muss sich der Spieler einem Boss stellen. wenn er diesen anschließend besiegt ist das Spiel gewonnen.

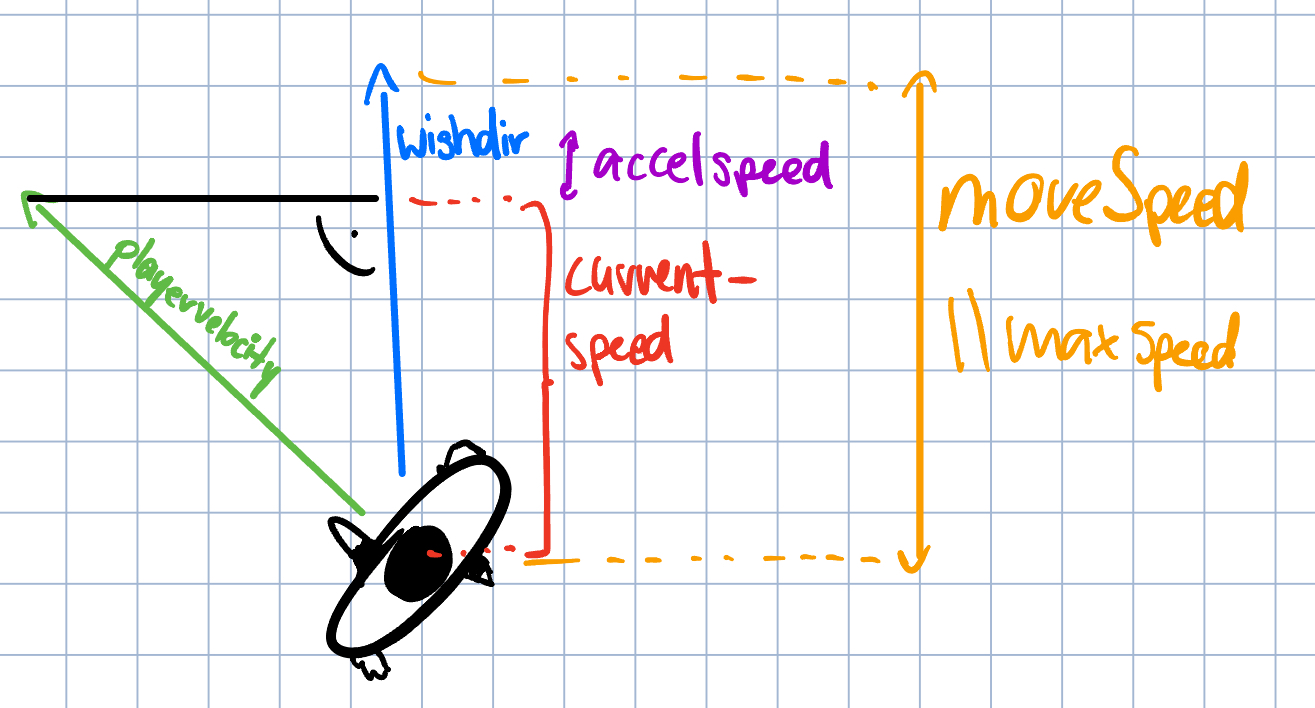
## Wichtiger Code

Der gesamte Code wurde dynamisch gecodet.

Bewegungssteuerung – Unser Movement Script

Eines der wahrscheinlich wichtigsten Stücke Code in unserem FPS. Dieses setzt zum einen unsere Spieler-Kamera an ihre richtige Position, kümmert sich um den Zustand des Mauszeigers z.B. Fest während dem freien Bewegen oder frei, wenn man gerade sein Inventar begutachtet. Anders wie bei den meisten modernen FPS, kann man nicht Sprinten. Wobei man hier annehmen könnte, dass das sehr Lamm sei. Jedoch, da keine maximale Geschwindigkeit unbedingt definitiv festgelegt ist hat man immer irgendwie Luft nach oben. Es ist eher ein Movement – Script, welches auf Momentum (wir arbeiten mit beschleunigten Bewegungen und Reibung) basiert.

*Bewegung auf dem Boden*

Bsp.: 

In diesem Fall werden die Tasten W (nach Vorne) und D (nach Rechts) gedrückt, welche resultierende Richtung durch den „Wishdir“ wiedergegeben wird, dieser aktualisiert die horizontale Bewegung des Spielers. Die vertikale Bewegung des Spielers wird mithilfe eines einfachen Schwerkraft Codestücks berechnet. Die Geschwindigkeit unseres Spielers wird mit der Reibung skaliert und danach mit unserer „Wishdir“ verrechnet, welche davor mit der „accelspeed“ skaliert wurde. Die „accelspeed“ ist abhängig von der momentanen Geschwindigkeit, sowie der gewollten maximalen Geschwindigkeit. Absichtlich in unserem Script, wird jedoch nicht die „playervelocity“ als „currentspeed“ genommen, sondern eher die „skalare Projektion“. Diese Änderung kann ausgenutzt werden um über die „maximale Geschwindikeit“ zu beschleunigen.

Z.B.: durch “zig-zagging”, “circle jumps”, “wall-running” oder “strafe-jumping”

*Bewegung in der Luft*

Bewegung in der Luft wird genauso berechnet wie auf dem Boden, nur das es in der Luft keine Reibung gibt und so die Geschwindigkeit so auch nicht skaliert wird.

Inventar

AI

# Betrachtung

## Erreichte Meilensteine und Probleme

Die genannten Namen entsprechen den Namen der Scripts in Unity

**Der Spieler kann sich Bewegen:**

Das PlayerMovement war sowohl eines der längsten als auch eines der kompliziertesten Teile des Programms, deshalb waren wir auch sehr lange damit beschäftigt und hatten demnach schon vor dem Projekt damit angefangen bzw. es schon davor beendet.

**Das Spiel sollte die Bunnyhop Funktion haben**

Diese Funktion konnte einfach in die Bewegungssteuerung gecodet werden.

**Der Spieler sollte mit Items und Waffen interagieren können**

**(Waffen sollen durch Munition Beschränkt sein)**

Der ItemManager war neben dem PlayerMovement eines der aufwändigsten Scripts. Dieses Item-PickUp-System, besteht aus 3 Klassen

**Der Spieler sollte die Möglichkeit haben Waffen zu benutzen**

**Der Spieler sollte verschiedene Skills erwerben können:**

Als der grundlegende Code schon fertig war wie zum Beispiel die Bewegung des Spielers oder die Lebensanzeige, war das Programmieren der Skills recht simpel. Es gab zwar ab und zu kleine Probleme aber bei keinem davon haben wir lange gebraucht um es zu beheben.

**Der Spieler sollte die Möglichkeit haben sich zu heilen**

**Gegner sollen Bewegungen anhand der KI ausführen können**

**Es sollte einen Bosskampf am Ende des Spiels geben:**

Nachdem ungefähr die Hälfte unserer verfügbaren Zeit vergangen war haben wir feststellen müssen, dass wir nichtmehr genug Zeit hatten um den Boss einzubauen. Stattdessen haben wir uns Aufgaben zugewandt die eine Höhere Priorität hatten.

**Weitere Probleme:**

* Beim Einfügen der Animationen gab es viele Schwierigkeiten, weil

## Fazit