LA1500: Projektdokumentation

Impostors
Jakob
Neiger
Studer
Burgherr

Datum	Version	Änderung	Autor
08.03.2022	0.1.0	Erste Version	Jakob
15.3.2022	0.2.0	Zweite Version	Jakob
22.3.2022	0.3.0	Dritte Version	Jakob
29.03.2022	0.4.0	Vierte Version	Burgherr
05.04.2022	1.0.0	Projektabschluss	Burgherr

1. Informieren

1.1 Ihr Projekt

Wir haben uns für ein 1-gegen-1 Spiel entschieden, da wir alle Interesse an "Fighting Games" haben und wir erhoffen, dass wir etwas auf den Weg dadurch mitnehmen können.

Sussy Fighters

Wir wollen ein Spiel erstellen, in dem zwei Leute am selben PC gegeneinander kämpfen können. ein Spieler steuert mit den Pfeiltasten und der andere mit WASD steuert. Das Spiel soll in der Side-On-Perspektive sein. Die Spielfiguren können verschiedene Schläge und jeder eine Spezialfähigkeit. Die Figuren aus dem Spiel werden bekannte Meme Figuren sein. Das Spiel endet, wenn einer besiegt wird.

Wir erwarten, dass wir erfahren, wie man Menus kreiert und diese Benutzen kann. Wir erwarten auch, dass wir erfahren, wie man sich mit der 2D-Umgebung Charakter erstellen und sich mit diesen bewegen kann. Dazu wäre schön, dass wir lernen, wie man im Spiel schlagen kann, sowie auch Schaden nehmen kann.

1.2 Quellen

Unity Tutorials: Schritt für Schritt Anleitung wenn man ein neues Projekt erstellt.

YouTube Tutorials: (Von Brackeys und Rosdiva)

https://www.youtube.com/watch?v=dwcT-Dch0bA

https://www.youtube.com/watch?v=hkaysu1Z-N8

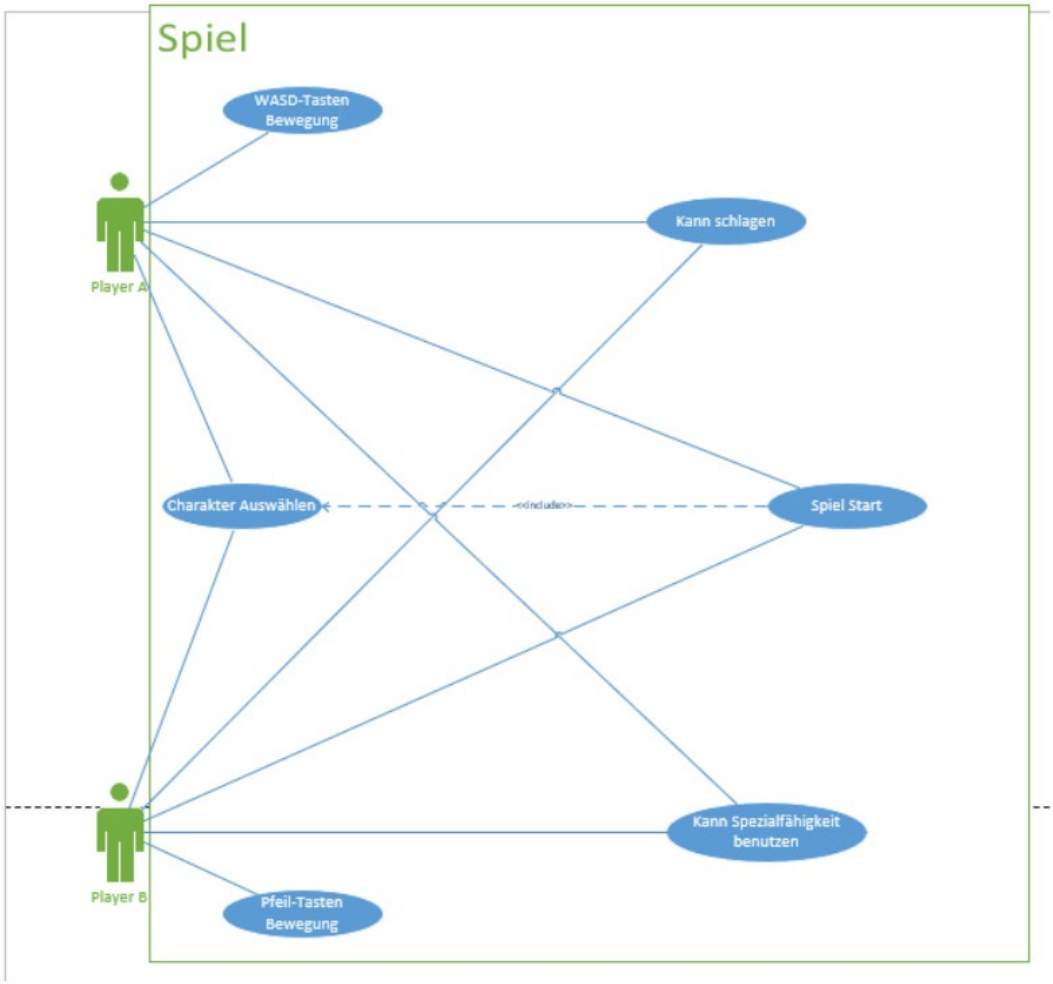
Playlist:

https://youtube.com/playlist?list=PLfxtrTpSBuuCCpWFmv9hMe89wjW1WF1Mb

1.3 Anforderungen

Nummer	Muss / Kann?	Funktional? Qualität? Rand?	Beschreibung
1	Muss	Funktional	Steuerung Charakter A mit WASD
2	Muss	Funktional	Steuerung Charakter B mit Pfeiltasten
3	Muss	Funktional	Meme-Figuren auswählbar als Charakter
4	Muss	Funktional	Man kann schlagen mit beiden Charakteren
5	Muss	Funktional	Man kann Spezialfähigkeiten einsetzen bei beiden Charakteren
6	Muss	Funktional	Spiel in 2D
7	Muss	Rand	In Unity programmieren
8	Kann	Qualität	Charakter im 2D-Design
9	Muss	Funktional	Charakteren sollen hüpfen können
10	Muss	Funktional	Charakteren sollen ducken können
11	Muss	Funktional	Bei K.O. schlag das Spiel Fertig
12	Kann	Qualität	Menu bei Spielstart
13	Kann	Qualität	Alte Kanti Hintergrundbild

1.4 Diagramme



1.5 Testfälle

Nummer	Voraussetzung	Eingabe	Erwartete Ausgabe
1.1	Spiel gestartet, Charakter ausgewählt	W	hüpfen (Charakter A)
1.2	Spiel gestartet, Charakter ausgewählt	A	Nach Links laufen (Charakter A)
1.3	Spiel gestartet, Charakter ausgewählt	S	Ducken (Charakter A)
1.4	Spiel gestartet, Charakter ausgewählt	D	Nach rechts laufen (Charakter A)
2.1	Spiel gestartet, Charakter ausgewählt	Pfeiltaste oben	hüpfen (Charakter B)
2.2	Spiel gestartet, Charakter ausgewählt	Pfeiltaste links	Nach Links laufen (Charakter B)
2.3	Spiel gestartet, Charakter ausgewählt	Pfeiltaste unten	Ducken (Charakter B)
2.4	Spiel gestartet, Charakter ausgewählt	Pfeiltaste rechts	Nach rechts laufen (Charakter B)
3.1	Spiel gestartet		Charakterauswahl
4.1	Runde gestartet	E	Schlagen (Charakter A)
4.2	Runde gestartet	Shift-rechts	Schlagen (Charakter B)
5.1	Runde gestartet	Q	Spezialfähigkeit (Charakter A)
5.2	Runde gestartet	"-"	Spezialfähigkeit (Charakter B)
6.1	Spiel gestartet		2D Ansicht
8.1	Runde gestartet		Charakter in 2D
9.1	Runde gestartet	W	Springen (Char A)
9.2	Runde gestartet	Pfeiltaste oben	Springen (Char B)
10.1	Runde gestartet	S	Ducken (Char A)
10.2	Runde gestartet	Pfeiltaste unten	Ducken (Char B)
11.1	Schlagen (Mehrmals)		KO
11.2	1 Charakter KO geschlagen		Spiel Fertig, Gewinner anzeigen
12.1	Spiel gestartet		Menu sichtbar
13.1	Runde gestartet		Hintergrund sichtbar

2. Planen

Nummer	Frist	Beschreibung	Zeit (geplant)	Zeit (effektiv)
12.1	22.3	Menu bei Spielstart	135	180
8.1	29.3	Charakter Designs (2D) inkl Animationen	180	230
13.1	22.3	Hintergrund sichtbar	45	45
13.2	22.3	Runde starten	45	90
3.1	22.3	Bei Rundenstart Charakterwahl sichtbar (2x)	135	180
3.2	22.3	Kann Charakter auswählen (2x)	45	90
6.1	22.3	2D Ansicht (Side-On-Perspektive) bei Spielstart	45	45
1.1	29.3	Kann Charakter A im Spiel bewegen	90	90
2.1	29.3	Kann Charakter B im Spiel bewegen	45	45
4.1	29.3	Kann schlagen (Char A)	90	45
4.2	29.3	Kann schlagen (Char B)	45	45
5.1	29.3	Spezialfähigkeit benutzbar (Char A)	90	-
5.2	29.3	Spezialfähigkeit benutzbar (Char B)	45	-
9.1	29.3	Charakter kann springen (Char A)	45	45
9.2	29.3	Charakter kann springen (Char B)	45	45
10.1	29.3	Charakter kann ducken (Char A)	45	-
10.2	29.3	Charakter kann ducken (Char B)	45	-
11.1	5.4	KO-Funktion im Spiel	180	90
11.2	5.4	Sieger anzeigen	90	90

3. Entscheiden

Code wird immer hin und her geschickt, Hauptcode besitzt Matteo Jakob und fügt den auch zusammen mithilfe anderen Mitgliedern.

Wir lassen die Ducken und Spezialattacken Funktion weg, da uns die Zeit nicht reicht.

4. Realisieren

Das Projekt befindet sich auf github unter https://github.com/BetterMJ/SussyFighters

5. Kontrollieren

5.1 Testprotokoll

Es gibt keine Fehler, wir haben absichtlich keine Spezialattacken und ducken eingefügt.

Nummer	Datum	Resultat	Durchgeführt
1.1	**05.04.2022**	**OK**	Matteo Jakob
1.2	**05.04.2022**	**OK**	Matteo Jakob
1.3	**05.04.2022**	**NOK**	Matteo Jakob
1.4	**05.04.2022**	**OK**	Matteo Jakob
2.1	**05.04.2022**	**OK**	Amon Burgherr
2.2	**05.04.2022**	**OK**	Amon Burgherr
2.3	**05.04.2022**	**NOK**	Amon Burgherr
2.4	**05.04.2022**	**OK**	Amon Burgherr
3.1	**05.04.2022**	**OK**	Finn Neiger
4.1	**05.04.2022**	**OK**	Finn Neiger
4.2	**05.04.2022**	**OK**	Finn Neiger
5.1	**05.04.2022**	**NOK**	Finn Neiger
5.2	**05.04.2022**	**NOK**	Finn Neiger
6.1	**05.04.2022**	**OK**	Randon Studer
8.1	**05.04.2022**	**OK**	Randon Studer
9.1	**05.04.2022**	**OK**	Randon Studer
9.2	**05.04.2022**	**OK**	Randon Studer
10.1	**05.04.2022**	**NOK**	Randon Studer
10.2	**05.04.2022**	**NOK**	Amon Burgherr
11.1	**05.04.2022**	**OK**	Matteo Jakob
11.2	**05.04.2022**	**OK**	Matteo Jakob

Das Ganze Spiel funktioniert einwandfrei. Wir haben zwei Funktionen weggelassen, was das Spiel aber nicht beeinträchtigt.

Zum Aktuellen Zeitpunkt könnte das Spiel verkauft werden.

6. Auswerten

Wir hatten schnell eine Funktionierende Steuerung, das hat gut funktioniert. Danach hatten wir aber Probleme, da wir die Animation mit der Steuerung verknüpfen mussten und das recht kompliziert war. Dann kam das Problem, dass wir die Charakteren auswählen und dann in die Spielszene übertragen können. Dieses Problem haben wir dann mit verschiedenen Szenen gelöst.