Semesterprojekt zu Interaktion 3

von

Julius Meier

Matrikelnummer: 1522833

E-Mail: julius.meier@stud.hs-hannover.de

Ein 2D Arcade Highscoregame mit dem Namen:

LoFi SunJu

Das Spiel:

LoFi Sunju ist an sich ein simples Spiel, bei dem es darum geht Geschossen die vom Himmel herunter kommen auszuweichen und durch das Überleben Punkte zu sammeln. Das Spiel wird mit zunehmender Zeit (mehr Geschosse, öfterres erscheinen der Boss Geschosse) und mit zunehmender Geschwindigkeit und Lautstärke der Musik (Geschosse werden größer umso lauter und schneller der Beat) schwieriger. Dem Spieler stehen dabei zwei Fähigkeiten zur Verfügung. Zum einen kann der Spieler einen kurzen Dash ausführen, der ihn währenddessen kurz unverwundbar macht (Heist man kann auch durch die Geschosse durch dashen) und zum andern kann man sich für eine kurze Zeit zu einem Glitch werden lassen und die Geschosse fliegen so durch einem durch ohne Schaden zu machen. Beide Fähigkeiten lassen sich auch während der Ausführung der jeweils anderen aktivieren. Wenn der Spieler gestorben ist, wird öffnet sich ein Fenster, in dem man einen Namen Eintragen kann und seinen Highscore so in die Lister einträgt. Die Liste wird lokal auf dem Pc gespeichert und nicht in den Spielfiles. Somit gibt es immer nur einen lokalen Highscore der auch nach dem Neuinstallieren des Spiels nicht verloren gehen würde.

Zeitlupenfähigkeit wurde aus Gameplay Gründen entfernt und abgewandelt zur Pausefunktion bei der das Spiel nicht nur pausiert, sondern erst verlangsamt und bei Wiederaufnahme beschleunigt.

Es gibt ein Hauptmenü mit Animiertem Logo und dem abgebildetem Highscore und ein Level, das sich endlos spielen lässt, bis man kein Lebenspunkte mehr übrighat. Die Sonne, die Sonnenstrahlen, Boss Geschosse, einige Ui Elemente und ein Teil des Spielercharakters reagieren auf die Musik und werden dadurch während des Spielens animiert.

Steuerung:

Bewegen: „A“ und „D“ oder Pfeiltasten „links“ und „rechts“
Dash: „Leertaste“
Glitch: „E“
Pause: „ESC“

Die Story:

Alles am Arsch…. Angefangen hat es mit der Erfindung des Marskatapults. Doge Father Elon Musk der Hund dachte es wäre eine gute Idee nicht nur den Dogecoin too the Moon zu schießen, sondern auch es sei schlau ein Katapult zu entwickeln das Ressourcen und Vorräte gut verpackt mit einem Katapult direkt von der Erde zum Mars zu seiner Marsstation schießen kann. An sich keine gänzlich schlechte Idee, wenn es denn funktioniert hätte…. Zumal die kosten, um Ressourcen zur Station zu befördern mit dem Herkömmlichen weg jeden Rahmen sprengten.

Allerdings entwickelten die Dinge sich etwas anders als erwartet. Da der Flug für abgeschossene Wahren 9 bis 11 Monate dauert konnte man lange Zeit nicht sagen ob die Sachen auch wirklich dort ankamen, wo sie ankommen sollten. Naja, wie soll ich sagen…. sie sind es nicht. Jede Woche beluden sie die 50 Hektar große Ladefläche mit allem Möglichem gut verpackt in einer Großen Hitzeresistenten Kugel und schossen es los…. So ging das 11 Monate lang…. Am Anfang dachten sie nur dass sie sich verrechnet haben und die Zeit bis zur Ankunft der Geschosse länger dauern könnte. Doch falsch gedacht irgendwo auf halben weg hatte sich ein Stern so weit von seiner Ursprünglichen Position bewegt (zumindest behaupten sie das… Ich vermute eher, dass sie die Flugbahn nicht richtig berechnet haben) dass die Geschosse an ihm zerschellten und sich der gut geschützte Inhalt der in Kapseln verpackt war um beim Aufprall auf dem Mars nicht kaputt zu gehen freigelegt wurden und sich nun durch die Gravitation eines naheliegenden Planeten sich auf direktem Weg zurück zu uns befinden.

Jetzt möchte man meinen das diese sowie beim Eintritt in die Atmosphäre verglühen… Naja falsch gedacht… zumindest zum Teil. Denn der Angestellte, der für die Legierung dieser Kapseln verantwortlich war, dachte sich es wäre eine gute Idee den Inhalt, obwohl er durch die Große Kugel darum herum schon vor Hitze Geschütz war auf Nummer sicher zu gehen und selbiges auch für die Kapseln zu tun.

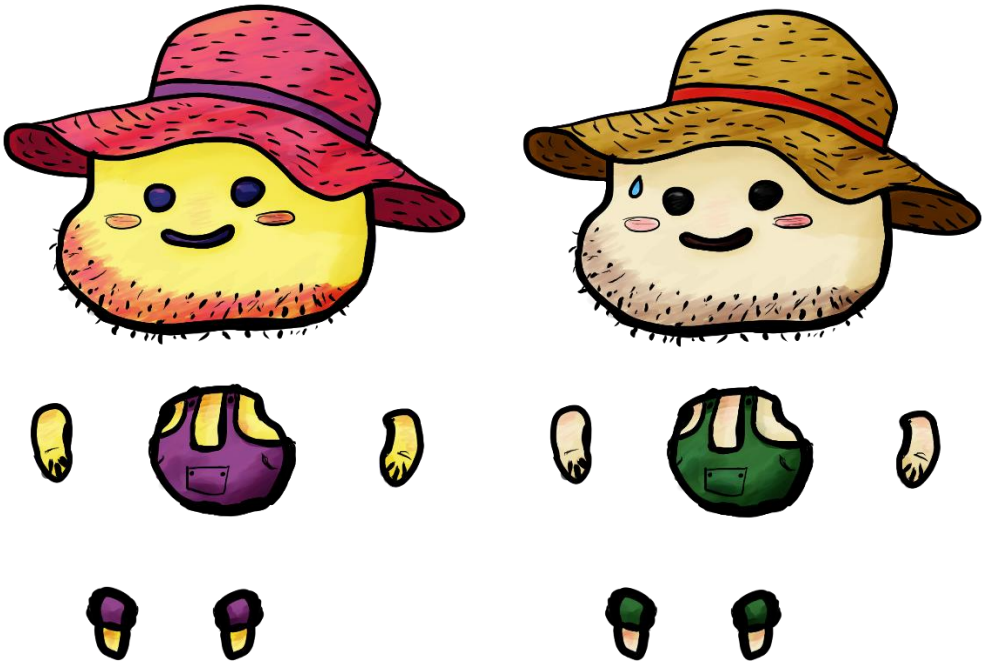
Und genau das wird in 23 Stunden das erste Mal passieren und dann für geschlagene 11 Monate so weiter gehen…. Was für uns Bedeutet, dass wir die nächsten 11 Monate größtenteils in einem Bunker verbringen dürfen, um nicht von Versorgungskapseln getroffen zu werden.

Also bin ich gerade draußen und genieße den letzten Tag in der freien Natur. Doch wie soll ich es am besten sagen…. Sieht es so aus als hätten die Wissenschaftler bei SpaceX sich „wieder einmal“ ein bisschen verrechnet!

(Die Story war so angedacht das man sie vllt. In gekürzter Form als Intro beim Start des Spiels, bevor das Hauptmenü auftaucht, wiedergibt. Im Stil wie es bei Star Wars vor dem Film getan wird nur das ich es gelesen wiedergeben Würde und es skipbar wäre.)

Da dieser Vorspann aber nicht Gameplayrelevant ist, habe ich meine verfügbare Zeit lieber für andere Aspekte des Spiels verwendet.

Der Charakter:



Der Charakter sollte ein bischen an einen Amerikanischen Farmer aus dem Westen errinern, nur eben als stilisierte Comicversion. Er sollte aber auch zu dem restlichen Stil passen also wurde wurde es letztendlich die Farbenfrohere Variante.

Gezeichnet habe ich alles in ClipStudio und Photoshop.

Zusammengesetzt und animiert wurde der Charakter komplett in Unity, auch alle anderen Animation, bis auf der Glytcheffekt fuer die Buttons den ich in Aftereffects erstellt habe, wurden alle mit dem Unity Animator erstellt.

Geschosse und Partikel:

Die ursprünglich Ausgewählten Geschosse waren einmal der Reiskocher und die Kapsel. Später habe ich diese um den Pokeball und den Todesstern als Boss Geschoss erweitert. Mit nur zwei Geschosstypen wurde es gerade im späteren Spielverlauf optisch sehr schnell einseitig und während des Arbeitens an dem Spiel kam die Idee auf auch Boss Geschosse mit einbauen zu wollen.

Spielablauf und Scripts:

Im generellen war der Fokus bei dem Projekt möglichst viel auch für zukünftige Projekte zu lernen.

Automatische Animation durch Sound:

SoundScale:

Ich wollte unbedingt das die Sonne, die im Hintergrund ist, sich nicht nur einfach bewegt, sondern auf die Musik reagiert. Dazu habe ich ein Script erstellt, das die Loudness der Musik abgreift und diese in Scale werte umwandelt die Objekte innerhalb des Spiels nach Festgelegten Zeitabständen verformen.

Die Regelmäßigkeit des Updatens der Größe wird durch den UpdateStep Wert festgelegt.

SoundVerfaerbung und Rotation funktionieren von der Basis her fast gleich.

Die Rotation der Sonnenstrahlen werden mir dem Soundrota Script gesteuert und SoundDiscolor ist dazu da die Transparenz eines Overlays zu steuern das so den Effekt erzeugt das das Spiel immer farbiger wird umso lauter die Musik.

Um die Scripts für alle möglichen Objekte im Spiel nutzbar zu machen kann man den Scalefaktor mit dem auf die Musik reagiert wird, die Startgröße, die Mindestgröße und die maximale Größe in Unity festlegen. So lässt sich ganz einfach bei jedem Objekt während des Testens die Reaktionsmenge und Variablen anpassen.

Schwierigkeitsgrad:

Der Schwierigkeitsgrad wird einmal über das GegnerSpawner Script gesteuert, indem es die Spawnabstände nach jedem gespawnten Objekt um einen Festgelegten wert verkürzt und dann noch ueber die ein SoundScaler Script das die Spawnpunkte je nach Musik grösser und kleiner macht.

Somit wird das Spiel einmal mit zunehmendem Voranschreiten der Spielzeit Schwieriger, zum anderen aber auch durch Schnelligkeit, Bass und Lautstärke der Musik.

Somit lassen sich einfach neue Level erstellen, die sich stärker voneinander differenzieren, indem man einfach den Song ändert.

Lauter Schneller Song = schwieriger

Leiser langsamer Song = leichter

Buttons:

Für die Buttons habe ich ein extra Script erstellt, da ich mit dem in Unity vorhandenen Button Funktion, das Schalten eines aktiven Objekts beim über dem Button hovern und automatischen wieder Abschaltens so nicht realisieren konnte, wie ich es wollte. Letztendlich benutzte ich in diesem Projekt aber nun eine Mischung aus beidem wobei das ButtonScript von Unity lediglich die Verfärbung des Buttons steuert.

Die Buttons, die mit meinem ButtonScript steuerbar sind, lassen sich dann ganz einfach mit einem Script das man in der Scene oder auf dem zu steuerndem Element ablegt definieren. In diesem Fall liegen die Steuerfunktionen immer auf den Ui Scripts da alle von ihnen UI Elemente steuern.

Der Vorteil bei einem erstellten Script ist außerdem, dass sich das Verhalten eines Buttons viel präziser steuern lässt und ich es ehrlich gesagt einfacher fand Ui-Elemente und Szenenwechsel mit meinem Buttonscript und dazu erstellten UI_Scripts zu steuern.

Ich wollte bei diesem Script lernen wie ich die Aktivierung und Deaktivierung von UiOberflächen oder Spielelementen per Script regeln könnte und habe auf diesem Wege viel damit herumprobieren können.

Spieler:

Auf dem Spieler liegt ein Script „Spieler“ das für die Steuerung, Fähigkeiten und Leben des Charakters zuständig ist. Des Weiteren stellt es den erreichten Highscore für das Eingabefeld bereit, wenn der Spieler gestorben ist.

Der Spieler kann den Charakter mit der Standard-Achsensteuerung von Unity bewegen „A“ und „D“ oder Pfeiltaste „links“ oder „rechts“. Der Dash wird mit der Leertaste ausgeführt und der GlitchModus in dem man keinen Schaden nehmen kann wird mit „E“ aktiviert. Das ermöglicht eine einhändige Steuerung auf der Tastatur.

Der Dash funktioniert, indem er bei Aktivierung die Geschwindigkeit des Charakters vorübergehend stark erhöht. Das hatte zur folge das er ursprünglich sehr verbugt war und beim spamen der Taste oder bei zu kurzem drücken Bugs verursachte. Zudem funktionierte der Dash so nur wenn man gleichzeitig im Spiel läuft. Das konnte ich alles beheben, indem ich bei der Aktivierung der Taste diese sofort wieder auf Cool down setzte und einmal die letzte Achseneingabe abrufe und das Ausführen der Bewegung in die letzte Bewegungsrichtung automatisch Ausführen lasse. Das hatte auch den Vorteil das sich das Gameplay stark verbesserte da sich das Dashen nun viel präziser einsetzen lässt. Dann habe ich durch das swappen der Hitbox während des Dashes des Charakters noch hinzugefügt das der Charakter durch Geschosse durchdashen kann ohne Schaden zu nehmen. Dadurch ist das dashen viel variabler einsetzbar und fühlt sich wirklich nach einem präzisen Gameplayelement an.

Die Glytch Fähigkeit wird recht simpel durch das swappen der Hitboxen geregelt.

Damit man auch während des Glytches dashen kann gibt es immer zwei Animationen für das Dashen, einmal mit Glytch, einmal ohne.

Das Leben wird als simple int geführt und bekommt den Schadenswert eintreffender Objekte, die es ansprechen abgezogen.

Hinter dem Spieler liegt eine Art Silhouette die auf die Musik reagiert.

Zeitlupen Fähigkeit siehe unter Abschnitt Pause

Gegner:

Die Geschosse werden mithilfe eines Scripts gespawnt dessen Spawnpunkte über dem Spielbereich sitzen. Einmal für die normalen Geschosse über dem Spielfeld verteilt und einmal für den oder die Bosse. Die Scripts habe ich so angelegt dass man die Geschosse einfach tauschen oder neue hinzufügen könnte. Auf den

Geschossen selber liegt ein Script das die Geschosse steuert. Hier lässt sich in Unity einstellen zwischen in welcher Geschwindigkeit sich das Geschoss bewegt (zufallswert zwischen min. und eingestelltem max.) und wie viel Schaden es macht, wenn es den Spieler trifft.

Auf dem Boss liegt ein angepasster Soundscaler der sich automatisch das Objekt, auf dem er liegt, holt, um es zu beeinflussen. Aus Perfomance Gründen konnte ich nicht wie angedacht alle Geschosse auf die Musik reagieren lassen und das tut nun nur noch der Boss.

An einer Lösung dafür arbeite ich gerade, bin aber noch zu keinem gut funktionierenden Ergebnis in dieser Hinsicht gekommen, weswegen ich die Version des Spiels hochlade in dem ich die entsprechenden Versuche noch nicht eingefügt habe.

Pause:

Ursprünglich hatte ich eine Zeitlupen Fähigkeit vorgesehen, doch diese erwiess sich beim Anspielen als schlechte Idee. Problem 1 war das sie von den Vorteilen in brenzlichen Situationen dem Glytch sehr ähnlich war und deswegen kaum genutzt wurde, weil die Glytch Fähigkeit einfach einen tikken stärker ist. Problem 2 war sie sich einfach nicht gut anfühlte. Problem 3: Die Fähigkeit als aktivierbare stellte sich einfach für diese Art Spiel nicht passend heraus da sie das Spielgeschehen kurz verlangsamte und den Spielfluss zu stark stöberte. (Die Testenden starben meist kurz nach verwenden und ärgerten sich nicht den Glytch benutzt zu haben)

Da ich den Slowdown und das Anlaufen der Zeit mit Tonsteuerung an sich aber sehr passend für das Spiel empfand fand dies einen neuen Platz

Mit ESC lässt sich das Spiel pausieren und wird darauf hin erst verlangsamt, bis es zum Stillstand kommt. Danach lässt es sich mit ESC wieder Fortsätzen und die Zeit lauft ansteigend wieder an, bis sie wieder normal läuft.

Eingabefenster und Highscore:

Der Highscore wird als JSON lokal auf dem Pc gespeichert auf dem das Spiel abgespielt wird.

Vereinfacht gesagt rufe ich mit dem HighscoreTabelle Script einmal die Grafiken aus dem Spiel ab die Ich für den Highscore erstellt habe und füge in die entsprechenden Textfelder einmal Platz, Name, Score ein.

Beim ersten Mal starten des Spiels wird eine Tabelle mit im Script festgelegten Einträgen erstellt und gespeichert.

Immer wenn die Tabelle über das Script geladen wird werden die Scores nach der Platzierung geordnet und mit dem Platz betitelt den sie erreicht haben indem jeder Eintrag beim laden solange mit dem Eintrag über ihm oder unter ihm verglichen und getauscht wird bis alle an der richtigen Stelle stehen.

Bei dem Neu eintragen eines Scores wird das Highscore JSON einmal geladen und der neue Eintrag hinzugefügt und der letzte gemachte Eintrag wird farbig hinterlegt. Das erleichtert bei zunehmender Eintragszahl das Finden der gerade erreichten Platzierung. Dann wir das JSON wieder gespeichert.

Beim neu Laden im Hauptmenü wird immer der Erste Platz farbig hinterlegt. Ähnlich würden sich auch Pokale oder Medaillen auf den oberen Plätzen hinzufügen lassen.

Ausblenden der Maus:

Die Maus wird im Level, durch ein Script, bei der Nutzung der Tastatur ausgeblendet und sobald man die Maus bewegt, wieder eingeblendet.

Fazit:

Das Projekt hat mir viel Spaß gemacht und mir neue Ideen gegeben wie sich Funktionen in Unity umsetzten lassen. Einige Probleme, die während des Projekts auftraten, haben mir auf ihrem Lösungsweg neue Möglichkeiten aufgezeigt, die ich in zukünftigen Projekten nutzen kann.

Obwohl ich mir alle Mühe gegeben habe die innerhalb der Scripts benutzten Bezeichnungen so zu wählen, dass sie mir ermöglichen im Nachhinein schnell wieder ihre Funktionsweise nachvollziehen zu können, habe ich bei einigen Scripts Bezeichnungen gewählt die zum Teil nicht 100% zutreffend für das Stehen was an entsprechender Stelle definiert wird was das Spätere nachbearbeiten und anpassen etwas Zeitaufwendiger macht.

Zu einem Teil entstand dies aber auch durch den anfänglichen Anspruch ausschließlich deutsche Bezeichnungen zu verwenden, obwohl mir die Englische Sprache gerade im Videogamebereich viel mehr liegt um Funktionen oder abschnitte zu definieren. Manche Funktionsweisen lassen sich im deutschen nur sehr schwer kurz und gleichzeitig treffend definieren.