HOCHSCHULE ANSBACH

Fakultät Medien

Visualisierung und Interaktion in digitalen Medien (B. A.)

WPM – Creative Coding

Vorgelegt von:

Sebastian Loch – 00166778

Julia Sauter – 00165587

Fachsemester: 6.Semester

Sommersemester 2025

Creative Coding

Projekt Inhalt

Das Projekt ist ein Spiel, in welchem man als Biene Blütenstaub sammeln muss und diesen zum Nest zurückbringt. Der Spieler hat nur begrenzte Ausdauer, welche bei Bewegung reduziert wird, zusätzlich gibt es ein Zeitlimit, in welchem das Ziel von 100 Punkten erreicht werden muss. Im Level sind Gegner und Fallen verteilt, die das Erreichen des Ziels zusätzlich erschweren sollen. Es gibt ein Einstellungsmenü, mit Settings für Lautstärke und der Möglichkeit die Flugkontrollen zu invertieren und dynamische Events während des Spiels.

Gameplay

Die Spieler Steuerung soll an die Steuerung eines Flugspiels erinnern, *W* und *S* bewegen den Player vorwärts und rückwärts, *Space* und *Strg* hoch und runter. Mit Bewegung der Maus sieht man sich um und fliegt in Sichtrichtung, da sich der Player dadurch stets leicht um die eigene Achse dreht, kann man diese Rotation mit *A* und *D* ausgleichen und eine BARRELROLL! vollführen.

Bewegung kostet den Spieler Ausdauer, angezeigt durch einen sich leerenden Balken in der oberen linken Ecke, welche bei Rückkehr zum Nest wieder aufgefüllt wird. So wird der Spieler gezwungen regelmäßig zum Nest zurückzukehren und seine gesammelten Punkte dort abzuliefern. Bei 55 erreichten Punkten wird die Fähigkeit freigeschaltet einen Speed Boost mit *LShift* zu aktivieren, dabei wird die Fluggeschwindigkeit erhöht. Der Boost wird ebenfalls bei Rückkehr zum Nest erneut aufgeladen und kann insgesamt für 10 Sekunden verwendet werden.

Hauptziel ist, die pink farbigen Blumen anzufliegen und deren Blütenstaub bzw. deren Punkte zu sammeln, pro angeflogene Blume erhält der Spieler 5 Punkte, die man zunächst bei sich trägt. Bei Rückkehr zum Nest werden diese Punkte zur Gesamtpunktzahl addiert und sind sicher. Blumen, die bereits angeflogen wurden, färben sich weiß, um zu zeigen, dass ihre Punkte bereits gesammelt wurden. Insgesamt können bis zu 130 Punkte erzielt werden, wobei nur 100 Gesamtpunkte nötig sind, um das Spiel erfolgreich zu beenden.

Auf der Karte sind Gegner und Windböen verteilt, um dem Spieler zusätzlich zum Ausdauer- und Zeitmanagement eine Herausforderung zu liefern. Gegner verfolgen den Spieler, sollte dieser zu nahekommen und greifen bei Kollision an. Wird der Spieler von einem Gegner getroffen, verliert dieser eines von drei Leben und 10 der getragenen Punkte. Die Windböen sind Bereiche, in welchen der Spieler vom Wind zur Seite gedrückt wird, falls man währenddessen mit einem Hindernis kollidiert, wird der Player für 6 Sekunden verlangsamt. Das kostet Zeit und Ausdauer, außerdem vereinfacht es den Gegnern, den Spieler zu treffen.

Der Spieler kann unter drei Umständen Game Over gehen, zum einen, wenn die Zeit ausläuft, man keine Ausdauer mehr hat oder zu oft von Gegnern getroffen wurde. In diesem Fall hat man die Option das Spiel sofort neu zu starten oder zurück ins Menü zu gehen.

Umsetzung

Zu Projektbeginn haben wir einzeln an unseren Aufgaben gearbeitet und entsprechend unserer Aufteilung die Grundlagen für unsere Arbeitsteile aufgebaut. Zum Ende hin haben wir jedoch vermehrt zusammen an einem Gerät gearbeitet, besonders als es um das Verfeinern der Spielmechaniken ging. Die ersten Schritte waren das Unity Projekt anzulegen und Grundlagen wie Spielersteuerung, Level Greyboxing und Assets, wie den Player, Gegner und Blumen, zu erstellen. Danach wurden weitere Kernmechaniken wie Gegnerverhalten, Ausdauer- und Punktesystem hinzugefügt. Als diese Aspekte bereit waren liefen Leveldesign und Gestaltung der UI Elemente parallel zueinander. Schließlich

haben wir das Spiel nach und nach zusammengesetzt mit Punktesystem, Game Over Konditionen, Wind und HUD. Ab dem Punkt war das Spiel schon bereit sehr oberflächliche Playtests durchzuführen, auf Basis dieser wurden immer wieder Dinge angepasst, unter anderem der Verlauf des Levels und Balancing. Als das Gameplay insgesamt passte, haben wir uns um zusätzlich Szenen wie Hauptmenü, Intro und Ende gekümmert und diese mit neuen UI Elementen ausgestattet. Ein großer Teil der Zeit zum Ende bestand daraus, das UI interagierbar zu machen und die Möglichkeit Spielübergreifende Einstellungen zum Kontrollieren der Lautstärke und der Invertierung der Maus zu implementieren. Die letzten großen Arbeitsschritte waren die Einpflegung von Sound für alle wichtigen Ereignisse und Erstellen der Hauptgrafiken für Menü, Intro und Ende.

Während der Umsetzung sind immer wieder Schwierigkeiten angefallen, vor allem Scripts/Mechaniken, die plötzlich nicht mehr funktionierten, gerade das Gegnerverhalten hat eine lange Zeit nicht funktioniert. Eine große Hürde war auch das Einpflegen der Blender Assets, da diese stets falsch rotiert waren, sobald sie in Unity eingefügt wurden und mussten teilweise vier Mal neu exportiert werden, bevor die Rotation stimmte. Ein Problem welches bis jetzt besteht, da die Models für die Gegner eigentlich falsch rotiert waren, auf Grund von Zeitgründen aber die Notlösung gewählt wurde die Gegner unter ein Empty GameObject zu gruppieren, welches stattdessen mit Script und Collidern versehen wurde. Ein weiteres ungünstiges Problem ist der verwirrende Levelaufbau, beim Playtesten wussten Spieler häufig nicht wohin sie als nächstes mussten, dies liegt unter anderem an dem Größenunterschied zwischen Player und Environment. Erschwerend kommt hinzu, dass das Spiel in unregelmäßigen Fällen beim Umsehen stottert und der Spieler so die Orientierung verliert. Problem dafür könnte einerseits die aufwändige Umgebung sein, andererseits die Art wie der Player Controller aufgebaut ist.

Arbeitsteil – Seb

Ich habe mich in erster Linie um die technischen Aspekte des Spiels gekümmert und einen großen Teil der Engineering Aufgaben übernommen.

Arbeitsanteil:

- Spielersteuerung
- Gegnerverhalten
- Leveldesign & Environment
- Sound
- UI-Einbindung
- Menüs & Settings
- Animations

Anmerkung – Player Controller: Der Player Controller ist ein zweiteiliges Script. Das Hauptscript auf dem Player-GameObject besitzt sämtliche Funktionen und ist für Positionsänderung zuständig. Der zweite Teil des Controllers ist ein Script, welches die Rotation eines leeren GameObjects anhand der Mausbewegung setzt und an das Player-Object weiterleitet und dessen Rotation überschreibt. Gleichzeitig wird die Position des Players and das Empty übergeben, wodurch sich beide Objekte stets an der gleichen Stelle mit gleicher Rotation befinden. Dieser Informationsaustausch findet in der LateUpdate Methode statt. Die Kamera ist an das leere GameObject gebunden. Ehrlicherweise weiß ich selbst nicht mehr, warum ich mich entschieden habe den Controller so aufzubauen und nicht den Player direkt zu rotieren, im Nachhinein wirkt es dumm und umständlich.

Arbeitsteil - Jojo

Ich habe mich größtenteils um den kreativen Teil des Projekts gekümmert – die Intro-, Outro- und Main Menu-Bilder, sämtliche Sprites und die Gestaltung des UIs – und das Scripten des Punktesystems,

der Wind-Mechanic und der Shortcut-Mechanic. Außerdem übernahm ich das Platzieren von Collidern und das Modelieren der Blumen und des Bienenstocks.

- Sprites
- Menu-Design
- Intro/Outro-Bilder
- Punktesystem
- Windmechanic
- Shortcut-Mechanic
- 3D Asset Creation

Credits & Externe Assets

Environment: Synty Studios Nature Packs

Sound: Pixabay & freesound.org

Code und Engineering wurden mit Hilfe der Unity Scripting API und dem Buch *Spieleentwicklung mit Unity* (ISBN: 978-3-8362-9539-0) erarbeitet.