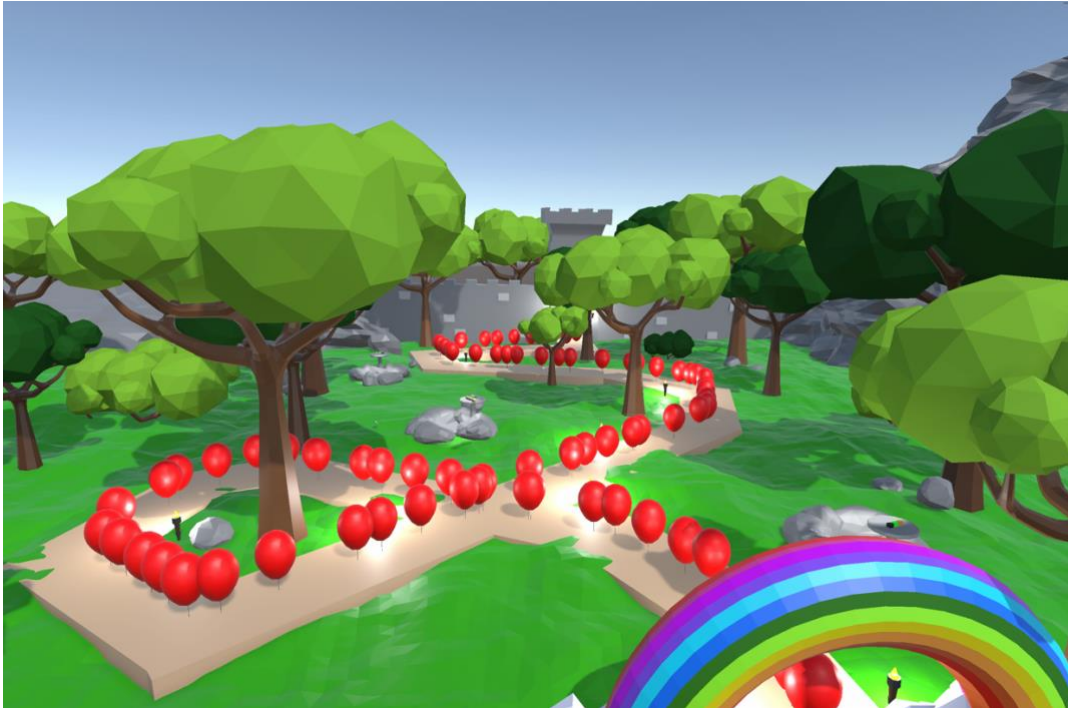


Virtual and Augmented Reality

Projekt 4



Hochschule Aalen

Projektarbeit, Gruppe 2

Kristina Buzko (67670)

Eric Flottmann (78198)

Alexander Hinz (85829)

Tamara Lauser (86633)

Inhaltsverzeichnis

1.	Einleitung.....	3
1.1	Aufgabenstellung.....	3
1.2	Idee und Titel	3
1.3	Unsere Projektziele	4
1.4	Aufteilung der Aufgabenbereiche	5
1.5	Verwendete Assets und Fremdmaterialien.....	5
1.6	Verwendete Tools	5
2.	Skripts	7
3.	Spielanleitung	14
3.1	Steuerung.....	14
3.2	Starten des Spiels	14
3.3	Spielablauf.....	15
3.4	Ziel und Ende des Spiels	17
4	Video und Builds.....	18
5.	Fazit.....	18

1. Einleitung

1.1 Aufgabenstellung

In Gruppenarbeit soll eine VR-Anwendung mit der Unity-Engine erstellt werden. Alle Assets sind erlaubt. Es muss dokumentiert werden, welche Assets benutzt werden.

Es soll mindestens eine Spielmechanik implementiert werden, die Feedback in Form von UI-Elementen (Punktestand, Treffermarkierung, Controller-Vibration, ...) bietet.

Es muss ein grundlegendes Menü gestaltet werden, das mindestens das Verlassen der Anwendung ermöglicht.

Bei der Gestaltung der Anwendung dürfen Objekte aus dem Blender-Projekt einbezogen werden. Weitere Objekte müssen nicht zwangsläufig im höchsten Detaillierungsgrad vorliegen; es reicht, wenn die Form/ Funktion erkennbar ist.

Das Thema der Anwendung kann frei gewählt werden, muss aber mit dem Dozenten abgesprochen werden.

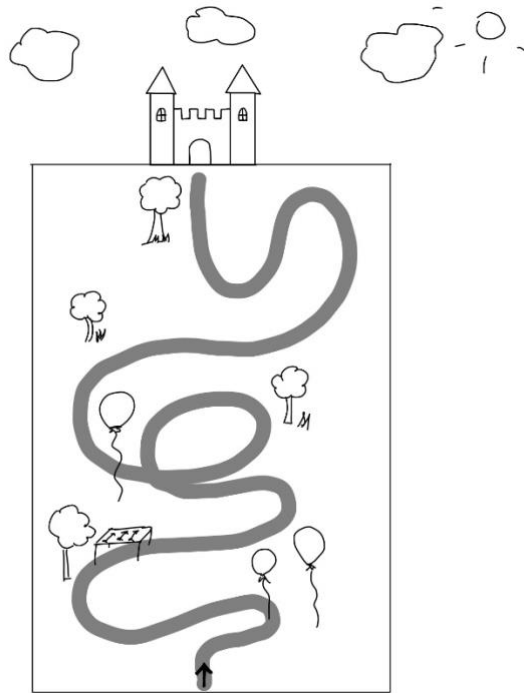
Die Abgabe erfolgt über Git und besteht aus:

- Kurze „Spielanleitung“
- Dokumentation
- Unity-Projekt mit allen Dateien
- Kurzes Demo-Video ihres Spiels (z.B. mit OBS aufgenommen)
- Im Labor lauffähiger Windows/Android-Build der Anwendung

1.2 Idee und Titel

Die Idee von unserem Spiel ist, mit einer Waffe, Darts auf Ballons (hier Bloons) zu schießen, um diese zum Platzen zu bringen, bevor sie die eigene Burg erreichen.

Dieses Spiel ist inspiriert vom bereits bestehenden Spiel „Bloons Tower Defence“. Den Titel des Spiels haben wir übernommen- „Bloons“.



Abweichend vom Originalspiel, wird in unserem Spiel der Dart selbst geschossen. Die Bloons haben verschiedene Stufen (hier Layer) und müssen Schicht für Schicht zum Platzen gebracht werden. Beim Zerstören von Bloons verdient sich der Spieler die Ingame Währung Dollar und mit dieser können Upgrades freigeschaltet werden, wie z.B. verbesserte Darts (mehr Schaden) und einer zweiten Waffe. Mit diesen Upgrades können die Schichten der Bloons schneller zerstört werden.

Das Spiel hat kein fest definiertes Gewinnziel; stattdessen ist es endlos und steigert kontinuierlich seine Schwierigkeit. Die Herausforderung besteht darin, den Punktestand so weit wie möglich zu erhöhen.

1.3 Unsere Projektziele

Neben der Aufgabenstellung haben wir uns verschiedene Schwerpunkte gesetzt, auf die wir bei der Umsetzung besonderen Weg legen wollten:

- Modellierung aller im Spiel integrierten Assets
- Einhaltung des Low Poly-Stils
- Umsetzung eines guten VR-Erlebnisses
- Gutes Spielerlebnis

1.4 Aufteilung der Aufgabenbereiche

VR-Movement	Alexander Hinz
VR-Interaktion	Alexander Hinz
Gamelogic	Eric Flottmann Alexander Hinz
Erstellen von Assents in Blender	Tamara Lauser Kristina Buzko Alexander Hinz
Mapbuilding	Tamara Lauser Kristina Buzko Alexander Hinz
Integration von Map und Logic	Alexander Hinz

1.5 Verwendete Assets und Fremdmaterialien

Wir haben keine Assets und Fremdmaterialien benutzt und alle Grafiken und Skripts selbst geschrieben. (Mit Ausnahme des XR Plugins von Unity und dem Standard TextMeshPro Plugin von Unity selbst, allerdings gehören diese zu Unity).

1.6 Verwendete Tools

Hardware	Index Valve VR-Brille Eigene PCs
Software	Unity C#

Git (Host: GitHub)

Notion

Discord

Obsidian

Blender

2. Skripts

Bloon Skript

- BloonSkript.cs

 BloonSkript.cs

Autor: Alexander Hinz

Dieses Skript verwaltet die Layer der Bloons und gibt die Möglichkeit einen Hit einzutragen (`hitThisBloon(int)`). Diese Funktion ruft `updateLayer()` auf, welche die Layer visuell aktualisiert.

Button Skripts


- BuyNextCoolDown.cs

 BuyNextCoolDown.cs

Autor: Alexander Hinz

Diese Klasse managend ein Upgrade Cube, welcher die die Angriffsgeschwindigkeit erhöht. Bei Aktivierung von `ActivateButton()`, was durch auswählen mit dem Controller passiert, wird überprüft ob es noch ein weiteres Level gibt und ob genug Geld vorhanden ist. Wenn beides zutrifft, dann wird das Geld abgezogen und ein Update an den Upgrade Manager geschickt, um die Anzeigen aller Upgrade Objekte zu aktualisieren

- BuyNextDart.cs

 BuyNextDart.cs

Autor: Alexander Hinz

Diese Klasse verwaltet wie *BuyNextCoolDown.cs* auch ein Upgrade und funktioniert im Prinzip gleich, nur das sie einen besseren Dart freischaltet.

- BuySecondWapon.cs

 BuySecondWapon.cs

Autor: Alexander Hinz

Diese Klasse verwaltet wie *BuyNextCoolDown.cs* auch ein Upgrade und funktioniert gleich, nur das sie eine weitere Waffe freischaltet.

- GroupButtons.cs

 GroupButtons.cs

Autor: Alexander Hinz

Dieses Skript ist nur da, um der Gruppe von Buttons den Zugriff auf den UpgradeManager zu gewähren

- StartButton.cs

 StartButton.cs

Autor: Alexander Hinz

Dieses Skript hat nur die Funktion `activateStartButton()` welche vom Controller ausgelöst wird und die Spielszene (Szene 1) lädt.

Dart Skripts

- Dart.cs

 dart.cs

Autor: Alexander Hinz

Hier werden die Daten von Darts wie Haltbarkeit und Kraft gespeichert.

Zusätzlich wird hier ein Trigger verwaltet, welches checkt ob ein Bloon getroffen wurde und dann `hitThisBloon(int)` von dem Bloon aufruft.

- HitEnvironment.cs

 HitEnvironment.cs

Autor: Alexander Hinz

Dieses Skript prüft ob Darts unter die y: -15 Grenze kommen und löscht es dann. (Ursprünglich mit Collision mit einem Environment, aber diese Version ist performanter.)

- LookInMovementDirection.cs

 LookInMovementDirection.cs

Autor: Alexander Hinz

Dreht ein Objekt in die Richtung, in die es sich bewegt. Sorgt dafür, dass die Darts gerade Fliegen.

Manager Skript

- UpgradeManager.cs

 UpgradeManager.cs

Autor: Alexander Hinz

Diese Klasse verwaltet alle Upgrade Module, welche sich bei diesem selbst registrieren. Die UpdateBuyStuff() ruft die Updatefunktion aller UpgradeSkripts auf. Dies ist eine Optimierung um nicht jeden Frame diese Updates durchführen zu müssen, sondern nur wenn sich was ändert.

- BloonsGoal_BackUp.cs

 BloonsGoal_BackUp.cs

Autor: Alexander Hinz

Diese Klasse verwaltet das Event, wenn ein Bloon das Ende erreicht, updatet die HP und beendet das Spiel. Die Backup Benennung ist entstanden, da diese Klassen ursprünglich, als Backup erstellt wurden, da diese ursprünglich von einem anderem Gruppenmitglied geplant waren.

- `GameData.cs`

 `GameData.cs`

Autor: Alexander Hinz

Diese Klasse hat nur Statische Elemente und ist für die Speicherung aller Statistiken wie Geld, HP und Score verantwortlich. Da dies Statisch ist, kann jedes Element und jede andere Klasse jederzeit diese Statistiken lesen.

- `GameManager_BackUp.cs`

 `GameManager_BackUp.cs`

Autor: Alexander Hinz

Diese Klasse speichert alle Wichtigen Einstellungen des Aktuellen Spiels wie z.b. CoolDown / StartHP / Costen / Upgrades etc. Zudem hat diese Klasse eine `updateScore(int,int)` und eine `updateMoney(int)` Funktion, welche immer zum ändern des Scores und des Geldes verwendet wird. Dies Funktionen updaten auch zugehörige Displays.

- `SpawnManager.cs`

 `SpawnManager.cs`

Autor: Alexander Hinz

Diese Klasse speichert aktuell nur die Verschiedenen Materialien für die Bloon Schichten.

- `StatDisplayManager.cs`

 `StatDisplayManager.cs`

Autor: Alexander Hinz

Diese Skript verwaltet alle Displays. Diese Tragen sich von selbst hier ein. Wenn sich etwas ändert ruft dieser Manager alle Update Methoden er einzelnen Anzeigen auf. Dies ist aus Optimierungsgründen so umgesetzt, damit nicht jedes Display, jeden Frame, aktualisiert.

Dark Skript

- GoDark.cs

 GoDark.cs

Autor: Alexander Hinz

Diese Klasse ändert die Tageszeit zu Abend ab einem Bestimmtem Score.

Path Skripts

- PathHolder.cs

 PathHolder.cs

Autor: Alexander Hinz

Ein Skript, was auf ein Gameobjekt gelegt werden kann um es zu einem Path zu machen. Hierfür müssen die Eckpunkte nur als Gameobjekte in das Array gelegt werden und jedes Objekt mit dem FollowThePath.cs Skript wird diesen Punkten Folgen.

- FollowThePath.cs

 FollowThePath.cs

Autor: Alexander Hinz

Dieses Skript lässt Objekte eine Path folgen.

Spawner Skript

- SpawnBloons.cs

 SpawnBloons.cs

Autor: Alexander Hinz

Diese Klasse ist für das Spawnen der Bloons verantwortlich. Dies ist extrem fein einstellbar. Einstellbar ist:


- Wann dieser Spawner Startet
- Mit Welchem Layer er Bloons spawnt
- Mit welchen Abständen er diese spawnt

- Ab welchem Score er stärker wird
- Wie viele größer der Abstand zu nächsten Score Stufe ist
- Wie viel Layer für die nächste Stärke dazu kommen
- Wie viel schnelle die Bloons Spawnen bei der nächsten Stärke

Es können beliebig viele Spawner gleichzeitig mit verschiedenen settings existieren.

Display Skripts

- MenuSkript.cs

 MenuSkript.cs

Autor: Alexander Hinz

Diese Klasse ist für das Anzeigen des Scores und des Highscores verantwortlich.

- StatDisplay.cs

 StatDisplay.cs

Autor: Alexander Hinz

Diese Klasse verwaltet ein "Display", welches folgende Daten zeigt:

- HP
- Score
- \$
- Letzte HP Änderung
- Letzte Score Änderung
- Letzte \$ Änderung

Die `UpdateStats(bool)` Methode aktualisiert die Anzeige und wird von außerhalb aufgerufen, wenn sich etwas ändert.

Waffen Skript

- Fire.cs



Autor: Alexander Hinz

Dieses Skript Verwaltet den Cool down für das schießen und beinhaltet die `fireDart()` Methode, welche eine Dart feuert und von einem Controller, welcher die Waffe in der Hand hat, durch den Trigger aufgerufen.

3. Spielanleitung

3.1 Steuerung

Teleportieren: Joystick nach oben

- Das Bewegen des Joysticks nach oben startet einen Bogen, welcher auf das Ziel zeigt
- Beim Zielen auf einen Ankerpunkt und gleichzeitig das Loslassen des Joysticks wirst du zum Ziel teleportiert (Ankerpunkte sind durch die Steininformation + Tisch erkennbar)

Drehen: Joystick nach Links/Rechts

Greifen: Grip Key

- Zeige auf ein Objekt und drücke den Grip key
- Das Greifen von Objekten kann theoretisch von Beliebiger Entfernung geschehen

Aktivieren/Auswählen: Greifen + Trigger

- Das Aktivieren oder Auswählen kann nur mit bereits gegriffenen Objekten durchgeführt werden
- Wenn das Objekt gegriffen wurde, kann der Trigger gedrückt werden um seine Funktion auszulösen (z.B. Schießen der Waffe, oder das Upgraden)

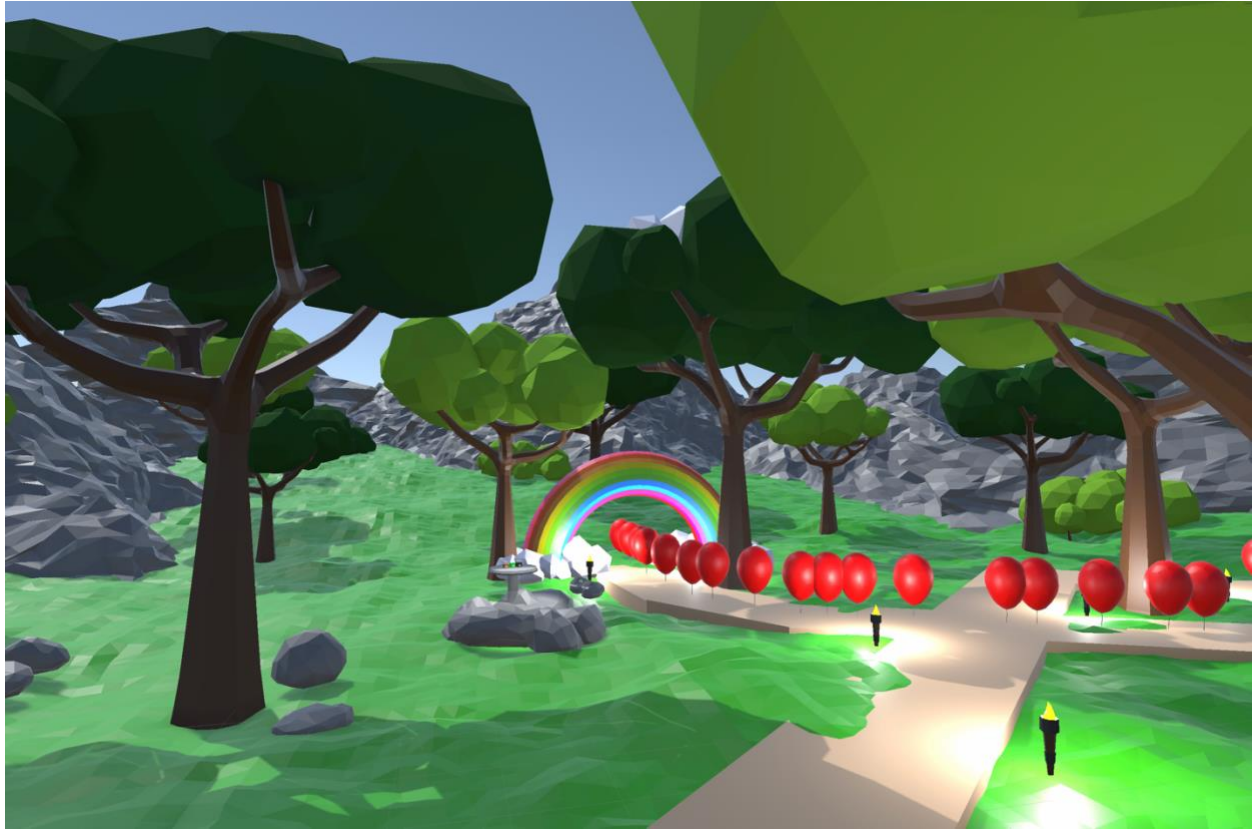
3.2 Starten des Spiels

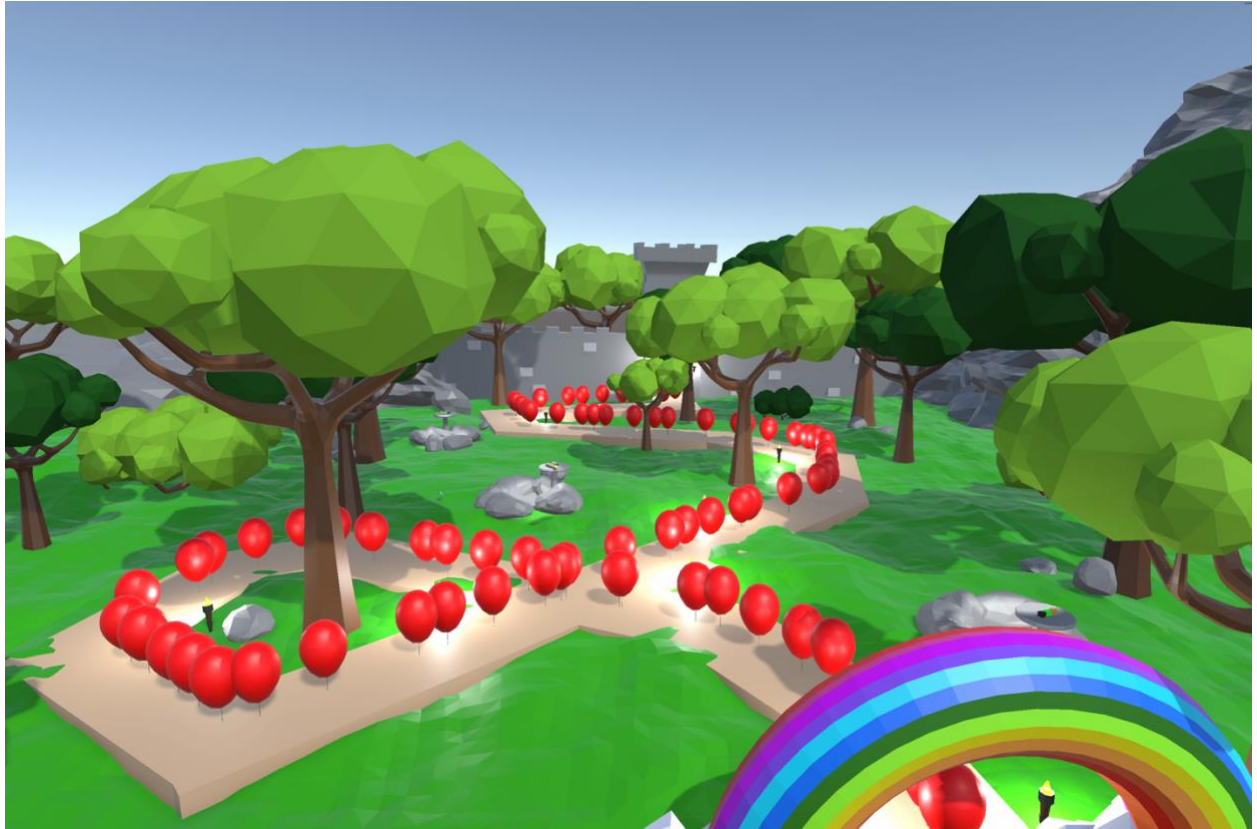
Beim starten des Spiels gelangt man automatisch in die Menu Szene. Dort steht man direkt vor einem riesigen Regenbogen und der Burg. Generell sind alle Elemente, die dort zusehen sind auch im Spiel enthalten, bis auf die Ballons. Um das Spiel nun zu starten, muss man die graue, ovale Tafel mit dem Play Symbol auswählen. Von dort gelangt man in das eigentliche Spiel. Dies ist zusätzlich der Ort, an den man gelangt, wenn man keine Leben mehr hat und somit verliert. Neben dem Play Symbol ist zusätzlich der zuletzt erzielte Score zusehen und der aktuelle Highscore



3.3 Spielablauf

In der Hauptszene angekommen wird man direkt von viel Natur begrüßt. Die Map besteht aus einer grünen Grundfläche, vielen Bäumen, Steinen und Büschen. An dem einen Ende der Map ist ein Regenbogen zusehen und am anderen eine Burg, diese beiden Elemente sind durch einen Pfad verbunden. Aus genanntem Regenbogen erscheinen balloons (bloons) die den Pfad entlang zur Burg gelangen wollen. Bei Spielstart wird man vor einen Tisch gespawnt, auf dem die erste Waffe liegt. Der Sinn des Spiels ist es, sich zum Tisch zu teleportieren, die Waffe zugreifen und auf die Bloons zu schießen, sodass sie nicht die Burg erreichen. Direkt vor einem befindet sich ein schwebender Stein, der den aktuellen Score zeigt, die übrigen Leben und das erzielte Geld. Denn das treffen der Bloons wird mit Münzen belohnt, damit kann man sich im Laufe des Spiels dann eine neue Waffe und bessere Darts kaufen. Dadurch werden mehr Schichten des Ballons zerstört und somit mehr Geld erzielt. Zusätzlich zu den Münzen gibt es natürlich auch Score Punkte.





3.4 Ziel und Ende des Spiels

Das Ziel des Spiels ist es einen höchstmöglichen Score zu erreichen.

Der Score sowie der Highscore der aktuellen Session wird im Menü, nachdem das Game vorbei ist angezeigt. Die Bloons kommen unendlich lange, bekommen immer mehr Layer und kommen in immer höhere Anzahl. Je mehr Layer ein Bloon hat, desto mehr Schüsse braucht man um ihn zu zerstören, oder eine bessere Waffe und Darts, die diese neuen Schichten mit weniger Schüssen zerstören können als die Standard Waffe.

Die Herausforderung besteht darin schnellst alle Ballons zu treffen, bevor sie die Burg erreichen, denn wenn sie die Burg passieren, werden Leben abgezogen. Bei allen verbrauchten Leben kommt man wieder ins Hauptmenu (3.2). Dadurch **endet** das Spiel, der HP sinkt auf 0 oder weniger. Je mehr Layer ein Bloon hat, der die Burg passiert, desto mehr HP werden abgezogen. Wie stark welcher Bloon ist, wird im Spielverlauf bekannt.

4 Video und Builds

Hier sind zwei Builds, einmal der Build der Abschlusspräsentation Vorführung und der Final Build zum Testen

Präsentations Build (kosten leicht angepasst)

<https://1drv.ms/u/s!AizbmdiWEv0rupUDty1rdgCb72e7RQ?e=O4au6M>

Final Build (Werte und Kosten massive angepasst, sodass alles getestet werden kann)

<https://1drv.ms/u/s!AizbmdiWEv0rusQSmcl3gUsqhNEzYg?e=eKp0l6>

Hier ist das ein Video des Spiels zu sehen:

<https://1drv.ms/v/s!AizbmdiWEv0rusNt5PCjOuGwpcuF8w>

5. Fazit

Das Spiel hat sich als funktionsfähig erwiesen und zeigt großes Potenzial für weitere Entwicklungen. In der Zukunft können wir, das Spiel durch die Implementierung verschiedener Ideen erweitern und das Spielgeschehen verbessern. Wir möchten klar definierte und abgrenzte Level erstellen, die den Spielern verschiedene Herausforderungen und Schweregrade bieten. Je höher das Level ist, desto anspruchsvoller, soll es für den Spielenden werden. Die Einführung verschiedener Maps macht die Spielumgebung vielfältiger und interessanter. Ein Online-Score-Board fördert den Wettbewerbsgeist und regt die Spieler an, sich ständig zu verbessern. Wir planen zusätzlich Out Game Progress und Vorteile außerhalb des Spiels einzuführen, um die Anzahl der Spieler zu erhöhen und den Spielern Ziele zu bieten als Meilensteine, auf die sie hinarbeiten können. Die Einführung verschiedener Waffentypen verbessert die strategischen Aspekte des Spiels und bietet den Spielern mehr Kontrolle und Flexibilität in ihren Spielstilen. Somit entstehen infolgedessen, einzigartige Spielmöglichkeiten. Wir möchten spezielle Fähigkeiten einführen, die den einzelnen Charakteren einzigartige Vorteile verleihen, um die Spielmechanik zu erweitern. Die Einführung von „Boss Bloons“ würde das Spiel herausfordernder und aufregender machen und den Spielern zusätzlich ein klares Ziel geben. Zusammenfassend lässt sich sagen, dass wir glauben, dass diese Verbesserungen das Spielerlebnis gewinnbringend bereichern könnten und das Spiel ansprechender machen würden.