Shadowboy

Bachelorarbeit aus Computermusik und Medienkunst

Timo Edelbauer

Betreuung: DI Johannes Zmölnig

Graz, 27. September 2019

institut für elektronische musik und akustik

Zusammenfassung

Diese Arbeit beschäftigt sich mit Computerspielen die primär Sound als Spielelement verwenden, sogenannten „Audio Games“. Im Rahmen des Seminars wurde außerdem ein Audio Game namens *Shadowboy* erstellt. Dabei handelt es sich um ein „2D Jump 'n' Run“ Spiel, in dem man mit Hilfe von Geräuschen, welche über ein Mikrofon aufgenommen werden, die Spielwelt verändern und Spezialattacken ausführen kann.

Der erste Teil der Arbeit befasst sich mit der Definition von Spielen, bis hin zur Entwicklung der ersten Videospiele und dem Unterschied zu Audio Games. Der zweite Teil der Arbeit fokussiert sich auf die Entstehungsgeschichte und der Entwicklung von *Shadowboy*. Abschließend werden Verbesserungen diskutiert, welche zukünftig implementiert werden könnten.

Abstract

This work deals with computer games, which mainly use sound as a gameplay element, so called „Audio Games“. As part of the seminar, an audio game called *Shadowboy* was created. It is a „2D Jump 'n' Run“ game, in which the player can control parts of the enviroment and perform special attacks, with the help of sounds, that are beiing picked up via a microphone.

The first part of the thesis deals with the definition of games, the development of the first video games and their difference to audio games. The second part of the work focuses on the development of *Shadowboy*. Finally, improvements, which could be implemented in the future, are discussed at the end of the thesis.

Inhaltsverzeichnis

1 Hintergrund **6**

1.1 Das Spiel . . . 6

1.1.1 Merkmale . . . 6

1.2 Computer- und Videospiele . . . 8

1.2.1 Die Entwicklung der Videospiele . . . 9

1.3 Audio Games . . . 13

1.3.1 A Blind Legend . . . 13

1.3.2 Lurking . . . 15

2 Shadowboy **16**

2.1 Idee . . . 16

2.2 Inspiration . . . 17

2.2.1 Spielmechanik - Super Mario Bros. . . . 17

2.2.2 Grafik Design - Pinstripe . . . 19

2.3 Vom Konzept zum Spiel . . . 20

2.4 Finale Version . . . 22

2.4.1 Spielwelt . . . 23

2.4.2 HUD . . . 23

2.4.3 Hauptcharakter . . . 24

2.4.4 Boni . . . 25

2.4.5 Standardgegner . . . 25

2.4.6 Endgegner . . . 27

2.4.7 Objekte . . . 29

2.4.8 Steuerung . . . 31

2.4.9 Spielstart und Menüs . . . 31

2.4.10 Sounds . . . 33

3 Technische Realisierung **34**

3.1 Grundsätzliche Informationen 34

3.1.1 Unity 34

3.2 Implementierung in Unity 36

3.2.1 Szene - Main Menu 36

3.2.2 Spielobjekte und Skripten 36

3.2.3 Szene - Level 1 38

3.2.4 Spielobjekte und Skripten 38

4 Fazit und Zukunft **42**

1 Hintergrund

1.1 Das Spiel

Definition

Einleitend ist es notwendig, den Begriff „Spiel“ zu definieren, sowie die Merkmale, welches ein Spiel besitzt, zu beschreiben. Um jedoch näher darauf eingehen zu können, muss zunächst geklärt werden, was Handlungen sind.

Das erste Merkmal einer Handlung ist die *Intentionalität*, die sich zum Beispiel bei Kindern schon im ersten Lebensjahr zeigt. Gestik und Gesichtsausdruck lassen auf die Absicht des Kindes schließen. Der *Gegenstandsbezug* beschreibt das zweite Merkmal. Gegenstände sind der Ausgangspunk für jegliche Handlungen. Diese müssen jedoch nicht materiell sein, sondern können auch als Ideen,Werte oder Regeln verstanden werden.

Der Begriff Spiel wird für eine besondere Form einer Handlung verwendet, bei der die Tätigkeit selbst in den Vordergrund rückt. Auf diese Art werden neue Fertigkeiten geübt, die sonst vermutlich nicht ausprobiert würden. [Oer97, vgl.]

1.1.1 Merkmale

Um nun die Frage beantworten zu können, bei welchen Handlungen es sich tatsächlich um ein Spiel handelt, schrieb Scheuerl in seinem umfassenden Buch „Das Spiel - Theorien des Spiels“ sechs grundlegende Merkmale nieder. *Freiheit*, *innere Unendlichkeit*, *Scheinhaftigkeit Ambivalenz*, *Geschlossenheit* und *Gegenwärtigkeit* bilden die Grundlage, auf denen ein Spiel aufgebaut ist. Die Merkmale formulieren einerseits aus welchen Gründen gespielt wird und andererseits die Rahmenbedingungen und den realen Implikationen eines Spiels.

Freiheit

Die Freiheit des Spiels bezieht sich auf dessen Zweck. Huber nennt beispielsweise das Spielen im Kindesalter, da es dort keinen Zweck erfüllt, sondern das Spiel deshalb betrieben wird, weil es unterhaltend ist. Keinem Kind wird befohlen zu spielen, sondern es tut dies, weil es entlastend wirkt. [Hub17, vgl. Huber, zitiert nach Scheuerl] Anders ist es wiederum zum Beispiel bei sportlichen Veranstaltungen und Wettkämpfen, da, in diesen Fällen, die Freiheit, durch den nun vorhanden Zweck, zerstört wird.

Innere Unendlichkeit

„Sechs von zehn Österreicherinnen und Österreichern in Internethaushalten spielen regelmäßig (mehrfach im Monat). Und fast jeder zweite Gamer (46 Prozent) spielt täglich oder beinahe täglich.“ [fU] Die vom Österreichischen Verband für Unterhaltungssoftware durchgeführte Studie zeigt, was Scheuerl als innere Unendlichkeit in seinem Buch beschreibt. Sobald das Spielbedürfnis befriedigt wurde, kommt es zur Beendigung des Spiels. Da es jedoch nur temporär gestillt werden kann, kommt es zur Wiederholung. Doch Scheuerl schreibt nicht nur über die Wiederholung des Spiels, sondern auch darüber, dass man das Zeitgefühl währenddessen verliert. Offensichtlich ist das vor allem bei Kindern der Fall. Diese werden zum Beispiel von Spielen regelrecht gefesselt, sodass sie nur durch äußere Faktoren bzw. Eltern aus dem Spiel gerissen werden können.

Scheinhaftigkeit

Scheuerls drittes Merkmal beschäftigt sich mit der Abgrenzung von realer und Fantasiewelt. Der Spieler gibt sich während des Spielens einer anderen Welt hin und löst sich somit von der Realität ab. Losgelöst vom Alltag kann er seine Fantasien nach belieben umsetzen. Dennoch kann diese Scheinwelt nicht vollkommen vom Realen getrennt werden. Nimmt man als Beispiel Mutproben oder Risikospiele, so können diese ernsthafte Konsequenzen in der Realität nach sich ziehen.

Ambivalenz

Mit dem Merkmal der Ambivalenz beschreibt Scheuerl die Unberechenbarkeit des Spiels. Damit ein Spiel möglichst interessant für den Spieler ist, darf es nicht zu vorhersehbar sein. Gegensätze müssen stets in einem Spannungsverhältnis zueinander stehen. Die Ambivalenz ergibt sich daher, dass sich das Spannungsmoment immer wieder in der Mitte einpendelt, das bedeutet, dass es kein vorgegebenes Muster gibt. Die Spannungslosigkeit würde das Ende des Spiels nach sich ziehen. Man stelle sich als Beispiel einen Spieler vor, der bereits sehr gut mit einem Spiel vertraut ist, da er es bereits einige Zeit allein gespielt hat. Kommt nun ein Mitspieler hinzu, steigt die Unberechenbarkeit und somit der Spannungsmoment des Spiels.

Geschlossenheit

Scheuerls Merkmal der Geschlossenheit des Spiels, scheint auf den ersten Blick im Widerspruch zu dessen Freiheit zu stehen. Auf der einen Seite ist das Spiel frei, auf der anderen braucht es jedoch Grenzen um zu bestehen. Es benötigt Gesetze oder Regeln, die ein Feld aufspannen, in welchem der Spieler sich frei bewegen darf. Dadurch nimmt das Spiel Gestalt an und behält seine Ordnung. Somit ist es zum Beispiel egal, welche Spieler in einem Spiel gegeneinander antreten, da das Spiel immer innerhalb der definierten

Grenzen abläuft. Gäbe es diese nicht, würde das Spiel im Chaos enden.

Gegenwärtigkeit

Als Gegenwärtigkeit beschreibt Scheuerl die veränderte Wahrnehmung der beteiligten Personen eines Spiels. Betrachtet man ein Spiel von außen, verläuft die Zeit innerhalb einer messbaren Spanne, doch für die Spieler existiert durch das Spielen ein anderes Zeitgefühl. Egal an welcher Stelle im Spielszenario sich der Spieler befindet, ist das Spiel an den Moment gebunden, in dem es stattfindet. Durch diese Gegenwärtigkeit findet das Spiel also quasi Zeitunabhängig statt.

Unter Berücksichtigung verschiedenster Spieltheorien beschreibt Scheuerl die grundsätzlichen Eigenschaften eines Spiels. Wie man erkennt, befassen sich die Merkmale mit dessen Aufbau, als auch der menschlichen Wahrnehmung. Sie finden sich außerdem in den unterschiedlichsten Erscheinungsformen von Spielen wieder. Im Gegensatz zu Scheuerl, befasst sich Huizinga in seinem Buch „Homo Ludens“, vor allem mit dem Spiel aus kultureller Sicht. Ebenfalls werden von Huizinga Merkmale zur Beschreibung eines Spiel genannt, die jedoch ähnlich wie bei Scheuerl ausfallen. Der große Unterschied ist jedoch, das Huizinga immer wieder auf den Unterschied zwischen Spaß und Ernst eingeht. „Der Mensch spielt als Kind zum Vergnügen und zur Erholung unterhalb des Niveaus des ernsthaften Lebens. Er kann auch über diesem Niveau spielen: Spiele der Schönheit und Heiligkeit.“ [Hui38, S.28] Er beharrt darauf, dass das Spiel sehr wohl ernsthaft betrieben werden kann und geht sogar so weit, dass er das Spielen als „heiliger Ernst“ bezeichnet, da die Kultur darin aufbewahrt bzw. weiterentwickelt wird.

1.2 Computer- und Videospiele

Definition

Spiele, die auf computerbasierten Geräten verwendet werden, tragen verschiedene Namen. Egal ob man sie nun Computer oder Videospiel nennt, handelt es sich bei den digitalen Spielen um Computerprogramme, durch die der Benutzer in eine virtuelle Welt abtauchen kann. Gespielt werden die „Games“ wie die Spiele umgangssprachlich bezeichnet werden, mittlerweile vor allem auf PCs, Konsolen und mobilen Geräten. Mit Hilfe von Tastatur, Controller oder Touchscreen können heutzutage dreidimensionale Charaktere durch atemberaubende Welten geführt und spannende Abenteuer erlebt werden. Der nachfolgende Teil fasst wichtige Entwicklungen der Videospielindustrie kurz zusammen, um anschließend näher auf die Besonderheiten von Audio Games einzugehen.

1.2.1 Die Entwicklung der Videospiele

Ende des 20. Jahrhunderts, durch die Verbesserung der technischen Gegebenheiten, entwickelten sich die ersten Arcade- Automaten. Spiele wie *Pong* oder *Space Invaders* wurden immer populärer und waren in den 1970er Jahren in zahlreichen Spielhallen zu finden. War es in *Pong* zunächst nur möglich einen Ball hin und her zu schießen, brachte *Space Invaders* zahlreiche Neuerungen. Es revolutionierte mit der ersten lenkbaren Spielfigur, dem Raumschiff und den zerstörbaren Gegnern, den Außerirdischen, die Spielindustrie.

Abbildung 1: Pong Arcade- Automat [ac]

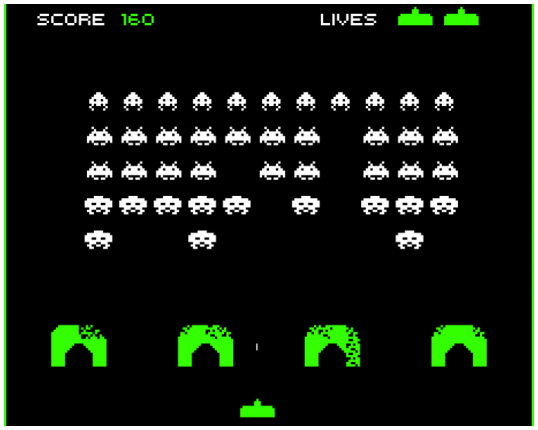

Abbildung 2: Space Invaders [Gra]

In den 80er Jahren dominierten vor allem *Pac-Man* und *Super Mario Bros.* die Bildschirme. Die Hauptcharaktere erhielten nun endlich eigene Namen, wodurch sie weitaus besser vermarktet werden konnten. Beiden gelang es, über die Grenzen des Bildschirms hinaus, bekannt zu werden. Doch auch die Spielmechanik trug zum Erfolg der Spiele bei. Die Idee des beweglichen Charakters, die schon in *Space Invaders* zu finden war, wurde weiter verbessert. Außerdem sorgten neue Handlungsmöglichkeiten und buntere Spielwelten dafür, dass Spieler noch tiefer in das Erlebnis eintauchen konnten. In *Pac-Man* musste die kleine gelbe Scheibe in einem Labyrinth vor Geistern fliehen, während in *Super Mario Bros.* ein Klempner Gegner und Hindernisse überwinden musste, um eine Prinzessin zu befreien (die Spielmechanik zu *Super Mario Bros.* wird in Abschnitt 2.2.1 näher besprochen).

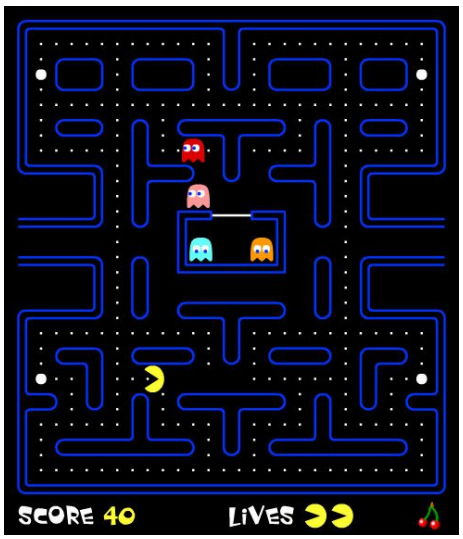

Abbildung 3: Pacman [Fis]

Abbildung 4: Super Mario [Web]

Mitte der 90er Jahre wurden mit einer neuen Generation von Konsolen schließlich dreidimensionale Spiele verwirklicht. Die höhere Leistungsfähigkeit und fortschreitende Technik ermöglichten eine bessere Grafik und umfangreichere Levels. Auch *Super Mario* gelang mit *Super Mario 64* der Sprung in die dritte Dimension.

Abbildung 5: Super Mario 64 [imd]

Zwar entstanden zu dieser Zeit auch die ersten Massively Multiplayer Online-Spiele, doch waren diese nur wenig erfolgreich. Erst Anfang des 21. Jahrhunderts war es aufgrund von Breitband-Internet möglich, dass tausende Spieler rund um die Welt zusammen Abenteuer erleben konnten. Eines, der auch heute noch bekanntesten Beispiele hierfür ist, das 2004 für den PC veröffentlichte Online-Rollenspiel, *World of Warcraft*. Die von Activision Blizzard, dem Publisher des Spiels, veröffentlichten Spielerzahlen in Abbildung 7, zeigen einen deutlich Anstieg der Spielerzahlen bis ungefähr 2010, ab diesem Zeitpunkt entwickelte sich jedoch ein deutlicher Abwärtstrend. Grund hierfür könnte sein, dass seit 2010 die Verbreitung mobiler Geräte wie Smartphones stark zunahm (Abbildung 8) und der Markt zunehmend auf mobile Geräte wechselte.

Abbildung 6: World of Warcraft [cho]

Aufgrund der starken Verbreitung mobiler Geräte, existiert heutzutage ein riesiger Markt für zahlreiche Spiele-Apps. Dennoch ist das Spielen auf anderen Plattformen alles andere als ausgestorben. Fortschrittliche Techniken, wie zum Beispiel Virtual Reality Brillen, werden auch in Zukunft dafür sorgen, dass sich Spieler noch tiefer ins Spielgeschehen einbinden können. Die Videospielindustrie entwickelte sich von einem Nischenhobby, welches nur in Spielhallen betrieben wurde, zu einem der weltweit lukrativsten Geschäftszweige.

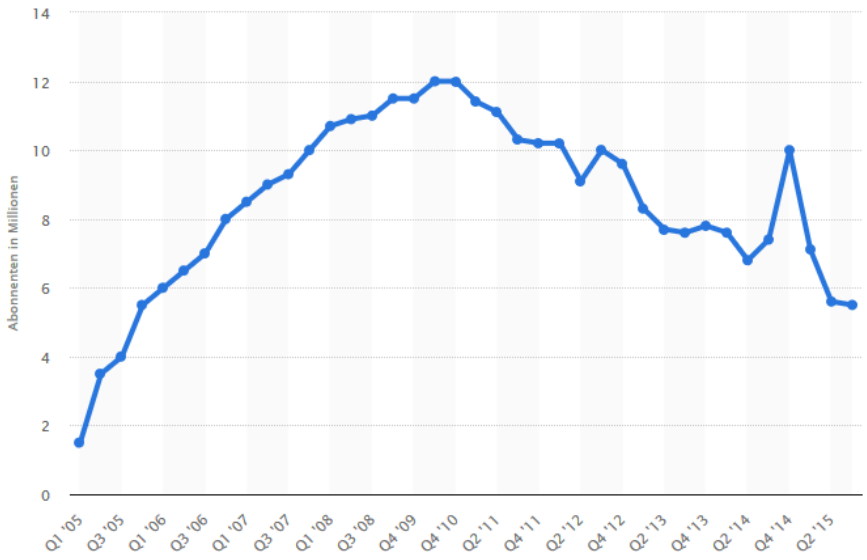

Abbildung 7: World of Warcraft Spielerzahlen [Bli]

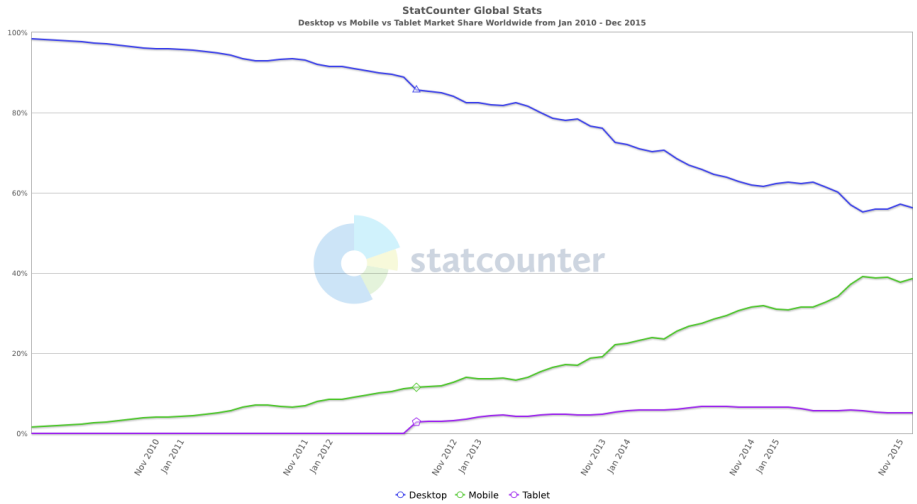

Abbildung 8: Marktverteilung [sG]

1.3 Audio Games

Was Audio Games von herkömmlichen Videospielen unterscheidet ist, dass sie oftmals problemlos von Menschen mit Sehbehinderungen spielbar sind. Bei dieser Art von Videospielen verlagert sich das Gameplay nämlich vorwiegend auf die Ausgabe von Klängen. Der Verzicht auf eine grafische Komponente ist zwar kein muss, doch sollte die Audioelemente des Spiels im Vordergrund stehen.

Im Gegensatz zum gigantischen Videospielmarkt (erwähnt in Abschnitt 1.2.1), sind Audio Games immer noch ein Nischenprodukt. Die Datenbank von *AudioGames.net* beinhaltet zum Beispiel nur etwa 600 Titel, welche sich jedoch immerhin über die verschiedensten Genres verteilen. Die ersten Audio Games konnten noch mittels Text-to-Speech-Systemen in Audio umgewandelt werden, da in den 1980er Jahren vor allem textbasierte Computerspiele weit verbreitet waren. Der Fortschritt der Technik führte aber dazu, dass Audio Games speziell konzipiert werden mussten. Im nachfolgenden Teil werden, um die Vielfalt der möglichen Implementationen aufzuzeigen, zwei Audio Games vorgestellt, die komplett unterschiedliche Ansätze verfolgen.

1.3.1 A Blind Legend

Da *A Blind Legend* ohne grafische Komponente entwickelt wurde, ist es auch für Menschen mit Sehbehinderungen spielbar. Das Gameplay wird ausschließlich über Klänge vermittelt, welche, durch die Implementierung von binauralem Audio, dem Spieler ein seltenes Spielvergnügen versprechen. Das Spiel ist „free to play“ und wurde für Windows, Mac und mobile Geräte entwickelt.

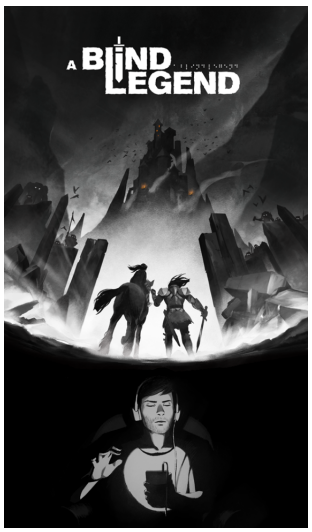

Abbildung 9: A Blind Legend [ulu]

Story

Als Edward Blake, einem legendären blinden Ritter, wird man von seiner Tochter Louise durch virtuelle Räume gelotst. Ziel des Spiels ist es, Edwards entführte Frau zu retten. Im Laufe der Geschichte begegnen Edward und Louise verschieden Gefahren. Die einzige Möglichkeit die Hindernisse zu überwinden und die Levels erfolgreich zu beenden, liegt darin, dass der Spieler Louises Anweisungen genau folgt. Der Bildschirm ist schwarz und die Anweisungen werden über Kopfhörer ausgegeben. Bewegungen und verschiedene Aktionen werden durch Gesten auf dem Touchscreen (mobile Geräte) oder der Maus (Windows, Mac) ausgeführt (die nachfolgende Erklärung bezieht sich auf das Gameplay mobiler Geräte).

Gameplay

Da der Spieler kein visuelles Feedback bezüglich der Umgebung bekommt, liefert Louise Informationen darüber. Durch ihre Anweisungen hilft sie dem Spieler sich im Raum zu orientieren. Gesteuert werden Edwards Bewegungen durch Gesten am Bildschirm. Wird ein Finger über den Bildschirm gezogen, bewegt sich der Charakter in Richtung des Fingers. Sobald der Finger vom Bildschirm genommen wird, stoppt der Charakter. Drehungen werden durch eine Bewegung des Fingers nach links oder rechts durchgeführt.

Ist Gefahr im Verzug muss Edward entweder seinen Schild oder sein Schwert benützen. Benützt man zum Beispiel den Schild während man von Steinen beschossen wird, verändert sich der Klang der Soundeffekte in den Kopfhörern. Der Effekt hält jedoch nur solang an, wie das Schild aktiviert ist. Damit eine Attacke während Kampfszenen ausgeführt wird, muss der Finger möglichst schnell über den Bildschirm bewegt werden. Auch währenddessen hilft Louise mit Anweisungen, da zur richtigen Zeit, die passende Geste am Bildschirm ausgeführt werden muss.

Zu Anfang des Spieles hat der Spieler mehrere Leben zur Verfügung, die durch ein Herzklopf Geräusch verdeutlicht werden. Nimmt Edward Schaden, wird sein Herzschlag schneller. Glücklicherweise regeneriert sich die Gesundheit mit der Zeit, was durch einen langsamer werdenden Herzschlag repräsentiert wird.

A Blind Legend ist ein gutes Beispiel dafür, dass Audio Games auch komplett auf die visuelle Komponente verzichten können. Die Herausforderung besteht darin, genau hinzuhören und allzeit bereit zu sein Louises Anweisungen zu folgen. Das Spiel nutzt die Tatsache, dass der Spieler sich „im dunklen“ über den weiteren Verlauf des Spiels befindet, um immer mehr Spannung aufzubauen. Ebenso sorgt die einfache Steuerung dafür, dass sich selbst unerfahrene Spieler schnell zurechtfinden.

1.3.2 Lurking

Bei *Lurking* handelt es sich um ein weiteres Beispiel für ein Audio Game, welches jedoch, im Vergleich zu *A Blind Legend*, nicht auf die visuelle Komponente verzichtet, sondern diese auf interessante Weise nützt. Das Spiel ist ebenfalls „free to play“ und wurde für Windows, Mac, sowie OculusVR entwickelt.

Abbildung 10: Lurking Titelbildschirm [lg]

Story

Der Spieler schlüpft in die Rolle einer Figur, die ohne Informationen in einem Krankenhauszimmer aufwacht. Ziel ist es, den Charakter möglichst leise durch die dunklen Gänge des Krankenhauses zu bewegen, um nach und nach herauszufinden was in den Stunden davor passiert ist. Die Herausforderung besteht darin, nicht zu viel Aufmerksamkeit von den geräuschempfindlichen Gegner auf sich zu ziehen.

Gameplay

Gesteuert wird der Charakter, mittels Tastatur und Maus, aus Sicht der Ego Perspektive. Des weiteren kann ein Mikrofon verwendet werden, dass die Geräusche des Spielers in das Spiel einbindet. Die einzige Möglichkeit sich in dem dunklen Krankenhaus zurecht zu finden, ist indem man Klänge erzeugt. Die Schritte des Hauptcharakters, als auch jegliches Geräusch das vom Mikrofon aufgenommen wird, erzeugt Impulse, die den Raum in dem man sich befindet, kurz umreißen. Abhängig von der Lautstärke, variiert der Radius des erzeugten Impulses. Je lauter ein Geräusch ist, desto stärker ist der Impuls. Um zu verhindern, dass der Spieler wahllos Impulse erzeugt und durch die Gänge irrt, gibt es Gegner, die angelockt werden, je lauter der Spieler ist. Wird der Spieler von einem Gegner gefangen, endet das Spiel in einem „Game Over“.

Gleich wie bei *A Blind Legend*, ist auch bei *Lurking* die volle Aufmerksamkeit des Spielers gefragt. Durch die eingeschränkte Sicht und den lauernden Gegnern, muss der Spieler

die Geräusche mit Vorsicht erzeugen. Einerseits muss er probieren, ein klares Bild der Umgebung zu bekommen, andererseits muss er aber auch leise genug sein, um nicht die Aufmerksamkeit der Gegner zu erregen. Der dadurch entstehende Konflikt, sorgt für das hohe Maß an Spannung im Spiel.

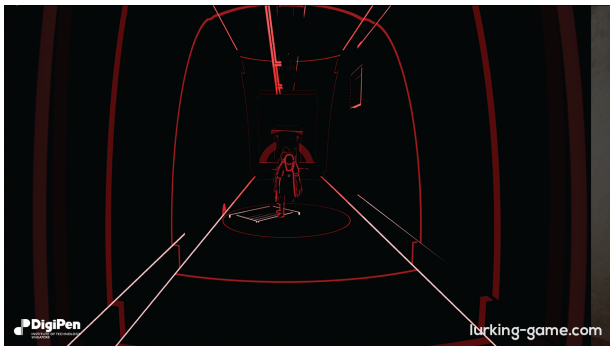

Abbildung 11: Lurking Gameplay [lg]

2 Shadowboy

Im Zuge des Seminars wurde mit *Shadowboy* ein Spiel geschrieben, welches sich stark an klassischen 2D Jump 'n' Run Games orientiert. Man schlüpft in die Rolle einer namenlosen Figur, die in einer dunklen Nacht von Haus zu Haus springt und dabei einige Hindernisse, als auch Gegner überwinden muss. Die herkömmlichen Elemente des Generes wurden mit denen eines Audio Games erweitert. Durch Nutzung eines Mikrofons können bestimmte Objekte in der Spielwelt manipuliert, sowie Spezialattacken ausgeführt werden.

2.1 Idee

Im Sport oder bei anderen Wettkämpfen, bringt man zum Beispiel durch einen Jubelschrei, Emotionen zum Ausdruck. Ähnlich drückt man beim Spielen von Videospielen mittels Geräuschen oder Lauten, ebenfalls Gefühle aus. Je nachdem wie das Spiel verläuft, äußert man sich währenddessen frustriert oder erfreut.

Aus dieser Überlegung entstand die Idee, die Emotionen des Spielers in das Spielgeschehen einzubinden. Ziel war es, Geräusche, die von der spielenden Person abgegeben werden, ins Spiel zu integrieren und als Steuerelement zu verwenden. Es sollte nur möglich sein das Ende des Spiels zu erreichen, indem man an manchen Passagen laut und an anderen Stellen wieder möglichst leise ist. Die Hindernisse sollten auf die Lautstärke der Geräusche reagieren und bei Bedarf als taktisches Hilfsmittel verwendet werden können.

2.2 Inspiration

Da es sich bei *Shadowboy* um ein 2D sidescroller Game handelt, orientiert es sich bezüglich des Gameplays stark an den populären Videospielen der neunziger Jahre. Besonders *Super Mario Bros.* wurde als Inspiration für die Spielmechaniken verwendet.

Ein anderes Ziel war es, dass sich das Spiel durch das Gameplay zwar bekannt anfühlt, die Grafik jedoch, sollte sich stark von klassischen und bekannten Jump 'n' Run Spielen abheben. Anstatt starrer, verpixelter Charaktere, sollten sie hochauflösend aussehen und lebendig wirken. *Pinstripe* inspirierte das Grafik Design der Spielwelt und Charaktere.

2.2.1 Spielmechanik - Super Mario Bros.

Super Mario Bros. wurde 1985 von Nintendo für die NES veröffentlicht. Im Spiel schlüpft man in die Rolle des Helden Mario, der es sich zur Aufgabe gemacht hat, die gefangen gehaltene Prinzessin zu befreien.

Bewegt man Mario in eine Richtung, rückt der Bildschirm immer allmählich nach. Um die Prinzessin zu retten, muss er es in der vorgegebenen Zeit nach ganz rechts, bis zum Ende des Levels, schaffen. Auf dem Weg gilt es Berge, Gruben, Gewässer, Schildkrötensoldaten und eine Vielzahl von Fallen und Rätseln zu überwinden.

Abbildung 12: Super Mario Screenshot [igc]

Startposition und Ziel

Zu Beginn jedes Levels startet der Spieler ganz links am Anfang des Gebiets. Außerdem beginnt die Uhr, oben rechts auf dem Bildschirm, zu ticken. Jede verbleibende Zeit auf der Uhr führt zu Bonuspunkten, sofern man das Ende des Levels erreicht hat.

Am Ende eines jeden Levels befindet sich eine kleine Burg. Das Level gilt als abgeschlossen, sobald man auf den Fahnenmast vor der Burg gesprungen ist. Je höher man auf

dem Fahnenmast landet, desto höher sind die Bonuspunkte, die man erhält.

Bewegung

Um Mario nach links oder rechts zu bewegen, benützt man die jeweiligen Tasten auf dem Control Pad. Drückt man die A-Taste, hüpft der Spielcharakter. Wie hoch Mario springt, hängt davon ab, wie lange man die A-Taste gedrückt hält. Hält man die B-Taste gedrückt, beginnt Mario schneller zu laufen. Durch die größere Geschwindigkeit, ist es auch möglich höher und weiter zu springen.

Boni

Im Verlauf des Levels, ist es möglich nützliche Gegenstände aufzusammeln. Durch das Zerschmettern von Steinen, indem Mario mit seinem Kopf dagegen hüpft, erscheinen Pilze, Blumen oder Sterne. Darüber hinaus erhält man beim Zerstören von Steinen Bonuspunkte.

Pilze, Blumen und Sterne verändern neben dem Aussehen des Charakters auch seine Eigenschaften. Ein Pilz vergrößert Mario, wird daraufhin eine Blume aufgesammelt, kann er durch kurzes drücken der B-Taste, Feuerbälle schießen. Sobald man durch einen Gegner jedoch verletzt wird, schrumpft Mario und seine speziellen Fähigkeiten gehen wieder verloren. Eine Ausnahme besteht dann, wenn Mario einen Stern aufgesammelt hat, da dieser ihn für einige Sekunden unbesiegbar macht. Dargestellt wird dies dadurch, dass Mario zu blinken beginnt und jede Berührung mit Gegnern, diese zerstört, ohne das Mario selbst Schaden nimmt.

Gegner

Die Gegner, auf die Mario im Spiel trifft, können auf unterschiedliche Weise besiegt werden. Wenn Mario gerade Feuerbälle schießen kann, ist es meistens möglich sie dadurch zu besiegen. Grundsätzlich reicht es jedoch aus, dass Mario auf den Kopf des Gegners hüpft. Schaden nimmt Mario nur, wenn ihn eine gegnerische Spielfigur von einer Seite berührt oder er von einem, vom Gegner geworfenen Objekt, berührt wird.

Gesundheit und Leben

Sobald Mario einen Gegner seitlich berührt oder von einer Plattform fällt, stirbt er und man startet wieder vom letzten Checkpoint. Falls Mario gerade einen Bonus aufgesammelt hat, tötet ihn eine einmalige Berührung nicht, sondern entfernt nur den Bonus.

Jedes mal wenn Mario stirbt, wird dem Spieler ein Leben abgezogen. Sobald man das letzte Leben verbraucht hat, startet man nicht mehr vom letzten Checkpoint, sondern muss das gesamte Level nochmals vom Anfang spielen.

2.2.2 Grafik Design - Pinstripe

Pinstripe handelt von einem Ex-Pfarrer namens Teddy, der die Hölle erkunden muss, um seine Tochter Bo zu finden, die von einem fremden Wesen namens Mr. Pinstripe entführt wurde.

Im Gegensatz zu *Super Mario Bros.* wurde bei *Pinstripe* ein großes Augenmerk auf die Grafik gelegt. Was man natürlich darauf zurückführen kann, dass *Pinstripe* 2017 veröffentlicht wurde, mehr als 30 Jahre nach *Super Mario Bros.* Vergleicht man die beiden Spiele jedoch trotzdem, fällt sofort auf, dass die einzelnen Level in *Pinstripe* lebendiger und detailreicher sind.

Zwar handelt es sich bei *Pinstripe* um ein 2D Spiel, durch den Parallax Effekt wirkt es jedoch so, als ob sich der Hauptcharakter in einer 3D Welt befindet. Der Effekt beschreibt, dass sich Gegenstände die näher im Bild sind, langsamer bewegen, als Gegenstände die weiter entfernt sind. Des Weiteren ist jedes Level so aufgebaut, dass die Gegenstände die näher im Bild sind, einen dunkleren Farbton haben, als jene die sich im Hintergrund befinden. Diese farbliche Abstufung verstärkt das Gefühl der Tiefe und lässt die Welt deutlich größer erscheinen. Oftmals erkennt man auch eine Nebelschwade hinter Objekten, wie zwischen den einzelnen Vordergrundebenen in Abbildung 14, die dieses Gefühl nochmals erhöhen.

Das Design des Spiels scheint zum Teil auf geometrischen Formen zu basieren. Speziell der Hauptcharakter ist minimalistisch gehalten und besteht im wesentlichen aus einem Kreis und ein paar Strichen. Durch die ständige Bewegung der Arme oder der Haare, wirkt die Figur jedoch nicht wie ein starres Strichmännchen, sondern wesentlich lebendiger. Doch nicht nur der Hauptcharakter verfügt über einige Bewegungsanimationen, auch die Welt in der er sich bewegt ist nicht bloß ein starres Gebilde. Es fällt Schnee, es blinken Lichter und Fahnen wehen im Wind. Das alles trägt dazu bei, dass die Spielwelt einen dynamischen Eindruck hinterlässt und extrem interessant auf den Spieler wirkt.

Außerdem ist es auffällig, dass die Level immer mit einer begrenzten Anzahl von Farben kreiert wurden. Die Welt in der sich der Hauptcharakter bewegt, ist nicht übermäßig bunt, sondern hält sich stets an eine Palette von Farben. In Abbildung 13 überwiegen zum Beispiel warme, rötlichere und in Abbildung 14 kalte, bläulichere Farben.

Abbildung 13: Pinstripe Screenshot [Mak]

Abbildung 14: Pinstripe Screenshot [Mak]

2.3 Vom Konzept zum Spiel

Zunächst war es die Aufgabe, das passende Grafik Design für die Spielwelt und die Charaktere zu entwerfen. Um weitere Ideen zu bekommen, wurden die in Abbildung 15, 16 und 17 zu sehenden Konzepteichungen angefertigt.

Farben

Ähnlich wie in *Pinstripe*, wurde das Konzept auf wenige Farben begrenzt. Die Spielumgebung sollte vorwiegend in dunklen Violett und Blautönen, sowie schwarz gehalten werden.

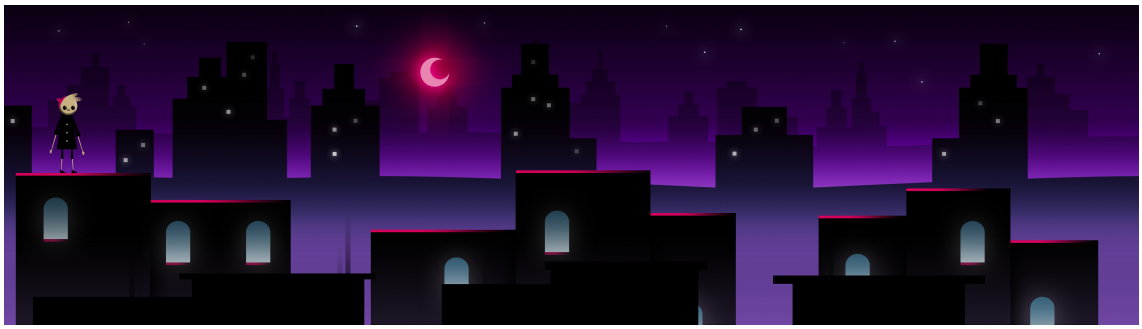

Abbildung 15: Konzeptzeichnung: Umgebung

Spielwelt

Beginnend bei den grundlegenden Elementen eines 2D Jump 'n' Run Spiels wurden zunächst Plattformen erstellt. Dabei sollte das Design möglichst simpel gehalten werden. Die rechteckigen Plattformen, wie sie auch in *Super Mario Bros.* zu finden sind, dienten als Inspiration. Kurz darauf entstand die Idee, aus den rechteckigen Plattformen Häuser zu machen, weshalb vereinzelt leicht abgerundete Fenster eingefügt wurden. Um der Spielwelt den Eindruck von Tiefe zu geben, wurden immer heller und kleiner werdende Gebäude im Hintergrund platziert. Der Vordergrund wurde von großen schwarzen Gebäuden besetzt, von welchen nur die Dächer sichtbar sein sollten. Um den Effekt der Tiefe zusätzlich zu verstärken, wurde violetter Nebel zwischen den einzelnen Gebäuden im Vorder- und Hintergrund eingefügt. Ähnlich wie die Gebäude, besitzt der Nebel im Hintergrund des Bildes auch einen helleren Farbton als im Vordergrund.

Um für den Spieler zu verdeutlichen welche Gebäude als betretbare Plattformen dienen, mussten sie gekennzeichnet werden. Die Idee war es, die jeweiligen Plattformen farblich vom Rest der Welt abzuheben. Veranschaulicht wurde dies, durch einen pinken Schimmer am Dach des betreffenden Gebäudes. Alle betretbaren Plattformen sollten also den Eindruck erwecken vom Mond beleuchtet zu werden, alle anderen nicht.

Charakter und Gegner

Die grundsätzliche Idee bei der Erschaffung des Hauptcharakters war es, ihn ähnlich wie den Hauptcharakter von *Pinstripe* zu designen. Das bedeutete, er sollte gleich wie die Spielwelt die ihn umgibt, aus simplen geometrischen Formen bestehen und nicht allzu viele Details besitzen. Die wirklich auffälligen optischen Attribute des Hauptcharakters sind seine Augen und seine langen Gliedmaßen, die so kreiert wurden, um sie in einem späteren Entwicklungsstadium zu animieren und den Charakter damit lebendiger wirken zu lassen.

Abbildung 16: Konzeptzeichnung: Hauptcharakter

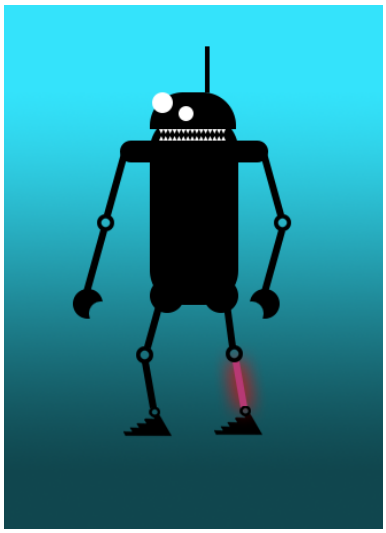

Abbildung 17: Konzeptzeichnung: Standardgegner

Ähnlich wie der Hauptcharakter, sollten auch die Standardgegner große Augen und lange Gliedmaßen besitzen. Um sie angsteinflößender wirken zu lassen, wurden außerdem spitze Zähne hinzugefügt. Ziel war es, die Standardgegner wie Roboter aussehen zu lassen.

2.4 Finale Version

Im Laufe der Entwicklung stellte sich heraus, dass die Designs der Konzeptzeichungen teilweise angepasst, als auch neue Elemente geschaffen werden mussten, um ein besseres Spielerlebnis sicherzustellen.

2.4.1 Spielwelt

Das Grundkonzept der Spielwelt wurde größtenteils übernommen, es wurden noch zusätzlich Nebelschwaden vor den Plattformen eingefügt, um das Gefühl zu vermitteln, dass es sich dabei um Hochhäuser handelt. Außerdem wurden an manchen Stellen im Spiel Hinweistafeln eingefügt, um dem Spieler in kniffligen Situationen ein wenig zu helfen.

2.4.2 HUD

Das Heads Up Display („HUD“) befindet sich in der linken oberen Ecke des Bildschirms und gibt Auskunft über einige wichtige Eigenschaften. Es besteht aus 4 Teilen:

- Pegelanzeige
- Lebensanzeige
- Münzanzeige
- Modusanzeige

Die *Pegelanzeige* zeigt an, wie sehr das gerade verwendete Mikrofon ausgesteuert ist. Je weiter der Balken nach rechts ausschlägt, desto mehr Geräusche werden gerade vom Mikrofon aufgenommen.

Die *Lebensanzeige* zeigt an, wie viele Herzen der Hauptcharakter gerade besitzt.

Die *Münzanzeige* gibt an, wie viele Münzen im Verlauf des Spiels bereits gesammelt wurden. Mit jeder aufgesammelten Münze erhöht sich der Zähler um eins.

Die *Modusanzeige* weist den Spieler darauf hin, ob der Charakter sich im „Super Attack Modus“ befindet. Der Text „SUPER ATTACK MODE“ erscheint nur, wenn der Modus aktiviert wurde.

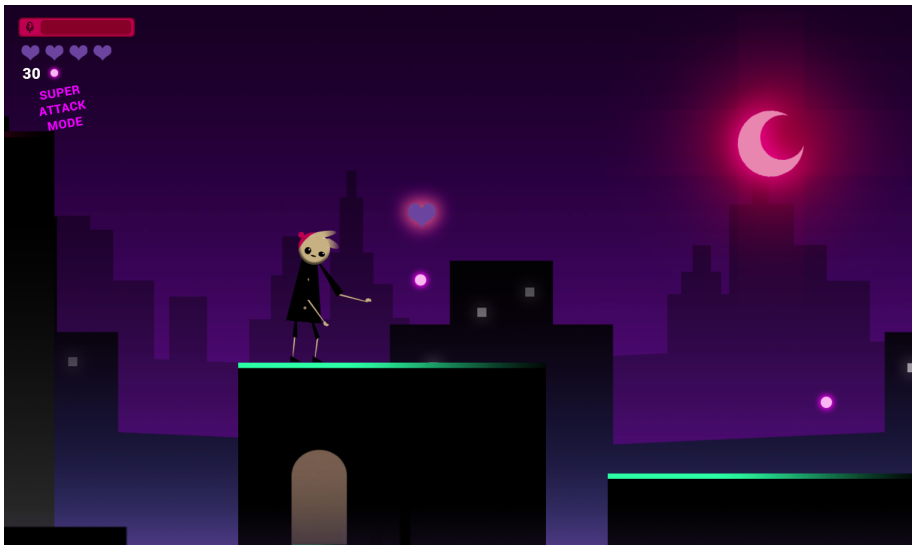

Abbildung 18: Heads Up Display mit aktivierter Modusanzeige

2.4.3 Hauptcharakter

Das Design des Hauptcharakters wurde direkt von der Konzeptzeichnung übernommen.

Eigenschaften

Der Hauptcharakter hat die Möglichkeit sich horizontal als auch vertikal zu bewegen. Um von Plattform zu Plattform zu kommen, hat er die Fähigkeit einen Doppelsprung auszuführen. Dieser ist am effektivsten und weitesten, wenn nach einem einfachen Sprung, an der höchsten Position, ein weiteres mal die Sprungtaste gedrückt wird.

Wenn mindestens 30 Münzen aufgesammelt wurden, befindet sich der Charakter im „Super Attack Modus“ und kann eine Superattacke ausführen. Dazu muss der Pegel des Mikrofons während eines Doppelsprungs einen Grenzwert überschreiten. Eine Superattacke kann jeden im Spiel zu begegnenden Gegner zerstören.

Der Hauptcharakter startet mit vier Herzen, die ihm durch Verletzungen von Feinden abgezogen werden können. Sinkt die Lebensanzeige auf null, stirbt er und das Spiel endet in einem „Game Over“. Ein Sturz von einer Plattform sorgt ebenfalls dafür, dass der Charakter stirbt und das Spiel vorzeitig endet.

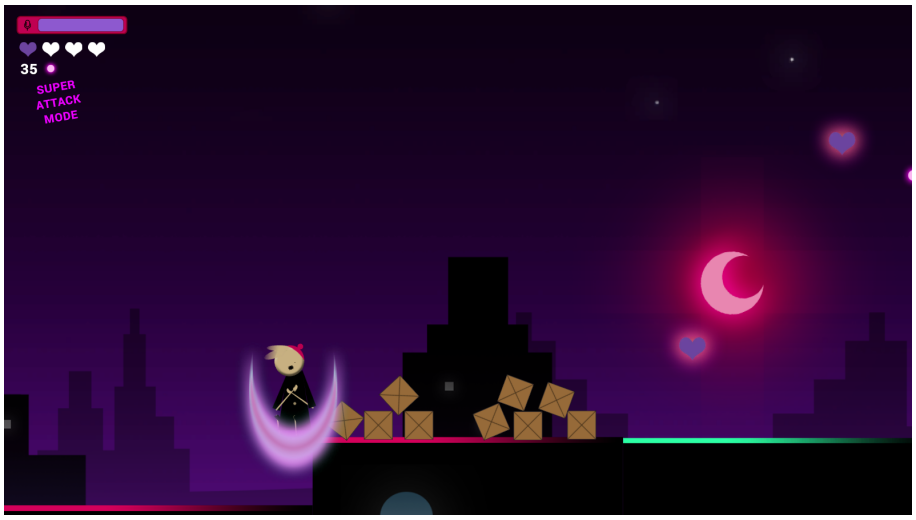

Abbildung 19: Superattacke

2.4.4 Boni

Wie in *Super Mario Bros.*, gibt es auch in *Shadowboy* Sammelobjekte, die gewisse Effekte hervorrufen. Gleich nach dem Start des Spiels ist es möglich sie einzusammeln. Im Gegensatz zum Vorbild, sind die Objekte in diesem Spiel jedoch frei verstreut. Es gibt zwei Arten von Objekten, die für den Hauptcharakter aufsammelbar sind:

- Herzen
- Münzen

Herzen sorgen dafür, dass der Hauptcharakter einen Teil seiner Gesundheit wieder regeneriert. Pro aufgesammeltem Herz, füllt sich die Lebensanzeige wieder um eins auf. Wird ein Herz aufgesammelt während die Lebensanzeige bereits voll ist, hat das Aufsammeln keine Auswirkungen.

Münzen sind zahlreich in der Welt zu finden. Sobald der Hauptcharakter mehr als 29 aufgesammelt hat, wird der „Super Attack Modus“ aktiviert. Ebenfalls wird dem Spieler nach erfolgreichem Abschluss des Levels angezeigt, wie viele Münzen vom Hauptcharakter im Verlauf des Spiels aufgesammelt wurden.

2.4.5 Standardgegner

Design

Die Konzeptzeichnung der Standardgegner unterscheidet sich stark von der finalen Version im Spiel. In einem späteren Entwicklungsstadium stellte sich heraus, dass die ursprünglich geplanten Gegner zu groß waren und zu bedrohlich wirkten. Vorallem, da es

der Plan war, sie wie in *Super Mario Bros.* durch einen Sprung auf den Kopf auszuschalten, musste das Design angepasst werden.

Abbildung 20: Finale Version: Standardgegner

Wie in Abbildung 20 zu sehen ist, sind die Standardgegner in der finalen Version wesentlich kleiner. Das Grundkonzept des Kopfes blieb erhalten, der gesamte restliche Körper wurde jedoch entfernt. Außerdem wurden noch ein paar farbliche Akzente hinzugefügt.

Eigenschaften

Ein Standardgegner patrouilliert ständig in einem bestimmten Abschnitt hin und her. Die Geschwindigkeit mit der die Standardgegner patrouillieren, nimmt zu, je weiter der Hauptcharakter im Level fortschreitet.

Berührt ein Standardgegner den Hauptcharakter seitlich, so zieht er diesem ein Herz ab. Gleichzeitig wird er selbst jedoch durch die Berührung nach hinten gestoßen. Das kann für den Spieler in manchen Situationen von Vorteil sein, da es möglich ist, dass der Standardgegner dadurch von der Plattform fällt. Die falsche Taktik ist es jedoch, zu probieren ihn absichtlich hinunter zustoßen, da dieser Versuch ihn zu überwinden, meist in den schnellen Tod des Hauptcharakters führt.

Um den Standardgegner zu besiegen, ist es notwendig senkrecht auf dessen Kopf zu springen. Ähnlich wie in *Super Mario Bros.*, schaltet diese Aktion den Gegner sofort aus und endet damit, dass er in etliche Teile zersplittert.

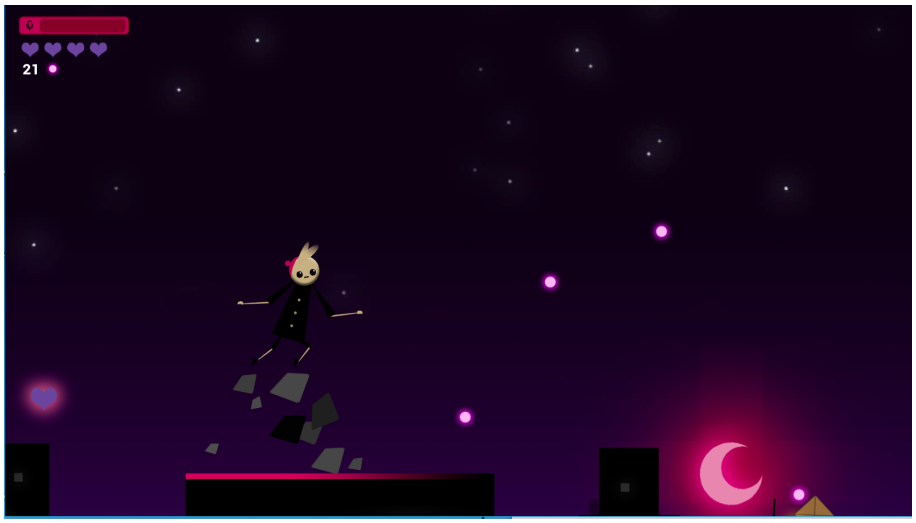

Abbildung 21: Zersplittern eines Standardgegners

2.4.6 Endgegner

Aus der Konzeptzeichnung in Abbildung 17 entstand unter anderem auch die Idee für einen Gegner, der am Ende des Levels besiegt werden muss, um es erfolgreich abzuschließen. Im Gegensatz zu den Standardgegnern, besitzt der Endgegner einen Körper und ist wesentlich größer. Außerdem befindet sich ein Feuerball auf seiner linken Hand.

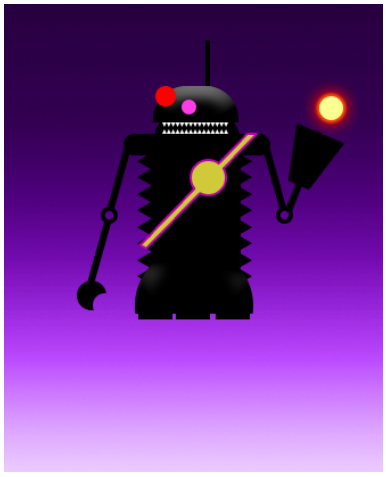

Abbildung 22: Endgegner

Eigenschaften

Im Gegensatz zu den Standardgegnern patrouilliert der Endgegner nicht, sondern schwebt ständig auf und ab.

Sobald der Hauptcharakter den letzten Abschnitt des Levels betritt, beginnt der Endgegner Feuerbälle zu schleudern. Die Herausforderung dabei ist, den Feuerbällen zu entkommen, da sie den Hauptcharakter verfolgen. Es ist jedoch möglich, hinter einer Wand Schutz vor ihnen zu suchen (siehe Abbildung 23).

Außerdem darf der Hauptcharakter den Endgegner nicht berühren, da er dadurch sofort alle seine Herzen verlieren und sterben würde. Die einzige Möglichkeit den Endgegner zu besiegen besteht darin, eine gezielte Superattacke, senkrecht auf dessen Kopf, auszuführen. Dazu muss der Spieler bis zum letzten Abschnitt des Levels jedoch mindestens 30 Münzen gesammelt und damit den „Super Attack Modus“ aktiviert haben. Ist der Endgegner besiegt, gilt das Spiel als gewonnen.

Abbildung 23: Charakter geschützt durch Wand

2.4.7 Objekte

Die Komponente, die das Spiel von anderen Jump 'n' Run Spielen abhebt, sind die lärmempfindlichen Objekte. Einerseits dienen sie als Hindernisse, die es zu überwinden gilt, andererseits können sie taktisch klug im Kampf gegen Gegner benützt werden. Es gibt 2 Arten von Objekten:

- Wände
- Spezialgebäude

Für *Wände* gibt es zwei Zustände: Aufgebaut oder zusammengestürzt (siehe Abbildung 24 und 25). Standardmäßig sind Wände zusammengestürzt, sobald der Hauptcharakter sich jedoch in der Nähe einer Wand befindet, baut sich diese auf. Die einzige Möglichkeit eine Wand zu überwinden, besteht darin, sie durch Geräusche zu Fall zu bringen. Überschreitet der Mikrofonpegel einen gewissen Grenzwert, fällt die Wand in sich zusammen und bleibt so lange in diesem Zustand, bis der Pegel wieder unter den Grenzwert fällt.

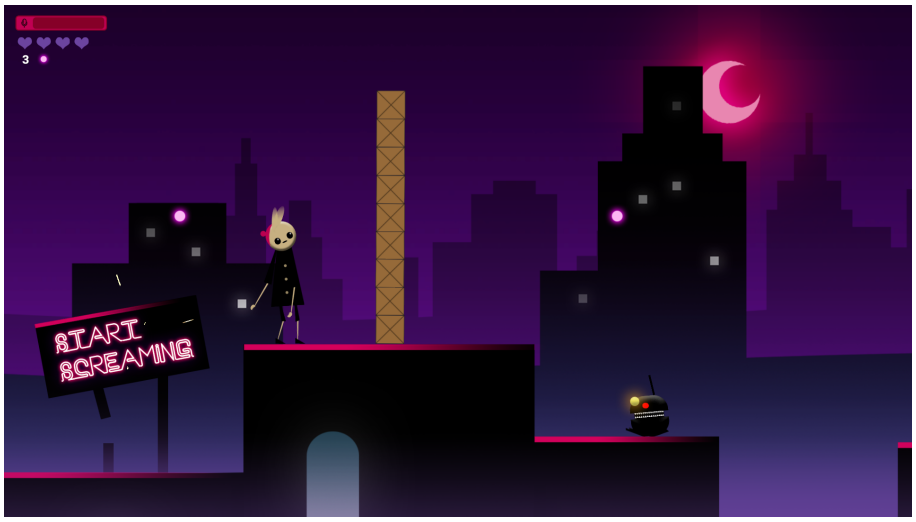

Abbildung 24: aufgebaute Wand

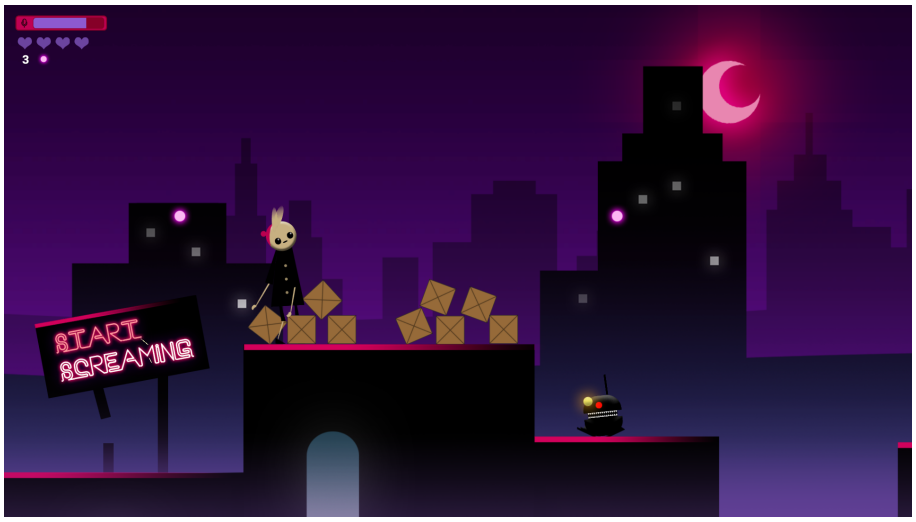

Abbildung 25: zusammengestürzte Wand

Spezialgebäude unterscheiden sich rein optisch von den herkömmlichen Plattformen, durch ihren grünen Schimmer (siehe Abbildung 26). Sie stellen für den Hauptcharakter ein gefährliches Hindernis da. Ähnlich wie Wände haben Spezialgebäude auch zwei Zustände: Aufgebaut oder zusammengestürzt. Standardmäßig sind Spezialgebäude aufgebaut, sobald sich der Hauptcharakter jedoch in unmittelbarer Nähe befindet, darf der Mikrofonpegel einen gewissen Grenzwert nicht überschreiten, da die Spezialgebäude sonst in sich zusammenfallen. Erst wenn der Mikrofonpegel wieder unter den Grenzwert fällt, bauen sich die Spezialgebäude wieder auf.

Abbildung 26: Objekt- Spezialgebäude

Zu Beginn befinden sich vor den Objekten Hinweistafeln, die dem Spieler helfen sollen zu begreifen, auf welche Art das nachfolgende Gebiet zu überwinden ist (siehe Abbildung 24, 25 und Abbildung 26).

2.4.8 Steuerung

Zum Spielen benötigt man eine Tastatur und eventuell ein Mikrofon, falls standardmäßig keines im Computer verbaut ist.

Zur Bewegung des Hauptcharakters werden die Tasten A und D verwendet. Taste A bewegt den Charakter nach links, Taste D nach rechts. Um zu springen oder einen Doppelsprung auszuführen, muss die Leertaste ein- beziehungsweise zweimal betätigt werden.

Mit Hilfe von Geräuschen, die über das Mikrofon aufgenommen werden, können Objekte beeinflusst und Superattacken ausgeführt werden (für nähere Details siehe Abschnitt 2.4.3 und Abschnitt 2.4.7)

2.4.9 Spielstart und Menüs

Hauptmenü

Nach dem Start der Applikation, befindet man sich im Startmenü. Ein Klick auf den orangen „Start Button“ führt ins Anleitungsmenü. In diesem Menü werden die grundlegenden Spielregeln sowie die Steuerung kurz erklärt. Im darauffolgenden Kalibrationsmenü werden die Lautstärke der Musik und Effekte, als auch die Empfindlichkeit des Mikrofons bestimmt. Der Spieler muss den Anweisungen am Bildschirm folgen und sobald er die Lautstärke eingestellt und das Mikrofon abgestimmt hat, ist er bereit das Spiel zu starten.

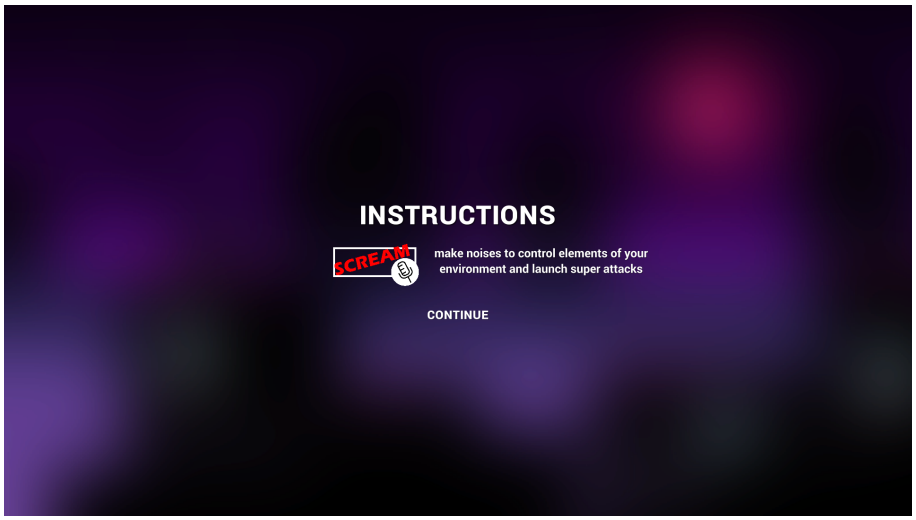

Abbildung 27: Anleitungsmenü

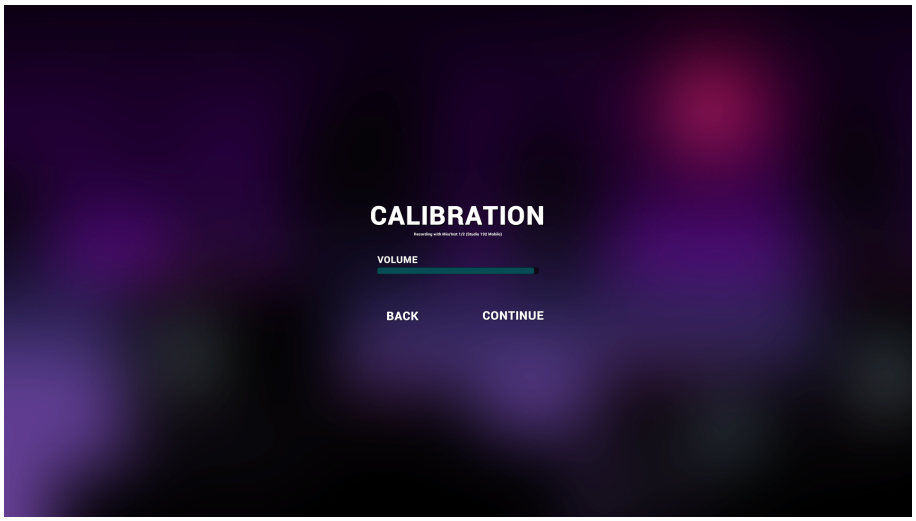

Abbildung 28: Kalibrationsmenü

Spielmenüs

Während des Spiels hat man die Möglichkeit das Spiel mittels der Escape Taste zu pausieren. Daraufhin erscheint das *Pausenmenü*, in welchem man sich entscheiden kann, zum Hauptmenü zurückzukehren, das Spiel wiederaufzunehmen oder das Spiel zu beenden.

Je nachdem ob man mit dem Charakter das Ende des Spiels erreicht oder auf dem Weg dorthin scheitert, erscheinen unterschiedliche Menüs. Beim vorzeitigen Tod des Hauptcharakters, erscheint das Verlierermenü, beim erfolgreichen Abschluss, das Gewinnermenü.

Im *Verlierermenü* hat der Spieler die Möglichkeit das Spiel von Beginn an neu zu starten, zum Hauptmenü zurückzukehren oder das Spiel zu beenden.

Im *Gewinnermenü* wird dem Spieler nicht nur mitgeteilt, dass er gewonnen hat, sondern auch die Anzahl der Münzen, die der Hauptcharakter im Verlauf des Spiels gesammelt hat. Im Menü gibt es die Optionen das Spiel noch einmal von Beginn zu spielen, zum Hauptmenü zurück zu kehren oder das Spiel zu beenden.

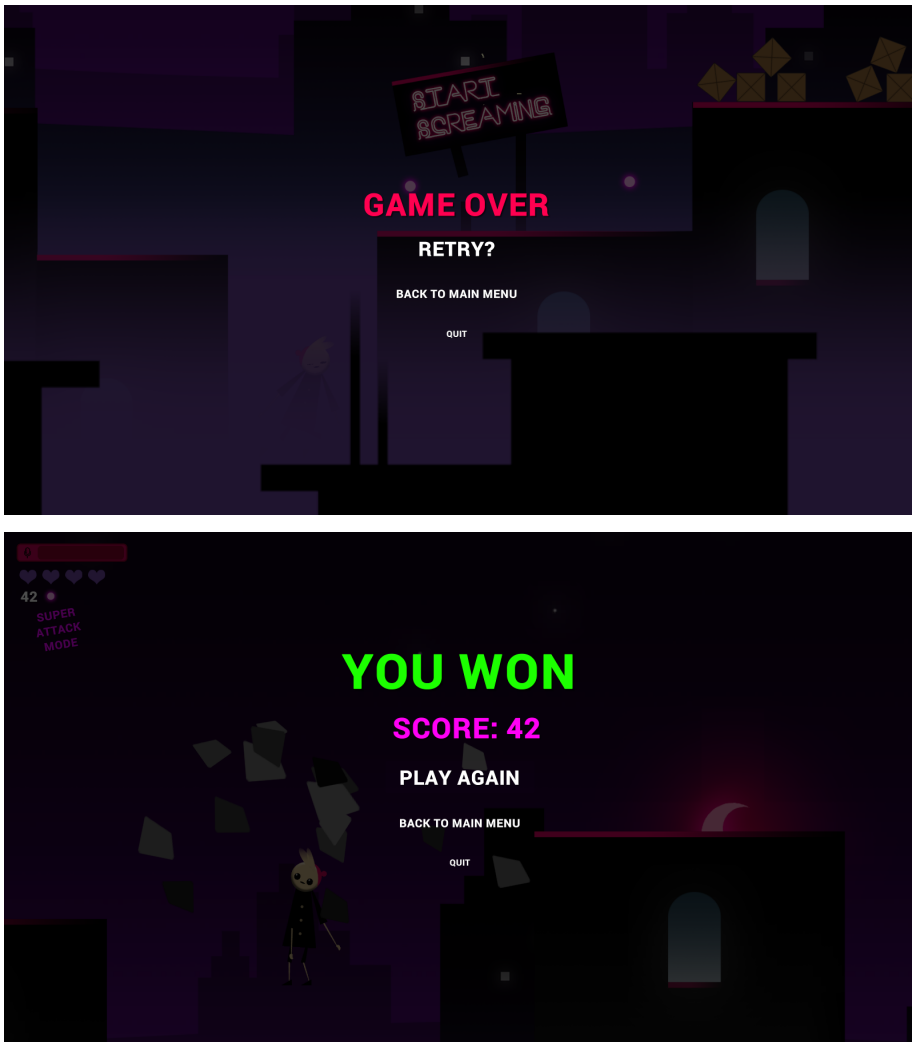

Abbildung 29: oben: Verlierermenü; unten: Gewinnermenü

2.4.10 Sounds

Hintergrundmusik

Bereits beim Start des Spiels wird man mit der Hintergrundmusik des Spiels vertraut. Es handelt sich dabei um ein kurzes Musikstück das sich ständig wiederholt. Angelehnt an alte, klassische Videospiele, ist die Musik monoton und beinhaltet nur elektronische Instrumente.

Hinweistöne und andere Soundeffekte

Neben der ständig spielenden Hintergrundmusik, werden bei gewissen Aktionen kurze Soundeffekte ausgelöst. Je nachdem ob der Hauptcharakter gerade springt, von einem Gegner berührt wird oder von einer Plattform in den Abgrund gefallen ist, sind un-

terschiedliche Sounds zu hören. Des weiteren werden Hinweistöne ausgegeben, wenn der Hauptcharakter Münzen und Herzen einsammelt. Hat der Hauptcharakter bereits 30 Münzen eingesammelt, wird zudem ein zusätzlicher Hinweiston ausgegeben, um zu signalisieren, das der „Super Attack Modus“ aktiviert wurde.

3 Technische Realisierung

Shadowboy wurde für den PC geschrieben. Aufgrund der übersichtlichen und intuitiven Arbeitsweise, wurde *Unity* als Game Engine verwendet. Die Grafik Designs wurden in *Adobe Photoshop* erstellt. Die erstellten Charakterdesigns, sowie die Hinweistafeln und Schilder, wurden anschließend in ein Animationsprogramm namens *DragonBones* importiert und weiterverarbeitet. Bei *DragonBones* handelt es sich um Open Source-Projekt für 2D-Skelettanimationen. Alle Knochen können vom Programm dynamisch gesteuert werden. Auch das Einstellen des Maßstabs und der Verzögerung der Animationszeit für einen einzelnen Knochen ist möglich. Somit können Frame für Frame Animationen erstellt für die unterschiedlichsten Charaktere erstellt werden. Alle im Spiel vorhandenen Sounds wurden in *Studio One* erstellt.

3.1 Grundsätzliche Informationen

3.1.1 Unity

Unity ist eine Entwicklungsumgebung für Programmierer, die es ermöglicht 2D oder 3D Spiele für die meisten gängigen Plattformen zu erstellen. Im nachfolgenden Teil werden kurz die nötigen Workflow-Konzepte besprochen.

Szenen

Szenen repräsentieren die einzelnen Levels des Spiels. Darin befinden sich alle Objekte, wie zum Beispiel die Umgebungen, Hindernisse und Charaktere, aus denen das Spiel aufgebaut ist.

Spielobjekte

Jedes Objekt in einer Szene, ist ein Spielobjekt. Von Charakteren und Sammelobjekten, bis hin zu Lichtern, Kameras und Spezialeffekten. Ein Spielobjekt kann jedoch nichts alleine machen. Sie benötigen Eigenschaften, bevor sie zu einem Charakter, einer Umgebung oder einem Spezialeffekt werden können.

Um einem Spielobjekt die Eigenschaften zu geben, die es benötigt, müssen Komponenten hinzugefügt werden. Abhängig davon, welche Art von Objekt man erstellen möchte,

gibt es verschiedene Komponentenkombinationen.

Man kann sich ein Spielobjekt als leeren Kochtopf und die Komponenten als die verschiedene Zutaten vorstellen, die das Rezept eines Spiels ausmachen. In Unity sind bereits viele verschiedene Komponententypen integriert, jedoch können, mit Hilfe von Skripts, eigene Komponenten erstellt werden.
[201a, vgl.]

Skripten

Das Verhalten von Spielobjekten wird von den Komponenten gesteuert, die an sie gebunden sind. Obwohl die in Unity integrierten Komponenten sehr vielseitig sind, benötigt man oft individuelle, um die eigenen Gameplay- Funktionen zu implementieren. Mit Hilfe von Skripten, kann man eigene Komponenten erstellen. Diese können zum Beispiel Spielereignisse auslösen, Komponenteneigenschaften im Laufe der Zeit ändern oder auf Benutzereingaben reagieren.

Unity unterstützt die Programmiersprache C#. Darüber hinaus können viele andere .NET-Sprachen mit Unity verwendet werden, wenn sie eine kompatible DLL kompilieren können.

Nach Erstellen einer Skript Datei im Projektordner, sieht der ursprüngliche Inhalt der Datei ungefähr so aus:

```
1  using UnityEngine;
2  using System.Collections;
3
4  public class MainPlayer : MonoBehaviour {
5
6      // Use this for initialization
7      void Start () {
8
9      }
10
11     // Update is called once per frame
12     void Update () {
13
14     }
15 }
```

Ein Skript stellt eine Verbindung zur internen Funktionsweise von Unity her, indem es eine Klasse implementiert, die von der integrierten Klasse namens MonoBehaviour abgeleitet ist. Man kann sich eine Klasse als eine Art Blaupause zum Erstellen eines neuen Komponententyps vorstellen, die an Spielobjekte angehängt werden kann. Jedes Mal, wenn eine Skriptkomponente an ein Spielobjekt anhängt wird, wird eine neue Instanz des, durch die Blaupause definierten, Objekts erstellt.

In der Update-Funktion befindet sich der Code für das Frame-Update des Spielobjektes. In dieser Funktion steht also alles, was im Laufe des Spiels mit der Zeit behandelt werden muss, wie etwa Bewegungen oder die Reaktion auf Benutzereingaben.

Damit die Update-Funktion funktioniert, ist es oft nützlich Variablen einzurichten, Einstellungen zu lesen oder Verbindungen zu anderen Spielobjekten herzustellen, bevor eine Spielaktion ausgeführt wird. Die Start-Funktion wird vor Beginn des Spiels aufgerufen (dh. bevor die Update-Funktion zum ersten Mal aufgerufen wird) und ist ein idealer Ort, um Initialisierungen vorzunehmen. [201b, vgl.]

3.2 Implementierung in Unity

3.2.1 Szene - Main Menu

Die Szene *Main Menu* beinhaltet alle Menüs, die nach dem Start der Applikation angezeigt werden. In der Szene wird darüber hinaus die Musiklautstärke festgelegt, als auch das Mikrofon kalibriert.

3.2.2 Spielobjekte und Skripten

Die wichtigsten, in der Szene *Main Menu* enthaltenen, Spielobjekte lauten:

- *Main Menu*
- *Instructions Menu*
- *Calibrations Menu*
- *Recorder*
- *SoundManagerMUSIC*

Jedes dieser Spielobjekte besitzt untergeordnete Spielobjekte, die für das Auslösen gewisse Aktionen zuständig sind.

SoundManagerMUSIC

Das Spielobjekt *SoundManagerMUSIC* ist im wesentlichen eine „Audio Source“. Eine Audio Source kann verschiedene Audioclips an einen davor erstellten Audiomixer weiterleiten. Sie ist also dafür verantwortlich, dass die Hintergrundmusik, sowie die Soundeffekte über die Lautsprecher ausgegeben werden.

Recorder

Recorder besitzt das Skript „Record.cs“. In dieser Szene wird der Teil des Skripts verwendet, der dafür verantwortlich ist, dass, aus den letzten vom Mikrofon erfassten Samples, die gemittelte Lautstärke ausgerechnet und in eine Variable gespeichert wird.

Main Menu und Instructions Menu

Dem Spielobjekt *Main Menu* sind vor allem Buttons untergeordnet, die für die Sichtbarkeit der verschiedenen Spielobjekte verantwortlich sind. Sie sorgen zum Beispiel dafür, dass das Anleitungsmenü erscheint, sobald auf den Button „Start“ gedrückt wird. Ebenfalls sind dem Spielobjekt *Instructions Menu* neben ein paar Textzeilen und Bilder, vorwiegend Buttons untergeordnet.

Calibrations Menu

Das Spielobjekt *Calibrations Menu* besitzt ebenfalls untergeordnete Spielobjekte, die dafür verantwortlich sind, dass der Benutzer die Lautstärke festlegen und das Mikrofon kalibrieren kann.

Mittels eines Schiebereglers ist es möglich, die Lautstärke der Hintergrundmusik anzupassen, da der Wert, welcher mit dem Schieberegler bestimmt wird, im Skript „OptionsMenu.cs“ den Wert der Lautstärke des zugewiesenen Mixers ersetzt.

Durch das Klicken des Buttons mit Aufschrift „START CALIBRATING“, werden mittels des Skripts „CalibrationButton.cs“, alle, vom Spielobjekt *Recorder* aufgenommenen Werte, in eine Liste eingetragen. Sobald man auf den Button gedrückt hat, ändert sich außerdem dessen Aufschrift zu „STOP CALIBRATING“. Es werden so lange die Werte in eine Liste eingetragen, bis der Button erneut gedrückt wird. Anschließend werden, mittels einer Schleife, der höchste und niedrigste Wert in der Liste ermittelt. Drückt man nun auf den Button „PLAY GAME“, startet die nächste Szene.

3.2.3 Szene - Level 1

In *Level 1* findet dann das eigentliche Spiel statt. Dort befindet sich die gesamte Spielwelt, sowie der Hauptcharakter und die Gegner.

3.2.4 Spielobjekte und Skripten

Die wichtigsten, in der Szene *Level 1* enthaltenen, Spielobjekte lauten:

- *Level*
- *Player*
- *Enemies*
- *BossEnemy*
- *Objects*
- *Bonus*
- *UI*
- *Sounds*
- *Menu*

Einige dieser Spielobjekte besitzen untergeordnete Spielobjekte, die für das Auslösen gewisse Aktionen zuständig sind.

Level

Im Spielobjekt *Level* befinden sich, in untergeordneten Spielobjekten, Elemente, wie zum Beispiel Plattformen oder Hinweistafeln, welche die Umgebung des Spielwelt aufbauen.

Damit etwa der Hauptcharakter nicht durch den Boden fällt, umspannen zahlreiche „Collider“ die Grafiken der Plattformen und machen sie dadurch betretbar.

Des Weiteren, sorgt das Skript „ParallaxLayer.cs“ dafür, dass sich Objekte, welche sich im Vordergrund befinden, mit einer anderen Geschwindigkeit bewegen, als Objekte, die sich im Hintergrund befinden.

Die Hinweistafeln, welche verteilt durch das Spiel zu finden sind, werden, wie alle DragonBones Skelette, durch das Skript „Unity Armature Component“ beschrieben. Es beinhaltet alle Eigenschaften, als auch die zur Verfügung stehenden Animationen des Skeletts.

Des Weiteren besitzen Hinweistafeln Partikeleffekte, die einen Funkenregen darstellen sollen. Die Schrift des zweiten Schildes, auf das man im Spiel trifft, bricht außerdem von der Tafel hinunter, sobald der Hauptcharakter einen gewissen Punkt erreicht hat. Ausgelöst wird diese Aktion durch das Skript „Sign2.cs“, welches dafür sorgt, dass, sobald der Hauptcharakter eine unsichtbare Box berührt, das „Unity Armature Component“ Skript die Animation wechselt.

Das Skript „object_in_frame.cs“, welches an das Spielobjekt *Moon* gebunden ist, sorgt dafür, dass der Mond sich, bei Bewegung des Hauptcharakters, in die gleiche Richtung bewegt. Dadurch bekommt der Spieler den Eindruck, dass der Mond sich an einem festen Platz befindet.

Player

Beim Spielobjekt *Player* handelt es sich um ein DragonBones Skelett, welches den Hauptcharakter darstellt. Die Eigenschaften des Skeletts sind ebenfalls wieder im Skript „Unity Armature Component“ zu finden. Außerdem wurden die Komponenten „Rigidbody 2D“ und „Capsule Collider 2D“ zum Spielobjekt hinzugefügt. Sie sorgen einerseits dafür, dass auf den Charakter die Schwerkraft wirkt und andererseits dafür, dass er nicht durch Plattformen fällt, sondern mit ihnen kollidiert.

Das Skript „animation_state.cs“, ist dafür verantwortlich, die im Skript „Unity Armature Component“ derzeit ausgeführte Animation bei Bedarf zu ändern. Insgesamt verfügt der Charakter über 5 Animationen: „Standing“, „Running“, „Jumping“, „Death“ und „Super-Attack“. Je nachdem, ob der Charakter also gerade etwa still steht, sich bewegt, hüpft, stirbt oder eine Superattacke ausführt, ändert das Skript also die dargestellte Animation.

Im Skript „Player.cs“ werden die Tastatureingaben verarbeitet, sodass sich der Hauptcharakter, beim Drücken der A oder D Taste, nach links, beziehungsweise rechts, bewegt oder, beim Betätigen der Leertaste, hüpft.

Enemies

Das Spielobjekt *Enemies* beinhaltet alle Standardgegner.

Gleich wie beim Hauptcharakter, handelt es sich bei den untergeordneten Spielobjekten um DragonBones Skelette. Diese besitzen die Komponenten „Rigidbody 2D“ und „Polygon Collider 2D“, damit auf sie die Schwerkraft wirkt und sie mit Plattformen kollidieren. Sie verfügen zwar über zwei Animationen, „Walking“ und „Death“, da aber immer nur die Erstere ausgeführt wird, benötigt man kein extra Skript zum Wechseln zwischen den Animationen.

Außerdem verfügen die Skelette über ein „Enemy.cs“ Skript, welches ihre Bewegungen definiert. Darüber hinaus wird darin festgelegt, dass, falls sie seitlich vom Hauptcharakter berührt werden, diesem ein Herz abziehen.

Das Skript „Enemy_Damage.cs“, ist dafür verantwortlich, dass, falls der Hauptcharakter auf den Kopf eines Standardgegners springt, diese zerstört werden. Dabei wird gleichzeitig ein Partikeleffekt initialisiert, der das Zersplittern der Gegner darstellen soll.

BossEnemy

Das Spielobjekt *BossEnemy* beinhaltet den Endgegner.

Der Endgegner wird ebenfalls wieder durch ein DragonBones Skelett dargestellt. Da sich der Endgegner nur auf und ab bewegt, benötigt er als Komponenten nur zwei „Polygon Collider 2D“ und kein „Rigidbody 2D“.

Das Skript „EnemyBoss.cs“ beschreibt neben der Bewegung des Endgegners, dass eine Berührung zwischen Hauptcharakter und Endgegner, zum Tod des Hauptcharakters führt.

Das Skript „EnemyDamageBoss.cs“, ist dafür verantwortlich, dass eine Superattacke, ausgeführt vom Hauptcharakter auf den Kopf des Endgegners, den Engegner zerstört. Dabei wird gleichzeitig ein Partikeleffekt initialisiert, der das Zersplittern des Endgegners darstellen soll.

„HomingMissle.cs“ beschreibt die Attacke des Endgegners. Es ist dafür verantwortlich, dass der Hauptcharakter, sobald dieser innerhalb einer gewissen Zone ist, von einem ein Feuerball verfolgt wird. Ebenfalls wird darin festgelegt, dass, falls der Feuerball mit einem Objekt kollidiert, ein Partikeleffekt ausgelöst und ein neuer Feuerball auf der linken Hand des Endgegners erzeugt wird.

Objects

Im Spielobjekt *Objects*, sind in den untergeordneten Spielobjekten alle lärmempfindlichen Hindernisse zu finden.

Wände, sowie Spezialgebäude sind DragonBones Skelette. Die Eigenschaften der Skelette sind wieder im Skript „Unity Armature Component“ zu finden. Über das Skirpt „animation_Wall.cs“, beziehungsweise „Animation_Building.cs“ werden die darzustellenden Animationen gesteuert. Grundsätzlich funktionieren beide Skripten so, dass, sobald sich der Hauptcharakter nahe genug an einem der lärmempfindlichen Hindernisse befindet

und der Grenzwert des Mikrofonpegels unter oder überschritten wird, eine Animation ausgelöst wird. Je nachdem, ob es sich bei dem Hinderniss um eine Wand oder ein Spezialgebäude handelt, wird gleichzeitig ein „Box Collider 2D“ ab oder aufgebaut. Nur mit dem richtigen Lautstärkepgel lassen sich die Hindernisse also überwinden.

Bonus

Das Spielobjekt *Bonus* beinhaltet, in untergeordneten Spielobjekten, alle Herzen und Münzen.

Die Skripten „BonusHeart.cs“ und „BonusCoin.cs“ funktionieren so, dass, sobald der Charakter das Objekt berührt, eine Aktion ausgeführt wird. Wird ein Herz aufgesammelt, erhöht das die Variable der derzeitigen Leben. Sammelt der Hauptcharakter eine Münze auf, erhöht das die Variable der Anzahl der derzeitigen Münzen. Anschließend wird in beiden Skripten das jeweilige Spielobjekt zerstört.

UI

Das Spielobjekt *UI*, ist für die Darstellung des HUD's verantwortlich.

Im Skript „MicBar.cs“, wird festgelegt, dass der Balken die Änderung, der vom Mikrofon aufgenommenen Lautstärke, anzeigt. Die Füllung des Balkens wird durch eine Variabel bestimmt die von 0 bis 1 reichen kann.

Die Skripten „CoinsUI.cs“ und „HeartUI.cs“ sind dafür verantwortlich, dass die derzeitige Anzahl von Münzen, beziehungsweise Herzen, angezeigt wird. Darüber hinaus, ist das Skript „CoinsUI.cs“ dafür verantwortlich, dass ab einer Anzahl von 30 Münzen, die Modusanzeige aktiviert wird.

Sounds

Im Spielobjekt *Sounds* findet man, ähnlich wie in der Szene *Main Menu*, ein untergeordnetes Spielobjekt, dass als Audio Source fungiert. Darüber hinaus befindet sich auch das untergeordnete Spielobjekt Recorder mit dem „Record.cs“ Skript darin.

Der höchste und der niedrigste Kalibrationswert, aus der vorangegangenen Szene, wird nun dazu verwendet, um zu erkennen, ob die Geräusche, die derzeit vom Mikrofon aufgenommen werden „laut“oder „leise“ sind. Dazu wird nun in einem anderen Teil des Skripts „Record.cs“ die, aus den Samples errechneten Lautstärkenwerte, auf eine Skala von 0 bis 1 normiert. Das bedeutet, sobald das Mikrofon nun ein Geräusch aufnimmt, dass mindestens dem höchsten Kalibrationswert entspricht, wird als Lautstärkenwert eine 1

ausgegeben. Liegt das aufgenommene Geräusch hingegen unter dem niedrigsten Kalibrationswert, wird eine 0 ausgegeben. Für alle Geräusche, die zwischen dem Maximal- und Minimalwert der Kalibration liegen, werden Werte von $0 < 1$ ausgegeben. Der Grund für die Normierung der Werte liegt darin, dass, wie im Abschnitt UI erwähnt, die Füllung des Balkens, nur Werte von 0 bis 1 annehmen kann. Um nun optisch zu repräsentieren, wie hoch die Lautstärke eines Geräusches ist, kann nun der normierte Lautstärkenwert, die Füllung des Balkens angeben.

Menu

Das Spielobjekt *Menu* beinhaltet alle Menüs, die während des Spielens angezeigt werden können. Sie bestehen grundsätzlich alle aus Textfeldern und Buttons, die mit verschiedenen Funktionen belegt sind.

4 Fazit und Zukunft

Mit *Shadowboy* wurde ein Audio Game kreiert, dass sich, durch die Verwendung eines Mikrofons und Geräuschen zur Erweiterung der herkömmlichen Steuerelemente, von den bekannten Videospielen abhebt. Es handelt sich beim Spiel trotzdem immer noch eher um ein „Proof of Concept“, da es für ein vollwertiges Spiel zu wenig Abwechslung bietet. Weitere Level, komplexere Spielmechaniken oder vielfältigere Gegner zu implementieren, wären ein nächster Schritt, um es zu einem vollwertigen Spiel zu machen.

Im Kern ist das Spiel durch simple Regeln aufgebaut, die selbst unerfahrene Spieler schnell verstehen. Die ersten Praxistest zeigten jedoch, dass der hohe Schwierigkeitsgrad sich für viele Personen als frustrierend herausstellte. Obwohl das Spiel dahingehend anschließend zwar angepasst wurde, stellte es immer noch eine große Herausforderung für einige Spieler dar. Der Grund dafür, dass der Schwierigkeitsgrad nicht weiter herabgesetzt wurde, ist, dass das Spiel absichtlich eine gewisse Herausforderung darstellen sollte, um das Erfolgserlebnis, nach Beendigung des Levels, größer ausfallen zu lassen. Eine weitere Idee für die Zukunft wäre, dass man die Fähigkeiten des Hauptcharakters nach jedem gescheiterten Versuch, mit den bis dato gesammelten Münzen, aufwerten kann. Somit wäre der Spieler motivierter das Level nach einem gescheiterten Versuch zu wiederholen, um den Hauptcharakter weiter zu verstärken, sowie die verbesserten Fähigkeiten auszuprobieren.

Das Spiel wurde zwar für Desktop PCs und Laptops hergestellt, es bietet sich im Nachhinein jedoch vor allem für mobile Devices an. Die Tests zeigten nämlich, dass, speziell bei Laptops, die Tastengeräusche teilweise vom Mikrofon aufgenommen wurden, falls das Spiel ohne externes Mikrofon betrieben wurde. Würde man das Spiel auf einem Tablet oder Smartphone spielen, gäbe es dieses Problem nicht.

Während der Entwicklungsphase kam auch die Idee eines Multiplayer Modus auf, die dann jedoch wieder verworfen wurde. In zukünftigen Versionen des Spiels könnte man den Modus jedoch verwirklichen, da sich damit zahlreiche neue Möglichkeiten für Levels und Funktionen ergeben würden. Besonders gut geeignet wäre ein Koop-Modus, in welchem die Spieler gemeinsam laut oder leise müssen, um die Levels zu überwinden. Mit einem derartigen Modus würde sich das Spiel besonderes für Parties eignen.

Unity und C# sind einfach zu erlernende Werkzeuge, mit denen sogar ein kleines Entwicklerteam, spannende Ideen verwirklichen kann. Ich persönlich startete das Projekt ohne Vorwissen und durch zahlreiche Foren und Internetvideos, gelang es mir, mich in beiden Umgebungen zurecht zu finden. Unity, als auch C#, haben eine große Menge an vorgefertigten Funktionen, die es dem Entwickler erleichtern verschiedene Ideen auszuprobieren und auch Zeit zu sparen. DragonBones bietet den Vorteil, dass es gratis zur Verfügung steht, sowie unkompliziert in Verbindung mit Photoshop und Unity arbeitet. Das einzige Hindernis an DragonBones war, die teils verwirrende Dokumentation. Grundsätzlich bin ich mit der Endversion des Spiel zufrieden, das einzige was ich jedoch im Nachhinein ändern würde, wäre, das Spiel von Beginn an für Smartphones und Tabletts zu entwickeln. Der Grund hierfür ist, dass man kein externes Mikrofon zur Behebung des Problems der Tastengeräusche benötigen würde. Außerdem bin ich der Ansicht, dass speziell bei einem zukünftigen Multiplayer Modus, Smartphones und Tablets im Vorteil wären. Da die meisten Personen heutzutage ein Smartphone bei sich tragen, wäre es möglich einen Multiplayer Modus mit mehr als 2 Leuten kreieren, bei welchem die Handys mittels Bluetooth gekoppelt werden. Auf diese Art hätte jeder Spieler auch automatisch sein eigenes Mikrofon im Smartphone verbaut. Natürlich wäre es auch möglich mehrere Laptops, zum Beispiel mittels Bluetooth, zu koppeln, doch bestünde dann das Problem der Tastengeräusche und im direkten Vergleich, würde wesentlich mehr Platz in Anspruch genommen werden, als bei Smartphones oder Tablets.

Literatur

[201a] U. T. P. 2019.1-002V., "Unity documentation," https://docs.unity3d.com/Manual/GameObjects.html, 20.06.2019.

[201b] ——, "Unity documentation," https://docs.unity3d.com/Manual/CreatingAndUsingScripts.html, 20.06.2019.

[ac] atari computermuseum.de, "Die geschichte von atari," http://www.atari-computermuseum.de/history.htm, 17.09.2019.

[Bli] A. Blizzard, "World of warcraft spielerzahlen," https://de.statista.com/statistik/daten/studie/208146/umfrage/anzahl-der-abonnenten-von-world-of-warcraft/, 25.09.2019.

[cho] chordian.net, "World of warcraft photos," http://chordian.net/2017/05/20/my-endeavors-in-world-of-warcraft-part-3/, 28.04.2019.

[Fis] S. Fisher, "Play the original pacman online..." https://www.lifewire.com/play-the-original-pacman-online-for-free-1357974, 26.04.2019.

[fU] Ö. V. für Unterhaltungssoftware, "Fast 5 millionen österreicher spielen videogames," https://www.ovus.at/news/fast-5-millionen-osterreicher-spielen-videogames/, 25.09.2019.

[Gra] L. Grace, "The original space invaders..." https://www.smithsonianmag.com/science-nature/original-space-invaders-icon-1970s-America-180969393/, 26.04.2019.

[Hub17] C. Huber, *Das kindliche Spiel und seine Bedeutung für das elementarpädagogische Handeln*, 2017.

[Hui38] J. Huizinga, *Homo Ludens*, 1938.

[igc] igcritic.com, "Super mario screenshot," http://igcritic.com/blog/2016/04/13/best-classic-nintendo-games-n64-and-earlier/, 27.04.2019.

[imd] imdb.com, "Supermario64 photos," https://www.imdb.com/title/tt0204657/mediaindex?ref_=tt_mv_close, 28.04.2019.

[lg] lurking game.com, "Lurking website," https://www.lurking-game.com/, 17.09.2019.

[Mak] B. Makedonski, "Review: Pinstripe," https://www.destructoid.com/review-pinstripe-490354.phtml, 15.05.2019.

[Oer97] R. Oerter, *Psychologie des Spiels*, 2nd ed., 1997.

[Sch97] H. Scheuerl, *Das Spiel - Theorien des Spiels*, 12th ed., 1997.

[sG] statcounter GlobalStats, "mobile geräte benutzerzahlen," https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/worldwide/#monthly-201001-201512, 25.09.2019.

[ulu] ulule.com, "A blind legend photo," https://de.ulule.com/a-blind-legend/, 17.09.2019.

[Web] K. Webb, "A rare copy of 'super mario bros.' sold..." https://www.businessinsider.de/most-expensive-video-game-ever-sold-super-mario-bros-2019-3?r=US&IR=T, 26.04.2019.