Studien- und Prüfungsabteilung
Leonhardstraße 15, A-8010 Graz
Tel: 0043(0)316/389-1310, -1313, Fax: 0043(0)316/389-1311
Mail:studienabteilung@kug.ac.at

Bitte deutlich leserlich ausfüllen!

Deckblatt einer

wissenschaftlichen Bachelorarbeit

Vor- und Familienname	Matrikelnummer
Michael Reiter	01423196
Studienrichtung	**Studienkennzahl**
Elektrotechnik-Toningenieur	213

Thema der Arbeit:

Ear Training Adventure - Design und Umsetzung eines Audio Games

Angefertigt in der Lehrveranstaltung: Computermusik und Medienkunst
(Name der Lehrveranstaltung)

Vorgelegt am: 07.09.2019
(Datum)

Beurteilt durch: Johannes Zmölnig, Dipl-Ing.
(LeiterIn der Lehrveranstaltung)

Universität für Musik und darstellende Kunst Graz

Institut für Elektronische Musik und Akustik

Ear Training Adventure

Design und Umsetzung eines Audio Games

Bachelorarbeit zum Erlangen des akademischen Grades
„Bachelor of Science“

von
Michael Reiter

Betreuer
Johannes Zmölnig, Dipl.-Ing.

15. September 2019

Kapitel 1

Abstract

Diese Bachelorarbeit entstand im Zuge des Seminars „Computermusik und Medienkunst“ des Instituts für Elektronische Musik der Universität für Musik und darstellende Kunst Graz. Das zentrale Thema dieses Seminars war der Begriff des „Audio Games“. Diese Art von Spiel bildet eine Unterkategorie der digitalen Spiele. Im Kontrast zu Videospielen lenken sie den Fokus vom Bildschirm und visuellen Informationen fort und stellen das Gehör in den Mittelpunkt, sodass Audio über die sonst übliche Rolle zur Untermalung der Stimmung hinauswächst. Ein solches digitales Spiel, das Audio ins Zentrum stellt, sollte als Prototyp konzipiert, designt und umgesetzt werden. Anlass für das so entstandene Spiel „Ear Training Adventures“ gaben die im Studienplan Elektrotechnik-Toningenieur enthaltenen Lehrveranstaltungen, die für ein positives Absolvieren, ein gut trainiertes musikalisches Gehör voraussetzen. Im Umfeld von Studierenden, die derartige Unterrichtsfächer frequentieren, hört man regelmäßig von Schwierigkeiten mit diesen Fächern und Problemen sich effizient auf deren Prüfungen vorzubereiten. Die Möglichkeit die Thematik der Gehörbildung in ein Spiel zu verpacken, bietet die Chance den Lernaufwand mit der heiteren Natur von Computerspielen zu verbinden um Anreiz für gezieltes Üben zu schaffen. Das Ziel sei ein fesselndes Spiel zu kreieren, bei dem man „nebenbei“ ein musikalisches Gehör entwickelt. Die folgende Arbeit befasst sich zuerst mit den Begriffen „Spiel“ und „sinnvolles Spielen“ und diskutiert deren Umsetzung als „Audio Game“. Zusätzlich soll ein Einblick in die zugrundeliegende Theorie der Gehörschulung gegeben werden inklusive eines Exkurses in die Musiktheorie. Zuletzt soll der Schaffensprozess des „Audio Games“ beschrieben werden. Von der Idee, über die Konzeptionierung der Regeln und der Spielwelt bis hin zur finalen Umsetzung in Unity und den damit auftretenden Herausforderungen und Problemen.

This bachelor thesis was written in the context of the seminar „Computermusik und Medienkunst“ by the „Institut für Elektronische Musik“ at the University of Music and Performing Arts Graz. The central topic discussed in this seminar was „audio games“. This kind of game forms a subgenre of digital games. In contrast to video games they tend to focus less on visual components and emphasize on auditive perception. Thus the role of sound, which in most instances is limited to setting the mood in a game, becomes an essential element of the experience. Motive for creating the audio game „Ear Training Adventures“ were courses in the curriculum of „Electrical Engineering and Audio Engineering“ which require students to develop a musical sense of hearing. Preparing for exams in an efficient way is often a difficult task according to students attending those programs. The possibility to implement ear training exercises in the domain of computer games is a chance to combine learning and training with the joyful elements of a computer game. The aim is to create a compelling experience through which the users ear developes the desired skills as a byproduct.

The theoretical background of this thesis consists of the discussion of the terms „game“ and „meaningful playing or gaming“ which are also put in the context of video games. Furthermore a glimpse into the theory of ear training and music theory is given. The main part of the thesis describes the process of designing and creating the game itself. First the concept and the design of general rules and goals are introduced. Then the implementation in the game engine „Unity“ is depicted and eventually the ocurring challenges and problems are examined and some possible ideas for further development are listed.

Inhaltsverzeichnis

Kapitel 2

„Spiel“ und „spielen“

2.1 Definition

Zuerst gilt es den Begriff des Spiels und dessen Zusammenhang oder Unterschied zum „Spielen“ zu klären. Zuerst soll das Spiel von dem ziellosen und relativ regellosen „spielen“, welches etwa Kinder oder Tiere betreiben, abgegrenzt werden. Dieses erfüllt nicht die grundlegenden Kriterien, nach denen hier ein Spiel definiert wird. Als Ausgangspunkt für eine anwendbare Definition wird auf den niederländischen Kulturhistoriker Johan Huizinga verwiesen, der in seinem Werk *„Homo Ludens: Vom Ursprung der Kultur im Spiel“* das Spiel folgendermaßen festlegt:

> *Spiel ist eine freiwillige Handlung oder Beschäftigung, die innerhalb gewisser festgesetzter Grenzen von Zeit und Raum nach freiwillig angenommenen, aber unbedingt bindenden Regeln verrichtet wird, ihr Ziel in sich selber hat und begleitet wird von einem Gefühl der Spannung und Freude und dem Bewusstsein des “Andersseins” als das “gewöhnliche Leben”.*

- Zum einen ist zwar unerlässlich, dass es sich beim Spiel um eine freiwillig eingegangene Situation handelt, es ist für diese Definition eines Spiels aber auch notwendig, dass bindende Regeln vorhanden sind. Diese Regeln sind im klassischen Spiel immer und von allen teilnehmenden SpielerInnen zu befolgen. Sie definieren für alle TeilnehmerInnen den jeweils möglichen Handlungsrahmen und leiten den Verlauf des Spiels.

- Dieses hier beschriebene System hat im Gegensatz zu „kindlichem spielen“ auch ein festgelegtes und quantifizierbares Ziel, auf das die AkteurInnen hinarbeiten sollen um es erfolgreich zu beenden.

- Die nächsten wichtigen Merkmale eines Spiels befassen sich auch mit der Frage des „warums“. Hier führt Huizinga nämlich das Gefühl der Spannung und Freude und dem Bewusstsein des Andersseins als das gewöhnliche Leben an. Es gibt also eine wichtige emotionale Ebene beim Spielen. Hier kann man also noch einmal stark abgrenzen von reinen „Lernspielen“ oder Simulationen, deren „Sinn“ nicht in Emotionen, sondern eher im praktischen Erfolg liegt. Da sich solche Ergebnisse wiederum auf das „gewöhnliche Leben“ auswirken liegt es wieder nahe, den Begriff des Spiels hier nicht anzuwenden. Ebenso kann man mit dieser Definition eine Grenze zu Wettbewerben im Leistungssport ziehen, da sich unter anderem ein Sieg massiv auf die Spieler auswirken kann und die Vorbereitungen und Wettkämpfe auch als Schritt weg vom lustvollen Spiel, hin zur „ernsten“ Arbeit gesehen werden können.

2.2 Sinnvolles Spielen

In diesem Punkt soll es nicht um den emotionalen und bereichernden „Sinn“ des Spiels gehen, sondern um die sinnvolle Interaktion eines oder mehrerer SpielerInnen mit dem System. Aus der Beziehung von Handlungen eines Spielers oder einer Spielerin und der Reaktionen des Systems entsteht, was hier als „sinnvolles Spielen“ definiert wird. Denn ohne die Teilnahme von SpielerInnen und deren Entscheidungen, welche den Verlauf des Spiels maßgebend beeinflussen, kann es kein sinnvolles Spielen geben. In einem Spiel wie „Snakes and Ladders“ bei dem Würfeln die einzig mögliche Aktion ist, ist dies beispielsweise nicht möglich, da die Handlung des Spielers rein zufällig ist und er keinen Einfluss auf das Ergebnis und den Verlauf des Spiels nehmen kann. Allein die Tätigkeit des Würfelns erweckt den Schein einer Handlung, die die Partie beeinflusst. Kurz in dem Wort Interaktivität zusammengefasst versteht man hier den Prozess vom Wahrnehmen der Situation, über das Tätigen einer Aktion bis zur Reaktion des Systems und deren Darstellung. Beginnend mit dem Wahrnehmen der Situation ist hier die Kommunikation der Situation und der Regeln maßgebend. Kann der/die SpielerIn erkennen in welcher Situation er/sie sich befindet und einschätzen welche Konsequenzen seine/ihre Handlung haben wird?

Hat er/sie einen Mangel an Informationen, kann eine Aktion eine beliebige Reaktion hervorrufen. Besteht das ganze System aus beliebigen Reaktionen ist kein Sinn mehr gegeben. Tätigt ein/e SpielerIn nun eine Handlung, so soll das System sinnvoll reagieren. Dies passiert genau dann, wenn der Zusammenhang zwischen Handlung und Ergebnis nachvollziehbar ist. Also weder eine für den/die SpielerIn „unsichtbare“, noch eine „sehr komplexe“

und nicht nachvollziehbare Reaktion ist sinnvoll. Denn in beiden Fällen wird der/die SpielerIn in dem Glauben gelassen, seine Handlungen haben entweder keine oder gänzlich beliebige Konsequenzen. Um zu gewährleisten, dass ein nachvollziehbarer Zusammenhang zwischen Ursache und Wirkung vorliegt, muss das Spiel die Auswirkungen auch adäquat kommunizieren und den neuen Kontext für weitere Entscheidungen liefern. Dieses Zusammenspiel kann einerseits auf der Mikroebene als sofortige Rückmeldung erkennbar sein, Handlungen sollen jedoch auch im größeren Kontext Auswirkungen haben. Die vorhergegangenen Entscheidungen sollen im späteren Spielverlauf nicht irrelevant werden, da diese Entscheidungen sonst beliebig hätten ausfallen können, was wiederum dem Grundsatz für sinnvolles Spielen widerspricht. Da sich das Spiel auf die AkteurInnen und deren Entscheidungen anpasst und verändert, ist jedes Spiel einzigartig und die Erfahrungen auch für jede Person individuell.

2.3 Computerspiele

Mit dem Voranschreiten der Rechenleistung von Computersystemen fand in der zweiten Hälfte des 20. Jahrhunderts die Spielewelt auch Einzug in die digitale Domäne. Mit der digitalen Datenverarbeitung ergab sich eine ganz neue Art der Interaktion und des Spieldesigns:

2.3.1 Video Games

Auch wenn sich die direkten Handlungsmöglichkeiten in diesem Bereich meist noch auf wenige Tastendrücke reduzieren lassen, so ergeben sich doch schnell komplexe Systeme. Zum einen können nun, im Gegensatz zu den meisten rundenbasierten Brettspielen, Befehle ununterbrochen und in Echtzeit angenommen werden (ständiges Schießen und Ausweichen beim Arcade-Klassiker „Asteroids“ – der Zeitdruck liefert einen zusätzlichen Spannungsfaktor). Zusätzlich wurden „NPCs“ (= Non Player Characters)also computergesteuerte Gegenspieler eingeführt. Man benötigt für viele dieser Spiele keine tatsächlichen MitspielerInnen, was die Beliebtheit des Videospiels als Freizeitgestaltungsmöglichkeit enorm begünstigt.

Da sich nun das gesamte Regelwerk in Form von „Code“ im Spiel selbst befindet, ist zum einen kein Schummeln oder abändern der Regeln mehr möglich. Zum anderen ergibt sich eine neue Art den/die SpielerIn ohne Anleitung und Regelbuch an das Spiel heranzuführen. Viele Interaktionen muss der/die SpielerIn erst während dem Spielen selbst herausfinden. Sehr komplexe Zusammenhänge können hier im Hintergrund verarbeitet werden, ohne

dass dem/der SpielerIn diese bewusst gemacht werden müssen.

Mit weiter steigender Rechenleistung wurden dreidimensionale, physikalisch realistische, komplexe Welten erschaffen und Interaktion mit anderen SpielerInnen über das Internet ermöglicht. Diese und viele weitere technische Innovationen machten das Videospiel zu einer der gängigsten Freizeitgestaltungsmöglichkeiten in der heutigen Zeit. Heute kann man den Begriff Videospiel, wie folgt definieren: Ein auf elektronischen Logikschaltungen basierendes Spiel, das den SpielerInnen eine Form der Interaktion ermöglicht und diesen den Spielzustand und Informationen visuell vermittelt. Obwohl in dieser Definition keine Rede von Audio ist, so ist es doch in beinahe jedem Videospiel in gewisser Weise vorhanden, sei es als Hintergrundmusik oder als Geräuschkulisse. Hier ist es wichtig festzustellen, dass die Audiokomponente in klassischen Videospielen wenig bis gar keine spielrelevanten Informationen enthält und meist nur zur Untermalung der Stimmung dient. Mit dem Begriff des „Audio Games“ soll nun ein neuartiges Genre vorgestellt werden, das den Fokus stark auf den auditiven Bereich legt und teils sogar gänzlich auf visuelle Unterstützung verzichtet.

2.3.2 Audio Games

Hier wird sich von dem Begriff der Videospiele etwas distanziert und Spiele beschrieben, die ohne Audio nicht spielbar wären. Da die Videospielwelt jedoch schon sehr viele bewährte Konzepte für ein erfolgreiches Spiel hervorgebracht hat, wird meist auch nicht ganz auf die visuelle Domäne verzichtet. Man kann nun zwischen verschiedenen Arten von Audio Games unterscheiden. In einigen Videospielen wird bereits Audio als subtiles Mittel zur Kommunikation von Informationen verwendet. Beispielsweise kann sich das Auftauchen eines Gegenspielers oder einer Gegenspielerin durch das Ändern der Geräuschkulisse ankündigen oder man kann sogar mithilfe der 3D-Audio-Technologie erkennen aus welcher Richtung die nächste Gefahr kommt. Hier liegt der Fokus immer noch auf dem Bildschirm. Je weiter man die Informationsvermittlung jedoch auf die Lautsprecher verlagert, desto mehr geht man in Richtung eines reinen Audio Games bis man schlussendlich beim „Accessible Game“ landet. Hierbei handelt es sich um Spiele, die für Sehbehinderte problemlos spielbar sind, da ihre visuelle Oberfläche nicht vorhanden oder zumindest redundant ist.

Implementation und Beispiele

- Den Anfang machte 1974 das Spiel „Touch Me“ von Atari, bei dem man eine eigene Spielekonsole mitgeliefert bekam. Der/die Benutze-

rIn bekam eine Melodie vorgespielt, die er/sie sich mit visueller Unterstützung merken und über die Tasten wiedergeben sollte.

- Seitdem hat sich dank des technischen Fortschritts viel verändert und komplexere digitale Accessible Games wie „The Blind Swordsman“ wurden möglich. Hierbei handelt es sich um ein „Mobile Game“ ohne visuelle Oberfläche, bei dem man in den Charakter eines Schwertkämpfers schlüpft und durch die Erzählungen und Konversationen der Charaktere den Spielverlauf vermittelt bekommt. Sowohl der Bewegungsablauf als auch der Kampfmodus sind sehr intuitiv über den Touchscreen zu bedienen. Orientierung bieten wiederum rein akustische Hinweise. Man kann hören woher Stimmen kommen und folgt diesen. Man hört einen Bären brüllen und weiß, dass man zur Verteidigung das Schild verwenden muss. Man hört den eigenen Herzschlag schneller werden und erkennt, dass man Gefahr läuft zu sterben. So viel Information, die man gewohnt ist visuell zu erhalten, lässt sich also durchaus akustisch verpacken.

- Einen weiteren Aspekt liefern Spiele wie Singstar oder „Lurking“. Bei dem angesprochenen Punkt handelt es sich um das Implementieren von Audio als Input beziehungsweise Steuerungsmöglichkeit. Bei Lurking sieht man quasi „mit den Ohren“ beziehungsweise durch Klänge und Geräusche. Schreit man ins Mikrofon, so wird kurzfristig visuell mehr von der Spielwelt offenbart und man kann sich orientieren. Ist man jedoch zu laut, macht man die herumwandernden Zombies auf sich aufmerksam.

- Eine andere Herangehensweise liefern kunstorientierte Projekte und spielerische Klanginstallationen wie beispielsweise Elektroplankton. Da diese „Spiele“ jedoch einige der zuvor besprochenen Kriterien nicht erfüllen werden sie hier auch nicht weiter behandelt.

2.4 Ear Training Adventure

Das im Zuge dieses Bachelorsemninars erstellte „Ear Training Adventure“ wurde konzeptioniert um eine Hilfestellung für (zukünftige) Studierende, die sich auf Fächer wie „Gehörschulung“ oder einen Aufnahmetest an einer Kunstuniversität vorbereiten möchten, zu bieten.

2.4.1 Complete Ear Trainer

Auditive Übung statt Audio Game

Da diese Art von musikalisch geschultem Gehör in vielen musikalischen Bereichen und Ausbildungen gefordert wird, sind natürlich bereits einige Apps und Programme auf dem Markt und bieten durchaus die Möglichkeit gezielt und effizient zu üben. In diesem Zusammenhang sei die App „Complete Ear Trainer“ als vorbildhaftes Programm angeführt. Mit dieser Applikation kann man viele der gewünschten Übungen in verschiedensten Schwierigkeitsgraden durchführen und auch selbst Aufgaben erstellen. Auch wenn es Belohnungen in Form von Achievements für langes und erfolgreiches Üben gibt, kommt beim Trainieren keinesfalls ein „Gefühl der Spannung und Freude und dem Bewusstsein des Andersseins als das „gewöhnliche Leben“ auf, was für Huizinga jedoch essentiell für ein Spiel ist. Bezieht man sich auf Carolyn Handler Miller (*Digital Storytelling: A Creator's Guide to Interactive Entertainment*), so können Lernerfolge sogar effektiver gestaltet werden, wenn die eigentlich trockene Lernsituation als Spiel dargeboten wird.

2.4.2 Das „RPG“ Genre als erfolgreiches Vorbild

Um diesem ungerichteten und wenig erfüllenden freien Üben einen Kontrapunkt zu setzen sollen die Aufgabenstellungen in die fesselnde und spannende Umgebung eines „Role Play Games“ eingebracht werden. In diesem Genre wird die Rolle eines fiktiven Charakters eingenommen, durch dessen Augen dann eine mehr oder weniger vorgegebene Handlung erlebt und durchgespielt wird. Konkret wurde mit „Pokémon“ eines der beliebtesten Spiele dieses Genres als Vorbild gewählt. Mit Millionen an AnhängerInnen haben sich die Spielmechaniken mehr als bewährt.

Wie in dem Klassiker aus Japan, soll es eine 2-Dimensionale Spielwelt in „Top-Down“ Ansicht geben, in der man sich relativ frei bewegen, Gegenstände sammeln und mit „NPCs“ interagieren kann. Die zentrale Rolle des Spiels nimmt jedoch das Kampf- und Erfahrungssystem ein. Denn hier sollte schon bei Pokémon der Charakter durch ständiges, repetitives Kämpfen und Siegen Erfahrung sammeln. Dadurch konnte man höhere Levels erreichen, stärker werden und schlussendlich einen schwierigeren „Endgegner“ besiegen um in der Handlung voranzuschreiten. Genau dieses ständige Kämpfen mit den gleichen Gegnern entspricht dem, von Lehrenden empfohlenen Gehörtraining. Der richtige Weg ein gut geschultes Gehör zu entwickeln sei demnach durch regelmäßiges Wiederholen derselben Übungen. Da der Weg ein ähnlicher ist und das konstante „Grinden“ (= repetitive Aufgaben bestreiten um Erfahrungspunkte zu sammeln und dadurch im Spiel

voranschreiten zu können) bei den Pocket Monsters kein Hinderungsgrund für dessen Erfolg war, lag die Idee nahe, ein ähnliches Umfeld für das „Ear Training Adventure“ zu schaffen. Dabei galt es nach *Marc Prenskys „Digital game-based learning“* folgende Kriterien zu beachten um ein effektives Lernspiel zu gestalten:

- Erzeugung eines hohen Maßes an intrinsischer Motivation
- Vorgabe klarer Ziele und Regeln
- Bereitstellung eines reichhaltigen und gleichzeitig angenehmen Kontexts
- Einbindung einer „packenden“ Storyline mit Überraschungselementen
- Gabe unmittelbaren Feedbacks
- hoher Level der Interaktivität, Herausforderung und des Wettstreits

Es soll eine Balance gefunden werden zwischen erfolgreichem Lernen und fesselndem Spielspaß. Liegt der Fokus zu sehr auf einer Seite, ist das Ergebnis entweder ein reines Spiel ohne Lernfortschritt oder ein reines Übungsprogramm ohne den Freuden eines Spiels.

Kapitel 3

Gehörbildung und Musiktheorie

Bei der Gehörbildung oder Gehörschulung handelt es sich um den Prozess, in dem das Gehör darauf trainiert wird musikalische Details, wie beispielsweise Tonhöhenverlauf, Rhythmus und Harmonik, zu erkennen. Mit diesen Fertigkeiten soll es schlussendlich möglich sein, musikalische Stücke allein durch das Hören einer Aufnahme zu transkribieren. Gefordert und gebraucht wird ein solches Gehör in beinahe allen höheren musikalischen Ausbildungen und Berufen, denn es ermöglicht nicht nur das Verschriftlichen von Musik, sondern erhöht auch das Verständnis und Gefühl für die gespielten oder gehörten Klänge. So können MusikerInnen, DirigentInnen, LehrerInnen aber auch ToningenieurInnen die Qualität ihrer Arbeit massiv verbessern. In der hier besprochenen Gehörbildung geht es ausschließlich um relatives Gehör, also darum den Zusammenhang zwischen den Tonhöhen der gehörten Noten zu erkennen und nicht den absoluten Notenwert. Erkennt man den Melodieverlauf kann man einen beliebigen Notenwert als Startwert annehmen. Erhält man später die tatsächliche Tonhöhe zu Beginn des Stücks, beispielsweise durch die Kontrolle mit einem Instrument oder einer Stimmgabel, muss man die Melodie nur mehr richtig transponieren, da der relative Verlauf bestehen bleibt. Absolutes Gehör, also das Erkennen der Tonhöhe ohne Kontext ist sehr selten . Ab einem gewissen Alter gilt es als kaum bis nicht mehr antrainierbar. Deshalb wird in dieser Arbeit auch nicht näher darauf eingegangen. Mit dem Fokus auf dem musikalischen Schaffen der westlichen Welt wird mit der Unterteilung einer Oktave in 12 Halbtöne gearbeitet. Weitere Tonsysteme werden hier nicht diskutiert. Zusätzlich wird, wenn von tonalem Hören gesprochen wird, in erster Linie auf das Dur/Moll-System Bezug genommen während beispielsweise andere Kirchentonarten vernachlässigt werden. Zusätzlich sei angemerkt, dass Rhythmus eine wichtige Rolle in der

Gehörbildung einnimmt. Diese Arbeit beschränkt sich jedoch auf den melodischen und harmonischen Aspekt.

3.1 Intervallisches Hören

Wie bereits erwähnt wird die Oktave in 12 Töne unterteilt. In der folgenden Tabelle werden die Intervalle zum Bezugston „C“ durch die Anzahl ihrer Halbtonschritte die sie von diesem entfernt sind definiert:

Note	Halbtöne	Intervall
C	0	Unison
C#	1	kleine Sekund
D	2	große Sekund
D#	3	kleine Terz
E	4	große Terz
F	5	reine Quart
F#	6	übermäßige Quart/ verminderte Quint
G	7	reine Quint
G#	8	kleine Sexte
A	9	große Sexte
A#	10	kleine Septime
H	11	große Septime

Für *Roland Mackamul* ist es unerlässlich intervallisches Hören gleichzeitig und als als gleichwertig mit dem, auf unser Dur/Moll-System bezogenen, tonalen Hören zu erlernen. Denn:

> *Es ist schon gefährlich, einzelne Intervalle mit Hilfe tonaler Eselsbrücken beherrschen lernen zu wollen, weil der daran gewöhnte Hörer beim Erfassen einer tonal nicht gebundenen Linie immer wieder zu fiktiven (Dur-)Grundtönen ausbrechen muß und dadurch das Organische einer solchen Linie zerstört und nicht wahrnehmen kann.*

Enharmonische Verwechslungen werden hier der Einfachheit halber nicht berücksichtigt. Man unterscheidet, obwohl man von übermäßiger Terz und reiner Quint sprechen könnte, beispielsweise nicht zwischen E# und F (bezogen auf das C), da in beiden Fällen ein Intervall von fünf Halbtonschritten gegeben ist und somit der selbe Klang ertönt. Ob es im tonalen Kontext Sinn

macht eines der beiden zu bevorzugen ist hier nicht von Bedeutung, da sich diese Art von Aufgaben nur im atonalen Bereich bewegt.
Zu Beginn des Intervalltrainings werden oft nur ein paar („einfache“) Intervalle geübt und auch nur ein Intervall nach dem anderen. Hier ist ein verbreiteter, jedoch umstrittener Ansatz sich an markanten Stellen aus populären Liedern zu orientieren. So kann man sich für die Quarte zu Kontrolle „Amazing Grace“ oder für kleine Terz das Kinderlied „Guten Abend, Gute Nacht“ vorsummen. Für den Einstieg scheint diese Methode durchaus brauchbar zu sein, kommt man aber in ein tonales Umfeld oder folgen mehrere Intervalle aufeinander, so treten schnell Probleme auf. Da es sich bei dem entworfenen Spiel um eine „Einstiegsdroge“ handelt, bei der die ersten Schritte möglichst angenehm ausfallen sollen, wird als Hilfestellung auf diese Methode verwiesen obwohl sie durchaus kontrovers ist.

Das Ziel des Spiels ist auf das Format des Tests der Lehrveranstaltung „Gehörschulung“ vorzubereiten. Im Bereich intervallisches Hören bedeutet dies eine „Viertonübung“, bei der auf dem Klavier jeweils vier Töne in beliebiger atonaler Folge gespielt werden. Die Intervalle können aufwärts als auch abwärts vorkommen, gehen jedoch nicht über die Oktave hinaus. Um darauf vorzubereiten werden im „Ear Training Adventure“ steigende Schwierigkeitsgrade verwendet und nach und nach werden die unterschiedlichen Intervalle, zuerst aufwärts, danach auch abwärts, vorgestellt. Die Anzahl der Töne kann beliebig angepasst werden und auch Zusatzübungen wie Intervalle harmonisch erkennen und das Nachsingen der Einzeltöne sind möglich.

3.2 Tonales Hören

Die Fähigkeiten jedes Intervall für sich erkennen zu können sind immer hilfreich und oft notwendig. Da zumeist auf das Dur/Moll-tonale System Bezug genommen wird, macht es auch Sinn, sich speziell darauf zu fokussieren. Dies wird mit dem tonalen Hören oder auch Stufenhören gefestigt. Erkennt man die Tonart und vor allem den Grundton eines Stückes, kann man jeden anderen skaleneigenen Ton bestimmen, indem man ihn zurück auf den Grundton referenziert. Hierbei wird sich auf das *Lehrbuch der Gehörbildung - Band 1* von *Roland Mackamul* bezogen. Dieser schreibt:

> *Der erstrebten Sicherheit und Schnelligkeit wegen sind gleichbleibende, standardisierte Auflösungswege Zweckmäßig. (...) Sie dürfen nie anders geführt werden, damit sie sich deutlich unterscheiden und schon der Ansatz zur Auflösung genügt, sie zu erkennen.*

Hier sei die Dur-Skala als Beispiel angeführt. Sie besteht wie alle diatonischen Skalen aus sieben Tönen. Diese werden auch als die sieben Stufen bezeichnet und werden meist mit römischen Ziffern bezeichnet. Auf den Grundton bezogen ergeben sich von Stufe I bis VII die Intervalle Unison, große Sekund, große Terz, reine Quart, reine Quint, große Sext und große Septime. Für jede dieser Stufen soll man nun gewisse Wege zurück zum Grundton, laut oder nur in Gedanken, singen. In der folgenden Tabelle sind die „Auflösungen" für die sieben Stufen der Dur-Skala aufgelistet. Diese werden zu Beginn des Spiels um sie zu verinnerlichen bei jeder tonalen Übung vorgespielt.

Stufe	I	II	III	IV	V	VI	VII
Auflösung	I	II-I	III-II-I	IV-III-II-I	V-I	VI-VII-I	VII-I

Hört man beispielsweise in der C-Dur-Skala ein „E", also die dritte Stufe, so soll man wie in der Tabelle angeführt vom „E" abwärts zum „D" zurück zum Grundton „C" singen.
Auch hier soll wieder auf die Abschlussprüfung vorbereitet werden. Die aufbauenden Schwierigkeitsgrade lassen den/die SpielerIn nach und nach die verschiedenen Skalen kennenlernen, erweitern die Register über eine Oktave hinaus und schlussendlich können wieder beliebig viele aufeinanderfolgende Noten abgefragt werden. Mit dem Wahrnehmen des tonalen Zentrums begibt man sich auch schon in die Richtung des Hörens und Erkennens von Harmonieverläufen und Kadenzen.

3.3 Harmonisches Hören

Eine weitere Dimension für das Hörverständnis bietet das vertikale oder harmonische Hören. Gemeinsam mit dem tonalen Hören kann man so nicht nur Melodien, sondern auch den zugrundeliegenden Harmonieverlauf analysieren. Auch hier muss man wieder bei den Grundlagen beginnen und zuerst die Unterschiede der vier verschiedenen Dreiklangsarten erkennen lernen. Diese entstehen durch die Schichtung von zwei Terzen übereinander. Aus der Kombination von kleinen und großen Terzen ergeben sich die folgenden Klänge:

Dreiklang	*Terz*	*Quint*
Dur	große Terz	reine Quint
Moll	kleine Terz	reine Quint
Vermindert	kleine Terz	verminderte Quint
Übermäßig	große Terz	übermäßige Quint

In dieser Form, auch Grundstellung genannt, sind die vier Klänge noch recht einfach durch ihre einzigartige Klangfarbe zu identifizieren. Dur als eher fröhlich, Moll klingt eher traurig. Verminderter und übermäßiger Dreiklang klingen beide sehr spannungsgeladen und sind schwieriger zu unterscheiden. Schon hier macht es Sinn die Akkordtöne herauszufiltern und die einzelnen Intervalle zu benennen. Bei einem spannungsgeladenen Akkord aus dem man eine kleine Terz heraushört handelt es sich beispielsweise um einen verminderten Dreiklang.
Mit den Fertigkeiten des intervallischen Hörens kann man nun den nächsten Schritt wagen und die Akkorde „umkehren“. Erklingt nun beispielsweise ein C-Dur Dreiklang nicht in Grundstellung (C-E-G) sondern in erster Umkehrung (E-G-C) handelt es sich noch immer um einen Dur Akkord obwohl das „tiefe“ Intervall E-G eine kleine Terz bildet und der Klang etwas „molliger“ wirkt. Man kann zwar immer noch darauf bauen, dass man den Charakter des Akkords erkennt, jedoch wird die Kombination mit dem intervallischen Hören immer wichtiger. Hierfür kann man den Grundton des Akkordes bestimmen und danach das Intervall zum tiefsten Ton analysieren. Da intervallisches Hören im Kontext von harmonisch gespielten Klängen durchaus eine andere Herausforderung ist wie das Erkennen aufeinanderfolgender Töne, soll im „Ear Training Adventure“ das Heraushören der einzelnen Töne zuerst mit Zweiklängen geübt werden.

Als Krönung dieser Disziplin werden als nächstes auch Vierklänge in das Repertoire aufgenommen. Diese werden durch das Schichten einer weiteren Terz auf die bekannten Dreiklänge gebildet, wodurch jeder dieser Akkorde eine kleine, große oder verminderte Septime erhält. Doch nicht alle Vierklänge sind gleich wichtig und häufig in der westlichen Musik. Ein Akkord nimmt hier eine sehr dominante Rolle ein: Der Dominant-Septakkord. Mit seinen Leittönen bildet sein Klang eine starke Spannung, die sich in die Tonika (der Akkord aufbauend auf dem Grundton einer Skala) auflösen möchte. Da dieser Vierklang mit Abstand am bedeutendsten in der westlichen Musik ist und weil er so klare Auflösungstendenzen hat, ergeben sich zusätzlich Möglichkeiten dessen Umkehrungen zu erkennen.

- Ist der Grundton im Bass, strebt der Basston des Dominant-Septakkords eine Quint nach unten in den Grundton der aktuellen Tonart (vgl. Stufenhören V – I).
- Die Terz im Bass löst sich einen Halbton nach oben auf, ebenfalls in den Grundton der Tonika.
- Die Quinte geht mit einem Ganzton nach unten, auch in den Grundton der Tonika.

- Einzig die Septime geht nicht zum Grundton, sondern mit einem Halbton nach unten in die Terz der Tonika.

Da dieser Akkord und sein Auflösungsverhalten enorm wichtig sind und so häufig vorkommen, soll auch im Spiel ein großer Wert auf dem Dominant-Septakkord liegen. Es wurden Übungen allein zum Erkennen dieser Umkehrungen implementiert. Ein akustisches Auflösen wie es beim Stufenhören eingebracht wurde, wäre noch eine wünschenswerte Funktion für das Audio Game.
Zusammenfassend sei hier noch eine Tabelle mit den wichtigsten Vierklängen und deren Aufbau angeführt:

Akkordname	Dreiklang	Septime	Beispiel
Dominantseptakkord	Dur	klein	C - E - G - B
Großer Septakkord	Dur	groß	C - E - G - H
Mollseptakkord	Moll	klein	C - Eb - G - B
großer Mollseptakkord	Moll	groß	C - Eb - G - H
Halbverminderter Septakkord	vermindert	klein	C - Eb - Gb - B
Verminderter Septakkord	vermindert	vermindert	C - Eb - Gb - Hbb
Übermäßiger Septakkord	übermäßig	groß	C - E - G# - H

Prüfungsaufgabe in dieser Kategorie ist es, zuerst festzustellen welche Art von Dreiklang vorliegt. Danach soll, falls vorhanden, die Septim bestimmt werden (also große ,kleine oder verminderte Septime). Zuletzt soll die Umkehrung benannt werden. Diese Unterpunkte sind im Audio Game möglichst auch getrennt voneinander trainierbar. So kann man beispielsweise zuerst mit Dreiklängen in Grundstellung, also dem Grundton im Bass, beginnen. Danach gibt es die Möglichkeit Umkehrungen in die Übungen einzubauen ohne sie benennen zu müssen. Man muss also beispielsweise einen Dur Dreiklang in zweiter Umkehrung nur als Durakkord erkennen ohne sich auf die Umkehrung konzentrieren zu müssen. Zuletzt kann man auch gezielt die Umkehrungen trainieren, zuerst mit nur einem Akkordtyp, danach mit mehreren und schlussendlich auch mit Vierklängen.

3.4 Kadenzen

Kombiniert man nun das tonale Stufenhören mit dem harmonischen Akkordhören, so kann man den Harmonieverlauf analysieren später auch die Stufen und Funktionen der Harmonien benennen. Da die Funktionstheorie

recht komplex werden kann, sei der Inhalt dieser Arbeit auf einige wenige, jedoch sehr häufige Wendungen oder Kadenzen beschränkt. Wie im melodischen Stufenhören zuvor, wird jede Note der verwendeten diatonischen Skala mit einer Stufe von I bis VII bezeichnet. Wählt man nun eine beliebige Stufe und schichtet zwei oder drei leitereigene Terzen darauf, so erhält man den zugehörigen Drei- oder Vierklang. Nimmt man beispielsweise die IV. Stufe als Ausgangspunkt schichtet für den Dreiklang noch zwei leitereigene Terzen, also die Noten der VI. und I. Stufe. In C-Dur ergibt sich mit den Tönen F (IV. Stufe) – A (VI. Stufe) – C (I. Stufe) ein F-Dur Dreiklang. Nach diesem Schema kann man für alle sieben Stufen jeweils die zugehörigen Drei- und Vierklänge bilden. Diese ändern ihr Tongeschlecht, wenn man die Skala beliebig transponiert nicht. Die leitereigene Harmonik ist also für jede Dur-Tonart gleich. Für die (natürliche) Moll Skala ergibt beispielsweise zwar eine andere Harmonik, sie bleibt aber auch für jede (natürliche) Moll-Tonart wieder unverändert:

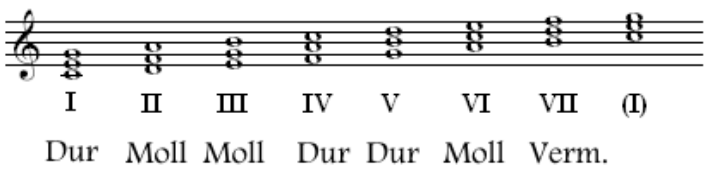

Abbildung 3.1: Dreiklangsakkorde aufbauend auf den Stufen einer Dur Skala

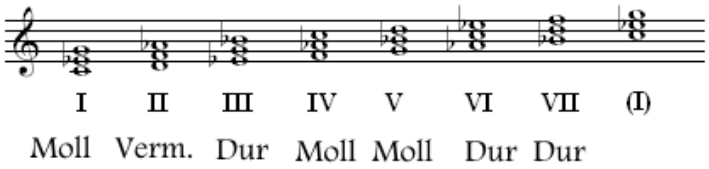

Abbildung 3.2: Dreiklangsakkorde aufbauend auf den Stufen einer Moll Skala

Bei einer Kadenz handelt es sich um eine Folge von Akkorden, die in unseren anfänglichen Beispielen ausschließlich, aus den eben besprochenen leitereigenen Akkorden gebildet werden. Sie bildet oft aber nicht notwendigerweise den Abschluss eines Abschnitts. Um nicht zu tief in die Materie zu tauchen, werden vorerst nur die wichtigsten Kadenzen oder Wendungen angeführt. Grundsätzlich wird zwischen drei Arten der Kadenz unterschieden.

- Am wichtigsten ist es die Folge von der V. Stufe in die I. zu erkennen. Dies wird auch als authentischer Ganzschluss bezeichnet.
- Als zweite wichtige Wendung sei der Plagalschluss, also der Übergang der IV. in die I. Stufe genannt. Dieser bietet weniger Spannung und somit eine schwächere Auflösungstendenz als der authentische Ganzschluss.
- Ein weiterer wichtiger Schluss ist der Trugschluss. Dieser deutet mit dem spannungsgeladenen Klang der Dominante (= Dreiklang der V. Stufe) ein Auflösen in die Tonika (= Dreiklang der I. Stufe) an, löst sich aber nicht dorthin, sondern in einen beliebigen anderen Akkord, auf. Eine Häufige Auflösung für den Trugschluss bildet die VI. Stufe.

Typische Kadenzen ergeben sich aus ebendiesen drei Kategorien. Ein paar beliebte Wendungen sind beispielsweise:

- I – IV – V – I
- II – V – I
- I – IV – V – VI

Natürlich gibt es sehr viele Varianten und Alterationen und es kann auch die Tonart kurzzeitig verlassen oder gar moduliert werden. Hier sei noch angemerkt dass,im Kontext einer Moll-Tonart die V. Stufe, die eigentlich einen Moll-Dreiklang bildet, zu einem Dur-Akkord umgedeutet wird. Denn so erst wird dasselbe Auflösungsbedürfnis zur Tonika, wie bei Harmonien in Dur, gewährleistet. Diese grundlegenden Kadenzen wurden zwar im Code implementiert, sie wurden jedoch im endgültigen Spiel nicht eingebaut, da die Spielwelt zu wenig Platz für die Fülle an vorhandenen Übungen bietet. Zusätzlich konnten die Kadenzübungen nicht in dem Ausmaß und der Genauigkeit behandelt werden wie die anderen Aufgaben.

Kapitel 4

Design des Spiels

4.1 Grundidee des Spiel

Die Idee eine Lernumgebung für ToningenieurInnen und andere StudentInnen, die Probleme damit haben ihr musikalisches Gehör effizient zu trainieren, zu schaffen gab es schon länger. Mit dem Angebot eines Seminars zum Thema „Audio Games“ war es naheliegend dieses Bedürfnis nach einer Lernhilfe als Bachelorarbeit umzusetzen. Zwar handelt es sich beim „Ear Training Adventure“ in der Umsetzung noch um ein Videospiel und es kann oder will nicht auf die visuelle Komponente verzichten. Dennoch spielt der klangliche Aspekt ebenfalls eine unverzichtbare Rolle. Der einzige Weg im Spiel fortzuschreiten, ist durch das Lösen von auditiven Aufgabenstellungen. In diesem Sinne handelt es sich um ein Audio Games insofern es ohne die Ausgabe von Audio nicht spielbar ist. Ein weiterer Teil des klanglichen Aspekts für das Audio Game ist die Steuerung des Spiels mit dem Mikrofon. Einige Aufgabenstellungen lassen sich nur durch richtiges Nachsingen oder summen der gehörten Töne lösen. Im Großen und Ganzen lag jedoch der Fokus darauf, das Spiel nicht zu einer reinen Lernplattform mit grafischer Oberfläche werden zu lassen. Davon sind im Internet und in Form von Apps bereits unzählige vorhanden. Das Ziel war es, sich darauf zu konzentrieren, dass es wirklich ein „Spiel“ wird, bei dem man auch Spaß und Freude verspürt und man es deshalb wieder und wieder aufgreifen möchte. Da es zu Beginn nicht klar war, wie weit dieses Projekt kommen würde, wurden nach und nach neue Funktionen eingebaut. So ist das Ergebnis teilweise etwas unstrukturiert, gerade wenn man den Code betrachtet. Zusätzlich wurden aus Zeitmangel einige wichtige Funktionen wie Rhythmusübungen oder erweiterte Kadenzen nicht mehr implementiert.

4.2 Erstes Konzept

Ein erster Versuch wurde bereits einige Zeit zuvor gestartet und umgesetzt. Es wurde zwar, wie im endgültigen Spiel auch mit Unity gearbeitet und in C# programmiert, jedoch wurde für den Audio-Input und Output mit Pure Data gearbeitet. Da sich „Pd“ (womöglich aufgrund mangelnden Wissens) nicht kompilieren lies, war es schwierig den ersten Entwurf weiterzugeben. Oftmals konnte das Audio Game auf anderen PCs nicht einmal gestartet werden. Zu diesem Zeitpunkt war das „Ear Training Adventure“ noch als 3-Dimensionales Spiel angedacht bei dem physikalische Hindernisse durch Lösen der Aufgaben überwunden werden konnten:

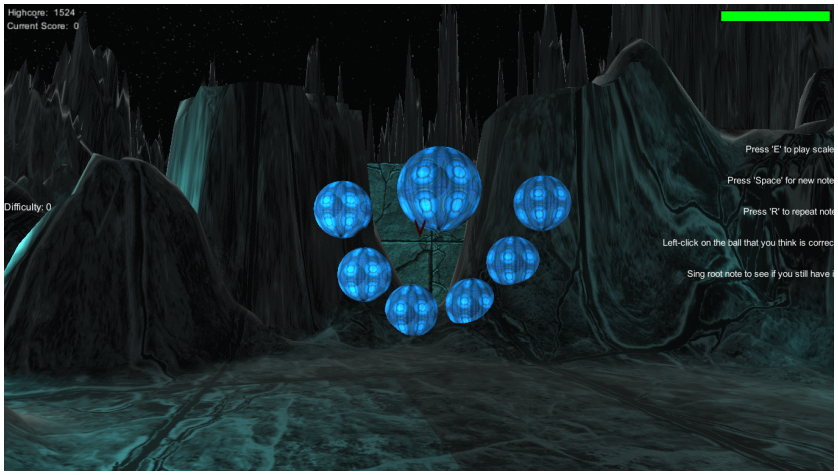

Abbildung 4.1: erster Entwurf des Spiels in 3D

Man konnte den Charakter wie im First-Person-Shooter im Raum fortbewegen und unter anderem mit dem „Visier“ auf die richtige Antwort klicken. In 4.1, welches eine Übung zum Stufenhören darstellt, bedeutet dies, das Klicken auf eine der sieben „Bälle“, da jeder „Ball“ repräsentativ für eine Stufe ist. In einer anderen Übung konnte man mit den Ziffern auf der Tastatur plus einer Maustaste das Intervall bestimmen (siehe Abbildung 4.2).

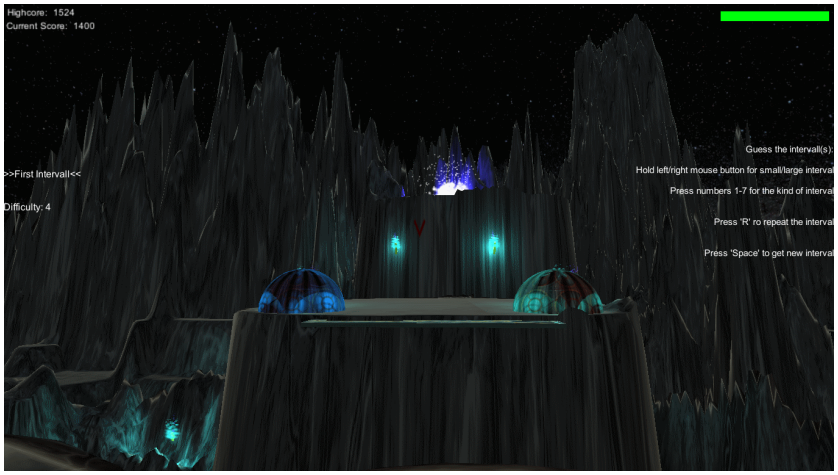

Abbildung 4.2: Intervallübung mit unintuitiver Handhabung

Trotz geschriebener Hilfestellung am Bildschirm, war es nach etwas Playtesting klar, dass diese unintuitive und variable Spielweise nicht sinnvoll war. Zusätzlich konnte man das ganze Spiel nur eine Übung nach der anderen absolvieren und lief so im Kreis und sammelte Punkte für den Highscore bis man oft genug falsche Antworten geliefert hatte und „starb“. Spaß ist beim Spielen kaum aufgekommen, die Übungen wurden schnell eintönig und der Code war aufgrund mangelnder Programmiererfahrung nicht einfach zu erweitern oder bearbeiten. Aus diesen und noch mehr Gründen wurde der erste Versuch verworfen und das ganze Konzept neu gestaltet mit dem Wissen im Hinterkopf auf welche Fehler und Mängel zu achten sei.

4.3 Neues Konzept - 2D RPG

Oftmals ist es förderlich für die Kreativität, die Möglichkeiten einzuschränken. Nach diesem Motto wurden aus drei Dimensionen zwei und aus First-Person wurde Top-Down, also eine Ansicht aus der Vogelperspektive. Man sollte nun, wie im „Role-Playing-Game“ üblich, in die Rolle des Protagonisten Florian schlüpfen und mit ihm oder eher durch ihn die Welt erkunden. Wie bereits erwähnt, wurde die neue Version stark von dem Klassiker „Pokémon“ inspiriert. Kaum ein anderes Spiel hat so viele Menschen in seinen Bann gezogen und diese stunden- oder tagelang für eigentlich recht repetitive Aufgaben an den Gameboy gefesselt. In Zahlen gesprochen ist die Spieleserie mit über 300 Millionen verkauften Exemplaren das meistverkaufteste RPG der Welt. Diesen Suchtfaktor galt es in das „Ear Training Adventure“ zu integrieren.

Es stellt sich die Frage, was macht nun diesen Klassiker so reizvoll? Welche Aspekte sind sinnvoll in ein Gehörbildungs-Lernspiel zu übernehmen?

- Das Design

  Das mittlerweile etwas nostalgische Design der ersten Generationen wurde mit einem Tilesystem, das 32x32 Pixel große Felder aneinanderreiht, nachgeahmt. Die Spielwelt, in der man von Stadt zu Stadt gehen kann, wurde auch mit dem Design der alten „Maps“ im Hinterkopf erstellt. Hier kann man frei durch die Städte schlendern, mit NPCs reden oder sich auf den Weg über gefährliche Routen, auf denen Gegner lauern, zur nächsten Stadt aufmachen.

- Die vermeintliche Freiheit

  Der/die SpielerIn kann sich in der Welt fast frei bewegen. Er/sie kann Stunden damit verbringen mit den Charakteren zu reden, mit Gegnern zu kämpfen oder einfach die Hintergrundmusik genießen. Das Spiel ist natürlich trotzdem stark geleitet. So kann man beispielsweise erst in die nächste Stadt wenn man dem Menschen, der einem den Weg versperrt etwas Bestimmtes bringt oder man muss stark genug sein um ihn/sie im Kampf zu besiegen.

- Das Kampfsystem

  Es gibt viele verschiedene Gegner und diese werden stärker, je weiter man fortschreitet. Man kann jedoch seinen eigenen Charakter so lange trainieren, bis man selbst die starken Gegner mit Leichtigkeit besiegt.

- Das Menü

  Etwas weniger offensichtlich aber doch wichtig ist zum einen das Inventar, also die Möglichkeit „Items“ zu sammeln und zu verwenden. Diese kann man beispielsweise verwenden um Gegenstände zu sammeln, die notwendig sind um Hindernisse zu überwinden. Zusätzlich ist eine Speicherfunktion im Menü unerlässlich. Ein Spiel wie dieses ist kein kurzweiliges Erlebnis, das man immer wieder von vorne starten möchte. Der Fortschritt des Charakters ist ein zentrales Element der Motivation zu spielen.

4.4 Spielwelt und Grafik

Für die visuelle Oberfläche wurden Grafiken die unter freien Lizenzen stehen von verschieden Websites (siehe Verweise) verwendet. Aus diesen PNG-

Dateien konnte man mit dem „Unity Sprite Editor“ 32x32 Pixel große Quadrate ausschneiden und in das in Unity integrierte „Tilesystem“ einbringen.

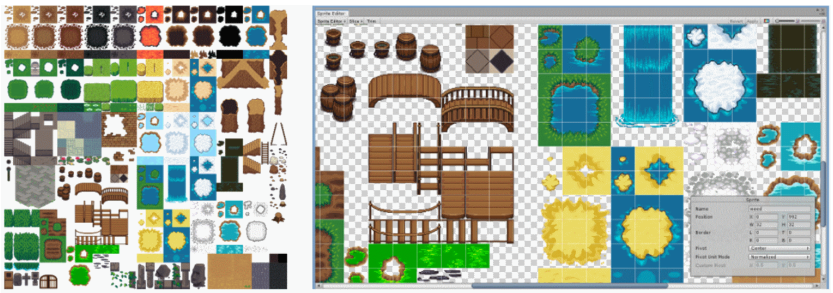

Abbildung 4.3: Unity Tilesystem und verwendete Sprites

Mit diesem Editor war es möglich in mehreren Ebenen diese Grafiken zu platzieren und ihnen Eigenschaften zu geben. Als „tiefste Ebene“ war es sinnvoll den Untergrund zu „verlegen“. Dabei muss noch nicht darüber nachgedacht werden, was schlussendlich wo platziert wird. Die nächsten Ebenen überdecken die tiefergelegenen einfach. Darauf konnte man eine Ebene höher problemlos weitere Details auftragen, wie zum Beispiel einen Weg auf dem verschneiten Untergrund oder eine Brücke.

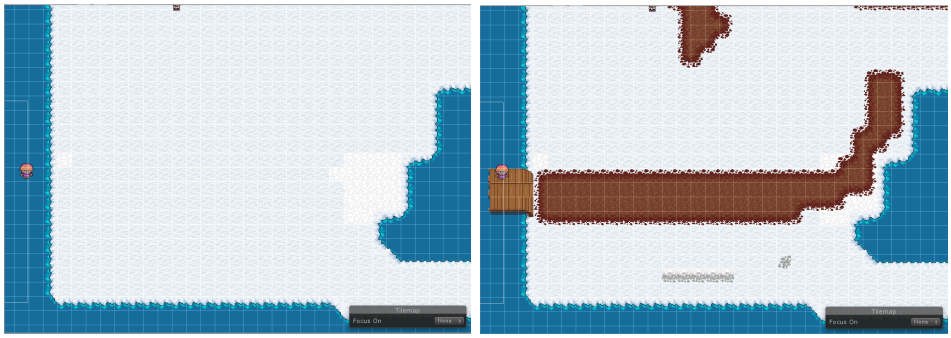

Abbildung 4.4: Layers 1 and 2

Diese Grafiken haben noch keinen Einfluss auf das Spielgeschehen, sie sind reine Dekoration und problemlos begehbar. Als nächste Ebene kommt das aus Pokemon berüchtigte „hohe Gras“ zum Einsatz. Die Idee dabei ist, dass im hohen Gras Gegner lauern und wenn man lange genug darin herumwandert wird man von einem dieser wilden „Monster“ angegriffen. Hierfür wurde ein eigenes Script benötigt, das an das „Game Object“ dieser Ebene angeheftet wurde.

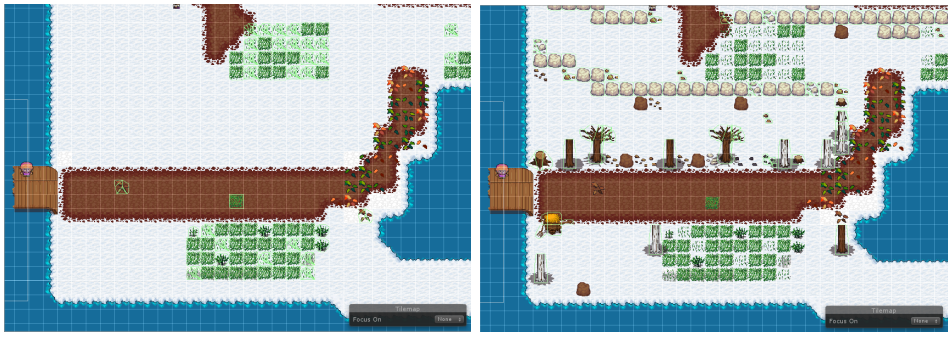

Abbildung 4.5: Layers 3 and 4

Im nächsten Schritt und der nächsten Ebene war es wichtig den Collider (= Komponente eines Spielobjekts, welche physikalische Berührungen mit anderen Objekten detektiert) zu aktivieren. Mit diesem konnten die platzierten Pixelquadrate als nicht passierbare „Hindernisse“ verwendet werden. Die damit platzierten Häuser, Bäume oder Zäune konnten somit die Grenzen der begehbaren Welt bieten.

Abbildung 4.6: Endergebnis aller überlagerten Ebenen

Als letzter Feinschliff brauchen nun auch Objekte, die zwar keine Interaktion oder Einfluss im Spiel haben aber „über“ dem Protagonisten platziert werden müssen, eine weitere Ebene. Dabei handelt es sich beispielsweise um Baumkronen oder Hausdächer „hinter“ denen man vorbeigehen kann. Somit ist die Grafische Oberfläche der Spielwelt geschaffen. Noch nicht besprochene Elemente, wie NPCs, sammelbare Items oder der Spielcharakter selbst werden abseits des Tilesystems in das Spiel eingebunden.

Zuletzt wurde noch damit begonnen Animationen in das Spiel einzubauen, um es so lebendiger werden zu lassen. Aktuell ist nur der Hauptcharakter selbst animiert. Hierfür wurde ein Set von Grafiken des Spielers in verschiedenen Positionen erstellt.

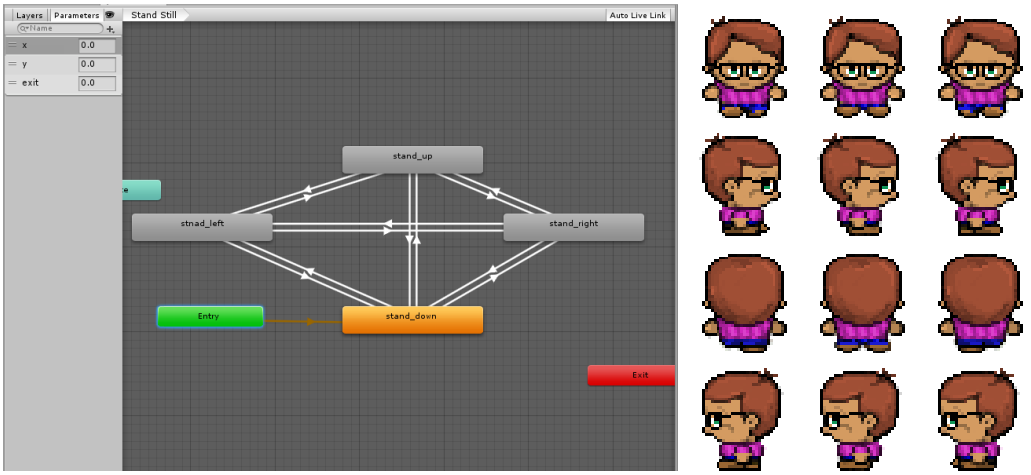

Abbildung 4.7: Animationssystem der Unity Engine

Im in Unity vorhandenen Animationssystem konnte man diese Grafiken dann beliebig ansteuern. Dafür wurden über das Skript, das auch für die Bewegung des Protagonisten zuständig ist, die Parameter x und y im „Animator“ verändert. Ist beispielsweise $x = 1$ und $y = 0$, so bewegt sich der Charakter nach rechts und die zugehörige Animation wird gestartet. Die Pfeile in Abbildung 4.7 zeigen von welchem Zustand in welchen übergegangen werden kann. Jeder dieser Übergänge hat auch Bedingungen, die für diesen Übergang erfüllt werden müssen.

4.5 Struktur der Spielwelt

Die Spielwelt lässt sich grob in zwei unterschiedliche Teile gliedern. Zum einen die Städte (bis zur Abgabe konnte nur eine Stadt implementiert werden) und Routen.

Stadt

Zu Beginn des Spiels findet sich der Protagonist in einer Stadt wieder. Hier läuft alles sehr sicher und kontrolliert ab. Die Stadt ist in erster Linie der Ort, an dem man neue Informationen erhält, meist dadurch, dass man mit den

ortsansässigen Menschen redet. Dafür kann der Spieler auch fremde Häuser betreten. Man soll grundlegende Fertigkeiten erlangen und kurze Einschulungen zur Musiktheorie und Tipps zu den Übungen erhalten. Des Weiteren kann man in der Stadt auch seine Lebenspunkte, die man im Kampf verloren hat, wieder auffüllen.

Abbildung 4.8: Stadt und inneres eines Hauses

Routen

Bei diesen Gegenden handelt es sich um Übergangspassagen, die durchquert werden müssen, um in der Geschichte voranzuschreiten. Die Routen sind gefährliche Orte an denen es vor Gegnern nur so wimmelt. Ist das Level des Spielers/der Spielerin nicht hoch genug und verliert er/sie einen Kampf, wird er „sterben“ und in der letzten Stadt wieder „spawnen“ (= Wiedereinstieg der Spielfigur mit vollen Lebenspunkten an einem festgelegten Ort). Die Lebenspunkte werden wieder aufgefüllt, jedoch verliert man jegliche gesammelten Erfahrungspunkte. Der Wechsel zwischen den Gegenden wird mit zwei unsichtbaren „Collider“ Objekten verwirklicht. An jedem dieser Objekte hängt ein Skript, das dem Spiel mitteilt, welche Region gerade verlassen und welche neu erschlossen wurde. Durch das Verwenden von zwei anstatt nur einem Collider konnte ein kritischer Fehler vermieden werden der dazu führen konnte, dass Regionen (und damit auch die Gegner in diesen Regionen) falsch erkannt wurden. Der Wechsel der Gegend selbst wird dem Spieler zum einen durch eine Informationsbox, das für ein paar Sekunden auftaucht, gekennzeichnet. Zusätzlich hat jedes Areal seine eigene Hintergrundmusik, die auch wechselt sobald man die Grenze überquert.
Was dem/der SpielerIn nicht mitgeteilt wird, ist die Tatsache, dass jede Region und auch jeder Teil einer Route unterschiedliche Gegner im hohen Gras enthält. Diese werden auch mit dem Betreten eines Gebiets neu definiert.

Abbildung 4.9: Der Spieler betritt Route 1

Welche Monster genau im hohen Gras angreifen ist zwar zufällig, welche Möglichkeiten für die „Zufallsauswahl“ zur Verfügung stehen und in welchem Bereich sich deren Level bewegen kann, wird in einem Skript vordefiniert. Auch die Menschen oder NPCs, die einem auf den Routen begegnen sind für gewöhnlich aufs Kämpfen aus. Ihre Stärke und der Typ der Gegner wird ebenfalls vordefiniert, jedoch direkt im Unity Editor. Um diese Gegner nicht einfach umgehen zu können und um „Endbosse“ erstellen zu können, wurde ein „Raycasting“ System verwendet.

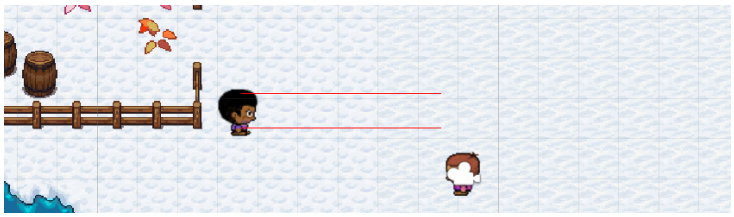

Abbildung 4.10: Raycasting System

Hierbei wird ein Skript an jeden „feindlichen“ NPCs angehängt und eine Richtung und Länge definiert in die er „sieht“. Kommt nun der Spieler in sein „Sichtfeld“, so wird er gestoppt. Der Gegner läuft ihm entgegen und fordert ihn zum Kampf. Doch diese Kämpfe sind nicht das einzige Hindernis am Weg, es wurden auch andere Hürden in die Spielwelt eingebaut. So ist dem Protagonisten zu Beginn der Weg aus der Stadt durch eine zerstörte Brücke versperrt. Der/die SpielerIn muss sich auf die Suche machen und Holz sammeln um die Brücke zu reparieren.

4.6 Controls

Wie spielt man nun das Spiel? Es wurde versucht die Steuerung möglichst einfach und intuitiv zu halten.

Fortbewegung – „W, A, S, D“

So wurde beispielsweise die, für Computerspiele übliche, Fortbewegungsart mit den Tasten „W, A, S, D“ für die Aktionen „oben, links, unten, rechts“ verwendet.

Menü – „Leertaste“

Geöffnet und geschlossen wird das Menü mit der klassischen Belegung der Leertaste. Im Menü selbst kommt dann zum ersten Mal die Maus ins Spiel. Die Unterpunkte wie beispielsweise „Save Game“ oder „Inventory“ sind durch einfaches Klicken des Knopfes mit der linken Maustaste bedienbar.

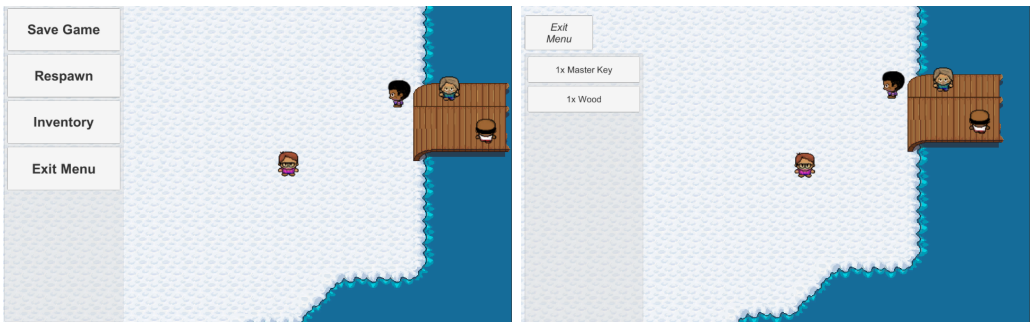

Abbildung 4.11: Ingame Menü und Inventar

Interaktion - E

Möchte man mit anderen Menschen reden, mit ihnen kämpfen, ein Item sammeln oder ein Schild lesen, so muss man zu demjenigen Objekt gehen und „E“ klicken. Auch dies ist eine übliche Taste für Interaktionen.

Navigieren im Menü und Kampf – „Mausklick“

Im Kampfmodus ändert sich die ganze Umgebung. Man kann sich in dieser Situation nicht mehr bewegen. Die einzigen Interaktionsmöglichkeiten bieten wiederum die „Buttons“ via Mausklick. Sobald der Kampf jedoch vorbei ist, kehrt man wieder in die gewohnte Umgebung zurück.

Singen – Bei vorhandenem Mikrofon

In gelegentlichen Übungen muss der/die SpielerIn die gespielten Töne eines Akkordes nachsingen. Dafür muss er/sie bei aktiviertem Mikrofon anfangen zu singen und in dem Moment, in dem er/sie glaubt die richtige Note zu halten, einen „Button“ klicken um den Ton als Antwort „einzuloggen“.

4.7 Kampfsystem und Gegner

Grundsätzlich gibt es 4 verschiedene Typen von Gegnern denen man begegnen kann. Intervallübungen, tonale Übungen, Akkordübungen und Kadenzen. Das Prinzip, nachdem so ein Kampf abläuft ändert sich dabei kaum. Anhand eines Gegners vom Typ „intervallisches Hören“ wird dessen Ablauf erörtert:

Begegnung

Zuerst muss man sich in der oben gesehenen 2D Welt einen Gegner finden. Dies kann durch langes gehen im hohen Gras oder treffen eines NPCs passieren.

Neue Benutzeroberfläche

Wie bereits erwähnt tritt man nun in ein anderes Bediensystem ein. Man kann sich nicht mehr bewegen und sieht nur einen starren Hintergrund mit einem Bild von dem Protagonisten und dem Kontrahenten, die sich gegenüberstehen. Über den beiden Charakteren sind noch Name, Level und eine Lebensanzeige zu sehen. Außerdem sieht man bei dem Gegner noch die Art der „Übung“ und bei dem Protagonisten den Fortschritt der „Experience Bar“ (= Erfahrungspunktanzeige). Die Interaktionsfähigkeit beschränkt sich auf das Klicken der Knöpfe im Menü. Zu Beginn des Kampfes erscheint nur ein einziger Button mit dem Titel „Play Attack“ welcher die erste Übung startet. In diesem Fall werden zwei Töne in Folge wiedergegeben deren Intervall es zu erkennen gilt.

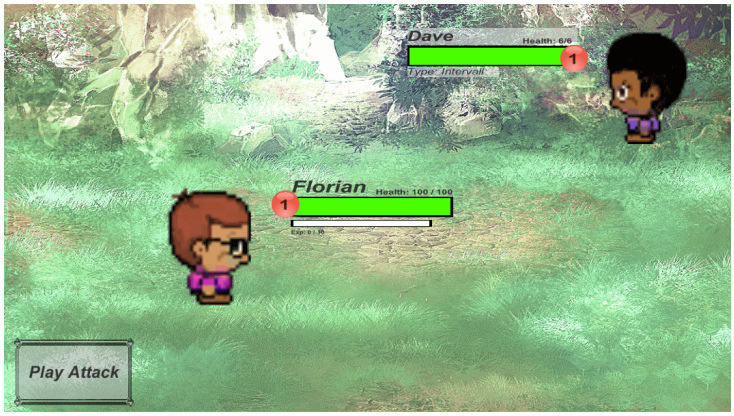

Abbildung 4.12: Kampfbeginn

Das Menü

Nun erscheinen mehrere Knöpfe, einige davon ausgegraut um die relevanten Buttons hervorzuheben. Die sind sind „Repeat Attack“ und „Name Intervall“. Ersterer wiederholt die exakten Töne, die bereits gespielt wurden. Klickt man auf den letzteren, so öffnet sich ein weiteres „Dropdown“ Menü, welches einem die Antwortmöglichkeiten vorgibt. Mit einem weiteren Klick ist die Antwort „eingeloggt“ und mit dem Klicken des „Solve-Buttons“ wird sie überprüft.

Abbildung 4.13: Menü in der Kampfumgebung

Bei einer falschen Antwort färbt sich der Knopf rot, man verliert Lebenspunkte und muss seine Antwort ändern. Klickt man nach dem Ändern der Antwort noch einmal „Solve“ und die Antwort ist richtig, so verliert der Gegner Lebenspunkte und der erste Bildschirm mit „Play Attack“ taucht für eine neue Übung wieder auf. Wird der Schwierigkeitsgrad höher, kommen mehr

Intervalle hinzu und damit auch mehr Knöpfe zum lösen der Intervalle. Erst wenn alle Intervalle richtig benannt wurden gilt die Antwort als korrekt und der Gegner verliert Punkte.

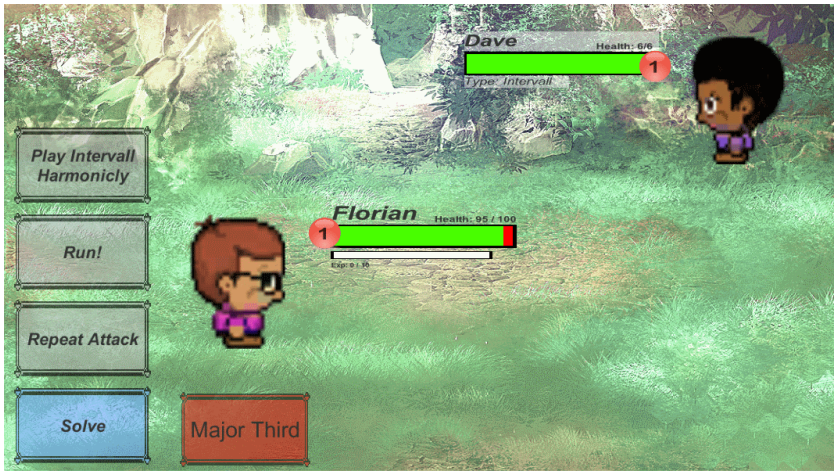

Abbildung 4.14: Darstellung einer falschen Antwort

Die anderen beiden, noch nicht besprochenen Knöpfe bewirken zum einen eine je nach Übung unterschiedliche Hilfestellung zum lösen der Aufgabe („Play Intervall Harmonicly“), mit dem anderen Button („Run“) kann man vor dem Kampf flüchten. Man erhält dafür aber auch keine Erfahrungspunkte und man kann auch nur in kämpfen mit „wilden“ Gegnern aus dem hohen Gras fliehen. Um die Intuitivität zu verbessern, wäre es womöglich sinnvoll manche Buttons in ein Untermenü zu verschieben.

Schadensberechnung

Die Schadensberechnung hängt von drei Faktoren ab. Zum einen von einer Konstante. Diese kann beliebig festgelegt werden um Kämpfe länger oder kürzer zu halten. Als nächstes hängt der Schaden davon ab, wieviel Prozent der Antworten falsch waren. Bei einer Antwortmöglichkeit und einer falschen Antwort ergibt sich ein Faktor von 1, bei einer nur teilweise falschen Lösung wird der Faktor kleiner und der Spieler verliert weniger Lebenspunkte. Als dritter Faktor geht die Kampfstärke der Charaktere ein. Diese verhält sich proportional zu dem jeweiligen Level. Die genauen Werte werden bei jeder Begegnung zufällig generiert um nicht zu eintönig zu werden. Eigenschaften wie das Level selbst, Name, Lebenspunkte und Aussehen der Gegner variieren ebenfalls zufällig in einem vorgegebenen Rahmen.

Experience Points und Level Up

Sind nun alle Aufgaben richtig gelöst worden und hat der Gegner keine Lebenspunkte mehr übrig, so ist der Kampf vorbei und der/die SpielerIn erhält Erfahrungspunkte proportional zu dem Level des Gegners. Visuell wird dies mit der Experience-Bar (unterhalb der Lebenspunktanzeige) veranschaulicht. Ist diese voll, so steigt der Spieler ein Level auf und wird stärker. Die Experience Bar geht zurück auf null und man benötigt um das folgende Level zu erreichen mehr Erfahrungspunkte als für das vorherige. So muss man früher oder später zu „stärkeren“ Gegnern weiterziehen um merkbare Fortschritte zu machen. Verliert man im Kampf jedoch alle Lebenspunkte, so fällt man in Ohnmacht und wacht am letzten Ort an dem man gespeichert hat mit einer leeren Experience-Bar auf. Der Verlust der Erfahrung is sozusagen die Bestrafung für das Verlieren eines Kampfes.

4.8 Intrinsische Motivation

Dieses Kampfsystem und vor allem die damit einhergehenden Belohnungen sind ein wichtiger Bestandteil des Konzepts eines spannenden Spiels. Auf jede Handlung des Spielers/der Spielerin folgt direktes Feedback in Form der Lebensanzeige. Doch auch nach einem Kampf lassen sich die Auswirkungen durch das Fortschreiten des Erfahrungspunkte-Balkens erkennen. Dieses schrittweise Sammeln von Erfahrungspunkten endet damit, in das nächste Level aufzusteigen und somit stark genug zu werden die nächsten Gegner zu besiegen und so in die nächste Region zu ziehen. Weitere Ideen für eine Belohnung für gewisse, erreichte Levels wären unter anderem neue Instrumente, die Fähigkeit schneller laufen zu können oder ein neues Outfit für den Protagonisten. Ein weiterer Grund für den/die SpielerIn im Spiel zu versinken, ist die Möglichkeit den Spielstand zu speichern. Es ist kein kurzweiliges Spiel, das man täglich von neu startet und bei null anfängt. Abhängig davon, wo man im Spiel steht ist die Erfahrung des Spielers/der Spielerin mit jedem Tag, an dem er/sie das „Ear Training Adventure“ startet eine neue. Anders als die üblichen Übungsprogramme geht hier ein „Spiel“ über eine 5-minütige Trainingseinheit hinaus, bei der der größte Anreiz ist, den Highscore in puncto Geschwindigkeit und richtigen Antworten zu schlagen. Ebenfalls ein Argument für das Audio Game gegenüber den herkömmlichen Übungsspielen ist die Tatsache, dass es den Fokus etwas vom Üben wegnimmt. Weniger konzentrierte Gehörbildung soll für mehr Freude und Begeisterung sorgen. Die regelmäßigen Kämpfe werden unterbrochen durch Aufgaben wie beispielsweise das Sammeln von Holz zum Reparieren einer kaputten Brücke. Zuletzt

ist es förderlich zu vermeiden den/die SpielerIn mit zu vielen Regeln und Erklärungen zu überwältigen. Es soll alles, von der Steuerung, über das Levelsystem bis hin zur Musiktheorie entweder sehr minimalistisch erklärt werden oder freiwillig geschehen. Damit ist gemeint, dass es dem/der SpielerIn freisteht mit anderen NPCs zu reden, die ihm Informationen geben können. Tipps zum Spiel können so im Tempo des Spielers/der Spielerin eingeholt werden und es wird ihm/ihr nicht alles auf einmal aufgezwungen. Zusätzlich sind die verschiededenen Arten der Übungen zeitlich über das Spiel verteilt, was zum einen für Abwechslung sorgt aber auch eine gewisse Struktur in den Lernprozess bringt, ohne dass sich der/die SpielerIn selbst einen sinnvollen Übungsplan zurechtlegen muss.

4.9 Handlung

Um auch inhaltlich einen Grund zu haben, den steinigen Weg mit all diesen Kämpfen zu bestreiten, braucht das Audio Game schlussendlich auch eine Story. Zuerst soll dem/der SpielerIn die neue Umgebung vermittelt werden, in der er/sie sich befindet. In dem Einführungsteil wird erklärt, dass er/sie nun eine Welt betritt in der musikalisches Gehör die wichtigste und mächtigste Fertigkeit ist, die man erwerben kann. Das Ansehen einer Person und dessen Familie ist in erster Linie dadurch definiert, wie talentiert man darin ist Intervalle zu benennen und Akkorde zu erkennen. Denn in dieser Welt kämpft man nicht mit Fäusten und körperlicher Gewalt, sondern mit Musik. Um die eigene Familie vor den wilden Bestien beschützen zu können, muss man also diese Fertigkeiten besitzen. Mit der Zeit hat sich diese Kampfform jedoch auch zu einem Wettstreit entwickelt und sogar Menschen kämpfen zur Belustigung miteinander. Mit dieser Prämisse schlüpft der/die SpielerIn nun in die Rolle des jungen Protagonisten Florian. Diesem steht gleich zu Beginn des Spiels eine wichtige Prüfung bevor, die wegweisend für seine Zukunft sein wird. Nachdem er beim Test erbärmlich versagt, ist er für seine angesehene Familie untragbar und sie wollen ihn nicht mehr als Schandfleck in ihren Reihen haben. Sie verstoßen ihn. So zieht der kleine Florian in die Welt hinaus mit nur einem Ziel. Er will lernen. Er will allen beweisen, dass er kein Versager ist und der beste Gehörgebildete werden, der jemals gelebt hat. Mit dieser kleinen Geschichte im Hinterkopf startet man in das Spiel und soll sogleich mit gesteigertem Ehrgeiz und Empathie für den Spielcharakter loslegen.

Kapitel 5

Technische Umsetzung

Für derartige Ideen liegt es nahe, eine Entwicklungsumgebung zu wählen, in der man möglichst keine Einschränkungen für die Implementation hat. Vorgefertigte abstrahierte Entwicklungsumgebungen, die einem zwar ein schnelles Erstellen von Spielen ermöglichen, eine individuelle Anpassung jedoch erschweren oder gar verhindern, sind nicht zielführend.

5.1 Game Engine - Unity

Aus diesem Grund wurde mit Unity eines der bekanntesten und beliebtesten Programme zum Umsetzen von digitalen Spielen ausgewählt. Die Möglichkeiten eigene individuelle Ideen umzusetzen, sind, unter anderem Dank einer aktiven Community, beinahe endlos. Der Editor bietet für viele gewünschte Funktionen vorgefertigte Editoren. Für die „UI“ (User Interface) gibt es ein eigenes System, arbeitet man in 2D und möchte „Tilemaps“ verwenden, gibt es einen separaten Editor dafür. Selbst für Animationen gibt es eine spezielle Umgebung. Jedes „Game Object“, sei es 2D oder 3D, kann mit vordefinierten Eigenschaften, sogenannten „Components“ versehen werden. Dazu zählen Position, Form, Aussehen, physikalische Eigenschaften, wie Gewicht und Elastizität oder auch „Collider“, die entweder bei Berührung als Auslöser für ein Skript dienen können oder einfach ein physikalisches Hindernis bilden. Auch Skripten können als Komponenten angefügt werden. Alle diese Eigenschaften sind im Editor selbst frei bestimmbar. Möchte man jedoch Aufgaben lösen, wie beispielsweise den Hauptcharakter zu bewegen, so muss man mittels eines Skripts diese Komponenten manipulieren. Dafür wird eine neue Skript-Komponente beim „Game Object“ des Protagonisten hinzugefügt. In dieser muss in einer regelmäßig (ein mal pro Frame) abgerufenen „Update“-Funktion die Tastatureingabe kontrolliert werden und dementsprechend die

anderen Komponenten wie beispielsweise die Position, das Aussehen und die Animation verändert werden.

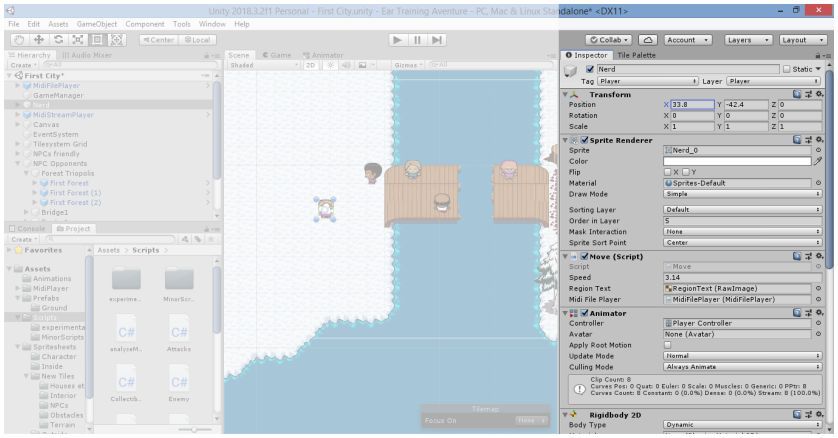

Abbildung 5.1: Unity Inspector - Komponenten eines Gameobjects

Auf der rechten Seite der Abbildung 5.1 sieht man einige dieser „Components“ des Protagonisten. Zuerst die Transform Komponente für Position, Rotation und Größe. Darunter erkennt man den für das Aussehen verantwortliche „Sprite Renderer“. Als nächstes folgt das selbst geschriebene Skript für den Bewegungsablauf des Spielers, gefolgt von dessen Animation und einem „Rigidbody 2D“, welcher dem Bild des Spielers auch einen physikalischen Körper für Kollisionen gibt.

5.2 Programmierumgebung

Die Skripte, die für die Logik und den Ablauf des Spiels verantwortlich sind, werden in der Objektorientieren Hochsprache C# verfasst. Als Umgebung diente zu Beginn „MonoDevelop“ und später, das mit dem Update von Unity mitgelieferte, „Visual Studio“. Dank diesen benutzerfreundlichen Programmen wurde der Prozess des Programmierens wesentlich vereinfacht. So bieten diese Dienste unter anderem die Möglichkeit auf Autovervollständigung, sogar bei selbst geschriebenen Funktionen und Klassen. Verschiedene Logikelemente werden unterschiedlich eingefärbt, man kann Sektionen zusammenklappen und den Stil für Klammernsetzung und Einrücken von Zeilen für jede Funktion vordefinieren. Neben diesen angenehmen Eigenschaften gibt es zusätzlich für jede Funktion, die man verwenden möchte eine meist recht ausführliche Dokumentation, wie genau sie funktioniert, welche Eingabeparameter möglich sind und was man als Ergebnis zu erwarten hat. Obwohl

mit Unity auch problemlos „mobile Games“ entworfen werden können, beschränken sich die Versionen für das „Ear Training Adventure“ auf computerbasierte Systeme, also PC und Mac, da die Steuerung dafür ausgelegt wurde. Eine Erweiterung auf Smartphones ist aber durchaus denkbar.

5.3 Umsetzung eines Skripts

Um die Umsetzung gewisser Spiellogiken in einem C# Skript darzustellen, soll hier anhand des Themas „wie entsteht ein Gegner“ ein Beispiel geliefert und durchbesprochen werden.

Definieren der möglichen Kontrahenten im hohen Gras

Es gibt zwei Arten von Gegnern. Wilde Gegner, die im hohen Gras erscheinen und „menschliche“ NPCs. Beide werden auf ähnliche aber nicht idente Weise initialisiert. Erstere werden schriftlich im Code festgehalten und variieren je nach Region, in der man sich befindet. Es können verschiedenste Arten von Gegnern in einer Region definiert werden:

```
5  public class RegionManager : MonoBehaviour
6  {
7      #region definitions and initialisations
8
9      public List<Region> listOfRegions = new List<Region> ();
10
11     public Region currentRegion = new Region();
12     public Sprite sprite1;
13     public Sprite sprite2;
14     public Sprite sprite3;
15      int common = 100;
16     int rare = 50;
17     int legendary = 10;
18     |
19     #endregion
20
21     void Start()
22     {
23         //input for constructor: string name, int averageEnemyLevel
24         listOfRegions.Add (new Region ("City1",15));
25         listOfRegions.Add (new Region ("Route1",1));
26         listOfRegions.Add (new Region ("Route1", 20));
27         listOfRegions.Add(new Region("Route1", 20));
28
29         comments
58
59         #region Enemies in Region 0
60         listOfRegions[0].enemiesInRegion.Add(new Enemy("Major", sprite3, "Tonal", common, new int[] { 0 }, 0, 1, new int[] { 1 }));
61         listOfRegions[0].enemiesInRegion.Add(new Enemy("Natural Minor", sprite3, "Tonal", rare, new int[] { 1 }, 0, 1, new int[] { 1 }));
62
63
```

Abbildung 5.2: Intialisierung von möglichen Kontrahenten im hohen Gras

Im Vorfeld werden Sprites, also die „Bilder“ der Kontrahenten und Konstanten für den Seltenheitswert einer Kreatur („common“ bedeutet beispielsweise, dass dieser Gegner oft vorkommt) festgelegt. Danach werden die Regionen definiert mit einem Namen und einem durchschnittlichen Level, der dort vorkommenden Monster. In den Zeilen 60 und 61 wird dann für die erste Region

definiert welche Möglichkeiten für Begegnungen hier bestehen. Die Eingabeparameter bestimmen nicht nur den Typ des Gegners, sondern auch alle Variationen von Schwierigkeitsgraden.

Eingabeparameter

- string EnemyName

  Spielerisch nicht sonderlich relevant, aber dem Gegner einen Namen zu geben, macht den Kampf interessanter.

- Sprite EnemySprite

  Hier wird eines der zuvor definierten „Bilder“ für diesen Gegner festgelegt.

- string TypeOfEnemy

  Dieser „string“ setzt die Art der Übung fest, also ob es sich um tonales, intervallisches, harmonisches Hören oder um Kadenzen handelt.

- Int Rarity

  Die „Rarity“ definiert die Wahrscheinlichkeit dieses Gegners aufzutauchen.

- int[] BaseDifficulty

  Hier beginnen die Variationen der Schwierigkeitsstufe. Dieses Array legt fest welche Intervalle, Skalen, Töne oder Kadenzen für die Übung zur Auswahl stehen. Da es ein Array ist, können mehrere Auswahlmöglichkeiten getroffen werden. Der Zufall entscheidet dann bei jeder Begegnung mit einem Monster, welche Übung exakt genommen wird.

- int AdditionalDifficulty

  Dieser Parameter ist für jeden Typ von Übung unterschiedlich. Bei den Intervallen wird so gewählt, ob sie aufwärts oder abwärts gespielt werden. Bei tonalem Hören kann damit das Register über eine Oktav hinaus erweitert werden. Bei den Akkorden wird zwischen den verschiedenen Umkehrungsformen unterschieden.

- int NumberOfNotes

  Hier wird die Anzahl der Noten festgelegt. Bei Akkordübungen wird dafür der Grundakkord genommen und so viele Töne oktaviert bis diese Zahl erreicht wird.

- int[] AdditionalInformation

  Auch im letzten Parameter gibt es für jede Übung wieder starke Unterschiede. Bei der Intervallübung bedeutet es, dass man die Töne harmonisch, also zur gleichen Zeit, hört und sie danach einzeln nachsingen muss. Beim tonalen Hören entscheidet diese Variable ob die „Auflösung“ zum Grundton beim Lösen der Übung abgespielt wird. Für die Akkordübung legt sie die Noten pro Oktav fest, also ob eine enge oder weite Lage gespielt werden soll und bei den Kadenzen bestimmt sie welche Skalen als Basis für die Akkordfolge gewählt werden sollen.

Definieren der NPC Gegner

Im Gegensatz zu den wilden Gegnern werden NPC Gegner direkt im Unity Editor über eine Komponente am jeweiligen Charakter definiert. Dies ermöglicht eine angenehmere Handhabung und bessere Übersicht. Im Gegensatz zur vorherigen Initialisierung im Code können hier nicht beliebig viele Variationen von Übungen für einen NPC festgelegt werden. Man soll immer ähnliche Übungen erwarten können, wenn man wiederholt mit dem selben Charakter kämpft.

Initialisieren eines Kampfes

Sind die Gegner nun definiert und begegnet man einem dieser Geschöpfe, so werden über ein anderes Skript die Zufallswerte eingesetzt und ein konkreter Gegner gewählt. Zusätzlich wird die Kampfumgebung mit den Bildern, Lebensbalken und Namen der Charaktere erstellt und die Knöpfe für das Menü kreiert. In diesem Beispiel ist man in einer Intervallübung. Der Spieler klickt den Knopf „Play Attack“. Hier sei das nächste Skript angeführt.

```
35      int[] returnNotes;
36
37      switch (typeOfAttack)
38      {
39      #region case: Intervall
40      case "Intervall":
41          highestPitch = 80;
42          lowestPitch = 40;
43          // marker!!! root note must be randomized again and note tempo should not be adjusted?
44          //rootNote = Random.Range(40, 65);
45
46          Manager.sustainMelodic = 1450;
47          Manager.noteTempo = 1500;
48          rootNote = 53;
49
50          UI_FightScreen.possibleAnswersString.Clear ();
51
52          if (!difficultyIsSet)
53              // when an opponent appears it sets difficultyIsSet (in FightManager) to false so one of the random possible opponents gets set,
54              // then difficultyIsSet is set to true so it does not get changed every attack (the possible intervalls stay the same)
55          {
56              possibleIntervallsLocal = possibleIntervalls (baseDifficulty);
57              difficultyIsSet = true;
58          }
59
60          numberOfNotes = additionalDifficulty == 3 ?  2 : numberOfNotes;
61          //if notes are played harmonicly, there can only be two notes + one intervall (maybe i change that later), just for avoiding initialisation mistakes
62
63          int[] consecutiveNotes = new int[numberOfNotes];
64
65          int iterator = 0;       //two break while loop if it does not find a reasonable result
66
67          do {
68              outOfBounds = false;
69              UI_FightScreen.solution.Clear ();
70
71              consecutiveNotes [0] = rootNote;
72
75              for (int i = 1; i < numberOfNotes; i++) {
76
77                  //randomize if negativ or positiv intervall: the  index "0" is always +1, the index "1" is always -1, the index "2" is random
78                  //the index "3" is for harmonic intervals and is also always +1
79                  makeIntervallsNegativ [2] = makeIntervallsNegativ [Random.Range (0, 2)];
80                  int nextIntervall = possibleIntervallsLocal [Random.Range (0, possibleIntervallsLocal.Count)] * makeIntervallsNegativ [additionalDifficulty];
81
82                  consecutiveNotes [i] = consecutiveNotes [i - 1] + nextIntervall;
83
84                  if (consecutiveNotes [i] < lowestPitch ) {          //if one of the notes is too low or too high, it genereates a new number of notes with higher or lower root
85                      outOfBounds = true;
86                      rootNote++;
87                  }
88                  if ( consecutiveNotes [i] > highestPitch) {
89                      outOfBounds = true;
90                      rootNote--;
91                  }
92
93                  UI_FightScreen.solution.Add (nextIntervall);
94  //                  Debug.Log ("rootNote " + rootNote + "currentNote" + consecutiveNotes [i] + "iterator" + iterator);
95              }
96              iterator++;
97              if(iterator>=249){Debug.Log("too many while loops for intervall attack");}
98          } while(outOfBounds && iterator < 250);
99
100         setSolutionAndAnswers (possibleIntervallsLocal.ToArray());
101         returnNotes = consecutiveNotes;
102         break;
103     #endregion
104
105     case: Chord
187
188     case: Tonal
255
256     case: Cadence
302
```

Abbildung 5.3: Code zum Laden eines konkreten Gegners und Laden aller Grafiken und Werte

In diesem Teil des Codes werden alle Informationen der Initialisierung verarbeitet. Zuerst werden Konstanten wie der Dauer einzelner Noten, die Zeit dazwischen und die Tonhöhe der ersten Note (Zeilen 46- 48) definiert. Danach werden ein paar Sicherheitsmaßnahmen getroffen, die sich beim Debuggen als notwendig erwiesen haben. Ihre Funktion steht in grau dabei. Es wird ein Integer-Array namens „ConescutiveNotes“ erstellt. Dieses soll mit den MIDI-Werten der Noten befüllt werden, die abgespielt werden sollen. In einer Schleife werden nun so viele Noten hinzugefügt wie in der Initialisierung festgelegt wurden. In den Zeilen 80 und 81 wird die zusätzliche Information eingebaut, ob Intervalle positiv oder negativ vorkommen sollen. Zusätzlich wird nur auf die, durch die „BaseDifficulty“ definierten Möglichkeiten an Intervallen zugegriffen. Die nächsten Zeilen dienen dazu, alle Noten wieder nach unten in den hörbaren Bereich zu verschieben für den Fall, dass ein Ton eine zu hohe Frequenz erreicht. Sind nun alle Töne, welche in Form von

MIDI-Tonhöhen verarbeitet werden, festgelegt, werden alle wichtigen Informationen an das „User Interface“ weitergegeben, da dieses die Knöpfe richtig erstellen und benennen muss. Danach werden die MIDI-Noten von der Funktion ausgegeben und über einen MIDI-Player abgespielt.

5.4 Midi Player

Wie erwähnt werden alle Klänge, die in diesem Spiel erzeugt werden über MIDI-Noten gespielt. Die Berechnung der zufälligen Tonfolgen und Akkorde ist so am einfachsten und das MIDI-Format ermöglicht einige weitere Eingabeparameter wie beispielsweise die Dauer der Note und deren Anschlagstärke. Der hierfür benutze MIDI-Player wurde aus dem Asset Store, dem Onlineshop für Erweiterungen von Unity, übernommen. Einerseits ist dieser für einzelne Noten ideal über ein Skript anzusteuern, andererseits kann man auch direkt vollständige, mehrspurige MIDI-Dateien abspielen. Für die Hintergrundmusik wurden deshalb MIDI-Dateien unter freier Lizenz heruntergeladen und in das Spiel integriert. Da man nur MIDI-Dateien speichern muss, wird für die Musik wenig Speicherplatz benötigt. Für das Erzeugen der Übungen wird mit folgendem Skript mit dem vorgefertigten MIDI-Player kommuniziert. Die Funktionen, die in anderen Skripten verwendet werden können, benötigen nur ein Integer-Array als Input. Zusätzlich muss noch unterschieden werden ob die Töne gleichzeitig oder nacheinander erklingen sollen, wobei der Unterschied dabei nur in einer kurzen „Wartezeit“ zwischen den Noten liegt, der in Zeile 44 implementiert wird. Ansonsten wird in einer Schleife eine neue MIDI-Note (Klasse „MPTKNote“ mit der Instanz „newNote“) nach der anderen kreiert. Diese benötigt die Tonhöhe und die Länge des Tons. Bei einem harmonischen Klang kommt zusätzlich die Anschlagdynamik und eine Zeitverzögerung, die im Millisekunden-Bereich liegt hinzu um die Töne besser differenzieren zu können und einen „humanize“-Effekt zu erzielen. Die Anschlagdynamik („velocity“ wird zum besseren erkennen der Bass Note auf den Wert „100“ und danach auf „80“ gesetzt. Diese MidiNote wird schlussendlich über den „midiStreamer“ ausgegeben. Die Klassen „MidiStreamPlayer“ und „MPTKNote“ mit den Instanzen „midiStreamer“ und „newNote“ sind die aus dem Asset Store entlehnten Inhalte.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using MidiPlayerTK;



public class PlayNotes : MonoBehaviour {


public MidiStreamPlayer midiStreamer;
MPTKNote newNote;


    public void PlayHarmonicly(int[] pitch)
    {
        int velocity = 100; // velocity of lowest note (or rather first indexed note) in pitch[]
        foreach(int note in pitch)
        {

            newNote = new MPTKNote (note, Manager.sustainChord, velocity, Random.Range(0,80));
            newNote.Play (midiStreamer);
            velocity = 80; // reduces velocity of all following notes so you cn hear the low note better

        }
    }


    public void PlayMelodicly(int[] pitch)
    {
        StartCoroutine (waitForIntervall (pitch, Manager.noteTempo));
    }







    IEnumerator waitForIntervall (int[] pitch, float  duration)
    {
        foreach (int note in pitch)
        {
            newNote = new MPTKNote(note, Manager.sustainMelodic);
            newNote.Play (midiStreamer);
            yield return new WaitForSeconds ((duration / 1000));
        }
    }


}
```

Abbildung 5.4: Wiedergabe von Noten via eines integrierten MIDI-Players

Kapitel 6

Conclusio

6.1 Erkenntnisse

Die offensichtlichste Erkenntnis nach über einem Jahr Arbeit an dem Projekt „Ear Training Adventure“ ist wohl, dass ein vollwertiges Computerspiel, das viele der wichtigen Elemente des Genres beinhalten soll zu viel für eine Einzelperson ist. Leider blieb es aufgrund der hohen Ambitionen bei einem „Proof of Concept“ und wurde zu keinem vollwertigen Spiel. Die Spielwelt wurde nicht groß genug umgesetzt um alle Elemente und Schwierigkeitsgrade der Übungen in einem sinnvollen Aufbau unterzubringen. Zusätzlich wurde noch keine richtige „Playtesting“-Phase durchlaufen. Die Einschätzungen zu dem Spiel kommen bislang zur Gänze aus der Perspektive des Designers. Da auch zuvor kaum Kenntnisse in der Arbeit mit Unity oder C# vorhanden waren und sich das Projekt immer weiterentwickelte, wurde der Code mit der Zeit sehr unübersichtlich. Sollten andere Personen an dem Projekt weiterarbeiten wollen, so müsste die Struktur des Codes verbessert werden und mehr auf Codingstandards geachtet werden. Als positive Erkenntnis sei noch angeführt, dass die verwendeten „To-Do-Listen“ eine großartige Idee waren. In diesen konnten kleine Aufgaben, die während dem Arbeiten auftauchten, eingetragen und nach und nach abgearbeitet werden. So hat man nicht nur das große, überwältigende Projekt vor Augen, sondern kann sich kleine Ziele stecken, sich gezielt auf jede Zeile Code konzentrieren und trotzdem neue Einfälle schnell festhalten, um sie nicht zu vergessen.

6.2 Probleme

Viele der Probleme, die im Laufe der Arbeit auftraten, wurden mittlerweile wieder gelöst. Einige wurden aber noch nicht behandelt, da das Spiel auch

mit den Fehlern funktioniert und andere Aspekte Priorität hatten. Bei den Übungen mit Gesang als Input gibt es beispielsweise Probleme mit gewissen Vokalen. Dort wird die Tonhöhe nur sehr ungenau und oft gänzlich falsch gemessen. Das kann daran liegen, dass diese genau in einem störenden Bereich Formanten haben. Weiters ist zu überlegen, ob es zu Leistungsproblemen kommen kann, da in der Spielwelt aktuell alles auf einmal geladen wird und nicht nur die Region, in der man sich befindet. Durch solche Informationen würde das Spiel vom Playtesting stark profitieren.

6.3 Ideen

Die Ideen, mit denen das Projekt erweitert werden kann sind zahlreich. Deshalb seien hier nur einige kurz erwähnt.

- Einen Rivalen integrieren
- Verschiedene Spielstände ermöglichen
- Sammeln von besiegten wilden Gegnern nach dem Stil von Pokémon
- Eine Klangumgebung abseits von Musik
- Klangliche Rätsel abseits von Kämpfen
- Geld und Shops
- Fähigkeiten zum Überwinden von Hindernissen (e.g. Schwimmen)
- Rhythmusübungen
- Melodiediktat (statt zufälligen Noten)
- Komplexere Kadenzen mithilfe von MIDI-Dateien
- Übungen selbst bestimmen können
- Aussehen des Charakters variabel machen
- Aufnahmen statt MIDI-Noten verwenden
- Spielwelt vergrößern

Man kann erahnen, dass in diesem Projekt noch sehr viel Potential steckt. Womöglich bietet sich noch die Chance einige dieser Punkte zu verwirklichen und ein fertiges Spiel zu veröffentlichen. Ob das „Ear Training Adventure“ in seiner aktuellen Form eine Bereicherung für die Landschaft an Lernhilfen ist, ist schwierig einzuschätzen. Um festzustellen, ob mit dem Spiel auch tatsächlich ein Lernerfolg einhergeht, wäre wohl eine ausführliche Evaluation nötig. Sicher ist jedoch, das Audio Game bietet Möglichkeiten und ein Umfeld, das es in dieser Form noch nicht gibt und es ist zumindest einen Versuch Wert diesen Weg zu testen.

Kapitel 7

Verweise

Bei allen verwendeten Grafiken und Dateien handelt es sich um freie Lizenzen.

Grafiken am Startbildschirm:

Kopfhörergrafik

https://pixabay.com/de/vectors/abstrakt-kunst-audio-aural-ohr-2027962/

Grafiken im Spiel:

Tiles

https://opengameart.org/content/lpc-submissions-merged

Spieler und Non-Player Characters

http://opengameart.org/content/rpg-character-bases-assets

Autor: Curt

Benutzername: cjc83486

Grafiken der Gegner

https://opengameart.org/content/owl-character

https://opengameart.org/content/zombie-elf

https://opengameart.org/content/elguaah

https://opengameart.org/content/cute-blob-creature

https://opengameart.org/content/googon-one-eyed-monster-rework

Created by Tracy & Jordan Irwin (AntumDeluge)

Hintergrund im Kampfmodus

https://opengameart.org/content/forest-background-art-2

MIDI-Player:

Midi Player Tool Kit von Bachmann (aus dem Unity Asset Store)

https://assetstore.unity.com/packages/tools/audio/midi-tool-kit-free-107994

Midi Dateien für die Hintergrundmusik

http://www.partnersinrhyme.com/midi/index.shtml

Grafiken im Dokument:

Dreiklänge auf den Stufen der Dur- und Moll-Skala

https://de.wikipedia.org/wiki/Datei:Stufentheorie2.PNG

https://de.wikipedia.org/wiki/Datei:Stufentheorie3.PNG

Literatur

Johan Huizinga

Homo Ludens: Vom Ursprung der Kultur im Spiel

Roland Mackamul

Lehrbuch der Gehörbildung - Band 1

Marc Prenskys

Digital game-based learning

Carolyn Handler Miller

Digital Storytelling: A Creator's Guide to Interactive Entertainment

Eric Zimmerman and Katie Salen

Rules of Play: Game Design Fundamentals