Bachelor's thesis

Information and Communication Technology

2021

Janne Soikkeli

CREATING A RHYTHM AUDIO GAME

TURKU AMK

TURKU UNIVERSITY OF APPLIED SCIENCES

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Information and Communication Technology

2021 | number of pages: 35

Janne Soikkeli

CREATING A RHYTHM AUDIO GAME

Audio games - as opposed to video games - are generally created for visually impaired players and are wholly playable without visual stimulus. Audio games are a diverse sector of games without a well-defined design architecture, and as such offer a fertile ground for experimentation.

The challenge of this thesis was to create a unique audio game, as a solo developer commissioned by myTrueSound game company, that would be appealing to both visually impaired -and normally sighted players. To accomplish this, I composed a genre blending idea that’s mechanics can be distilled into three main systems: rhythm gameplay, typing interaction and dynamic audio.

The co-operation of these main components resolved the success of the project, measured by the results of a testing period done on a game build with all systems implemented, performed by players ranging from blind to fully sighted. The results were compiled from a questionnaire regarding the three main areas of the game (rhythm gameplay, typing interaction and dynamic audio), yielding positive results in each area.

This thesis reports how the project advanced from the idea conception stage to a fully playable horror-themed rhythm audio game, to be published on various online marketplaces, including Itch.io and AudioGames.store.

In the scope of this thesis, the resulting game was a success, based on the feedback gathered in the testing phase.

KEYWORDS:

audio game, rhythm game, horror game, typing game, accessibility, solo development

OPINNÄYTETYÖ AMK | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tieto -ja viestintätekniikka

2021 | sivuja: 35

Janne Soikkeli

RYTMIÄÄNIPELIN KEHITTÄMINEN

Äänipelit – toisin kuin videopelit – ovat yleensä luotu näkövammaisille pelaajille ja ovat näin täysin pelattavissa ilman visuaalista stimulusta. Äänipelit ovat pelien monimuotoinen sektrori, jolla ei ole selkeästi määriteltyä suunnitteluarkkitehtuuria ja tarjoavat täten hedelmällisen maaperän kokeilulle.

Tämän opinnäytetyön tavoitteena oli uniikin äänipelin soolokehitys myTrueSound-peliyritykselle. Peli on kehitetty sokeille tai näkökyvyltään heikoille pelaajille yrittäen samalla luoda siitä vetovoimainen myös perusnäkökykyisille pelaajille. Tavoite pyrittiin saavuttamaan muodostamalla pelityylilajeja yhdistelevä idea jonka mekaniikat voidaan tiivistää kolmeen perussysteemiin: rytmiliikkumiseen, kirjoitusinteraktioon ja dynaamiseen audiojärjestelmään.

Näiden pääkomponenttien yhteistoiminnalisuus arvioitiin testausvaiheen tulosten perusteella, jotka ratkaisivat opinnäytetyön onnistumisen. Testauksessa käytettiin versiota pelistä, jossa kaikki sen perussysteemit olivat toteutettu. Testaajina toimi pelaajia sokeista täysin näkökykyisiin. Tulokset muodostettiin pelin pääkomponentteihin pureutuvan kyselylomakkeen vastausdatan perusteella. Data tuotti positiivisia tuloksia joka osa-alueella.

Projekti edistyi ideointi -ja tutkimusvaiheesta täysin pelattavaksi kauhuteemaiseksi rytmiaudiopeliksi, joka tullaan julkaisemaan useassa verkkopelikaupassa.

AVAINSANAT:

äänipeli, rytmipeli, kauhupeli, kirjoituspeli, esteettömyys, soolokehitys

TABLE OF CONTENTS

PICTURES

USED ABBREVIATIONS OR GLOSSARY

NPC	non-player-character
DAW	digital audio workstation
CLUT	colour lookup table
LMMS	an open-source digital audio workstation
FMOD	a proprietary sound effects engine for video games
FPS	first person shooter game genre
API	application programming interface

1 INTRODUCTION

This thesis chronicles the Unity game engine development of a rhythm-based audio game, targeted for visually impaired players, while being applicative for the conventional player base. The objective was to join rhythm gameplay, typing interactions and dynamic audio systems into a unique game that is fully playable without visual stimulus.

Typing as an interaction method is rarely used in games nowadays – it is mostly regarded as a relic of old point & click adventure games – however, it offers an interesting contrast activity to fast-paced rhythm gameplay and can be an engaging base game mechanic as the current game, *The Textorcist* has demonstrated by building its shoot-‘em-up gameplay around it.

The theme of the game is horror, and the rhythm is based on the player character’s heartbeat rate, which changes based on player input and dynamic game world events. Inspiration was taken from the game, *Crypt of the Necrodancer*, which featured a similar heartbeat rhythm element. The theme and gameplay mechanics were chosen because of the uniqueness of their combination and their supplementation of the audio game genre.

A stand-alone audio system, that reacts dynamically to game events, was created using the FMOD audio engine. This enabled the use of easily triggerable and customizable sound event parameters and audio timeline transitions, such as loop regions and rhythm management via timeline markers.

Although the game is defined as an audio game, visuals were added to make it more appealing to a wider audience. The visual representation follows game accessibility guidelines and includes colour blind modes, among other accessibility options. The pixel art indie game, *Lone Survivor* served as a visual example and blueprint for the game’s graphic style.

After research, planning, creation, and implementation of the base systems – as well as graphical and audial assets – an early build of the game was sent for testing, to determine the compatibility and functionality of the base game mechanics and the audio system.

In the theory part of the thesis, I will demonstrate the demands of the project, while introducing research data. I will also summarize the histories and mechanics of game genres used to design the game. The testing method will be introduced here also, which ultimately determined the success of the project.

The practical section will start by narrating the implementation of the base game mechanics; moving in rhythm and object interaction by typing in commands announced to the player. The visual presentation and accessibility options will also be shortly touched on here.

Audio design is allocated it's own chapter, as it is the by far largest segment of the project. The chapter starts by explaining the integration of FMOD in the Unity project, it's usage and the implementation of the game's general audio principles.

The last chapters will draw a conclusion based on analysed data acquired from two testing phases. Testing criteria, tester profiles and data analysation methods will be fully explained here.

2 LITERATURE REVIEW

This thesis aims to answer the question: how to design and create a compelling, rhythm-based horror-themed audial gameplay experience for visually impaired and sighted players alike.

Audio games have a special criteria for creating an appealing experience. Immersion is one of the most important aspects of the genre; it will motivate the player to proceed as well as add to the atmosphere of the experience (Pouru 2019). This also applies to the horror game genre, making them a fitting combination. A well-founded interactivity mechanic will add to player engagement also. The distinctiveness of sound cues is naturally important, as well as the intuitiveness of the way the game is played; the player should have a clear goal and execution strategy communicated to them.

2.1 Accessibility

The game was designed from the ground up according to accessibility features and practices described in International Game Developers Association's guide (IGDA 2004), tailored to the project's format.

The resulting game should follow these rules:

1. The game should be fully controllable without a mouse, with logical keyboard inputs.
2. The menu commands must be fully voiced and easily navigated.
3. The typography should be clear, and the in-game speech should be fully subtitled.
4. Colour-blind -and high contrast modes should be included.

By design, the first rule is fulfilled, as the gameplay relies entirely on the keyboard as an input device. The remaining rules also fit the game's premise fluently and would be fulfilled consequently as the game was being developed.

Colour Blindness

Although the game’s focus is on audio, the game should have a full graphical representation too. The graphical assets should be designed to be easily recognizable and to have distinct colorization. The game camera’s viewport should be laid out so, that it would have a very limited field of view, providing a focused play view, and simultaneously evening the playing field between visually impaired and sighted players.

To ensure that the game’s graphical presentation would fit the project’s premise, the inclusion of colour-blind modes was a logical continuum. Colour-blindness is a common visual impairment (Downtown Vision 2020), depending on region and sex, it can affect as high as 10% of a demographic.

For example, the most common colour blindness, deuteranomalous, will result in the loss or partial loss of the colour green, while protanomaly has the opposite effect. To meet the demands of different kinds of colour-blindness’, a colour conversion system, that could fluidly change the colour scheme of the entire game without the need for manual editing of the game’s graphical assets, would have to be created.

To accomplish this, colour lookup tables (CLUT) would have to be utilized. By changing the lookup table metadata of a graphical asset, its colour information can be changed. This procedure would have to be automated and expanded to include all graphical assets at once.

2.2 Combining Game Genres

In this sub-chapter I will shortly summarize the history and gameplay mechanics of each of the game genres used in the project, to give a more comprehensive picture of the game’s influences and inherited mechanics.

Rhythm Games

Rhythm games have historically required a special controller to play, such as a guitar or a dance mat. These games are played by inputting right commands at the right time, following audio and video cues (Webster 2009).

In the late 80s, home game consoles saw their first rhythm music games, sold in a bundle with dance mats. This trend of additional peripherals being required to play, would become a lasting trend with the genre. In the 90s, the genre was dominated by Sony's Playstation console with it's multitude of rhythm titles, the most notable being *PaRappa the Rapper*.

Rhythm and music games had a spike in popularity in the 2000s, followed by the success of the peripheral based *Guitar Hero* series, which captured the attention of players and non-players alike. The wave peaked by the mid of the century however, as the market had become oversaturated with similar games bundled with their own controllers and innovation was running low. The market was rekindled by 2006's *Rock Band*, which featured co-operative play, each player playing a different instrument of a rock band.

Currently rhythm games enjoy a steady regimen of demand with titles like *Beat Saber* and the *Just Dance* series. As the popularity of games like *Crypt of the Necromancer* proves, the genre isn't defined by it's special controller requirement anymore.

As stated, *Crypt of the Necromancer* was a direct influence in the conception of the dynamic heartbeat rhythm mechanic, which functions as the base, which the gameplay is designed around.

Audio Games

Audio games are a small genre of games made for visually impaired players. The genre had it's first mainstream release in the Sega Saturn's *Real Sound: Kaze no Regret*, published in 1997, directed by an innovative game designer and music composer, Kenji Eno (Campana 2016). The game is a fully narrated story, void of graphics, where the player makes choices, affecting the path the story will take. The other mainstream publications include games for the Game Boy Advance handheld console and for the iPad tablet. The main bulk of pure audio games are small indie game titles, their gameplay and style varying vastly.

Another chapter entirely are games not targeted for the visually impaired, but which still end up being accessible, either by accident or by including additional accessibility options. Fighting games such as *Mortal Kombat* are a good example of this.

Defining a game as an audio game – as opposed to a video game – sets it up for certain requisites. First, the game must be fully playable with audio feedback only. Second, the soundscape of the game must be rich, while maintaining a dependable clarity, as gameplay actions are performed based on audio cues. Also, the flow of the game must be logical and predictable.

Horror Games

The horror game genre is almost as old as video games themselves, the first notable title, *Haunted House*, dating back to the early 1970s. A version of the game was ported to the Atari 2600 home console in 1982. The game had the player navigate a haunted house, avoiding enemy encounters to survive, and as such, became the first survival horror game (Sinha 2016).

The 1990s saw the rise of survival horror with titles such as *Alone in the Dark* and *Resident Evil*. These games featured a fixed camera angle, designed to disorient the player. The management of a very limited inventory was a key element also, making backtracking an intrinsic part of the experience. In survival horror games, saving the game is usually restricted to appointed safe room locations, to increase suspense the further the player proceeded into the unknown.

1993 saw the release of the first-person shooter *DOOM*, which leaned heavily on the horror aesthetic, while retaining a heavily action-based gameplay.

In the 2000s and 2010s, the genre was re-invented by the first-person game *Amnesia: The Dark Decent* and it's prequels, the *Prenumbra* series. These games emphasized player helplessness, forcing the player to run away – rather than fight – when encountering an enemy.

Horror, as a long-lasting game genre, offers a well-established framework to build on. It supplements the audio game genre, as horror games are naturally audio focused. The uncanny effect (Mark Nicholas 2009, 5) was applied when designing the sound effects of the enemy character, to disorient the player by producing close to human, but not quite, sounds.

Choosing elements best suited for the project would be an on-going task during development, but the basic blueprint was clear from the beginning; the player must travel from a locked safe room to another, avoiding the enemy on the way by manipulating the environment and using items collected from the game world to proceed. The chosen approach exploits many of the main tropes of the survival horror genre; item management, avoidance and creating tension by limiting the game-saving capability.

Typing Games

Typing games are generally played by typing in announced commands correctly and as fast as possible.

The typing game genre is mainly populated by mini-games and educational titles. There are a few exceptions to this rule, such as the 2000 typing-rail-shooting hybrid, *The Typing of the Dead* and the more recent *The Textorcist,* which inspired the use of typing as an interaction method for the project.

The esoteric nature of these kind of games in today's game industry provides an exotic method for executing player actions. By establishing typing as a base mechanic beside rhythm movement, the game will possess a unique blend of gameplay.

2.3 Rhythm Gameplay

Rhythm gameplay functions as a suitable base mechanic for an audio game, as it supplements the genre intrinsically. By establishing the beat as the player's heartbeat, a horror element is introduced to the base of gameplay fluently, as the tempo of the heartrate will determine whether the player lives or dies.

Rhythm games usually lean on visual elements to help players stay on beat (Pichlmair, Martin, Kayali, Fares 2007), but can be converted into an audio format by diminishing the visual aspect.

By focusing on audio, it becomes imperative to communicate tempo changes to the player clearly, this calls for a special audio structure for the beat, adjusting its pitch and volume, while adding new percussive effects as the tempo changes. It should be noted that haptic feedback provided by some game controllers is ruled out, as the keyboard is required for textual input.

All in-game events should happen in rhythm with the beat, including player and non-player-character (NPC) movement. This establishes a consistent, foreseeable logic to the game’s flow, allowing the player to rely on it. To further reinforce the importance of the beat, other in-game sound events, such as voice commands, should play in sync with it.

As human reaction time averages on 15ms on audio cues (Backyardbrains.com, Jain et al. 2015), an accurate input detection system is mandatory to ensure a responsive gameplay experience.

After an extensive research and testing period, an accurate enough method for detecting input was found in J.Leaneys open-source Unity package EasyRhythm for FMOD. It simplifies FMOD’s callback system and allows Unity code to be triggered based on beat, by using a FMOD Studio timeline event’s beat or time value.

2.4 Typing as a Game Mechanic

Part of the game’s challenge stems from beat-dependent scenarios, where the player must type in exactly an announced command to proceed. An example scenario would be the opening of a door by typing in a command, while an enemy is getting closer by the beat. If the command is misspelled, it is re-voiced, and typing must be started from the beginning.

Research suggests, that typing errors in English typing games are most caused by incorrect operation of the keyboard rather than due to phonological causes (Tachibana and Komachi 2016), which was considered when designing the phrasing of the voice commands.

The words of a voice command should start on beat and their volume should correspond with the current tempo value, always keeping them in the foreground of the soundscape.

Textual Input and Visually Impaired User Interaction

Typing was elected as an interaction method because it adds to the immersivity of a gameplay experience compared to a choice-type approach (Mehta et al. 2010), as immersion is a critical aspect of both horror -and audio games. Inspiration was taken from the game *The Textorcist*, which demonstrated that typing could function as an engaging primary gameplay mechanic.

Touch typing – knowing the location of the keys on a keyboard by muscle memory – is an essential skill for any blind computer user, and thus, establishing typing as an interaction method, re-enforces the game's the audio game premise and acknowledges its main target audience as the visually impaired.

Many promising, forefront audio game interaction methods, such as kinetic movements or voice recognition, were discarded, as it was concluded, that the scope of the project and its solo development nature called for an already established, moderately easily implementable solution.

2.5 Dynamic Audio System

To provide descriptive and accurate information about the surrounding game world to the player, a versatile audio system with spatial capabilities must be developed. The requisite is, that the game world can be navigated solely by audio signals.

The layering of sounds should be done so, that much of the frequency scope of human hearing is utilized (Purves and others 2001, 149), creating a rich yet uncluttered soundscape, where essential sound effects are always prioritized to the surface.

The requirements of this system are:

- audio events should have optional, triggerable states that can hold different sounds
- audio events can interface with a global, varying beats-per-minute value, changing their characteristics proportionately, like raising their pitch value
- the sound properties must be easily modifiable, for example, adding a lowpass filter based on the sound's surroundings in the game world

Audio icons and earcons as defined in the paper by Dingler et al (Dingler et al 2008). will be used to distinguish important actions such as entering and leaving a game object's interaction zone. Associating helper sounds with certain actions will give the player unintrusive feedback on what is excepted in the current gameplay situation.

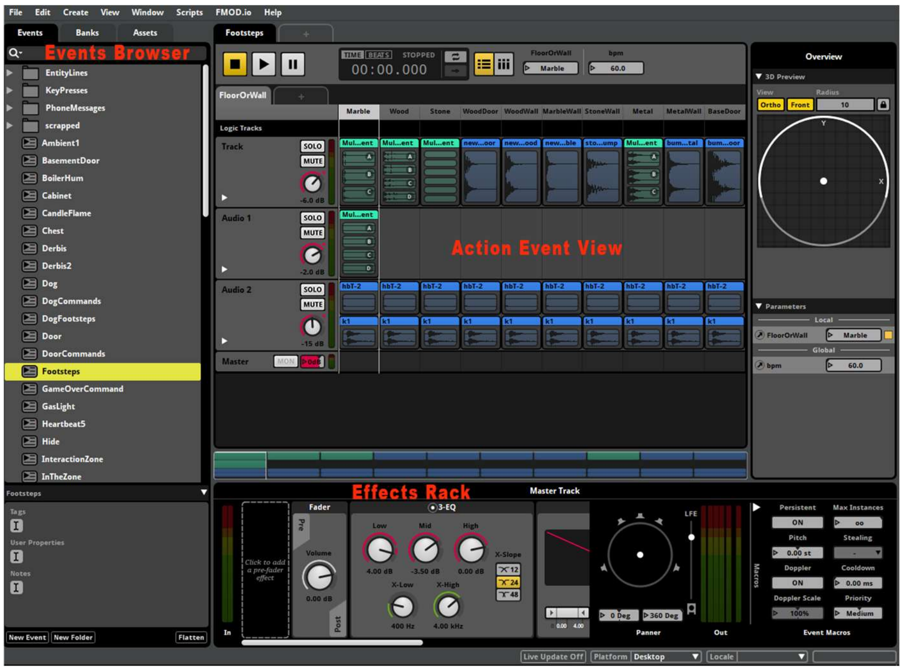
Earcons will be utilized to emphasize in-rhythm sounds, such as footsteps and heartbeat sounds, by mixing slight musical instrumentation in with these effects. With these tactics employed, the game will have more intuitive flow to the non-visual user.

Audio Engines

As Unity’s inherent audio system does not meet the above requirements out of the box (Takanen 2020), a replacement system had to be chosen. The solution was to replace Unity’s audio engine with a middleware audio software. There are a few professional middleware systems that fit the bill, but in the end, in addition to FMOD, only Wwise had the capacity and attributes required by this project.

Wwise, a powerful and well-known audio system, that is known to be difficult to get into – as it’s interface differs greatly from the standard of a digital audio workstation’s (DAW) – have been used in games as 2016’s *DOOM*. The steep learning curve combined with the software licence’s price for a commercial project made me lean toward FMOD.

FMOD’s application programming interface (API) integrates fluidly with Unity, and it’s sound authoring tool’s, FMOD Studio’s, interface (picture 2) was easily absorbable because I’ve had some experience with DAWs in the past. The simple interface hides beneath it a wealth of complexity and powerful features. FMOD is free to use even for commercial projects developed by small teams (Abbott 2018). This in tandem with it’s familiar interface made me decide to go with it.



Picture 1. FMODS Studio’s interface.

FMOD Studio has modulation and automation features which allow simple randomization of a sound’s properties, such as volume or pitch. This helps in creating a more variable soundscape effectively.

FMOD Studio sound events can be either action -or timeline events. Action events can be assigned local and global parameters that can be triggered from code. For example, a footstep event can be assigned a parameter which holds a value and a sound for each different floor type. Timeline events have marker and transition systems in their disposal and can be mixed fluidly using Studio’s interface. In addition, each type of event can be assigned postproduction effects easily using the effects rack.

Spatial Sound and Audio Occlusion

The game is visually represented as a two-dimensional side scroller. Research shows, that visually impaired individuals have a highly developed skill of pinpointing a sound’s location on a 2D map (Després, Candas, and Dufour 2005).

When audially communicating the game world’s map to the player, steps would have to be taken to represent a game object’s distance and height in relation to the player in a clear and accurate manner.

The audio listener of spatial sounds should be the camera, which is located close to the centered player character.

The 2D setting of the game simplifies the implantation of spatial audio, because one of the common errors regarding to spatial audio is confusing sounds emanating from the front with those emanating from overhead (Gardner 1999). By removing the depth dimension, the sound's source is left with only horizontal and vertical possibilities, thus streamlining it's localization.

A sound emitting game object's occlusion, with respect to the player, must also be distinctly communicated, as the gameplay relies on avoidance and environment manipulation. There is no need to differentiate an audio source's obstruction level; to keep the game's audio logic straightforward and clear, an audio source is either fully occluded or not at all. Occluded sounds should be easily identified as muffled, void of high tones.

2.6 Game World and Story Progression

A story encapsulates a game as a narrative experience with a clear beginning and an end, but implementing it is a balancing act between interactivity and plot (Majewski 2003). To keep a game's flow uninterrupted – naturally important for rhythm games – the storytelling methods should be as subtle as possible. Environmental storytelling is a fitting vehicle for this, as it keeps gameplay in the driver's seat, so to speak (Fernández-Vara, 2011).

This can be accomplished by embedding audial story elements into game world objects, such as, into an enemy character, as vocalizations that hint into its origins. The environment itself should audially and visually convey the story's premise, for example, by presenting clues of it's history. This concept could be reinforced by making the player character's beating heart an integral part of the story.

As stated, the story elements should minimally impede on gameplay and the flow of the game. This in mind the architecture of the levels should be designed as clear, compact spaces, relatively easy for the player to navigate.

2.7 Project Objectives

The goal of this thesis is to create an audio game that combines rhythm, horror and typing game genres. The unison of these genres is unexplored territory and as such, fertile ground for experimentation and testing.

The game will be developed from start to finish as a solo project for a myTrueSound game company and released on various online game marketplaces.

It was decided from the start, that the published version would function as a base, to which content - presuming it uses the same base mechanics - could be easily added. This in mind, movement in rhythm, typing interactions and dynamic audio were planned as modular systems. The final product should be an easy-to-add interface which to build on.

The success will be measured by the functionality of the game’s base systems; rhythm gameplay, typing interactions and dynamic audio. As an overall gameplay experience is not objectively measurable (Archambault et al. 2007), specific parameters must be determined to evaluate the success of the project. The testing chapter is dedicated for this subject.

3 IMPLEMENTATION

3.1 Conception

After an ideation and planning period, a fleshed-out framework of the project was presented for the commissioner, myTrueAudio, and was accepted. The resulting game would be a two-dimensional game, viewed from the side, where movement would happen in rhythm with a dynamic heartbeat sound, and game object interaction via typing in announced commands with a keyboard. This broke project development into three base sections:

1. create an accurate, compelling moment-to-moment rhythm gameplay
2. establish typing as an interaction mechanic
3. create a dynamic audio system that communicates all essential gameplay and navigational information to the player

The challenge rose from combining these components into a functional, interconnected whole and setting it inside the horror aesthetic, while – if all goes well – creating a unique, game, that has appeal for both visually impaired and regularly sighted players, all the while keeping in mind the requirements of game accessibility (Grammenos, Savidis, and Stephanidis 2009).

As this was a solo project with a six-month development cycle, it was imperative to create a detailed project plan to keep progress steady. To monitor progress, it was decided that bimonthly meetings with the commissioner would be held. These proved to be valuable sessions, as they shed light on various shortcomings with the current state of the project and in turn, highlighted what was working, while presenting me with new ideas and refining the path to proceed.

I assigned Csikszentmihalyi's components of enjoyment in his 1990 *flow* research (Csikszentmihalyi 1990) as personal guidelines as for what to strive for in regards of a player's gameplay experience:

1. Tasks with reasonable change of completion
2. Clear goals
3. Immediate feedback

4. Deep but effortless involvement that removes from awareness the frustrations and worries of everyday life.
5. Sense of control over our actions
6. No concern for the self
7. Alteration of the concept of time, hours can pass in minutes and minutes can look like hours.

When in a rut with a design decision, the above principles offered a grounding solution, that always in hindsight seemed like the right one.

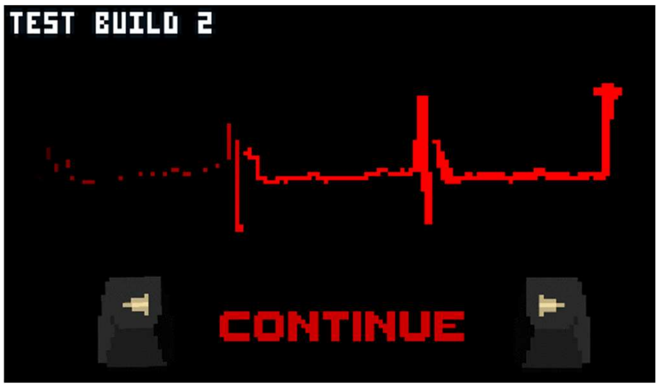
A Summary of the Final Product

The player can move left or right in sync with a dynamic heartbeat in a 2D-sidescrolling environment, out of sync actions raise the heartrate value and staying still, in turn, lower it. Interactions with game world objects are performed by typing them in with a keyboard, when manually narrated commands are presented to the player.

To progress, the player must find a way to a safe room, where, after interacting with a radio object, progress is saved, the next level loaded and the way backwards blocked. The levels are compactly built, usually introducing and teaching a new gameplay mechanic.

The challenge commonly lies in finding a safe room key and avoiding an enemy by listening its movement sounds, steering its patrol path by opening and closing doors and using a trap to hider its movement. For visually limited players, memorization of the level map is an important part of the trial.

The full game will have seven levels of varying difficulty and complexity, as well as an intro and an outro sequence. The fully-voiced main menu is controlled by the left and right keys (picture 2), and holds an accessibility sub-options menu.



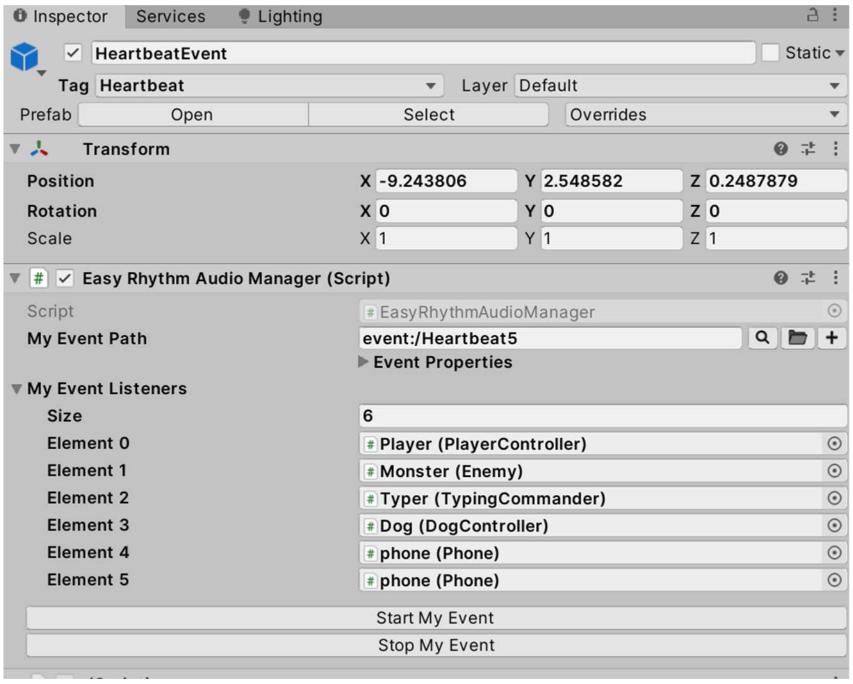
Picture 2. A screenshot of the main menu.

3.2 Movement in Rhythm

The heartbeat is the pivot point of the game, as player and NPC movement are tied to its rhythm.

To ensure lag-free, responsive movement, EasyRhythm for FMOD developed by J.Leaney was used. This solution allowed me to configure the press time window for player input in FMOD Studio by using its tempo marker system (picture 4).

To integrate EasyRhythm with an FMOD event, an Easy Rhythm Audio Manager component must be added to a Unity game object. Then, the FMOD event chosen, and the scripts using the rhythm system assigned into My Event Listeners array (Picture 3).



Inspector Services Lighting

HeartbeatEvent Static

Tag Heartbeat Layer Default

Prefab Open Select Overrides

Transform

Position	X -9.243806	Y 2.548582	Z 0.2487879
Rotation	X 0	Y 0	Z 0
Scale	X 1	Y 1	Z 1

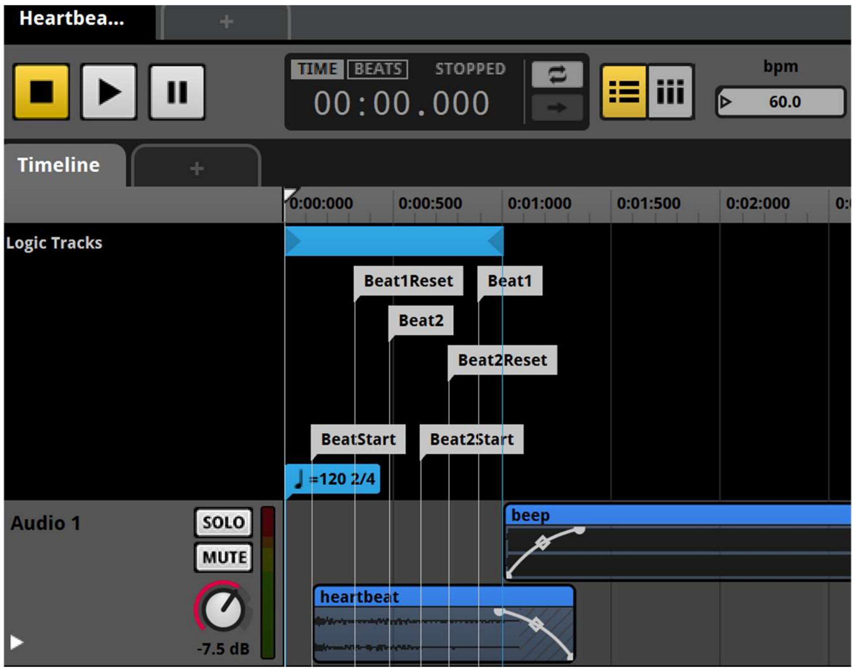
Easy Rhythm Audio Manager (Script)

Script	EasyRhythmAudioManager
My Event Path	event:/Heartbeat5
	Event Properties
My Event Listeners	
Size	6
Element 0	Player (PlayerController)
Element 1	Monster (Enemy)
Element 2	Typer (TypingCommander)
Element 3	Dog (DogController)
Element 4	phone (Phone)
Element 5	phone (Phone)

Start My Event

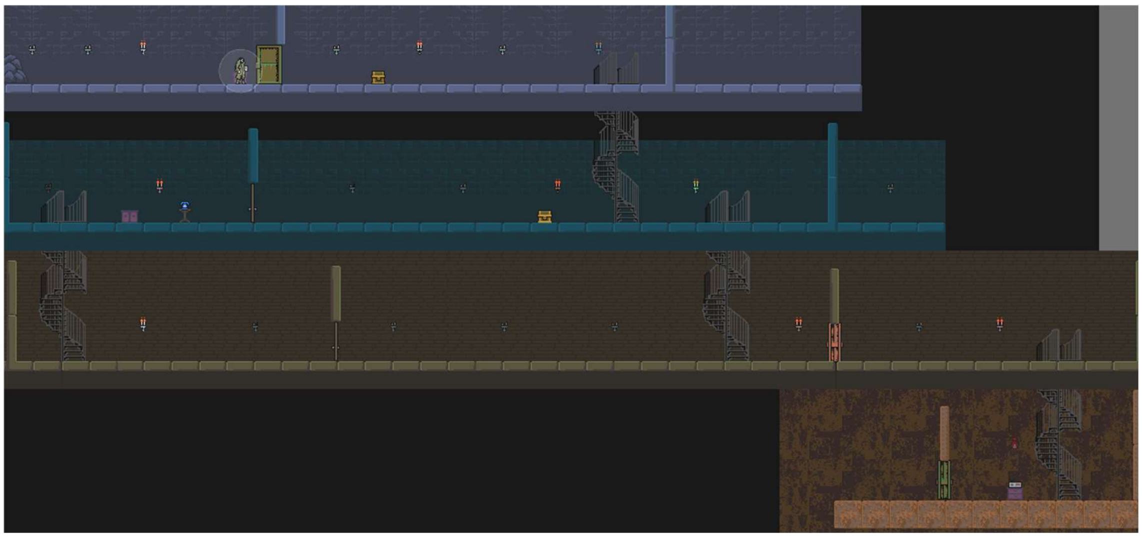
Stop My Event

Picture 3. A game object containing an EasyRhythm component, with scripts assigned that use the Heartbeat5 FMOD event's markers as triggers.



Picture 4. FMOD Studio markers used as triggers in Unity scripts.

The game environment is a grid of 32x32 pixel blocks, where movement happens from block to block instantly in rhythm of the beat. Each environment block was designed as an independent piece, which could be easily added or removed, without dependencies to global game systems.



Picture 5. A map of a level, as seen in Unity editor.

The beat rate changes are communicated to the player as inhalation and exhalation sound effects, as events or missed actions rise – and staying still – lower the rate.

In FMOD Studio, I created a global beats-per-minute parameter for the heartbeat event, which could be controlled via scripts in Unity. As the parameter is lowered or raised, it dynamically alters the tempo, pitch and volume of the event. The parameter being global, allowed other FMOD events be tied to its value and changing their properties accordingly.

The resulting rhythm system was easily configurable by visually changing the tempo markers in FMOD Studio and provided accurate, lag-free input.

3.3 Typing Interactions

The player interacts with usable objects in the game world by typing in commands, which are presented visually and aurally when the player moves into an interaction zone of such an object. When the command is typed in correctly a corresponding interaction is executed, for example, a door is opened.

The typing system consists of two main scripts: a word checker and a voice commander.

The word checker script gets a key input and verifies it against the current character of a selected phrase. The commander script holds references to all FMOD voice command events, selects the right one and plays its words in rhythm with the heartbeat, by the help of the EasyRhythm system.

The intractable objects themselves were created as homogeneous Unity prefabs, each inheriting an Interactable script interface that holds their defining base functions and handles communication to the voice commander script.

This way, the blueprint of each interactable object is identical and new ones could be easily added later.

3.4 Progression and World Events

A state machine script controls the enemy's default behaviour, also allowing the firing of special states based on world events. By utilizing the EasyRhythm interface, the enemy's movement is also synced with the heartbeat rate. The enemy's control script includes a chase function, making it move double-time if the player crosses it's field of view, giving the player only a short window of escape. When the player is in the unobstructed vicinity of the enemy, the heartrate is set to raise progressively, adding pressure to enemy encounters.

To allow for different tactics when facing the enemy, four mechanics were implemented:

1. Hiding. By pressing up on beat and holding the key down, the player object's layer is changed and thus becomes invisible to the enemy, continuously rising the heartrate value. This, in addition to a heartrate increase caused by an enemy's proximity, hiding only grants a brief shelter.
2. Changing floors. To avoid the enemy, the player can climb stairs up and down by typing in a correct voice command. This emphasizes the importance of observing enemy audio cues, because, for the evasion to succeed, changing floors must be done when the enemy is moving away from the stair's destination location (assuming it is on the same floor).
3. Setting a trap: A reusable trap can be set by pressing the down key on beat, assuming such an item has been picked up. Later in the game as the enemy AI becomes more aggressive, the player must hinder it's movement to escape.

4. Closable doors. Doors can be closed to confine the enemy's patrol route. This is encountered later by triggering more unpredictable states in the enemy's AI script.

The dog, which serves as a motivator for the player to progress, has simple mechanics. It spawns in pre-designated locations at pre-determined times, giving off random sounds chosen from a multitrack pool based on it's state.

Level Architecture

While the objective of the game stays identical from level to level – find a key to open a way into a safe room – the level architecture and interaction commands get more diverse as the game progresses, attempting to keep gameplay fresh from level to level.

For example, in later levels, a reusable trap object is introduced. First the player is taught the trap usage in a simple, yet hurried scenario, but in the next, multi floor level, the trap must be used and reused in a more extensive way, as the enemy AI is made more versatile and aggressive.

Storytelling Methods

Two main in-game methods of storytelling were used:

1. Story snippets fused into tutorial messages: Each level holds one or more phone objects, which, when interacted with, give the player a tip how to beat the current level with some subtle story telling mixed in, trying to keep the immersion intact.
2. Monster sounds: The enemy, which is present most of the game, was assigned voice lines that give an attentive player hints of it's origin.

3.5 Visual Presentation

Pixel-art graphical style was chosen because of its simplicity and for its relatively fast creation rate, so more time could be allocated to more important areas such as audio design.

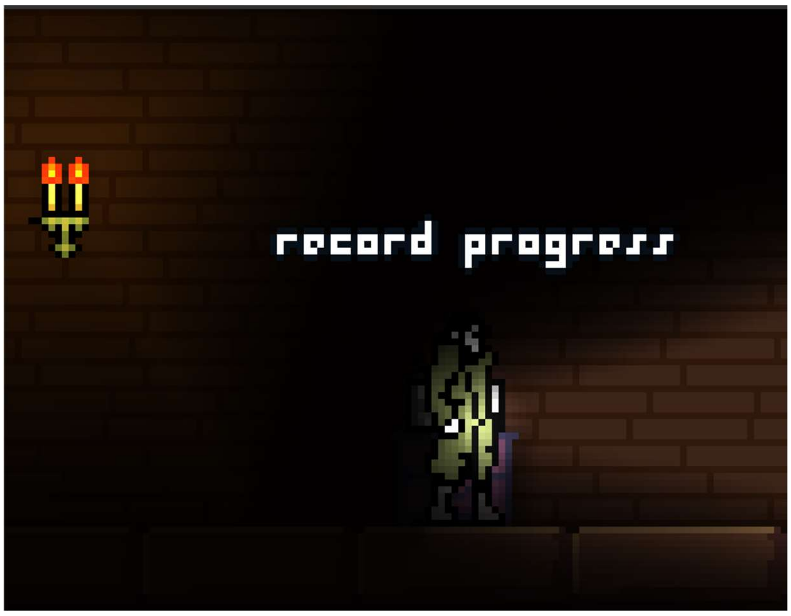
Pixel Perfect Camera and 2D-lights

Unity’s 2D Renderer was chosen because of its new 2D lightning capabilities.

To accomplish a uniform style, Unity’s Pixel Perfect Camera package was implemented. The package allows genuine, low-resolution graphics to be scaled accurately to high definitions while keeping the pixels snapped correctly.

The package comes with a Shadow Caster 2D component, which in tandem with the 2D renderer’s 2D light system, allows the creation of dynamic shadows and lights quite easily.

The lightning system was an important addition as it complements the horror theme and game flow, keeping out of view objects obscured and thus normalizing the gameplay experience between visually impaired and non-impaired players.

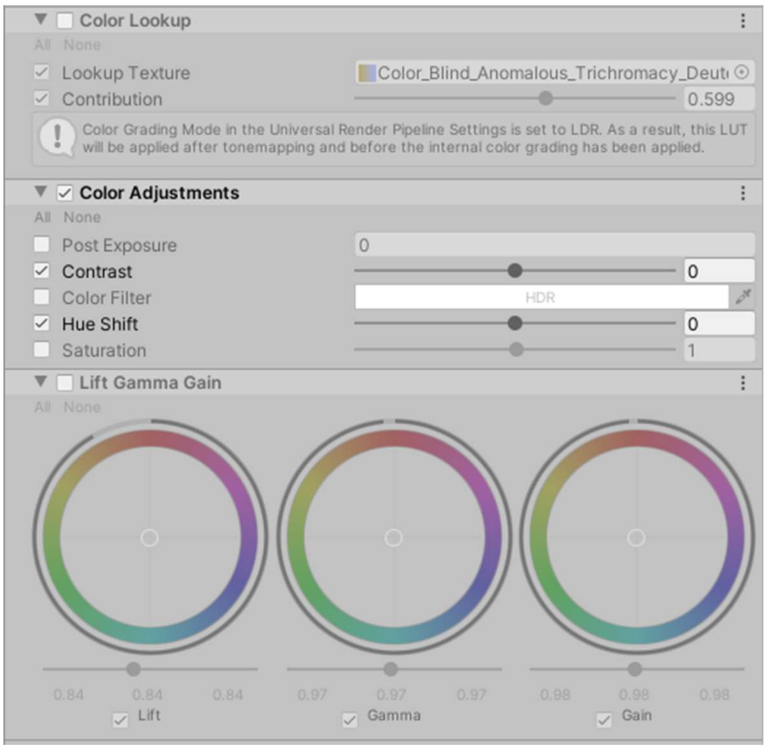


Picture 6. A screenshot of the game view.

3.6 Accessibility Options

The menu system was designed to be fully usable without the visual component, applying voice descriptions to each button and by adopting a side scrolling menu structure.

Optional colour-blind modes were implemented by utilizing Unity’s post processing system’s Colour Lookup component (picture 7). Several togglable colour lookup tables (CLUTs) for different types of colour-blindness – such as deuteranomaly or tritanomaly – were included.



Color Lookup
All None
✓ Lookup Texture Color_Blind_Anomalous_Trichromacy_Deut
✓ Contribution 0.599
Color Grading Mode in the Universal Render Pipeline Settings is set to LDR. As a result, this LUT will be applied after tonemapping and before the internal color grading has been applied.

✓ Color Adjustments
All None
Post Exposure 0
✓ Contrast 0
Color Filter HDR
✓ Hue Shift 0
Saturation 1

Lift Gamma Gain
All None
0.84 0.84 0.84 ✓ Lift
0.97 0.97 0.97 ✓ Gamma
0.98 0.98 0.98 ✓ Gain

Picture 7. CLUT component of Unity’s Global Volume game object.

Changing the colour mode by applying different LUTs was an effective solution, as manual colouring of game assets would have been a time-consuming process.

4 DYNAMIC AUDIO SYSTEM

4.1 FMOD Setup

FMOD audio engine was chosen to manage the game's audio because of it's extensive professional utility, DAW-like interface and free of charge availability.

When setting up FOMD with Unity, it is strongly recommended to use the automatic install option. Attempting manual install led to an extended troubleshooting period.

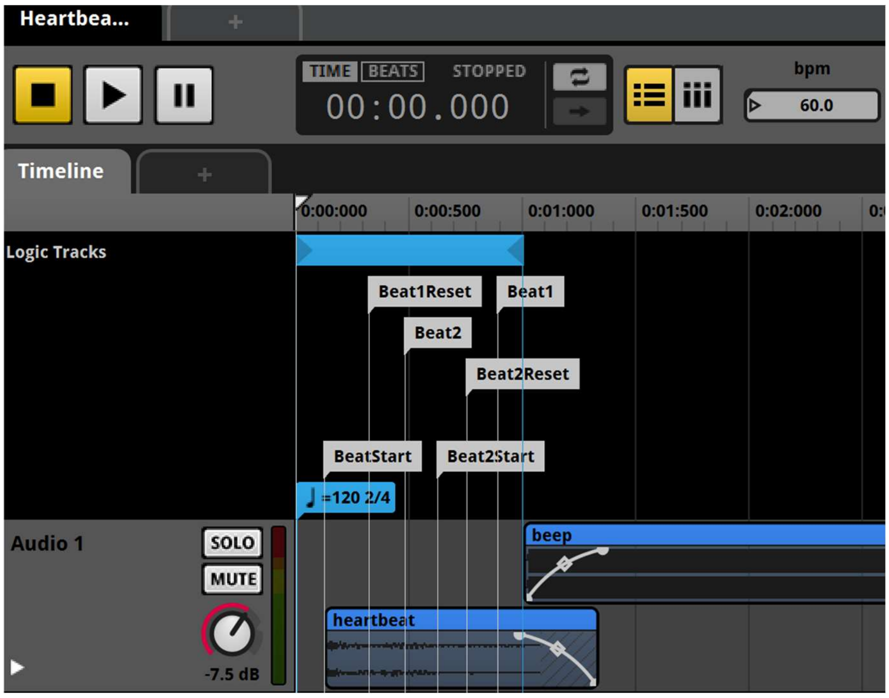
FMOD's additional Studio application has many DAW-like features, like pre- and post-effects, spatial audio, and extensive mixing capabilities. It's graphical interface is relatively easy to use, if one has some DAW experience.

4.2 FMOD and EasyRhythm

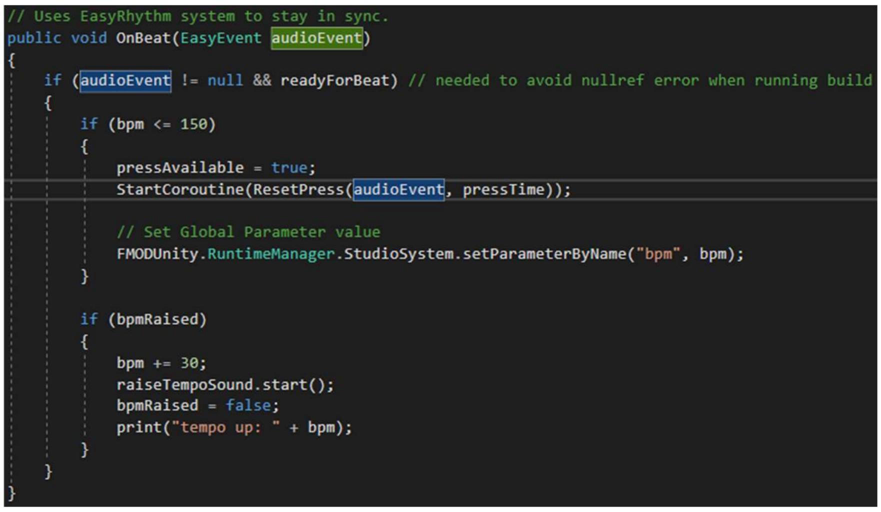
FMOD's most important feature though, as regards to this project, was it's dynamic audio parameters. By utilizing this feature, audio events could be fluidly adjusted via Unity scripts, creating an interconnected, dynamic audio system.

A global FMOD parameter for the heartbeat's bpm value was created. A majority of the gameplay sound events would derive this parameter, and it would determine the events' volume and pitch values. In game, when the heartrate is raised, volume of other events, such as footsteps and voice commands, is raised also, creating a more intense mood.

The heartbeat was created as a FMOD timeline event (picture 8), to utilize the marker system with EasyRhythm (picture 9). This way, as the bpm value raises, the press window of a beat is automatically adjusted.



Picture 8. Heartbeat event holds markers that define the press window for player input and other on-beat game world events, such as NPC movement.



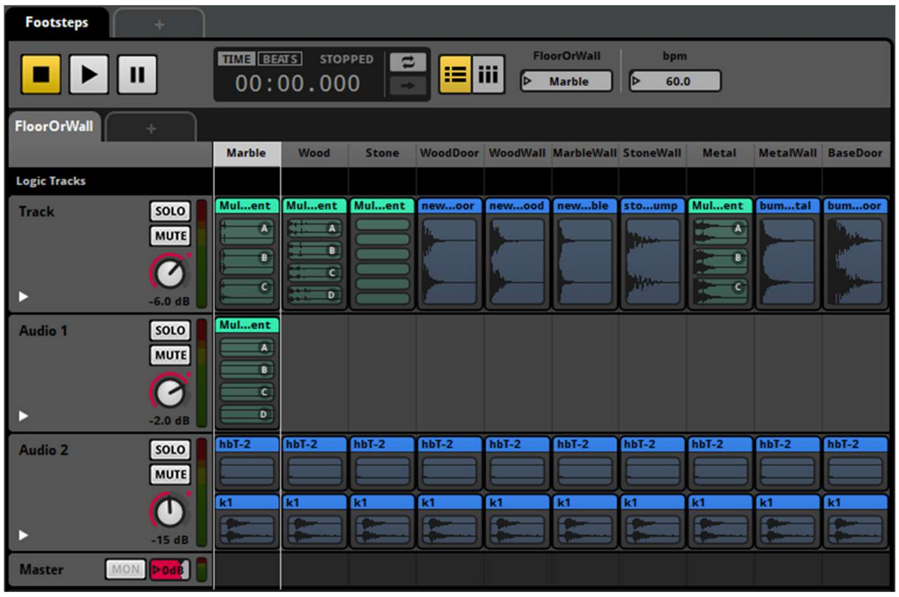
```
// Uses EasyRhythm system to stay in sync.
public void OnBeat(EasyEvent audioEvent)
{
    if (audioEvent != null && readyForBeat) // needed to avoid nullref error when running build
    {
        if (bpm <= 150)
        {
            pressAvailable = true;
            StartCoroutine(ResetPress(audioEvent, pressTime));

            // Set Global Parameter value
            FMODUnity.RuntimeManager.StudioSystem.setParameterByName("bpm", bpm);
        }

        if (bpmRaised)
        {
            bpm += 30;
            raiseTempoSound.start();
            bpmRaised = false;
            print("tempo up: " + bpm);
        }
    }
}
```

Picture 9. A code snippet demonstrating the usage of EasyRhythm to open a press window for player input.

Labelled parameters (picture 10) were used with many sound events, to select the right sound for the current situation.



Picture 10. Using labelled parameters to select right footstep -and wall bump sounds.

4.3 Work Flow

Zapsplat.com’s sound banks were initially used for majority of the sounds effects of the game, though they ended up being remixed or replaced along the way with more suitable, self-recorded sounds.

All recording was done with a fast and easy to use ZoomH6 device, so new sound effect could be recorded with short notice.

Attention was put into the musicality of the sound effects, so they would emphasize the rhythmicity of the game. This was accomplished by exporting percussive instrument sounds from LMMS DAW, using a Roland Sound Canvas plugin and mixing them with the recorded sound effects.

After basic normalization and noise reduction treatment, done in the Audacity program, sound files were imported into FMOD Studio as events and given the needed parameters.

Creating, playing, stopping and releasing events via Unity scripts was done through FMOD API’s namespaces.

4.4 Audio Occlusion and Spatial Sound

To create a dynamic soundscape, an occlusion system was created for sound sources, lowering their volume and applying a low-pass filter value based on their distance and surroundings in the game world.

This was accomplished by using Unity’s raycast system in tandem with FMOD’s Spatializer plugin and property automation feature. It was necessary to tweak the Spatializer Sound Size and Minimum Extent values to match the game’s 2D environment.

The functionality of these systems was imperative in terms of the goal of the game, and they were given special attention during testing phase.

4.5 Sound Design

General soundscape was designed to be clear of any obtrusive, non-gameplay affecting sounds, so all the sounds the player hears are of importance. This helps the player to, for example, pinpoint an enemy’s location and monitor it’s patrol path by sound, be the enemy in the next room or a floor below.

Environmental sounds, such as thunderstorm sounds, were added for flavour and to make navigating easier for visually impaired players. The principle became, that, if a visual element was to be added to the game world, like a candle or a window, it should have an audial component as well.

As safe rooms serve as checkpoints for the player, they were assigned a distinct, soothing music to indicate where the next objective lies. The safe room music was created using LMMS synthetization plugins. It was designed not to contain any clear lead instruments, to keep it rhythm independent, and as such, would not feel out of sync with the varying heartbeat.

Voice Commands and Tutorial Messages

To add variety to the voice commands, each line was recorded multiple times and added to a FMOD Studio event’s multitrack pool, from where sounds are randomly selected when triggered. To further emphasize this, a randomized highpass filter value was set up for each multitrack.

Interactable phone objects were added to communicate a story to the player, while simultaneously teaching new game mechanics. These were mixed using Audacity and voicechanger.io web page to fit their designed story origins.

5 TESTING AND ANALYZING DATA

To measure the functionality of the game's base systems, a game build with all components implemented was tested by visually impaired players of various states, ranging from non-visually impaired -to blind users.

5.1 Gathering Data

A questionnaire regarding the three main areas of the game was presented to the testers:

1. responsiveness of rhythm gameplay
2. the co-operation of typing and voice commands as an interaction method
3. the serviceability of the dynamic audio system as a navigation method

Each area was divided into sub questions, exposing the root causes for possible issues. Free discussion was also encouraged to get additional feedback and suggestions for improvement.

Unity Analytics was also enabled for the project, contributing additional data about specific gameplay events, such as level completion times and percentages, as well as dying statistics and typing command actions.

5.2 Scope

It is important to define the limits of the gathered data. The scope of this thesis is to evaluate the functionality and compatibility of the three main game systems, not to rank the subjective, overall gameplay experience of the testers, although such data is of value for the commissioner.

5.3 Testing Builds

Testing was done in two periods, by sharing subsequent test builds on Audiogames.net forum, to accumulate a sufficient data pool.

At first phase, a three-level game build with all base mechanics implemented was shared. For the second phase, the build was refined and re-released as a five-level version to generate more discourse and test data.

5.4 The Questionnaire

Along with the test builds, a questionnaire was shared. The questionnaire was designed so, that it fixates on the core components of the mechanics, keeping it terse and to the point, thus securing a maximal follow-up count.

The questionnaire is divided into four parts. The first part holds questions about the rhythm movement mechanic; the sufficiency of the press window and the sense of input lag. The second part focuses on the typing mechanic; its responsiveness and sound clarity. The third part centres around the dynamic audio system and the players' experience with the game's spatial soundscape. Part four inquiries about the testers' visual acuity, categorizing the compiled data.

5.5 Unity Analytics Data

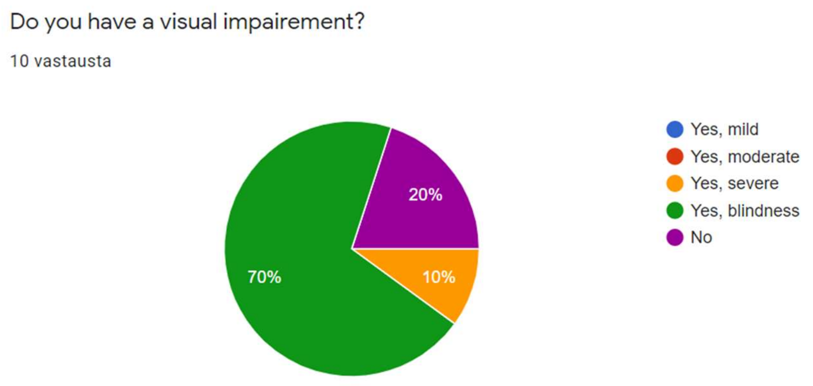
Unity Analytics allows high-level monitoring of player quantity and use cases. Custom analytics events were implemented to trigger on certain conditions:

- when a player completes a level: the time and scene id were saved
- player death: scene id is saved
- typing command completion: scene id and completed command text were saved

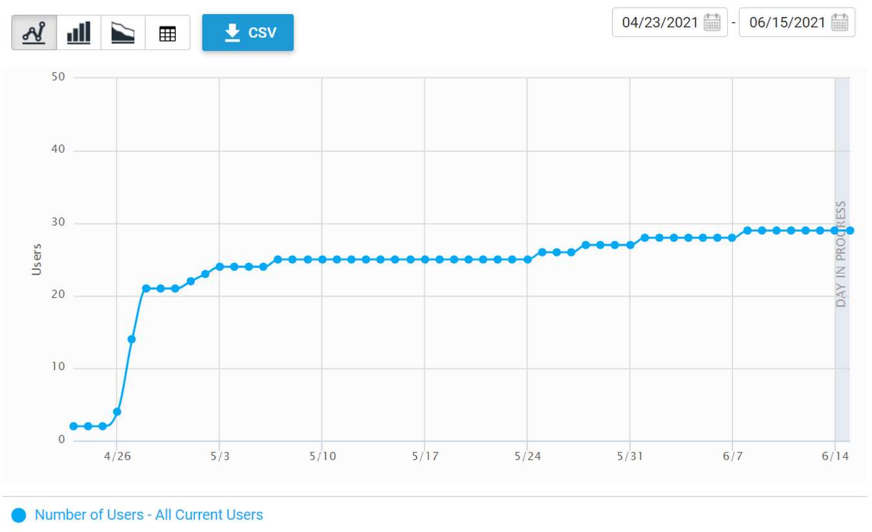
These events were created to give a substructure to the questionnaire answer data.

5.6 Analysing the Data

Tester Profiles and Unity Analytics Data



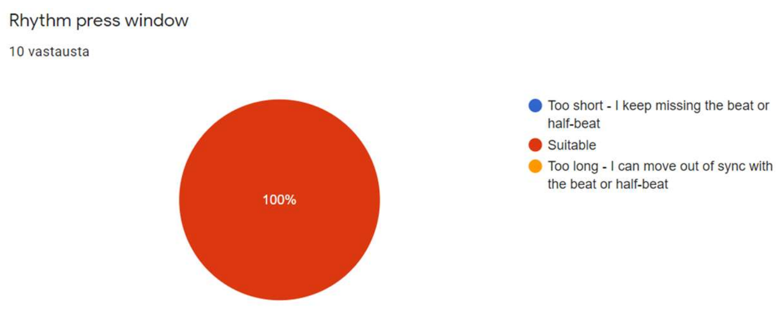
Picture 11. Research questionnaire answerer profiles.



Picture 12. Unity Analytics, total number of testers.

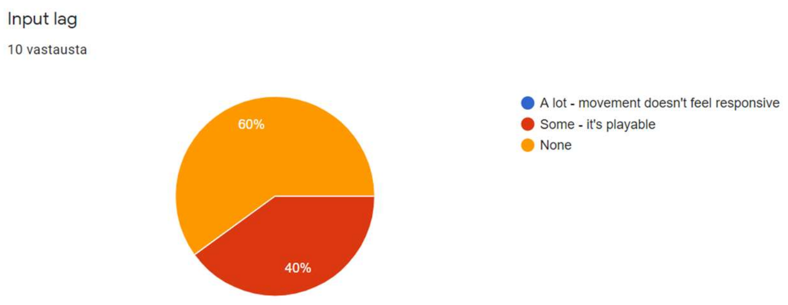
The total tester count of the two testing periods was 30, of which 33% participated in the questionnaire, providing a modest data pool to draw from.

Rhythm Gameplay



Picture 13. Results of research question 1: Is the window for key press sufficient?

Based on the results, the movement key press window is unambiguously the right length. This was balanced throughout development. The press window was implemented so, that it’s length varies commensurately with the heartbeat’s bpm value.



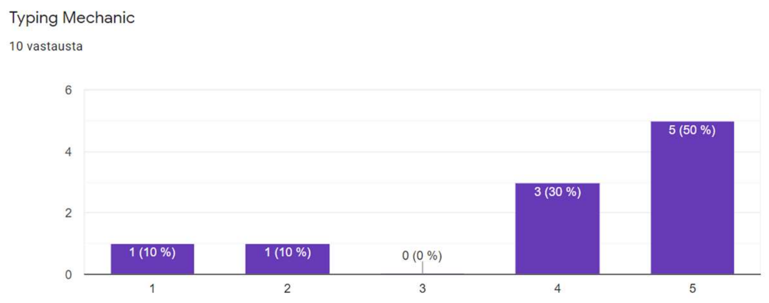
Picture 14. Results of research question 2: How much input lag did you experience when moving on beat?

Here, the outcome is more divided, as 40% of the testers experienced some input lag. Some of this could be explained by the input lag inherent in the modern flat screen monitors. This, however, does not explain the whole dataset, as 70% of the testers were blind.

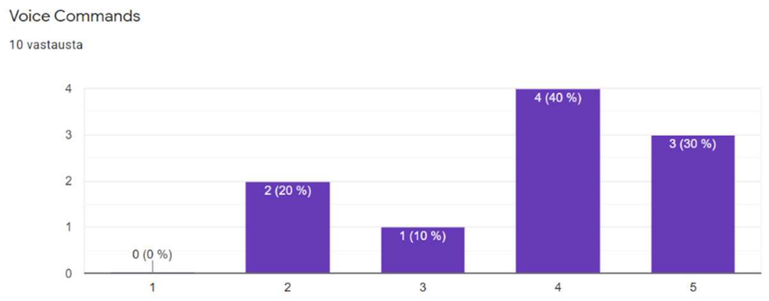
Another common cause for slight input lag is the usage of wireless devices, in this case a keyboard. The most probable issue is the usage of dated hardware and/or software. Before public testing, the game was tested with a five-year-old, low-end PC without issues. Compatibility issues can arise from a multitude of places, however; from device driver issues to physical faults in input peripherals.

In any case, if the project is continued after the game's launch, this area requires further examination, as it is imperative that the rhythm movement feels receptive to the player.

Typing Mechanic



Picture 15. Results of research question 3: How responsive is the typing mechanic? Range: from (1) not responsive to (5) responsive. Average score: 4.0.

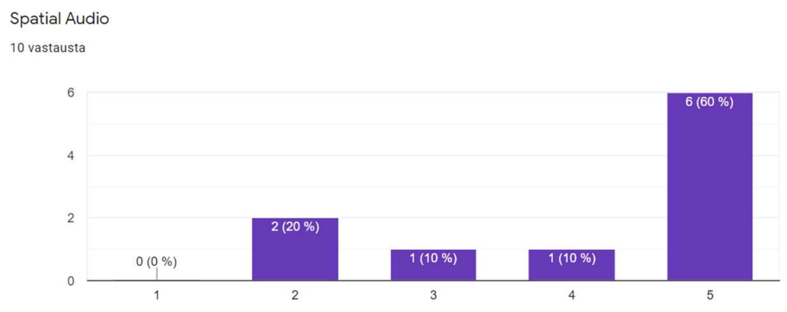


Picture 16. Results of research question 4: How clear are the voice commands? Range: from (1) unclear to (5) clear. Average score: 3.0.

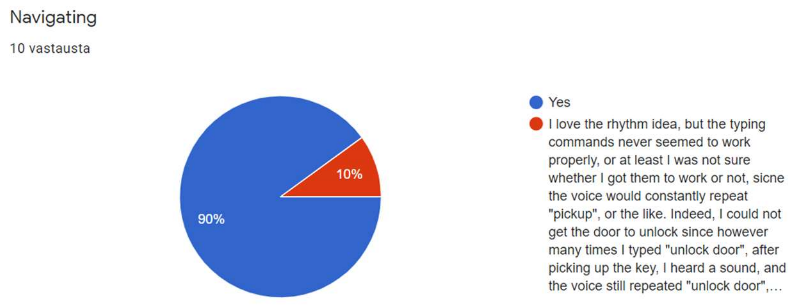
This result was anticipated, because, although much emphasis was put on preserving the voice commands in the foreground of the game's soundscape, it is still to be expected that in certain audially overloaded and gameplay-wise hectic situations the commands can be challenging to hear.

Audio System

The performance of the audio system is measured by two main functions: the spatial accuracy of audio sources and the facility of navigation by environmental audio cues.



Picture 17. Results of research question 5: Is it clear where each sound is coming from? Range: from (1) it isn't clear to (5) it is clear. Average score: 4.1.



Picture 18. Results of research question 6: Is navigating the game world intuitive?

As research data suggested, navigation of the two-dimensional map was almost unanimously intuitive for the testers. These results were encouraging, as the gameplay relies on these two factors to be predictable. Some thought it would be fitting if typing were tied to the heartbeat's rhythm. This was an intriguing proposal and could be tested out if development will continue after the game's launch.

6 CONCLUSION

The project was a success based on the test data and feedback gathered, as the goal of this thesis was to successfully solo-develop an audio game for visually impaired and sighted players based on three interconnected base game systems: rhythm movement, typing interaction and navigation using a dynamic audio system.

All the specified areas of the main game systems yield research answer results that are positively oriented. Additional, yet limited, written feedback suggests that the co-operation of these systems can create a compelling and uncommon gameplay experience. The gathered data reinforced the decision to publish the game.

The project will be continued based on the commercial success of the published game. As the implementation and integration of the base systems is complete, expanding the game with subsequent episodes would be relatively straightforward.

This thesis can be used as an example for what is required to create a unique game by combining different mechanics and systems with a very limited time window and resources. This thesis also demonstrates the audio game sector as a fitting area for experimentation with new game ideas, as it lacks the well-defined conventions of video game design. To fulfil the requirements of an audio game, a video game developer must adapt a fresh approach to game design: video game mechanics must be restructured to fit the audio game-format.

REFERENCES

Archambault, D., Ossmann, R., Gaudy, T., & Miesenberger, K. 2007. Computer games and visually impaired people.

IGDA. 2004. Accessibility in Games: Motivations and Approaches. http://www.igda.org/accessibility.

Color blindness: The most common, uncommon eye condition. 2020. Downtown Vision.

Takanen, Akseli. 2020. Analysis and Comparison of Unity and FMOD Sound Engines Degree Programme in Media and Arts Music Production. https://www.theseus.fi/bitstream/handle/10024/340930/Takanen_Akseli.pdf?sequence=3&isAllowed=y.

Jain, Aditya, Ramta Bansal, Avnish Kumar, and K. D. Singh. 2015. A Comparative Study of Visual and Auditory Reaction Times on the Basis of Gender and Physical Activity Levels of Medical First Year Students. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4456887/.

Tachibana, Ryuichi and Mamoru Komachi. 2016. Analysis of English Spelling Errors in a Word-Typing Game.

Experiment: How fast your brain reacts to stimuli. Backyardbrains.com.

Grammenos, D., Savidis, A., & Stephanidis, C. 2009. Designing universally accessible games.

Pichlmair, Martin, Kayali, Fares. 2007. Levels of sound: On the principles of interactivity in music video games. https://www.researchgate.net/publication/253005594_Levels_of_Sound_On_the_Principles_of_Interactivity_in_Music_Video_Games

Purves, D., Augustine, G. J., Fitzpatrick, D., & et al. 2001. Neuroscience (2.th ed.)

Mehta, Manish, Andrea Corradini, Santiago Ontañón, and Peter Juel Henrichsen. 2010. Textual Vs. Graphical Interaction in an Interactive Fiction Game.

Després, Olivier, Victor Candas, and André Dufour. 2005. The Extent of Visual Deficit and Auditory Spatial Compensation: Evidence from Self-Positioning from Auditory Cues. https://www.sciencedirect.com/science/article/pii/S0926641004003106.

Gardner, William G. 1999. 3D Audio and Acoustic Environment Modeling. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.8.1665&rep=rep1&type=pdf.

Dingler, Tilman, Jeffrey Lindsay, Tilman Dingler, Jeffrey Lindsay, and Bruce N. Walker. 2008. Learnabiltiy of Sound Cues for Environmental Features: Auditory Icons, Earcons, Spearcons, and Speech.

Nicholas Mark. 2009. The Audio Uncanny Valley. https://vbn.aau.dk/ws/files/61573698/audioUncannyValley_MG.pdf.

Csikszentmihalyi, Mihaly. 1990. Flow: The Psychology of Optimal Experience

Pouru, Lasse. 2019. The Parameters of Realistic Spatial Audio : An Experiment with Directivity and Immersion.

Webster, Andrew. Roots of Rhythm: A Brief History of the Music Game Genre. 2009. https://arstechnica.com/gaming/2009/03/ne-music-game-feature/.

Campana, Andrew. The Neglected History of Videogames for the Blind. 2016. https://kill-screen.com/previously/articles/real-sound-audiogames-blindness-shadow-history-gaming/.

Sinha, Ravi. A Comprehensive History of Horror Gaming. 2006. https://gamingbolt.com/a-comprehensive-history-of-horror-gaming.

Majewski, Jakub. 2003. Theorising Video Game Narrative.

Fernández-Vara, Clara. 2011. Game Spaces Speak Volumes: Indexical Storytelling.