UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Markus Hirsimäki

BETWEEN BEATS: LINKING PLAYER EXPERIENCE TO ADVERTISEMENT FREQUENCY AND INTRUSIVENESS

Master's Thesis
Degree Programme in Computer Science and Engineering
May 2021

Hirsimäki M. (2021) Between Beats: Linking Player Experience to Advertisement Frequency and Intrusiveness. University of Oulu, Degree Programme in Computer Science and Engineering, 79 p.

ABSTRACT

Advertising is common way to monetize mobile games and an increase in revenue is commonly tied to an increase in the number of advertisements. In this Master's thesis the relationship between player experience and advertising is explored in the context of mobile games.

This thesis set out to fill a research gap in existing literature by performing a survey study with constructive approach. I developed a game titled Between Beats which was used as a construct to study specific aspects of player experience in the field of game user research, which is a specified area of user experience research. The research questions explored were: *How an increase in advertising influences player experience in mobile games?* and *How can an increase in advertising be framed in such a way that it has minimal negative impact on player experience in context of mobile games?* Existing work was explored from the standpoint of related games and design choices that gave influence to the project and from the standpoint of research performed on advertising in the context of mobile games. The thesis also presents description on the development of Between Beats from the technical point of view in conjunction with information on project management methods and software tools used.

This study found that adding rewarded advertisements to a situation that already had interstitial advertisements, without removing the interstitial advertisements, resulted in reduction of perceived advertisement intrusiveness and increase in advertisements viewed. Furthermore, player engagement remained constant. The findings agree with the existing literature presented in the areas that have been explored before, such as player engagement and effects of using rewarded advertisements instead of interstitial advertisements.

Keywords: game user research, interstitial advertising, mobile game, player experience, questionnaire, rewarded advertising, survey study, user experience

Hirsimäki M. (2021) Between Beats: Pelaajakokemuksen yhdistäminen mainonnan tiheyteen ja häiritsevyyteen. Oulun yliopisto, Tietotekniikan tutkinto-ohjelma, 79 s.

TIIVISTELMÄ

Mainosten näyttäminen on tyypillinen tapa ansaita voittoa mobiilipeleillä, ja voiton kasvattaminen on usein sidottu mainosten määrän kasvattamiseen. Tässä diplomityössä tutkitaan pelaajakokemuksen ja mainonnan välistä suhdetta mobiilipelien kontekstissa.

Tämä työ pyrkii konstruktiivisen menetelmän ja käyttäjille toteutetun kyselyn avulla lisäämään tietämystä mainonnan vaikutuksesta pelaajakokemukseen. Pelaajatutkimus ja pelaajakokemuksen kartoitus ovat osa käyttäjäkokemustutkimusta, joiden aihepiiriin diplomityö asettuu. Pelaajakokemuksen tiettyjen aspektien tutkimiseen käytettiin tässä työssä kehitettyä konstruktia, mobiilipeliä nimeltä Between Beats. Työn tutkimuskysymykset ovat: ***Kuinka mainosten määrän kasvu vaikuttaa pelaajakokemukseen mobiilipeleissä?*** **ja** ***Kuinka mainosten määrän kasvu voidaan asettaa sellaiseen viitekehykseen, että sillä on mahdollisimman pieni negatiivinen vaikutus pelaajakokemukseen mobiilipelien kontekstissa?*** **Olemassa olevia pelejä tutkittiin niiden tälle työlle antamien vaikutteiden näkökulmasta. Kirjallisuuskatsaus on tehty myös pääosin mobiilipelimainostamisen näkökulmasta. Tässä työssä kuvaillaan laajasti Between Beats -pelin design-vaikutteet ja kehittäminen teknisestä näkökulmasta, sekä käytetyt projektinhallintametodit ja ohjelmistotyökalut.**

Tutkimuksessa todettiin, että palkittujen mainosten lisääminen tilanteessa, jossa on valmiiksi välisivumainoksia, johti koetun mainosten häiritsevyyden laskuun ja katsottujen mainosten määrän nousuun, kun lisäys tehtiin poistamatta välisivumainoksia. Pelaajien henkinen sitoutuminen pelin kanssa pysyi muuttumattomana mainosten määrän muutoksesta huolimatta. Löydökset yhtyvät aiempiin tutkimustuloksiin niiltä osin kun aihetta on tutkittu esimerkiksi pelaajavuorovaikutuksen sekä erilaisten mainosten ja niiden katsomisesta palkitsemisen kannalta.

Avainsanat: kysely, kyselytutkimus, käyttäjäkokemus, mobiilipeli, palkittu mainos, pelaajakokemus, pelitutkimus, välisivumainos

TABLE OF CONTENTS

FOREWORD

This is my thank you, directed towards all those who helped make the dream of a game come true. I would like to give thanks to the Godot Engine Discord community and the Closed Beta Hype Discord community for all the help they provided. Additionally, I would like to thank my supervisors Dr Paula Alavesa and Dr Leena Arhippainen, for their support.

Special thank yous go to Rémi "Akien" Verschelde for their help in producing an Android build, to Raffaele "Picster" Picca for their insight on Godot's undocumented audio server features, to an unnamed connection for their valuable industry insight, and to Veera for enabling me to spend a nearly unreasonable amount of time on this work.

Between Beats started with the intention of passing a university course by using recognition of prior learning. But as things do, the situation evolved as I lost the permission to use the game for recognition of prior learning, and the project became my personal venture into game development and the basis for a thesis. Eventually, all this culminated in Finland Games Job Fair 2021 with me securing a position in the game industry.

Finally, as a fair warning to any and all aspiring game developers, start with something smaller than this; here be dragons.

```
                                      ¨`-*,.
                                           ^B\
           A                                |e\
          /\\                               /^t\
         {\ \(                            ,:d* w},
        <  \ \\   \WwW/                _.:¨     #e
         \  \ \\  )  @___oo         ,/p’^  .A,   e\
          >  \ \\/ _) VwVwV~~   -*=#X#    < ¤ >  n@
         /  /  //  ) _/             `\b*.  ´V`   */
        <  /  //   _) _)               ¨`:,     #B
         {/ //\    _) _|                  ´:b  e}´
          \//  )  _) _/                     \>a/
           V  /  _) _|                      |t/
             /   _) _)                    ,:s/
/¨¨*-._.-*¨¨’  _) _/                  _.-*"`
\_.-.__.--{www}--{www}
```

Oulu, May 12th, 2021

Markus Hirsimäki

ABBREVIATIONS AND GLOSSARY

BPM	Beats Per Minute
PC	Portable Computer
PNR	Point of No Return
PPB	Pixels Per Beat
RNG	Random Number Generator

Banner Advertising - Advertisement format that occupies a part of the display without stopping gameplay. Commonly placed the very top or at the very bottom of the display.

Interactable - Any object within Between Beats that the player can interact with, such as a coin or a jump pad.

Interactive Advertising - Advertisement format that allows for interaction. Interactive advertisements are commonly served either as interstitial or rewarded advertisements. Commonly full-screen.

Interstitial Advertising - Interstitial advertisements are displayed commonly before transitions in the game. E.g: advertisement is shown to the player before they can continue after a death.

Light Baking - The process of including lighting information within images used to render the game. E.g. including glow effect directly in the image of a glowing object instead of dynamically creating the glow effect during gameplay.

Permission Based Advertising - Same as rewarded advertising, but the player is not rewarded for viewing the advertisement.

Pixels Per Beat - Concept used internally by Between Beats to match player movement and background music; the amount of pixels moved horizontally by the player character per a beat in the background music.

Rewarded Advertising - Rewarded advertisements give out in-game rewards, such as coins, to the player after being viewed. These advertisements are permission based.

The project - This phrase is solely reserved for referring to the particular game, Between Beats, whose implementation is described in the chapter 3.

Tweening - From the words "in betweening".Tweening refers to changing a variable's value smoothly over time from a starting point to an end point. E.g. changing the size of on an object within a game over time.

1. INTRODUCTION

This introductory chapter will first present the academic motivation behind the thesis and the related problem statement followed by the research questions explored. Subsequently, a brief description of the research method is given, and lastly authors contribution is presented, which is followed by an overview of the thesis structure.

1.1. Motivation

The academic motivation of this thesis was to perform novel game user research in the form of a survey study to fill a research gap. Existing literature presented in the related work chapter explores the effects of advertisement frequency, and the differences between interstitial and rewarded advertisements. Additionally, the impact of switching from interstitial advertisements to rewarded advertisements have been investigated in case studies by advertising mediators.

The literature presented, however, does not explore a situation in which the number of advertisements is increased by adding rewarded advertisements to a situation that already has existing interstitial and banner advertisements. This research gap paves the way to for the problem statement and research questions of the thesis, as presented below.

The problem statement for the thesis is the following: *Advertising is a common way to monetize mobile games, and an increase in revenue is commonly tied to an increase in the number of advertisements.*

1.2. Research Questions and Method

This thesis presents a survey study with a constructive approach [1]. I designed and developed a game, and used it as a construct to study specific aspects of player experience. The study falls into the field of game user research, which is a specified area of user experience research [2]. The aforementioned problem statement leads to the thesis' research questions, which are:

- How an increase in advertising influences player experience in mobile games?
- How can an increase in advertising be framed in such a way that it has a minimal negative impact on player experience in context of mobile games?

The product being deployed and explored, Between Beats, is a mobile game that was produced in conjunction with the thesis. The study collected mixed data in the form of an online survey. The data contained information on advertisement intrusiveness, player engagement, and general feedback.

1.3. Authors Contribution

In addition to the thesis written solely by the author, the entirety of the project being explored was developed and deployed by the author. In other words the construct, the game Between Beats was designed and developed by the author. This included asset creation, programming, marketing, testing, game design, and more. The singular exception to this rule are the background music pieces which were licensed from an independent music producer.

1.4. Thesis Structure

This thesis follows the standard formal thesis structure. The thesis begins with a description of the related work from the perspective related genres and games that gave inspiration for the project. Related work section then covers literature on methods used to conceptualize the game, on mobile advertising research, and on game user research.

The subsequent implementation chapter covers key aspects of the game; core gameplay mechanics, procedural level generation, procedural animation, and connection between music and movement. Additionally, the implications and justifications of using procedural generation and procedural animation are explored. The chapter then moves on to highlight essential aspects of project management used, such as the overarching design principle, the explorative and iterative design approach, and methods used to manage the complexity of the project. Finally, the chapter covers the most important software tools used.

The thesis then describes the experiments performed over the course of the project. The description of experiments starts with a look a into the very first player tests that were non-systematic. This is followed by look into later experiments performed and finally the Beta testing, which explored the research questions of this thesis. The look into Beta testing involves advertising implementation, the different test groups, and the definition of the research method and the survey used.

The chapter on results begins with information on the quantitative results and their limitations. This includes results on advertising intrusiveness, player engagement, and sources of error. The qualitative results are followed by quantitative results that cover gameplay problems, suggestions from the players, and the connections between this and other games from the player's point of view.

The discussion chapter reflects on the findings seen in the results chapter. The most important academic contributions are presented first, which are then followed by answering the research questions in detail. The chapter ends with an analysis of the known limitations and how they could have affected the results.

Finally, the thesis concludes with the a summary chapter highlighting the most important pieces of content, results, and academic contributions. The summary chapter presents the key content in the order of formal structure for thesis.

2. RELATED WORK

The project has taken inspiration from a variety of popular genres and games, not all of which are strictly related to the mobile platform. First, this section highlights the key sources of inspiration by introducing relevant genres followed by relevant games. After these, some related literature is presented. The literature focuses on the iterative method used in the development, the context of mobile game advertising, rewarded video advertising on mobile devices, and game user research.

2.1. Related Genres

Drawing clear-cut lines between different genres in video games is a task left outside of the scope of this thesis. As the genres presented here, and their relationships to each other, are subject to an ongoing debate, these categorizations are based on widely-accepted norms within video game culture. More specifically, the categorization is based on popular tags and names used to describe games on PC and Android platforms.

In this study, for PC games, the most popular tags from Steam are used [3]. Steam is major digital distribution service for video games [4]. As the list of most popular tags is under constant transformation, the version used is provided in Appendix 3. Classification of games on Android systems is less clear, so this thesis opts to provide screenshots of search results showcasing similar games within Google Play forming genres and styles.

Four major genres have given particular design influence for the project. In no specific order of importance, these are: the endless runner genre, the music games genre, the color switch genre and the battle royale [*sic*] genre. Below are brief descriptions of these genres along with relevant screenshots and explanations of how these genres influenced the project. The sections below provide background from the vantage point of genres; background on specific games is provided in section 2.2.

2.1.1. Endless Runner Genre

The endless runner genre is characterized on the mobile platform by games that feature seemingly endless tracks on which the player progresses. Figure 1 features two prominent games, Subway Surfers and Temple Run [5, 6] which have over 1.5 billion combined downloads. While this genre is less prominent on the PC platform, there exists a wide variety of mobile games mimicking the two popular aforementioned games. Icons and names for some of these similar titles can be seen at the bottom of figure 1. This sort of mimicking action is commonplace in the mobile games market, which is highlighted in figure 4. On the PC platform, some music games aim for similar gameplay; examples of these games can be seen in section 2.1.2.

The major inspiration that these titles gave was the idea of seemingly infinite progression. While the progression is limited by gameplay elements in the practical sense, a perfect player could theoretically attain a single round spanning several hours. Pre-Alpha tests of Between Beats also experimented with the "three lanes mechanic" seen commonly in this genre in which the player character avoids obstacles

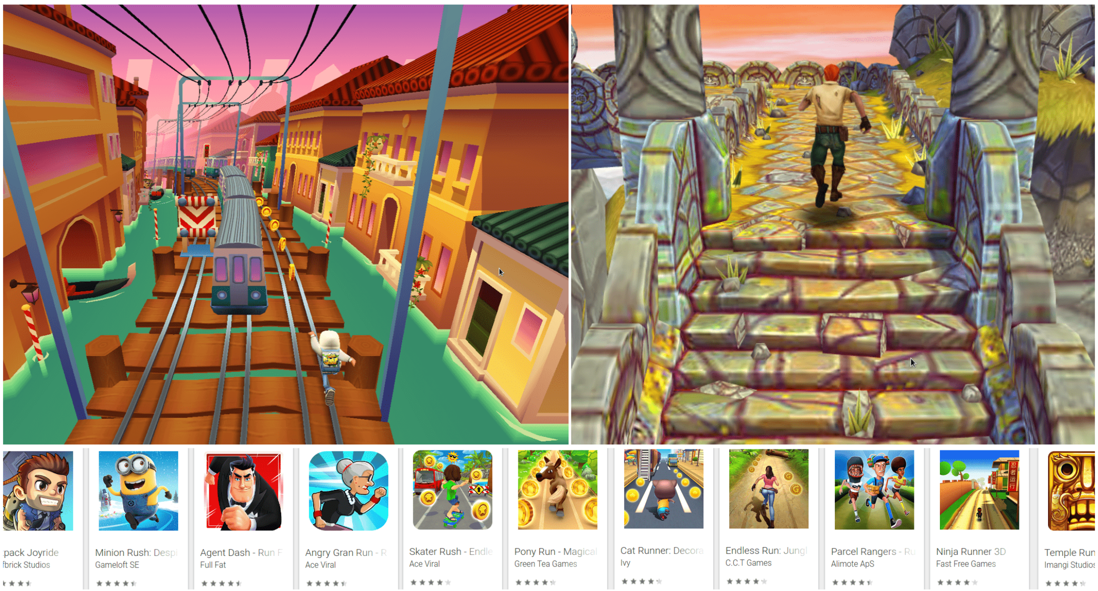


Figure 1. A collage of games related to endless runner genre. Top left: in-game screenshot of Subway Surfers [5]. Top Right: in-game screenshot of Temple Run [6]. Bottom: Search results for *endless runner* on Google Play. (Composition and screenshots © Markus Hirsimäki, all rights reserved. Content: each part owned by its respective copyright holder)

by switching between three lanes. Figure 1 shows Subway Surfers using this mechanic very clearly. Figure also features Temple Run using the same mechanic in less visible manner.

2.1.2. Music Games Genre

The genre of music games is very large and overarching as it spans several subgenres. Unlike in the case of endless runners, the games that had their influence on the project were mainly from PC platform. These games were Audiosurf2 [7], Super Hexagon [8] and osu! [9]. While these three games represent only a small sample of the genre they showcase three different approaches to matching music with player actions.

Audiosurf2 [7] has many similarities when comparing to the games presented in the endless runner section; the player moves on a track consisting of different lanes [7]. Their goal is to avoid obstacles and collect power-ups. A significant difference, when compared to the aforementioned genre, is that the game has its focus on music; Audiosurf2 generates the tracks based on the contents of a song which in essence allows for all audio files to act as playable maps. This limits map length to the length of the song used but also allows the music to closely match the obstacles. The player character is presented as a futuristic vehicle moving in a tunnel as seen in figure 2.

Super Hexagon [8] takes a different approach as the game uses fixed songs matched with sets of predesignated sets of obstacles. Figure 2 shows the player controlling a small triangle that moves in a circular manner around the center-most hexagon. The outer hexagons collapse in on the player, which requires the player to maneuver

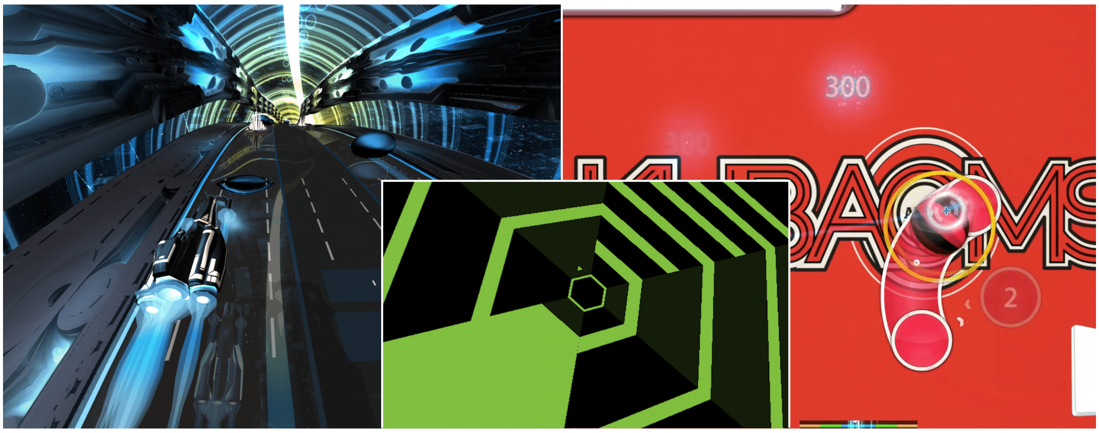


Figure 2. A collage of games related to the music games genre. Left: Steam Store preview of Audiosurf2 [7]. Right: Screen capture of official osu! tutorial [9, 10] . Center: Steam Store preview of Super Hexagon [8]. (Composition © Markus Hirsimäki, all rights reserved. Content: each part owned by its respective copyright holder)

through the paths present in the collapsing hexagons. Unlike Audiosurf2, Super Hexagon does not make use of procedural generation; while the starting point for required movement patterns is randomized, the patterns are always related to the same songs. The songs also denote different levels in the game.

Of the three games in this section osu! [9] has the least amount of similarity to the project both in the visual sense and gameplaywise. The gameplay in osu! consists of the player clicking within circles that appear as a constant stream during gameplay. The game also requires for the player to occasionally spin the mouse cursor rapidly in circle or perform drag-and-drop style motion with balls embedded into paths. Figure 2 shows a partial view of a round of osu!; on the screen, two obstacles are visible. One is a clickable circle that has the number two superimposed on it, while the other obstacle is a curved path that requires drag-and-drop motion.

The term beatmap was borrowed from osu! for this thesis to describe a different aspect of this game. Within osu! the term describes songs to which the obstacles have been manually mapped. Similarly to the other two games in this section a single map consists of one song in osu!. These three games present three different approaches to matching player input with background music within a game. Audiosurf2 aims to generatively match the *content* of the music with player movement along with the beats per minute (BPM). Super Hexagon does not match the content of songs with player actions but still matches their BPM. Lastly, osu! aims to match both the content and BPM similarly to Audiosurf2; this is achieved non-programmatically as the beatmaps of osu! are manually constructed.

The major inspiration drawn from these three games is procedural content generation and matching the players actions with the BPM of the background music. Matching the content of music programmatically would require overhauling the system used by the game, and matching large amounts of content manually, similarly to osu!, would break the design principle seen in section 3.5.1.

2.1.3. Color Switch Genre

The color switch genre is largely absent on PC platforms as it mostly consists of games trying to imitate the success of a game titled Color Switch, which released for mobile platforms [11]. The game saw immense success and has over 200 million downloads [11, 12, 13]. The success has spawned a multitude of games mimicking the visual style, name, and gameplay. A screenshot highlighting this phenomenon can be seen in figure 4, and at the bottom figure 1.

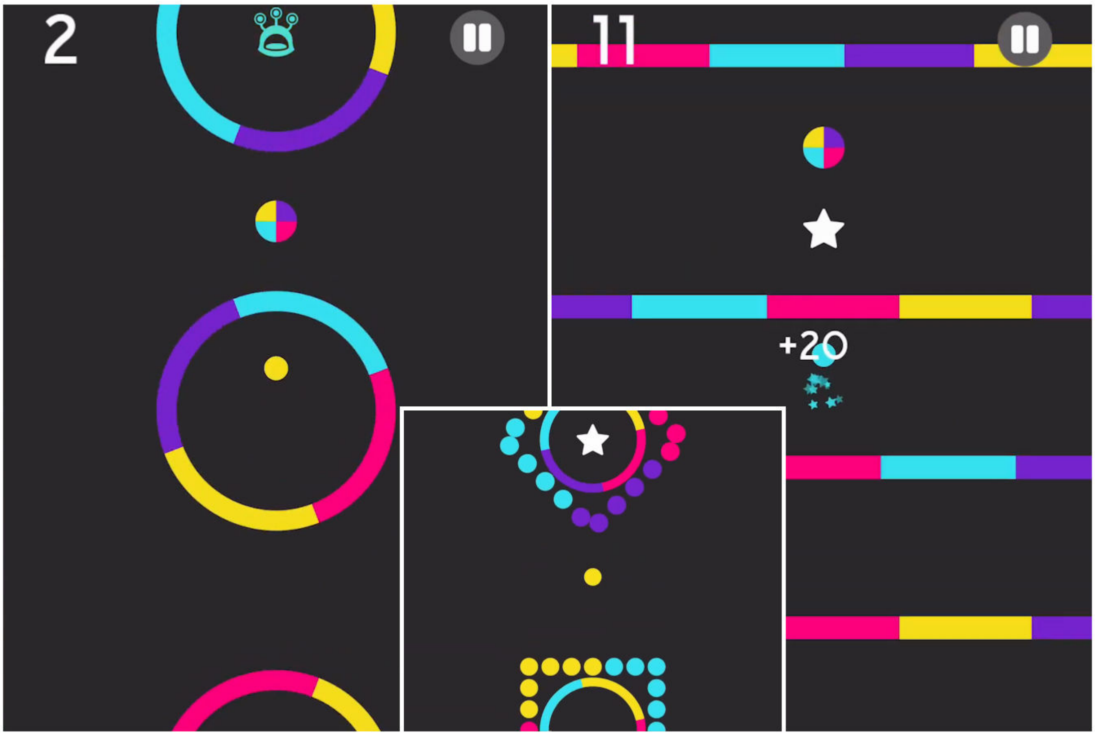


Figure 3. A collage screenshots from the game *Color Switch* captured from an official tutorial video [14]. (Composition and screenshots © Markus Hirsimäki, all rights reserved. Content: owned by *Color Switch*)

The gameplay loop of this genre focuses on the idea that the play environment has different sections which can only be moved through by having a player character with matching color. Figure 3 shows three different situations in which the player character is represented by the colored dot in the middle of each screenshot. Tapping the screen moves the dot upwards in a jumping manner, after which it will fall back down if no news inputs are given. The player character can only advance upwards through blocks that have the same color. Additionally, the left and right image show multicolored dots similar to the player which will randomize the player character's color.

The game, and other games in the genre, contain a multitude of other movement mechanics and environments, but they all focus on the idea of player character having to be of a specific color. A notable detail is that the player will generally not change the character's color, but rather control its movement; the environment changes the character color.

The major inspiration taken from this genre was the core mechanic in and of itself, switching the color. The project, however, takes a very different approach to it as the player will change the character's color with their input while being unable to control the character's movement. This is the polar opposite to the general color switch genre.

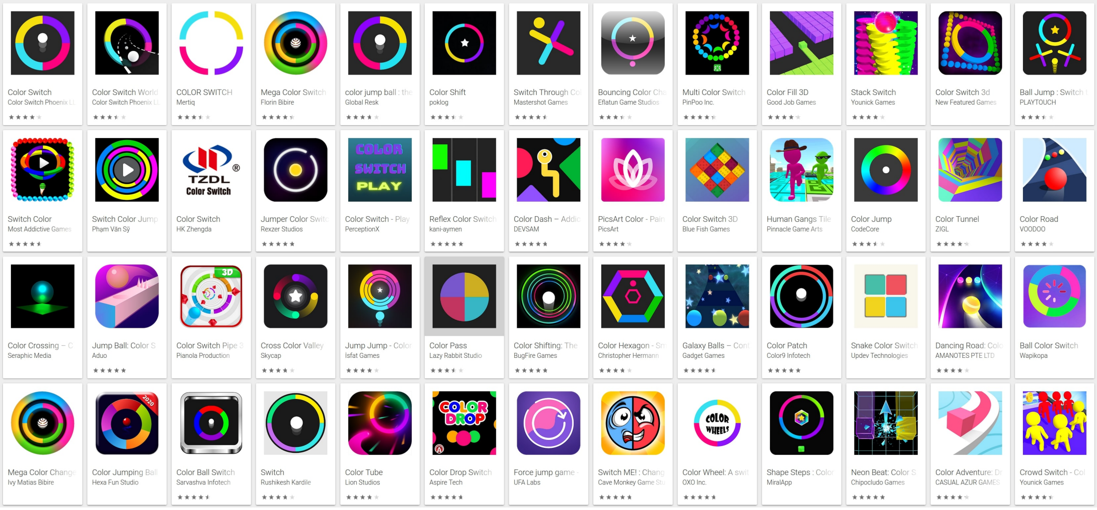


Figure 4. Search results for *color switch* on Google Play showing the genre spawned by success of the initial game. The original game is seen on top left corner. (Screenshot © Markus Hirsimäki, all rights reserved. Content: each part owned by its respective copyright holder)

2.1.4. Battle Royale Genre

The battle royale genre is named after a movie of the same name [15, 16]. The genre initially formed around modifications to multiplayer games, such as *Minecraft* and later evolved into popular standalone games such as *Player Unknown's Battlegrounds* and *Fortnite* [16].

The genre spans an extensive wide variety of different games all of which maintain the same core feature of last-man-standing style competition. In essence, a typical battle royale round consists of a number of players being eliminated one by one from the game until a single survivor remains who is then declared the winner. The eliminations typically consist of the players falling victim to different environmental hazards, such as shrinking play area, or to each other. [16, 17]

The inspiration provided by this genre is focused around its core mechanic, similarly to the one provided by the color switch genre. The project will contain a similar multiplayer element in its full release. The versions presented thorough this thesis do not contain implementation for multiplayer. The aim is to provide massive concurrency in the order of hundreds of players. This will either be achieved by traditional multiplayer solution, or an asynchronous implementation relying on tournament system, or a combination of these two.

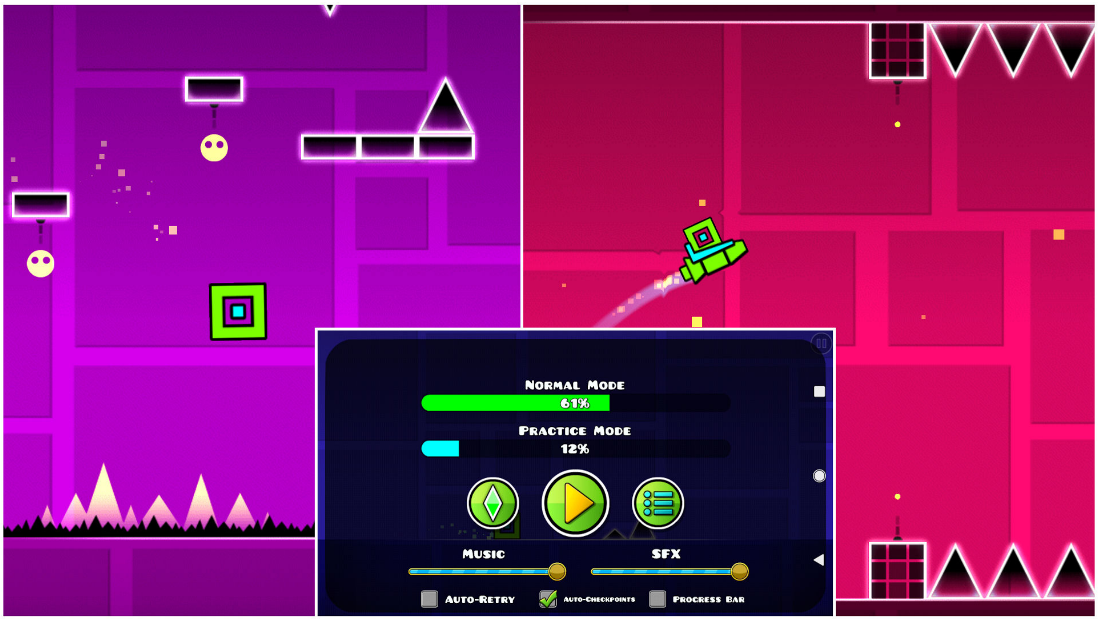
The concurrency will be based on the infinite level generation presented in section 3.2 which will be used to create increasingly difficult last-man-standing competition within the game. A traditional multiplayer approach would have all the players connected to a central server at the same time with each player playing a map generated from same initial value. The asynchronous solution would instead use a tournament-based approach where smaller elimination rounds are player consecutively to attain massive tournament size. In addition, the asynchronous approach would have a time window for each elimination round in the tournament so that each participant has a time window in which to play and submit their attempt to beat others in the round.

2.2. Related Games

The chapter will now move on to highlighting two specific games that were of particular importance and influence for the project. More specifically, they influenced the project when the different ideas were blended together using conceptual blending along with iterative and explorative design process.

2.2.1. Geometry Dash

Geometry Dash is presented here as a singular game instead amongst the genres above due to its considerable influence on the project, even though it has inspired a wave of mimics similarly to Color Switch. In-game screenshots of Geometry Dash can be seen in figure 5. Geometry Dash features a number of different movements modes two of which can be seen on left and rights sides of figure.



□ Auto-Retry ☑ Auto-Checkpoints □ Progress Bar

Figure 5. A collage of in-game screenshots from Geometry Dash. Left: Player character falling down and about hit spikes after missing jump. Right: A section of the game where the player controls a rocket instead of the character. Bottom: The in-game menu. (Composition and screenshots © Markus Hirsimäki, all rights reserved. Content: RobTop Games)

The default movement, seen on the left in figure 5, in Geometry Dash consists of the player character sliding forward automatically. When the player taps the screen, the character will jump, which can be used to dodge obstacles. The right side of the figure features a rocket which also moves rightwards automatically like all the movement types in the game do. The rocket will jump in large arcs similar to the player but can perform new jumps mid-air. The goal in Geometry Dash is similar to the one seen in the project; to avoid obstacles and allow for the character to pass through the entire map while restarting from the beginning on death.

Various graphical elements of Geometry Dash can be seen in figure 5. While the project has a distinct graphical style of its own, there exists a similarity due to the limitations of vector graphics and inspiration. In addition to graphical style, the horizontal orientation of the display and rightwards scrolling was inspired by Geometry Dash. While both the project and Geometry Dash feature seemingly similar style the other mechanics of the project provide their own influence to create a different feeling gameplay. This conceptual blending is explored in the section 2.3.

2.2.2. Crypt of The Necrodancer

Crypt of the Necrodancer [18] combines roguelike dungeon crawling with rhythm games by replacing the classical turn-based movement with the game automatically ticking forward. The tick speed is tied to the tempo of background music being currently played in the game.

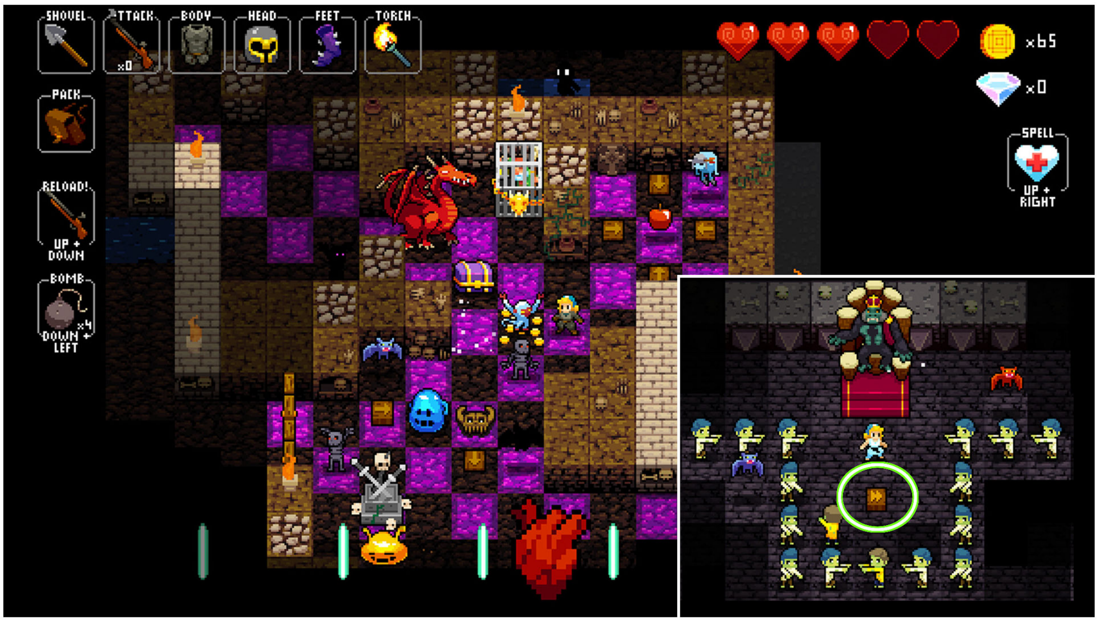


Figure 6. A collage of in-game screenshots from Crypt of The Necrodancer official Steam page [18]. Large picture: in-game view of the game showing menu elements and beat indicator at the bottom of the screen. Right: highlight showing a tile on the ground which increases the tempo of stepped on. (Composition and screenshots © Markus Hirsimäki, all rights reserved. Content: Brace Yourself Games)

Crypt of the Necrodancer, in combination with the general music game genre, influenced this thesis with how it matches player inputs with the BPM. A key difference is that the project allows for the player to give inputs in windows between beats, as opposed to most of the music games presented. Crypt of the Necrodancer also provided a major gameplay element: dynamic variations to the musical tempo during the gameplay. Lastly, a major point of influence was having the player character constantly move without inputs as opposed to a situation where the player has to take action to move the character.

Figure 6 has a circle highlighting the specific tile and the specific scene within the game which eventually grew to form the dynamic tempo variation system, which is elaborated on in the section 3.4.3.

2.3. Related Literature

This section is focused on highlighting relevant existing literature and research regarding conceptual blending and advertising in mobile games. First, the focus is on conceptual blending and its iterative use in the thesis project's development followed by a look into studies that have previously achieved results that are related to the research questions of this thesis.

2.3.1. Iterative Conceptual Blending

The key concepts from the different games were blended together in this thesis during the explorative and iterative design process, which is expanded on in the section 3.5.2. The conceptual blending that happened during the iterations began from the core idea of controlling a game solely by switching color as opposed to more traditional movement approaches, which can be seen in all of the related games presented. The other mechanics were then iteratively grafted onto this core.

Conceptual blending theory was developed by Mark Turner and Gilles Fauconnier [19]. The theory initially focused on linguistics and cognition [19, 20, 21]. In his book, Thinkertoys, Michael Michalko describes conceptual blending from a pragmatic standpoint as an automatic and subconscious process in which two ideas are merged so that the "...meaning isn't contained in either input; it's created by the combination of the two." [22]. The theory's falsifiability has later been questioned by Raymond W. Gibbs Jr. making theory more akin to a framework [23]. Conceptual blending, however, has been successfully applied to game development in practice, such as in the case of Color Switch [13]. Crypt of the Necrodancer [18] also provides an example of combining two unrelated mechanics into a novel game.

Using iteration in game design is advocated for in related literature. In his book, The Art of Game Design, Jesse Schell dedicates the entirety of chapter seven to encourage the use of iterative approaches and to highlight the dangers of the traditional waterfall model, even going as far as declaring "Rule of the Loop" as absolute truth; "The more times you test and improve your design, the better your game will be." [24]. Even outside of the context of game development, many popular software development methodologies are focused around looping and iteration. Some examples of such methodologies and frameworks are: Extreme Programming, Scrum, Feature Driven Development, Safe, Continuous Delivery, Continuous Integration, and to a limited extent the entire agile manifesto, which welcomes changes in plans [25, 26, 27].

In this thesis, the initial core mechanic of basing the gameplay around changing color was first blended with the ideas provided by Geometry Dash and Crypt of The Necrodancer. This resulted in the iteration of the game seen in figure 14. The gameplay in this state, however, lacked a steady flow as the game only progressed in "ticks" as the player gave their inputs. The influence that then followed was provided by games presented in the music games section and endless runner section. Later in the development, the influence of battle royale games resulted in the design choice to eventually include multiplayer based on infinite map generation. It is worth noting that while this free-form description of the order in which the elements were blended appears linear the act of implementation and testing is more dynamic. The end result of blending these concepts is best represented by figure 7.

2.3.2. Methods of In-Game Advertising

A variety of existing literature has already explored aspects closely related to the research question presented in this thesis [28, 29, 30]. In addition to academic research, there also exists commercial case studies focusing on similar themes. The thesis will now present some of the relevant results from both the academic sources and from the case studies that have been conducted by advertiser networks. Additionally, they will be linked with the research question of this thesis.

Context of Advertising

Burns et al. research how advertising density affected player retention in their study titled The Sensitivity of Retention to In-Game Advertisements: An Exploratory Analysis. They used a data set consisting of 21 free-to-play mobile games and propose metrics that are then used to analyze the effects of ad density in the data set [29]. Their findings go against the common wisdom that increasing advertising will reduce player retention due to the perceived annoyance [28]. Most importantly, they find a weak link between player retention and advertisement density, whereas the content of the game itself had a strong link to player retention. [29] Their study also highlights biases in the data set that could affect the results; none of the applications serve the end user with more than one ad per minute, which restricts the data set, and all of the games use their own implementation for actually fetching and displaying advertisements. Additionally, they do not model for different game types or mechanics such as scores or levels. Despite these shortcomings, the results point in a similar direction as other studies [29]; the way advertisements are framed has a correlation with how advertisements are perceived and interacted with, which ultimately has effects on profit from said advertisements.

In their study, Effects of the Types and Contents of Video Advertisements in Mobile Games, Hsuan-Yi Chou and Tsai-Chen Yang used two mobile games published on Android platform to study connections between reward-based and permission-based advertisements with successful and failed game outcomes. Additionally, they studied connections between reward-based, permission-based and interstitial advertisements in relation to dramatized, real, and mixed game scenes used in advertisements. [30] The study size was limited to two applications, both of which did not use English; additionally, the study was limited on the scope of game genres. In the context of this thesis their most important result was an indication that both rewarded and permission-based advertisements yielded better attitudes and download intents than interstitial advertisements. Additionally, there was no significant difference between reward-based advertisements and permission-based advertisements. [30]

The results of both of these studies point in similar a direction; the way the advertisements are framed has an effect on their effectiveness. This framing can occur both in the form of asking for the player's consent as well as in the sense of the game itself in which the advertisements are located. The next section focuses more closely on rewarded video advertisements as it has become a favored form for advertisements on the mobile games market [31, 32, 33].

Rewarded Video Advertising

In a study by Khai Chiong et al., Understanding the Effect of Incentivized Advertising On the Conversion Funnel, a large data set of nearly one million mobile advertising impressions was used to study rewarded video advertisement. The study tested several different hypotheses that focused around the length of the advertisement, its associated reward, and the temptation effect. [31] The data set covered a large variety of gaming applications, publishers, and advertisers; data set was randomly sampled from an advertisement mediation platform and contained 549 publishers with 398 advertisers. The findings most relevant to the research question at hand were as follows: "in three separate quasi-experimental approaches, we find that incentivized advertising leads to lower users' click-through rates, but a higher overall install rate of the advertised app" [31 p.1], and that "We conclude that the effect of switching to incentivized advertising increases the publisher's revenue by 3.10 USD per 1,000 impressions, which is a 70 % increase in revenue per impression. In addition, the install rate increases by 0.00229, which translates into a 116 percent increase". [31] They also to hypothesize that rewarding players for watching advertisements could reduce profits gained from selling other in-game items as the players have a different way of obtaining them via watching advertisements. Conversely, they also hypothesize that there could exist an opposing "halo effect" in which perceived goodwill from the publisher could increase sales. [31]

Similar results that support rewarded advertising over other formats have also been published directly by advertising networks themselves. Two such examples are two different infographics published by AdColony [32] and MoPub [33]. Both AdColony and MoPub are large advertisement mediating frameworks focusing on mobile platform.

The case study by MoPub achieved a 45 percent lower cost-per-install for a "social casino app" which allowed the advertiser to reach a positive return over investment of 5 %. The starting condition was negative return over investment, and the intervention used was switching to rewarded advertising as opposed to the initial interstitial advertising used. [32]

In the case of AdColony, the study, spanning one month and 60000 unique players, involved a mobile game that initially was already offering a "watch for coins" -type rewarded advertising. The intervention used was adding a different type of reward: continues. These allow the player character to respawn the start of the current level with all their bonuses and collectibles intact if the player chooses to watch an advertisement when the character has died. Adding this new reward type that is tightly coupled to the gameplay drove a 232 % increase in average revenue per daily active user. The change also caused a stable increase of 104 percent in session count and 4.6 increase in session length [33].

The effectiveness of rewarded advertising is further supported by a study that explored rewarded advertising as a tool for promoting unfamiliar confectionery brand towards children. The study, Advertising Placement in Digital Game Design Influences Children's Choices of Advertised Snacks: A Randomized Trial, by Rachel Smith et al. used an online game that lasted four minutes to test the reactions of 156 children that were split into four groups: control group, banner advertising group, advergame group, and rewarded video advertisement group [34].

The findings showed that the children, aged 7 to 12 years, were significantly influenced by the rewarded advertising. Additionally, the findings showed that the children were not significantly influenced by advergames or banner advertisements. Furthermore, the children were not significantly aware of the other forms of advertising besides rewarded video advertisements. [34]

2.3.3. Questionnaires and Player Experience

Psychometric questionnaires are a common way of structuring a questionnaire for mapping player experiences [35, 36]. In this thesis a survey is used to gain insight of the gameplay experiences and attitudes of test players towards the project. The survey is presented in chapter 4. This survey is partially based on existing work while the rest of the survey consists of questions and statements related to background and demography, and of questions deemed otherwise useful, such as: "How would you categorize the game?".

One of the questions related to background demography is "Recent mobile gaming experience in hours per week" which used the following categorization: "15 or less", "16-30", and "31 or more". This categorization was based on work by [37] Zaheer Hussain and Mark D. Griffiths. Their work explored attitudes, experiences, and feelings of the players of massively multiplayer online role-playing games. The survey used in evaluating the player experiences in section 4 is partially based on existing work by Amir Zaib A. et al. [35] from their research titled; The Role of Personality Factors Influencing Consumer Video Game Engagement in Young Adults: A Study on Generic Games. In their work they study the impact of personality dimensions on video game engagement. The relevant part of their work in the context of this thesis is the survey used along with other works that have validated the survey. More specifically, the survey they use contains two question sets; "Consumer Video Game Engagement Scale (Higher-Order/Third-Order Level Formative Construct" and "HEXACO Personality Inventory Model (100-Items scale)" [35]. From these two, the consumer video game engagement scale is used; the scale measures cognitive, affective, and behavioral engagement. These three categories contain a total of 29 statements such as "Anything related to this video-game grabs my attention" [35]. This thesis uses five-step scale for these statements, ranging from 1, strongly disagree, to 5, strongly agree. The authors who both developed and validated the video game engagement scale survey evidence that "consumer video game engagement has strong psychometric properties and is a valid instrument to measure engagement in video game play". Their study showed that this engagement can be measured in multiple dimensions, more specifically cognitive, affective, and behavioral dimensions. [35, 36]

3. IMPLEMENTATION

The following sections will first introduce the core mechanics of the implemented game, onto which more detailed explanations are then built. Primary focus is given to content generation, animation, music, and finally, management techniques used in the implementation.

3.1. Core Gameplay Mechanics

This section provides a general overview of the core gameplay mechanics by introducing the control scheme and interactables. While details of different game modes are subject to change in later iterations of the game, the core mechanics will remain the same. The mechanics are presented here to provide the basis for other implementation-related information.

3.1.1. Controls

The main gameplay loop is built on a very simplistic control scheme requiring only one input type, a tap, or interchangeably a click, or a button press. During the actual gameplay, the player navigates a level on which the player character moves automatically from left to right. The only influence the player has over the character is the ability to switch the character's color back and forth between blue and red by giving an input. The gameplay is synchronized with background music so that the speed at which inputs should be given is matched by the beats per minute (BPM) of the music.

The character's route within the level is determined by the character's color; when the character is blue it will interact with blue interactables such as jump pads or teleporters. When the character is red it will interact with red interactables. All of the interactables are either red, blue, or have both colors. This is excluding hazards, saw blades and spikes, which are green, and coins which are golden.

In figure 7 the player character is blue. If no inputs were to be given the character would slide into the pit separating the platforms. If the character's color were to be changed, it would slide over the pit using the red teleporter. Furthermore, the character would hit the saw blade on the right if the color were not switched again before the character reaches the blue jump pad.

Figure 7 also shows three other important gameplay elements. The coin counter of bottom left, which together with the score counter on the bottom right, are used to indicate the players performance within the current level. Additionally, a pause button is shown at the top right corner.

3.1.2. Interactables

Different interactables have a crucial role in the game as they define what the player can do and what can happen to them. While these interactables include hazards and

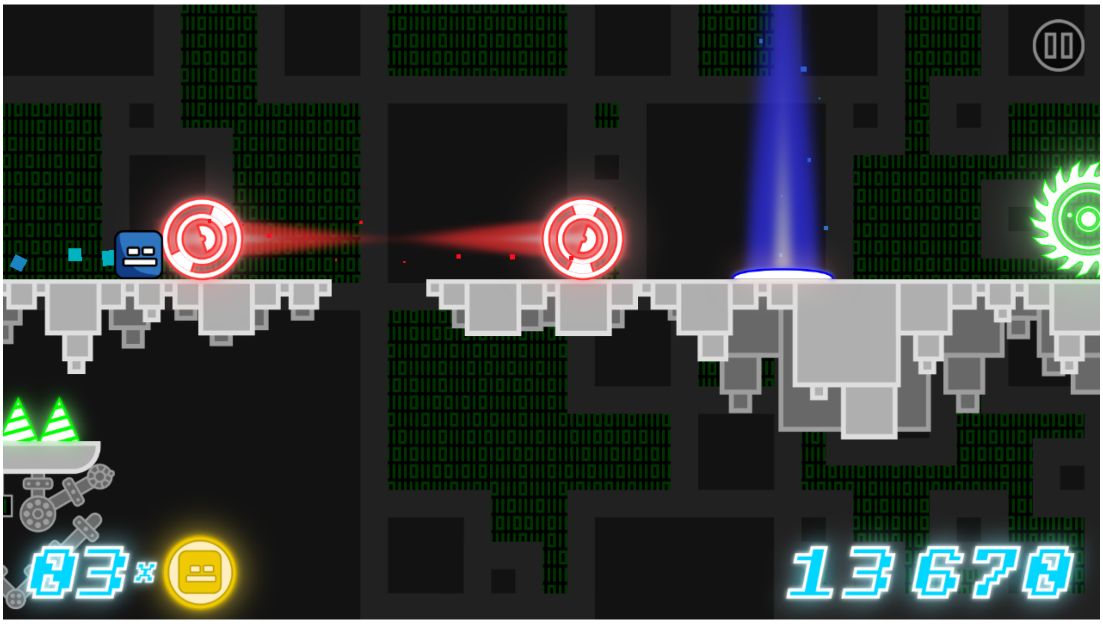


Figure 7. Screenshot featuring the gameplay in infinite mode during late Alpha phase. A video related to the screenshot is available on Youtube [38]. (Figure: CC BY 4 Markus Hirsimäki)

coins the most important ones are the ones that change the players path. Unlike in many other games, the player is not only moving on the map but is also being moved by the map itself.

This section provides a short description of all the movement affecting interactables that exist as of late Alpha phase. These are visible in figure 11. The technical details relating to changes in player's velocity and, more precisely, changes in the in beats per minute of music are elaborated on in the section 3.4.

Jump Pad

Jump pads are objects onto which the player slides when they are moving on the ground, or alternatively jumps onto. Jump pads cause the player to move in a predefined arc. This arc motion is produced by tweening the player's position via key locations stored in each jump pad.

Earlier implementations used the physics engine to perform the jumps, but this was later changed to keep the players path constant even if the player's speed and music BPM change. These predefined paths prevent the player from overshooting or undershooting the targets of the jumps even when the player's speed is varied.

Teleporter

Teleporters perform a function very similar to jump pads as they transport the player from one of their ends to an another one when the player character intersects the starting area of the teleporter.

They use similar basis for the transportation; players position is tweened during the transition between two predefined points. Furthermore, and similarly to jump pads, the player's horizontal speed during the transportation is consistent with the background music's BPM. A major difference when compared to jump pads is that the player is invulnerable during the transition. This, together with the visual differences, provides a major gameplay element by creating variance.

Anti- and Zero Gravity

The two interactables, anti gravity and zero gravity, create three distinct functions during gameplay. The anti gravity interactable is used for both changing the player's gravity to a negative value as well as back to a positive value. The zero gravity interactables start a distinct gameplay section when encountered as the player will begin floating in mid-air instead of traversing on the ceiling or floor.

Unlike the previous two interactables, the gravity related ones do not use tweening. Instead, they use simple physics based implementation by changing the player's gravity scale to either positive, zero, or negative value.

Speed Up

Speed up arrows can be used to both speed up or slow down the player. In addition to changing the players speed these affect a score multiplier as well as the tempo of the music. Similarly to the color switch interactable, speed ups cannot be avoided by switching color and are represented in the game as having both the red and blue color at the same time.

Color Switch

This interactable has only and indirect effect on the players movement as it will forcibly change the character's color. This mechanic is used to introduce additional difficulty to some sections of the game as it forces the player to either skip an input or duplicate an input depending on the situation.

Colored Platforms

These platforms perform a function similar to normal ground, allowing the player to slide on them when they are of the same color as the platform. They differ from normal platforms in the way that the player character can pass through them while being a different color. In addition to creating a new class of obstacles for the player to survive, they allow for the player to easily select a different path in some situations.

3.2. Procedural Level Generation

The game contains a system for procedurally generating playable content from a seed value using a random number generator (RNG). The system is intended to facilitate multiplayer and to reduce the amount of manual labor spent on creating playable levels. This section will provide a general overview of the map generation logic.

3.2.1. Beatmaps

Internally the game uses the notion of *beatmaps* to denote the hand crafted sections of infinite levels that can be strung together based on various rules to generate more level as the player moves through it. While the term beatmap has different meanings in different contexts, the meaning of the word is defined to be as above in the context of this thesis.

A beatmap can be seen in figure 8. The sizes of beatmaps vary wildly; while some beatmaps might be less than the width of the output resolution, 1920 pixels, some beatmaps span several screen widths. Each beatmap is self-contained as they include all the necessary elements to allow the player to pass trough by following the correct path. A noteworthy implementation detail is that the beatmaps do not contain their own metadata as the metadata and its usage is handled by the level generation system itself; the beatmaps are building blocks.

Most beatmaps start with blue interactable, such as a jump pad or teleporter, and end on a red one. These allow for seamless chaining, which creates a steady rhythm for the gameplay as the player character alternates its color between red and blue. This alternating pattern emerges even if a single beatmap is repeated, such as the one in figure 8. Gameplay and its relation to beatmaps is further elaborated on in the section 3.1.

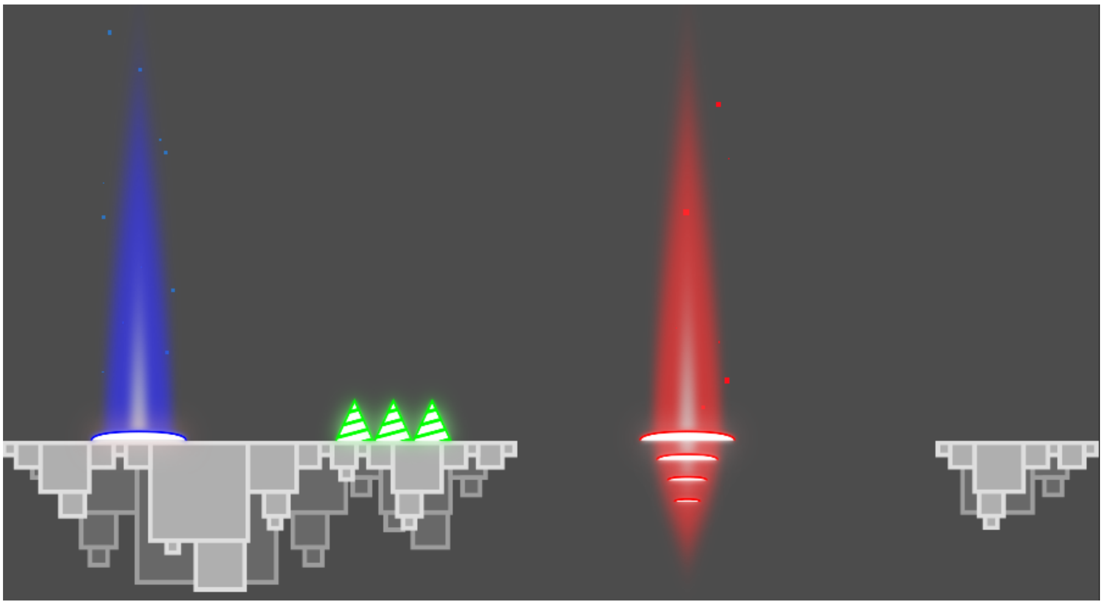
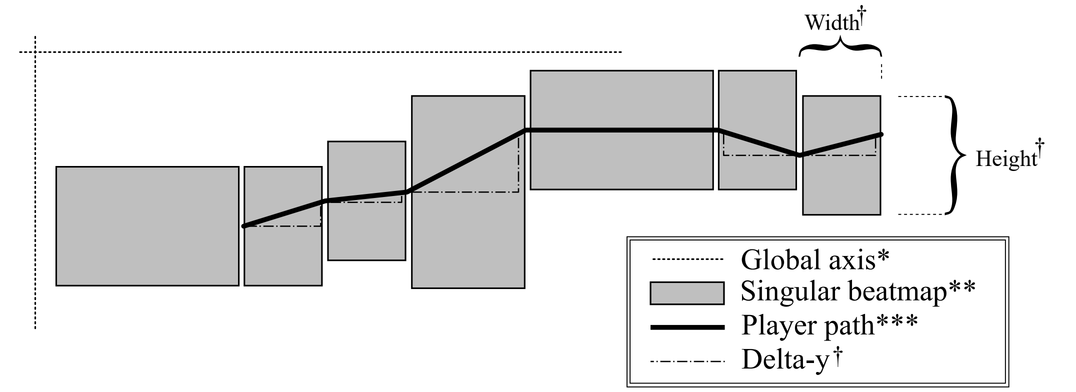


Figure 8. Screenshot featuring the contents of a singular beatmap outside of the game in late Alpha phase. The top and bottom edge of the image have been cropped, while the left and right edges denote the boundaries of the beatmap. (Figure: CC BY 4 Markus Hirsimäki)

Each one of these beatmaps has some metadata associated with it. This data is used by the level generation logic to piece the beatmaps together in such an order that the player can pass the level. This metadata describes the height and width of the beatmap along with delta-height. The metadata also includes three gravity related flags: start gravity, change in gravity, and flag denoting if the beatmap supports reversing gravity.

The width flag is used to describe the horizontal width of the playable content produced by the specific beatmap. This variable is measured in pixels similarly to height which describes the vertical height of the beatmap. The notion of delta-height is best described by figure 9, which shows a chain of multiple beatmaps forming a larger section of a level. The width and height can also been seen in the picture.

In practice, the sum of delta-heights describes the global vertical offset when the next beatmap is placed down, while the sum of widths describes the global horizontal coordinate. The height information is reserved for future use cases; each new beatmap is placed so that its horizontal axis lies at a position denoted by the cumulative sum of delta-height, which is the sum of all the previous deltas-heights. Similarly, the cumulative width is also a sum of all the previous widths from beatmaps that have been previously laid down. This cumulative width denotes the location of the vertical axis for each new beatmap.



* Global axis shown to highlight cumulative delta-y. New beatmaps set their horizontal axis to the current delta-y value. The metadata store the individual delta-y while generator keeps count of the cumulative delta-y.

** Leftmost beatmap has no player path; previous beatmap is not shown so start position can not be determined

*** Players internal path in beatmaps not shown; delta-y only concerns start value and end value. The internal path varies depending on gravity and interactables. This curve also describes the cumulative delta-y value.

† Width, height, and delta-y are stored as metadata per beatmap

Figure 9. An abstract description of the level generator implementation. Each one of the grey boxes is an unique beatmap, such as the one seen in figure 8. Each beatmap is placed in such a way that the player's path always starts in the middle of it. (Figure: CC BY 4 Markus Hirsimäki)

The three gravity related flags are used in the generator to keep track of the player's target gravity as interactables can change the players gravity to either negative value, zero, or back to a normal positive value. Beatmaps that work under normal gravity work exactly in the same manner under inverted gravity if the beatmap itself is mirrored on the horizontal axis. Not all beatmaps support this mirroring, so the data is stored in the gravity-flip metadata flag. During Beta testing phase, the only beatmaps that do not support this mirroring are the ones that are designed to work in zero gravity as the player will float in mid-air instead of touching the ceiling, or interchangeably, the floor. The start gravity flag describes the gravity type (normal, negative, or zero) that the player is supposed to have when entering the beatmap. The gravity change flag describes if the player's gravity changes to some other value before entering the next beatmap.

3.2.2. Generator Algorithm

This section provides a high-level description of the implementation both to protect the source code and to abstract away unnecessary details. The generator initializes itself by placing down the first hardcoded beatmap which is a long flat strip that requires no input from the player to allow for the player to orient themselves.

After the first beatmap has been instanced, the algorithm updates its state based on the metadata flags of the latest beatmap that had been laid down. The following information is updated based on metadata: the accumulating horizontal coordinate for placing next beatmap, the accumulating delta-height, and lastly, the generator's gravity state.

After these initial steps, the generator builds a subset from the total set of beatmaps and randomly chooses the next one from the subset. The subset has to fill requirements set by the generator based on the previous beatmap. In the generator, which is present in the Beta version of the game, the only requirement is a compatible gravity flag; a normal gravity beatmap can only be followed by a normal gravity beatmap, or a beatmap with negative gravity and a flag marking it as one that can be mirrored on the horizontal axis. Additionally, zero gravity beatmaps cannot follow normal gravity beatmaps and vice versa. It is worth noting that the gravity for beatmap is described by three flags and not just one. This allows the beatmaps to start with one gravity and end with a different one which in turn allows chaining otherwise non-compatible beatmaps by having some beatmaps act as a gate to move from one gravity state to another one.

Later in the development, the map generator will track its state in more detailed manner to keep track of variables like difficulty and limits for maximum, minimum cumulative delta-y. Unloading previous beatmaps and support for threaded operation when instancing new beatmaps is to be added to the generator in later iterations of the game.

Each one of the beatmaps registers itself to the generator and sends a signal when they detect player leaving their area. This signal triggers the process to instance the next beatmap. Additionally to the state information described above, the generator keeps track of a buffer variable. This buffer variable describes how much of the world ahead the player can see on their screen while adding a generous safety margin. This prevents the player from seeing the world being constructed as they move forward. The buffer variable is incremented by the width of each new beatmap as they are added into the game world and decremented every time the player leaves the area of the latest beatmap.

3.2.3. Justification for Procedurally Generating Playable Content

The need to procedurally generate content became apparent very early during development. This need arose due to practical limitations relating to time that could be spent on developing content manually. Similarly to many other design choices, the driving principle was *maximum impact with minimum effort*.

The second reason that influenced the choice to procedurally generate content was the increased replay value and player engagement arising from it. Procedural generation is a satisfactory approach to reaching these goals. As an example of games using procedural generation, Minecraft and Rogue are two games that heavily rely on it to create playable content [39, 40]. As of February 2021, Minecraft is the best selling video game of all time, with at least 200 million copies sold, while Rogue influenced the gaming industry to the degree of creating an entire genre titled Roguelike [40, 41].

3.3. Procedural Animation

The vast majority of all animations seen in the game are created procedurally. During Beta testing phase there, were two non-procedural animations which is contrasted by dozens of procedural animations. These procedural animations include menu and particle animations.

This thesis considers the differentiation between procedural and classical frame by frame animation to be such that any animation that is not explicitly comprised of a series of consecutive images is a procedural one. This section clarifies how these procedural animations are used within the game.

3.3.1. Tweening

Godot Engine offers a multitude of ways for implementing animations in a procedural manner. The chosen method that is used for nearly all such animations in the game is the tweening functionality of Tween nodes. Tweening is otherwise known as inbetweening from the words in and between. [42, 43]

The game's source code makes extensive use of the function `Tween.interpolate_property` (later referred to in the simpler manner of interpolate_property), which drives most of the procedural animations in the game. It is noteworthy that `Tween.interpolate_method` (later interpolate_method) is used in some edge cases to bypass the limitations of the interpolate_property function as it is unable to change variables that can only be changed using setter or getter functions, such as in the case of shader parameters.

Due to the function being so widely used in the source code, below is the full function prototype for interpolate_property, along with example arguments and explanations. This prototype is provided to help highlight how general purpose the function is and to explain its use cases in later sections.

```
bool Tween.interpolate_property(Object object,
                                NodePath property,
                                Variant initial_val,
                                Variant final_val,
                                float duration,
                                TransitionType trans_type=0,
                                EaseType ease_type=2,
                                float delay=0)

     object: Target object, of type Object.
        e.g: my_picture_sprite. Any object with variable
             properties can be targeted.

     property: Target property for object, of type NodePath.
          e.g: "position:x". Any property can be used.
               "position" and "position:y" are equally valid.

     initial_val: Initial value of property, of type Variant.
             e.g: 100.75. If "position" was used instead the
                  format would be Vector2(100.75, 0.0) instead

     final_val: Final value of property, of type Variant.
           e.g: 200.75. If "position" was used instead the
                format would be Vector2(200.75, 0.0) instead

     duration: Duration in seconds, of type float.
          e.g: 5.0. Describes how long it takes for variable
               to change from initial value to final value
               while representing all the values in between.
               Resolution depends on processing speed.

     trans_type: Curve type, of type Tween.TransitionType.
            e.g: Tween.TRANS_EXPO. Defines what type of
                 mathematical curve the variable will follow
                 as it gets values over time. Optional
                 argument. See picture below for details.

     ease_type: Easing type, of type Tween.EaseType.
           e.g: Tween.EASE_IN. Defines what type of easing the
                mathematical curve will follow as it describes
                the variable’s values over time. Optional
                argument. See picture below for details.

     delay: Delay in seconds, of type float.
       e.g: 5.0. Delay how long to wait before starting the
            changing of the variable value. Optional argument.
```

The transition types and ease types are best explained with help of figure 10. The curves can be thought to describe the value of the variable being tweened with the vertical axis representing the value of the variable on a scale from the initial value to the final value. The horizontal axis represents the time from zero to the end of the tween, as described by the duration argument.

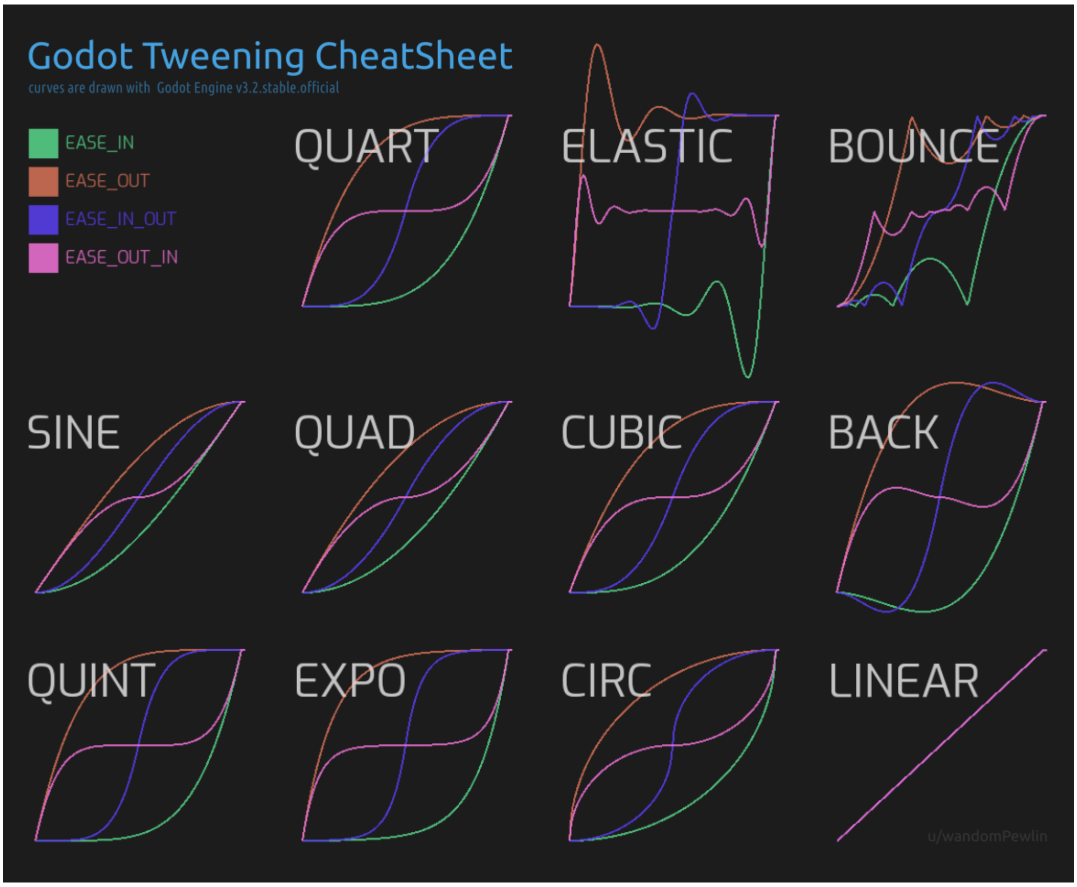


Figure 10. List of the transition types available for tweening with Tween nodes (quart, elastic, bounce, etc.). The four easing types can by applied to any transition. Image used under the Godot Engine license. Licence text is provided in Appendix 1. [42, 43]

These built in transition and easing types enable easy emulation of common behaviours described by curves such as a ball bouncing or object speeding up in a free fall. This allows for rapid development; a concrete example on this would be combining tweening of scale and position to achieve the effect of a menu panel jumping into players view in three dimensions. While these built ins allow for rapid development, they have their own limitations. A strong limiting factor is that the Godot Engine does not provide any easy way for implementing new curves using the existing Tween nodes.

3.3.2. Use Cases

In practice, most of the animations are achieved by tweening some combination of the following variables for images: scale, position, transparency, color tint and rotation. In addition to these some more advanced techniques are used, such as changing shader parameters and using an implementation detail of tween nodes to toggle boolean values or callback functions with delay. These delayed calls provide a framework for creating sequential procedural animation. The final key component of the procedural animation used is the Godot Engine's tree-like structure for organizing items in the game which allows for constructing complex movement of items with nested simple rotations and offsets

In essence, all of these methods boil down to tweening floating point variables with the interpolate_property function. While the function only has the ability to change seemingly simple variables, as seen in the function prototype, it allows for complex behaviour to arise from simple rules when different tweens are combined. These emergent effects can ultimately even produce rudimentary animations in three-dimensional space, such as the player character appearing to flip like a coin due to a combination of skewing and rotation.

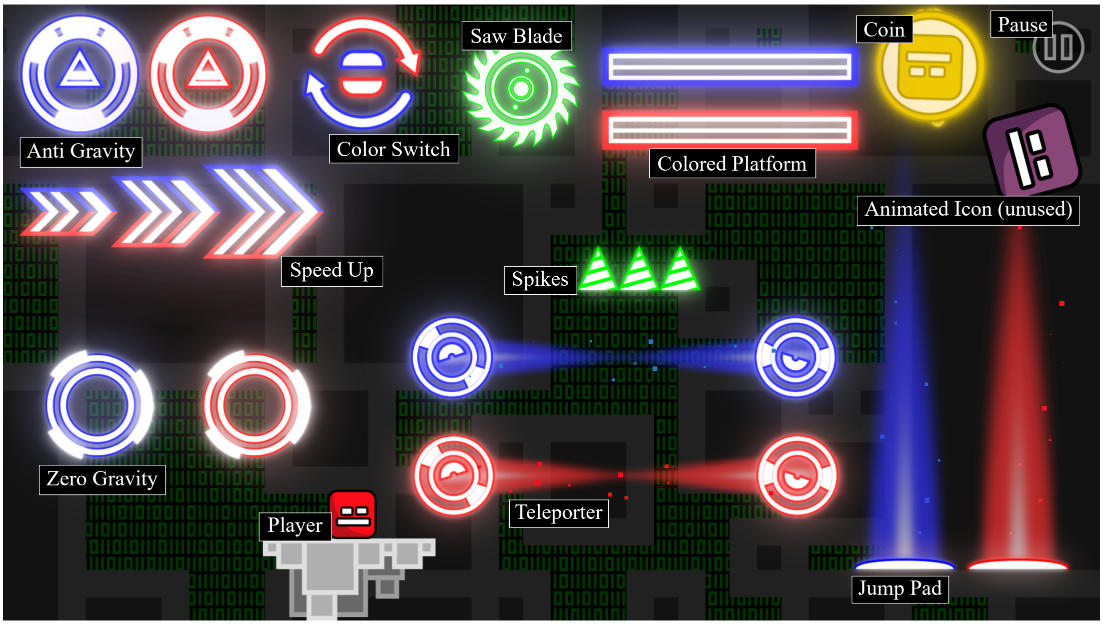


Figure 11. Screenshot of an animation test view in the game in late Alpha phase with superimposed captions. A video showcasing the animations is available on YouTube [44]. (Figure: CC BY 4 Markus Hirsimäki)

Figure 11 and its related video show a test view from within the game where various objects using procedural animation can be seen. Some of the more complex behaviours, such as the white stripes moving within the wide arrow on the left, are achieved by separating the object to smaller components. The red and blue border is static, as well as the highlights at the tips and tail ends of the arrow. The only moving parts are the middlemost white stripes within each section of the arrow, which all move together as they are part of the same image. Ultimately this animation only requires stacking multiple images on top of each other and moving some of them from left to right, after which they jump back to their initial position in a loop.

Some animations consist of multiple tweens. This can be seen in the circular arrow at the top of the image and the accompanying video; the final effect is achieved by having multiple animations playing at the same time in a loop. The first one is used to rotate the outer circle, the second one is used for the pulsating effect of the inner part and the third is used for the overall pulsing. The second one of these three animations consists of multiple sub-parts where tweens are timed to trigger consecutively by using the delay argument to achieve finer control.

Figure and its related video also feature the only two frame based animations that are present in the late Alpha version of the game. The player death animation (only seen in the video) uses frame based animation for the blinking effect, similarly to the set of the three spikes in the middle of the image, which use frame based animation for the moving white stripes. It is noteworthy that the design decision for using baked lighting effects and procedural animation arose early in the design; these two frame based animations were one of the first animations to be added into the game. This decision and its implications are further elaborated on in the next two sections.

3.3.3. Justification for Choosing Procedural Animation Style

The primary reason for preferring procedural animation over frame based animation relates to the driving design principle of the game: *maximum impact with minimum effort*. This is further discussed in the section 3.5.1. The labor intensiveness of creating frame based animations partially arises from the technique itself and partially from the design choice to bake in light effects. This is best explained by a concrete example involving the above mentioned animation of the spikes.

The spike animation uses only 5 frames in total. While the amount is small, arriving at the end result was extremely laborious. This included making multiple drafts that were up to 9 frames and baking the lights for each component in each frame. Unlike in procedural animation, a minuscule change requires manually applying the change to each frame and object and light map manually. Producing the final animation of 5 frames required: calculating reasonable frame count and speed for animation by hand, dividing the animation into frames, masking each frame manually to achieve the movement of stripes, duplicating each frame two times to create baked lighting effect, blurring each object in those extra layers manually a set amount, combining the objects into frames, carefully making each frame the same size in pixel perfect manner by having invisible frames, exporting the frames, and finally building an animated image in Godot Engine from those frames [43].

This is an extremely stark contrast to procedural animation where making changes to the feel of the animation might be as simple as changing one to two variables in the source code describing the animation. This style of animation, however, has its own problems; procedural animation can not easily create all effects that a designer could wish for as it is very limited in its capabilities. This shifts the work from the repetitive task of drawing frames to figuring out a way for having an animation emerge from atomically small components such as tweening a single floating point variable. Similarly to how frame based animations require a multitude of frames, this style of animation requires multitude of components that can be modified individually.

The practical reason for choosing procedural animation over frame based animation is quicker workflow; in the scope of the game, there exists an order of magnitude difference in time spent per animation. It should be noted that some animations are better suited to be produced in frame based manner. An example of such animation would be the player's death animation which produces a suitable stroboscopic effect with two frames. These types of animations, however, are rare in this game.

3.3.4. Implications of Animation Technique

Using procedural animation and the accompanying baked lighting effects has three major implications. These are as follows: significantly reduced final file size due to lack of storing individual frames, increased computation requirements for playing animations, and reduced computation requirements for post-processing.

The final file size reduction and the related need for increased computation power is a classical example of time–memory tradeoff [45]. The reduced amount of frames stored greatly reduces the final file size, which is especially important on mobile devices. This is, however, opposed by an increased need for processing power as the game uses a significant amount of tween nodes to play the procedural animations.

Additionally, the procedural approach allows for embedding the glow effects seen in most objects within the game into the bitmaps themselves. While this increases file-size somewhat, it is a minor problem compared to the cost of using the opposite choice of post-processing lighting. This is due to most low-end or old smartphones producing sub-par player experience when heavy post-processing is used. These sub-par experiences are largely caused by a 10 % market share of Android devices still using previous-generation technology, OpenGL ES 2.0, for post-processing. [46]

3.4. Matching Music with Player Movement

This section is dedicated to highlighting the relationship between the background music and player movement. Additionally, an example code snippet is provided to help facilitate the implementation of a similar solution.

3.4.1. Relationship Between Music and Player Movement

Background music plays a crucial role in the game. When the player's inputs match the BPM of the background music, the character will switch its color in the correct rhythm to pass all interactables. When these two become mismatched, it will generally result in the character's death due to it failing to pass some interactable.

As the player has to switch the character's color between interactables, there is a fairly large window in which the player can give their input. This allows for the game to have easier initial difficulty than many other rhythm games as they aim for the opposite where the player has to give their inputs on a beat rather than between beats [47]. Incidentally, this description of the gameplay was chosen as the game's name.

The above description of matching movement and music relies on both the BPM and the player movement speed staying constant. This is, however, not the case in the game. Variations in the player speed and music tempo provide a major gameplay element by increasing and decreasing the difficulty during the gameplay in a harmonic manner. This results in a situation where the background music's BPM must be changed in order to avoid mismatching it with player movement and the speed at which player encounters interactables.

3.4.2. Changing Tempo of Music at Runtime

The underlying system used to match players speed with music speed relies on two variables. These are the BPM of the song and an internal concept titled Pixels Per Beat (PPB). All parts of the game relating to map generation assume for two variables to never change; the base speed of the player, and the default BPM to which all beatmaps are designed to match into.

The players default speed remains constant during gameplay as changes in speed are applied via a secondary multiplier. The assumed default BPM and player speed give rise to the notion of PPB, which essentially describes how many in-game horizontal pixels there are between each interactable. When player's speed is increased, or decreased, via the aforementioned multiplier, the same multiplier is used to calculate a new BPM for the music. This allows the PPB, which was increased with the players speed, and the BPM to stay in sync.

Godot Engine allows easily manipulating playback speed along with pitch for an audio file at runtime [48]. This feature comes with a caveat as both the playback speed and pitch will change together instead of being separately accessible. This problem is circumvented in the game by using an audio effect to reduce the pitch without affecting playback speed after the pitch has been increased in conjunction with the tempo. Due to the missing documentation surrounding `AudioEffectPitchShift`, an example of the core solution used is provided in Appendix 2.

3.4.3. Dynamic Gameplay Tempo

To minimize the possible mismatch between BPM and player movement, when the tempo is updated, the change is applied instantaneously. The changes to tempo follow the harmonic scale seen in music theory by multiplying or dividing the current tempo by the twelfth root of two [49].

The design choice to use twelfth root of two as the factor is based on the players perceiving pitch shifting by this multiplier as a shift of one musical tone in the commonly used twelve note scale. The same multiplier is then used for player updating the player's movement speed.

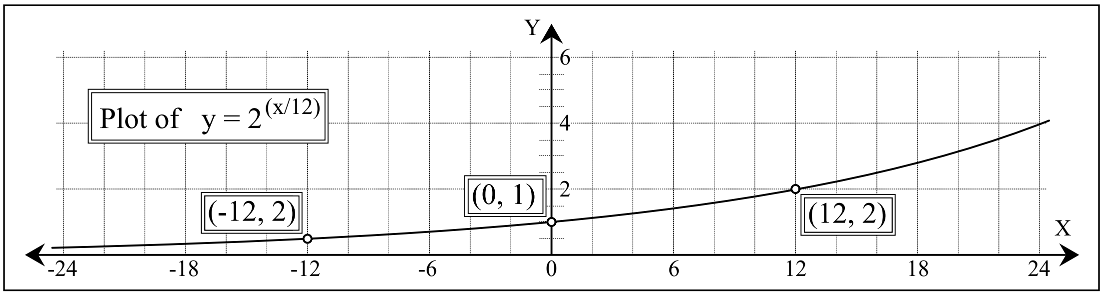


Figure 12. A plot of the curve used for calculating tempo with three key values shown (Figure: CC BY 4 Markus Hirsimäki)

In figure 12 the curve used can be seen. In practice, when the player's speed increases by one tick due to them interacting with speed up the, tempo shifts one whole integer on the X-axis seen in figure. Additionally, figure highlights the minimum, maximum, and starting value used for the multiplier. The same multiplier is used for both player speed and music speed. Using a power of X as the function results in a nonlinear curve, which requires twelve interactions to either double or half the total speed in logarithmic increments of multiplying by the twelfth root of two. Doubling or halving twice would require 24 interactions.

3.5. Project Management

This section will present methods and techniques that were used during the project to help manage its complexities. Without regard to the style of writing, the reader should consider this information as descriptive as opposed to prescriptive. The section is divided into three areas of focus; the driving design principle, description of the iterative approach used in the project, and managing complexity.

3.5.1. The Design Principle

Due to the project's relatively large scope as a task for one person, efficient time usage is a key element in upkeeping the overall feasibility. The overarching design principle, *maximum impact with minimum effort*, was not initially a conscious decision or learned knowledge. This design principle was emergent realization due to the constant time constraints relating to the project. Over time it has transformed into a conscious decision making process.

In practice, this principle could be understood as follows; *The project will eventually run out of time with implementation due to resource limitations. Thus all tasks should be prioritized based on their effort and impact. Implementation should focus on tasks that provide the highest impact with the lowest effort.* This inevitably leads to technical debt in the form of temporary solutions and hacks within the project, which in turn lead to a continuation on the previous: *Technical debt of the project should be considered to be a resource; as long as the interest for the debt can be paid the debt should be embraced.* The interest of technical debt could be considered to be the cumulative effects of the temporary solutions.

Following this principle, the project will, ultimately, end up in a finished state where high priority tasks have been finalized. There will also exist a number of unimplemented tasks that did not have high enough priority initially, nor did they gain an increase in priority over time. There will also exist an amount of temporary solutions that have not yet had a large enough cumulative effect to make them warrant the effort required to implement a more robust solution. The takeaway related to this, is that it should be intentionally, consciously, and constantly decided what to leave to be done later or ultimately not at all.

3.5.2. Explorative and Iterative Approach

The above principle relies heavily on an iterative approach and constantly making an effort to upkeep the feasibility of the project. Figure 13 shows the iterative approach used. Each one of the branching sections within the graph highlights a timeframe between two points of no return (PNR).

In practice, PNRs are major design choices that have heavy implications. Earlier PNRs offer a wider array of options as each one limits future decisions. The first major choice within the scope of the project was choosing a platform. While even this PNR is not final in the literal sense, it is final from the standpoint of feasibility as this choice has vast implications regarding fundamentals such as control scheme and monetization.

Examples of other such design choices within the project are graphical style, input system, and restricting the game to only using two different colors for the player character.

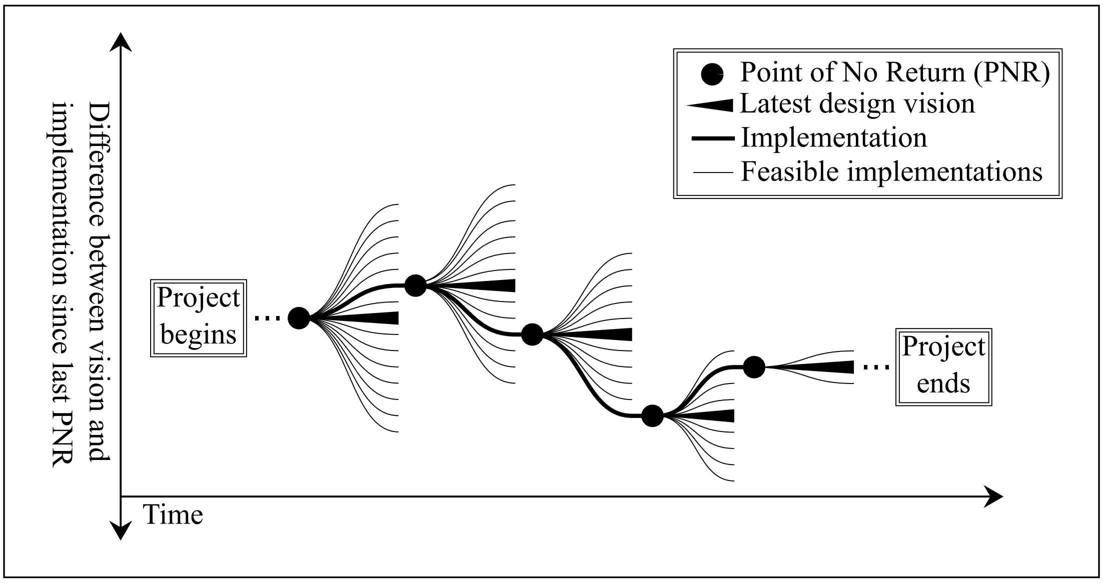


Figure 13. Description of the iterative approach used in the management of the project. Each brancing section marks a major design decision. Figure describes no specific project in particular; the total width of figure could be dozens of branching sections. (Figure: CC BY 4 Markus Hirsimäki)

The deviation between design vision and implementation in figure 13 is the result of the explorative nature of the development. In essence, this is part of the aforementioned design principle. While the design vision provides a general direction into which the game should next develop in the actual implementation will follow the path with the least effort and highest impact. It is worth noting that this exploration in and of itself can take an amount of effort varying from none to significant.

As an example of this, very early versions of the game contained a control scheme that resulted in the game "ticking" forward as the player gave inputs. This was forgone in the favor of a system using the physics engine provided by Godot [43]. Using physics engine allowed for *maximum impact with minimum effort* by leaning into the popularity of other similar music games via leveraging a familiar-looking visual flow and lower time cost in development as the physics engine provided ready made features [43, 47]. The approach provided higher impact per effort, which led to the deviation from the initial design vision.

The early test in question can be seen in figure 14 below. Other iterations of this test included control schemes using up to six on-screen buttons, which allowed the player to move in multiple directions on a grid-like world.

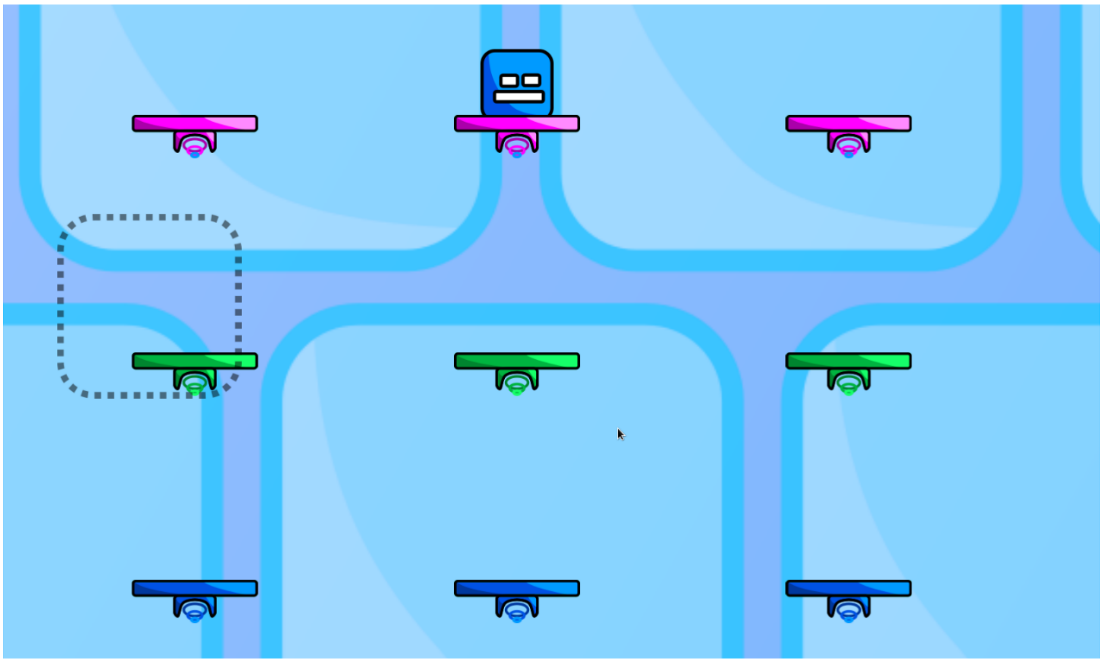


Figure 14. Cropped screenshot of one of the early movement and input tests. This version allowed the player to switch between three colors and move rightward only when the touchscreen button was pressed. (Figure: CC BY 4 Markus Hirsimäki)

The iterative system used arose as a practical solution to the challenges of game design. These challenges commonly presented themselves by making certain design options hard to choose from due to lack of information; iterating on other parts of the project, however, allowed for making educated decisions at later points in time. An example of such an decision would be delaying choosing the graphical style for colored platforms until all other interactables were designed, even though the colored platform was the first one to be conceptualized.

3.5.3. Managing Complexity

Most of the project's complexity management happened using an ad-hoc system similar to a Kanban board [50]. As new tasks arose during development they were written on sticky notes. The notes generally consisted of one to five bullet points sometimes followed by a small pictogram. These notes were placed on a bedroom wall into clusters, and eventually these clusters came to cover an area of approximately one square meter. Photograph of these clusters can be seen in figure 15.

In the order of importance, these clusters consisted of the following themes for different tasks: immediate future (programming), immediate future (marketing), far future, miscellaneous, and completed. In the some order as the previous list, examples of these tasks could be as follows: refactor audio manager, create an email newsletter, implement advertisement integration, add a new interactable (with a pictogram of a roller coaster loop), and refactor nested signals. [50]

Using this Kanban board like approach allowed the project to loosely follow Just In Time manufacturing ideology which is sometimes associated with Kanban. Similarly to the design principle, this approach emerged during development, rather than being a given from the beginning of the project and later developed into conscious effort. A noteworthy similarity to Kanban that spontaneously arose during development was a "pull system" which is also seen in Kanban [50]. During the project, only after a task was finished a new one would be pulled into a work-in-progress state along with its related tasks.

The thesis acknowledges that more sophisticated and purpose built software solutions exist for managing project complexity. This pen-and-paper approach was preferred throughout the project as it allowed for a highly visual and flexible solution. The most important function that the notes served was keeping track of unfinished work; organizing said work was of secondary importance. As the project had only one author working on it, this simplistic solution was sufficient; for a different person or team, a different solution could be better suited.

3.6. Software Tools Used

In this section, the thesis will briefly describe the most important software used during the development. Additionally, provided are screenshots of the programs in use and explanations on their use cases. All of the software used is free or open source, this, however, was not a design choice except for Godot Engine, which was preferred over commercial solutions due to being open source. The main reason behind using free software was the already existing experience with it. Finally, at the end of the section, the thesis will provide a list of otherwise important software used that is not strictly related to content creation.

3.6.1. Godot Engine

Godot Engine is a open source game engine that was used in the development. It has many similarities with the Unity game engine, which is widely used in the mobile games market. It is noteworthy that due to compatibility reasons, many game publishers prefer working with Unity based games specifically. The engine contains a fully integrated development environment and uses its own scripting language, GDScript [43].

The architecture of the Godot Engine is based on the concept of a tree that consists of nodes. The nodes are contained within scenes; scenes can be reused, instanced, inherited, and nested by other scenes and nodes. The engine loads all game resources, such as scripts and graphical assets, from the device’s file system as opposed to a

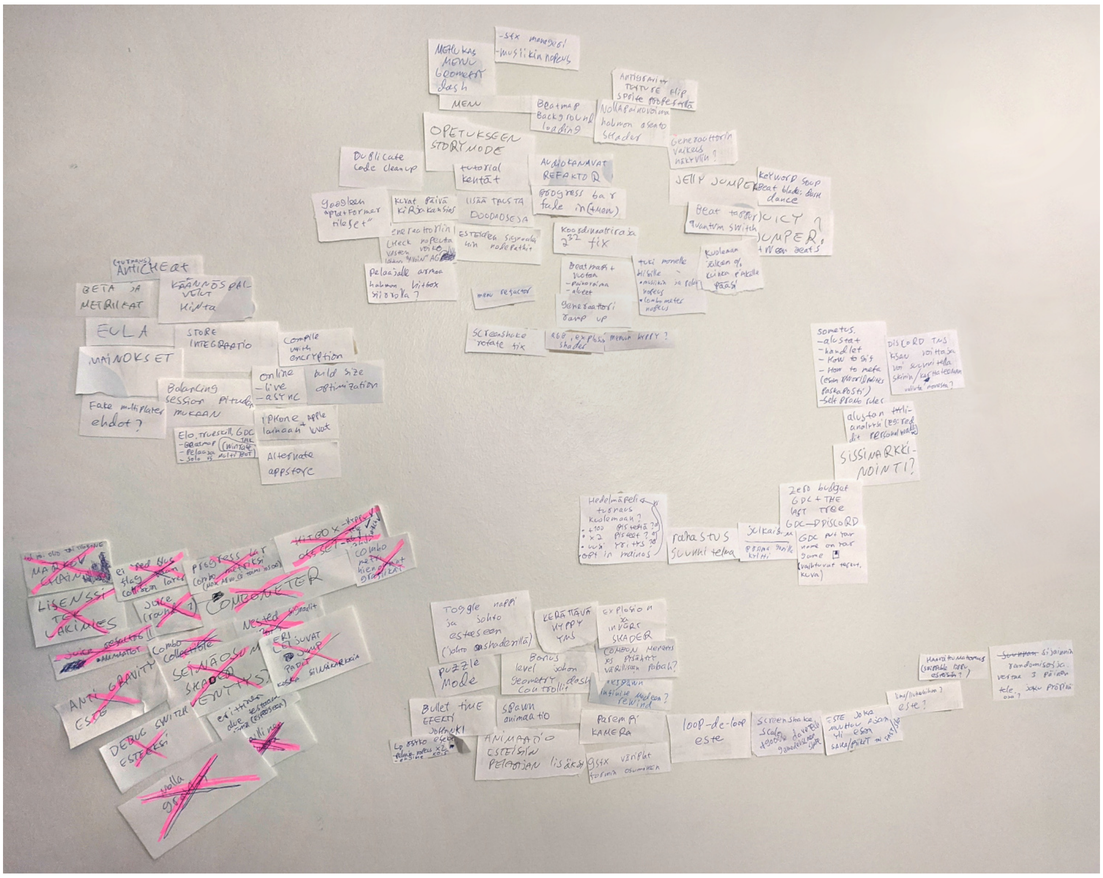


Figure 15. Photograph of the sticky notes on a wall during late Alpha phase (20 February, 2021). The themes of the five clusters, listed clockwise and starting from twelve o'clock, are immediate future (programming), immediate future (marketing), miscellaneous, completed, and far future. (Figure: CC BY 4 Markus Hirsimäki)

database file. Figure 16 shows a scene tree at the top of the left panel and assets at the bottom of the left panel. On the right hand side different properties of the currently selected node are seen. [43]

3.6.2. Inkscape

Inkscape is an open-source editor for vector graphics that was used to create and export graphical assets for the game. Inkscape can natively work with vector primitives such as circles and arcs as well as text. These objects can be colored, and they can have an outline, both of which can be patterned or filled with a gradient. The ability to embed other common image formats was sometimes used for tracing and modifying existing works.

Additionally, Inkscape was used for the creation of various figures in this thesis. Figure 17 has most of the menu elements hidden; a toolbar is seen on the left, and tool options for the currently selected tool are seen at the top. The right panel shows a list of various fine tuning tools such as skewing by matrix and exporting images. The rightmost bar shows snapping options.

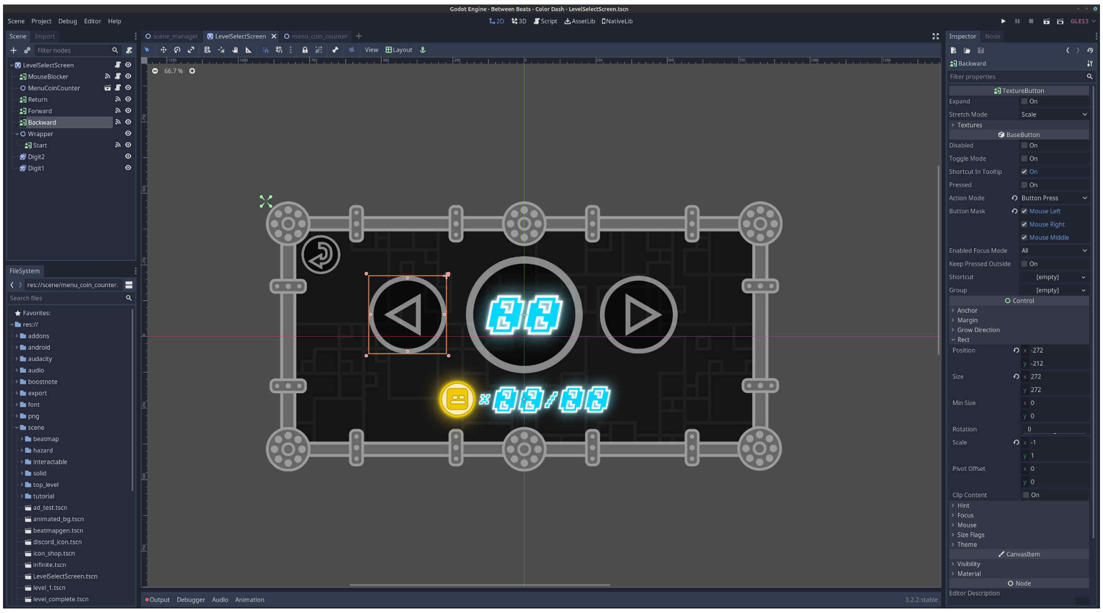

Godot Engine - Between Beats - Color Dash - LevelSelectScreen.tscn
Scene
Inspector
FileSystem

Figure 16. A screenshot of Godot Engine editor v3.2.2.stable.official. A scene containing the level selection screen is being edited. (Figure: CC BY 4 Markus Hirsimäki). [43]

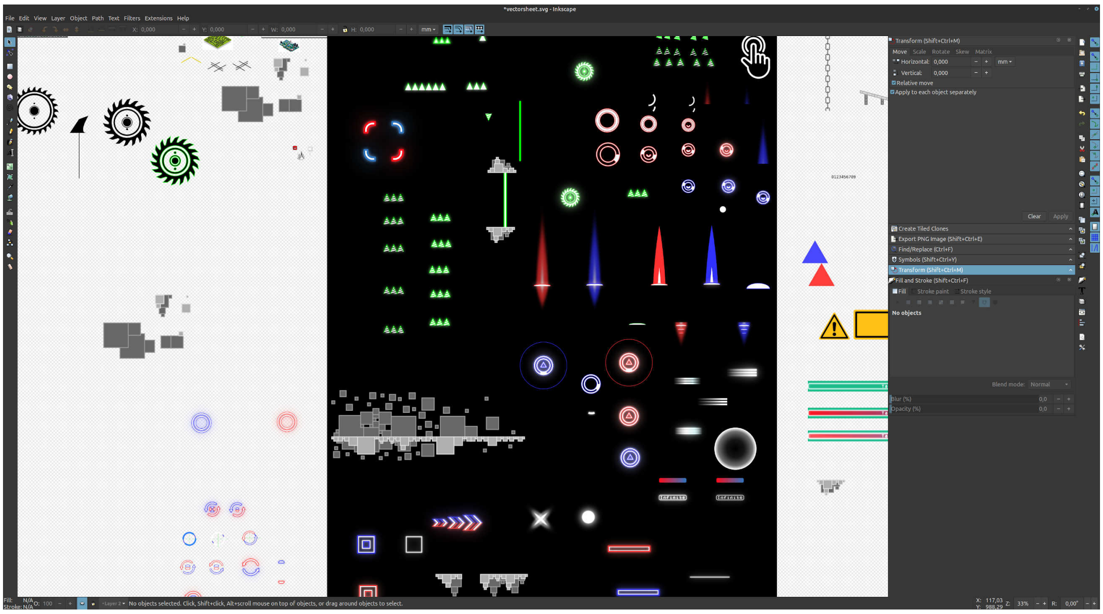


Figure 17. A screenshot of Inkscape 1.0. Within the editor a partial view of vector based sprite sheet is seen. (Figure: CC BY 4 Markus Hirsimäki). [51]

3.6.3. Audacity

Similarly to Godot and Inkscape, Audacity is free and open source. Audacity is the audio editor that was used to record, edit, mix, and master all the audio used in the project. Audacity contains most basic audio editing tools such as cutting, amplitude enveloping, mixing, and adding effects [52]. In figure 18 four tracks can be seen the last one of which was mixed from the three above tracks. The third track was created using Audacity's built in signal generators.

At the top of the editor a toolbar containing playback settings and some tools can be seen. On the left, information about the tracks can be seen; if the tracks are mono or stereo, their bit rate, their relative volumes, and track display type (spectrogram, waveform, decibels).

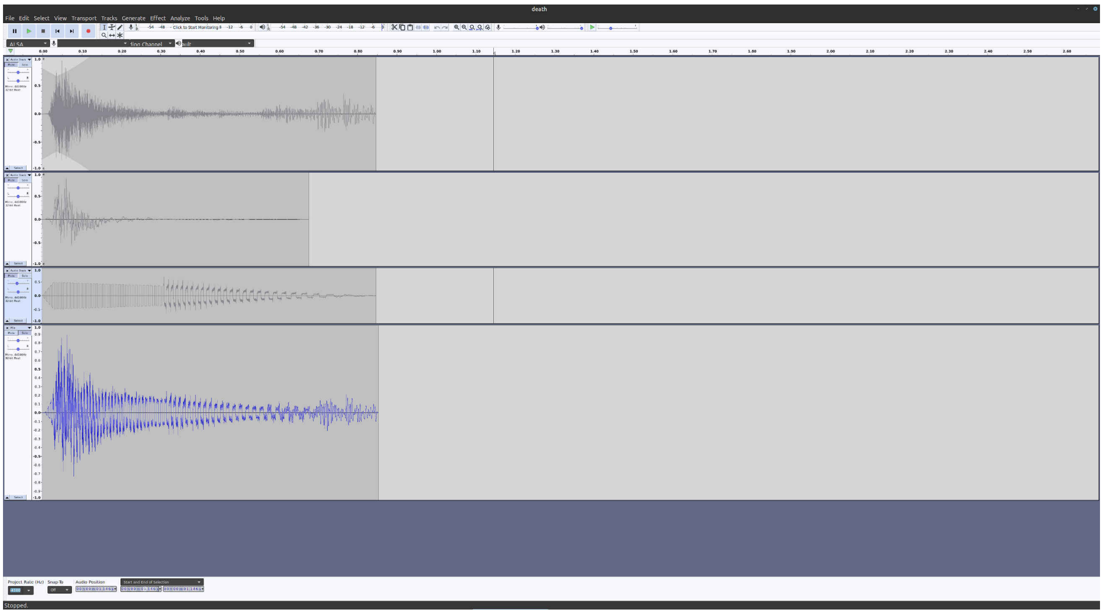

death
File Edit Select View Transport Tracks Generate Effect Analyze Tools Help
Project Rate (Hz)
Snap To
Audio Position
Stopped.

Figure 18. A screenshot of Audacity 2.3.3. Within the editor an Audacity project file containing the mastered sound effect for player's death is seen. (Figure: CC BY 4 Markus Hirsimäki). [51]

3.6.4. Other Software

In addition to the above software, other useful tools used during the project include: git, for version control; OBS, for recording videos; GIMP, for editing raster graphics; Ubuntu, the operating system used on the workstation; and lastly, Discord, the only commercial software used, for communication with the gaming community. These tools were not of lesser importance during development, but they are grouped into one category as they were used for auxiliary purposes instead of strictly content creation or development.

4. EXPERIMENTS

This chapter is a chronological description of the project from the point of view of the testing and eventually performing the main study of this thesis. The reader should note that while the sections for the Pre-Alpha and late Alpha are seemingly short compared to the Beta testing they spanned a timeframe in the order of half a year while Beta testing happened within weeks.

The vast difference arises from Pre-Alpha and Alpha stages being used for both developing the game and writing this thesis. This is in contrast to Beta testing phase, which entailed a code freeze until this thesis was finalized within a timespan of weeks after beginning large scale tests. Additionally, as the project reached a state in which this thesis was being finalized, I contacted a large amount of mobile game publishers and took part in Finland Games Job Fair [53] to showcase the project.

The distinction between Pre-Alpha, Alpha, and Beta stages of development is as follows; the Pre-Alpha stage included everything until the point at which development of menus and different playable levels started, the Alpha stage included what followed until the start of large scale player testing, and Beta includes everything starting from the large scale testing. During the writing of this thesis, the project did not leave the Beta stage and is instead in code freeze. These different stages are best pictured, in the aforementioned order, by figures 14, 19, and 22.

4.1. Pre-Alpha and Related Testing

The project has relied heavily on player feedback since the very beginning. In this section, the thesis provides a free-form description of the earliest player tests and some of their results. It is noteworthy that at this point, the testing was purely practical and informal. This was partly because the final form of the game was undefined and partly because compiling bullet point lists was sufficient to keep track of the project. Anonymized and translated samples of some early notes are provided in appendix 4 to show the style of the feedback received.

In the very early stages, the main way to gather feedback was to distribute a test build of the game to close friends. These test versions were shared via mobile instant messengers and email; these early version were compiled for both Windows and Android, as opposed to the later versions that were distributed to larger audiences via Google Play and targeted only Android. The first test distributed to a small group contained only a proof of concept for using procedural animation to move the player character to the point that was tapped on a screen.

A noteworthy detail is that most of the feedback, especially in the first rounds, seemed to focus largely on detail. While this was unexpected, it is easily understood to be caused by the small amount of playable content. Additionally, at this time, a license with a private individual was signed to licence the background music used in the game.

4.2. Late Alpha and Related Testing

In late Alpha stages, the testing expanded to cover, feedback gained from social media. While builds were still not shared with larger audiences feedback was regularly obtained by posting content such as videos and screenshots on social media. A vital part of feedback in this phase was building a community around the game on Discord.

Discord is a instant messaging application that offers voice calls, video conferencing, and file sharing on major platforms and web browsers. Discord is focused around communities colloquially called servers and the games Discord server later became a central hub for the game's community [54].

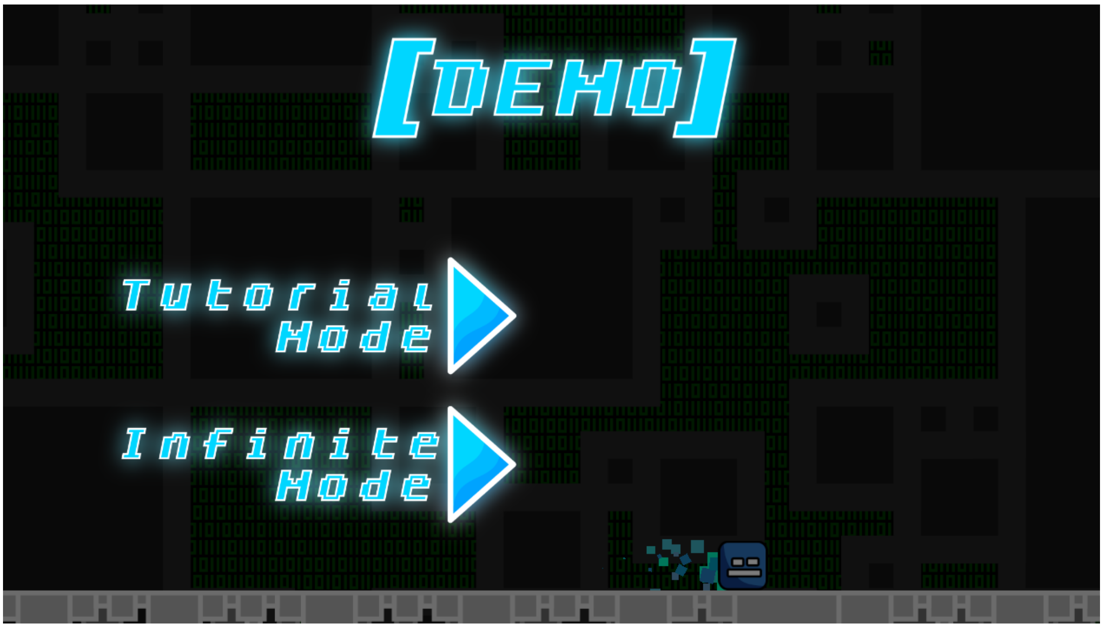


Figure 19. Screenshot of start menu used in late Alpha phase. (Figure: CC BY 4 Markus Hirsimäki)

The project gained an online following during its development and writing of this thesis. The formation of this community was the result of deliberate efforts to build up hype around the game; the initial marketing plan involved reaching a relatively large player base for the game through social media.

The three most important tools used to build this community were Discord, Reddit, and Twitter. Discord served as a hub towards which players were directed to whereas Reddit served as a platform to advertise the game to relevant audiences, such as fans of rhythm games or Geometry Dash. From a functional standpoint, Twitter shared a purpose very similar to Reddit, but each one of these platforms has its unique limitations, nuances, and etiquette.

Most of the feedback received in this phase of development came via the Discord community, as opposed to the earlier stages when the feedback was primarily received from close friends. While the early feedback in the Pre-Alpha was detail oriented, the feedback at this stage was broader and more focused around suggestions on how the project could move forward. This difference as compared to Pre-Alpha stage arose naturally as a consequence of discussion around the project. The discussion spanned various subjects and some noteworthy events were held on the server; brainstorming session to help decide a name for the project and three instances of voting to decide on different player icons that would be added first to the game. In hindsight, it can be said that both of these feedback types, detail oriented and general, were equally important but they targeted different aspects of the project.

An important detail of this phase is that the tutorial used in the game reached its final state at this point after several iterations. These iterations were based on feedback from both the Discord members as well as real life acquaintances, some of whom had no mobile gaming experience to speak of. The final iteration succeeds in explaining the game mechanics without any written language.

The use of social media for building a large enough player base proved to be infeasible for the purposes of this project. Instead of providing a detailed description of these efforts in social media that ultimately did not follow the design principle of maximum impact with minimum effort, the reader is encouraged to take a look at a different starting point that proved to be useful. This starting point was a guide to social media marketing by Oliver Peterson on Process Street [55]. The reader should, however, note that not all games fall within the target category of games seen in this guide.

4.3. Beta Testing

This section will explain in detail some of the key aspects relating to the Beta testing phase. This section will start with a description of the advertising implementation which is followed by description of the four groups used in testing. Finally, the section will define the research method from the practical standpoint.

4.3.1. Advertising Implementation

The advertising is composed of two different advertising types: banner advertisement and interstitial advertising. Both of these were implemented by using a Godot engine plugin, godot-applovin-max, by GitHub user DrMoriarty [56]. The plugin allows the game to fetch and display advertising provided by the AppLovin MAX mediator framework [57]. Examples of both the intersitial and banner advertising types can be seen in figure 20.

Implementing the advertising proved to be an extremely troublesome task. Solving these problems ultimately involved the following: trying two different plugins for advertisement implementation, creating console overlay to aid debugging, contacting both private individuals and corporate entities for support, and recruiting testers from Discord to troubleshoot bugs. This process caused a delay of one week in the project's schedule.

Excluding tutorial level, the banner advertisements are automatically triggered when player enters a level and automatically hidden when they leave a level. It should be noted that it is possible for the game to request to view and ad and receive a response from the network that there are no eligible advertisements to be displayed

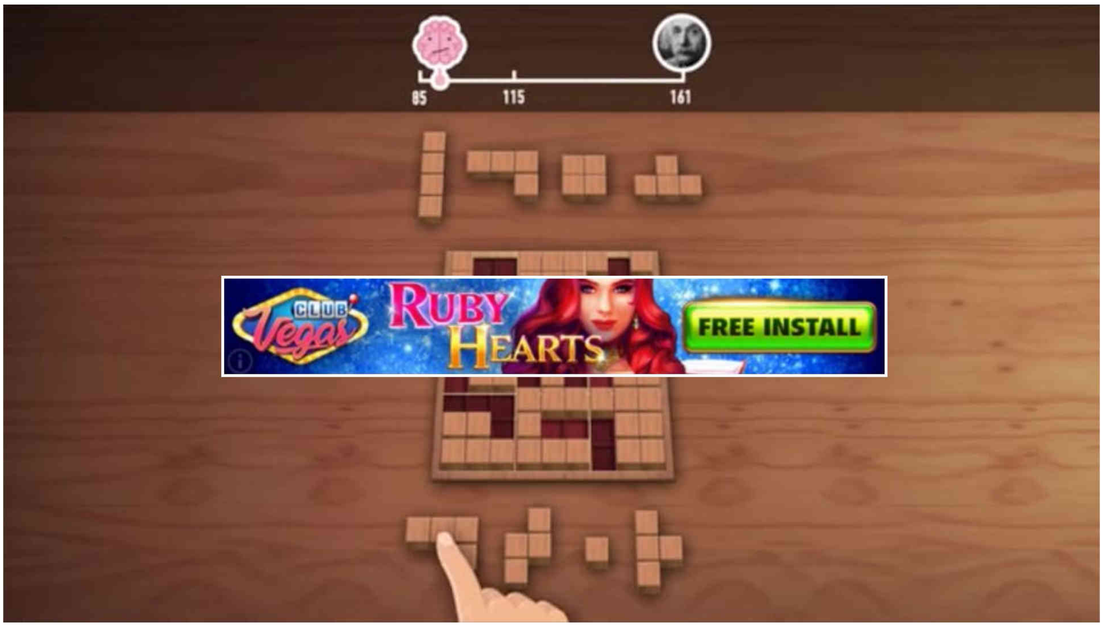


Figure 20. Examples of banner advertising (superimposed) and interactive interstitial advertising as seen in the game during testing. The banner advertising appears only during gameplay and covers only part of the display at the top edge. Interstitial advertising covers the whole display and can be either vertical, horizontal, or padded (Figure: CC BY 4 Markus Hirsimäki)

at the moment. This response proved to be the most common one in the context of banner advertising during the Beta testing.

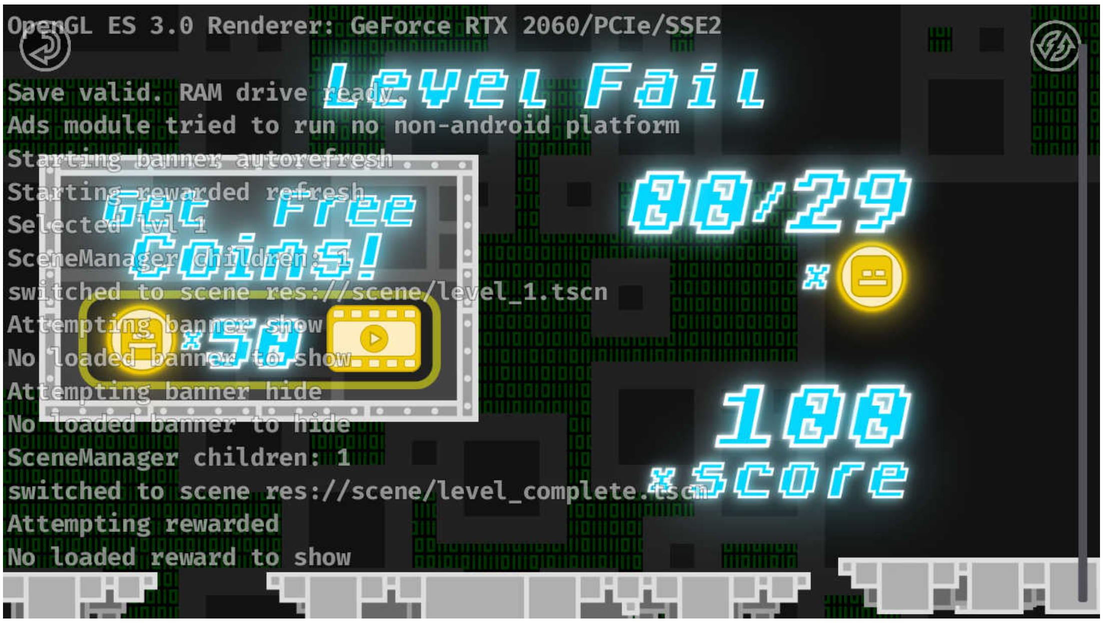
Interstitial advertisements generally had high fill rate and were always available to be displayed. These interstitial advertisements were used to split the test players into four groups, G1 through G4, according to table 1. Splitting the players into these four groups was achieved by requiring the player to enter a key code when they first opened the game; they code was used to dictate the player's group.

Figure 21 shows a window offering coins for watching an advertisement. In combination with non-rewarded interstitial advertisements these two choices were used to split the players into the four groups. The way how the test players were distributed among the different groups is explained below in section 4.3.2 with more detail.

The five minute cool down between interstitial advertisements can be considered arbitrary as the average session length and session count were not know prior to beginning of the tests. This arbitrarity is due to the industry standard being to balance the amount of advertising in terms of revenue. The amount of revenue is derived from the measure of advertisements per session which is used to define average revenue for daily user when the amount of sessions per day is known [33, 32].

4.3.2. Beta Test Groups

The Beta testers were distributed into four different categories that are seen in figure 1. The differentiation between the groups are best explained by a free form description of



```
OpenGL ES 3.0 Renderer: GeForce RTX 2060/PCIe/SSE2
Save valid. RAM drive ready.
Ads module tried to run no non-android platform
Starting banner autorefresh
Starting rewarded refresh
Selected lvl 1
SceneManager children: 1
switched to scene res://scene/level_1.tscn
Attempting banner show
No loaded banner to show
Attempting banner hide
No loaded banner to hide
SceneManager children: 1
switched to scene res://scene/level_complete.tscn
Attempting rewarded
No loaded reward to show
```

Figure 21. Screenshot of the screen a player will see when a level is completed or the character has died and rewarded advertising is available. Additionally, figure shows an overlay of console output used for debugging purposes. The console was not present in the version provided to testers. (Figure: CC BY 4 Markus Hirsimäki)

each group. These descriptions are provided below. Additionally to the advertisements described below, each group saw banner advertisements but these were commonly unavailable and many players saw them only once. The banner advertisements were used to establish a baseline for all the groups.

The group G4 saw both rewarded and interstitial advertising. Every time the character died, except for the first parts of the tutorial, an interstitial advertisement would attempt to display. For the group to actually see the advertisement two conditions hard to be true; the last automatically displayed interstitial advertisement must have been shown more than five minus ago, and the advertisement must have been loaded. If, and only if, this five minute cooldown was ongoing, a character's death would instead result in the game offering a rewarded advertising, as seen in figure 21.

The group G1 was otherwise similar to G4, but the players were not offered rewarded advertising even if the interstitial advertisements were on cool down. The counterpart of the G1 group was group G2 was only offered rewarded advertising on each death without any cool down period. Lastly, the group G3 provided a common ground for the other groups by having no rewarded advertisements and no interstitial advertisements.

In addition to this horizontal split between the players, the sample can also be split vertically based on date. The answer set contains results from two distinct player bases: the Discord community and participants from the University of Oulu's email lists. From the Discord player base, participants were manually assigned to each group so that each group received a near equal amount of potential candidates (total players from Discord and email lists: G1:12, G2:12, G3:11, G4:11). This assignation was done by giving players different keys for the game, which designated them into different groups. These Discord players obtained the game by downloading an installable file from Google Drive. This download link is provided in a footnote along with keys. [1] [2]

The other player base obtained their copy of the game via Google Play as it was determined to be a more straightforward solution for the general audience. As above, the download link is provided in the footnote with keys [3] [4]. A significant difference between the build distributed to the Discord player base, and the build found in Play Store was that instead of using four keys to designate the players to four groups the Play Store build only used one key, and the players were randomly assigned to one of the four groups.

Due to the game not automatically reporting any information about the players it is unknowable how many players proceeded to download the game via Play Store and how many players there were in each group. This could have caused some of the bias that is seen in figure 1. This possible bias is further elaborated on in the section 5.1.1. However, it is known that 22 players answered after information about the game was shared on email lists. The distinction between player bases originating from email lists and Discord is important as these two groups could have different initial opinions towards the game. This is further explained in the section 4.3.3. It is important to note that the players joining via email lists were randomly assigned to their respective groups whereas the members from Discord were evenly assigned to different groups in the order of requesting a Beta key.

Table 1. The distribution of answers between the 4 answer groups. Imbalance can be seen in the answer count towards less advertising receiving more answers. Players invited from the Discord community are over-represented in G4, whereas players that had no previous experience are under-represented in G1.

	Rewarded advertising	**No rewarded advertising**
Interstitial advertising	G3, n=9 (17.6%)	G1, n=11 (21.6%)
No interstitial advertising	G2, n=12 (23.5%)	G4, n=19 (37.3%)

4.3.3. Defining Research Method

The research was done by deploying a build of the game first to a known group of Discord members followed by the general audience, both as described in the section 4.3.2. The players were only given a brief instruction to allow for a natural exploration of the game.

Generally, these instructions can be summed up as follows: "Download and install the game from this source. When prompted, enter the key you were given. The game main menu contains a link to the survey you should complete". Additionally, a download link and relevant key were provided.

The testers were not aware of the existence of different groups and advertisement setups, nor were they aware of the fact that the testing was related to advertisements in

[1] https://drive.google.com/file/d/184CMTHfh5DmS9vhoDTbnY2UnlKFv4ejm/

[2] Drive keys; G1: countermissile, G2: infrastructure, G3: nonsymmetrical, G4: transportation

[3] https://play.google.com/store/apps/details?id=com.vivuta.betweenbeatsS

[4] Play Store key: bbopenbetaddemo

any way. The player base from Discord, however, was at least partially familiar with the game. The effects of this familiarity are explored further in the section 5.1.1. To get a proper understanding of the gameplay experience, the reader is encouraged to take a look at figure 22 and its accompanying YouTube video [58].

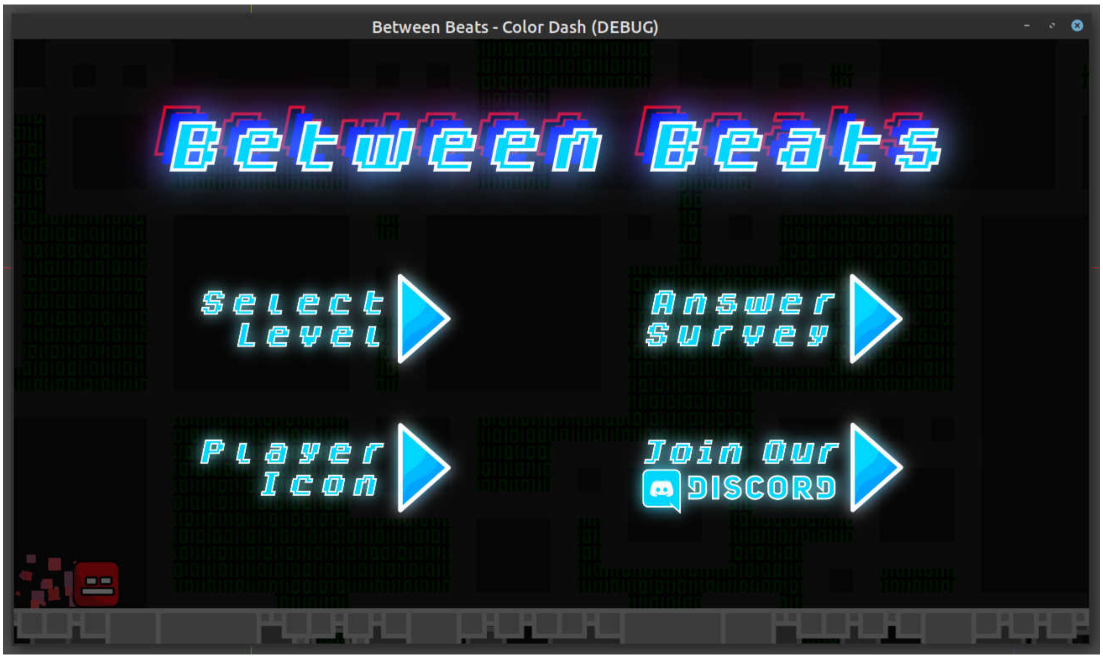


Figure 22. Screenshot of start menu used in Beta testing. A video showcasing the content of the version seen here is available on Youtube [58]. (Figure: CC BY 4 Markus Hirsimäki)

After the players had installed the game and entered their key they were met by the main menu seen in figure 22. From this main menu the player can either open the level selection overlay seen in figure 16, or open the icon selection overlay, or open an external link to the Discord server, or open an information screen about the survey. Next, the thesis will explain some parts of the game as experienced by a player.

From the level selection screen player is able to select either level 00, 01, or 02. Of these three, the first one is a tutorial meant to teach the player the game's basic mechanics. The tutorial achieves this goal by incrementally introducing more information to the player. Initially, the character is presented on a still platform accompanied by an animation of a finger tapping the screen.

When the player taps the screen a drop-down banner will show up with three circles. One of these circles is filled by a smiley face. The goal of this step is to encourage the player to tap the screen; each tap will add one more smiley face until all three slots are filled. Additionally, the character will switch color on each tap. After this this the animated finger will start to aggressively tap the screen to encourage player to tap even faster while the smileys are moved out of screen. After an amount of taps, the player will drop off the platform. At this point, it is assumed that the player has understood that tapping will switch the character's color.

The next screen within the tutorial will have the player pick the right color for the character to progress past a hazard within the level by using a blue teleporter. On failure, the character respawns on the edge of the static screen. Additionally, at the top of the screen a banner can be seen that has a blue player icon followed by a plus-sign and an picture of a blue teleporter. These are followed by an equals sign and a smiley face. After the player has progressed past these two static screens, it is assumed that the player has learned the basic mechanics. These assumptions are supported by the fact that the Beta test feedback contained no mentions of confusing game mechanics, and one player commented that the tutorial was well functioning.

The tutorial continues further after this point and after showcasing the mechanics of the game, the tutorial will take the player to a level-completed screen. On this screen, the player is able to see that they collected six or seven coins during the tutorial. After this, the player returns to the main menu and will likely explore the option to select player icon, answer the survey, or play another level.

Choosing the Player Icon button will open a drop down menu from which the player is able to switch how the player character looks and unlock additional icons with the coins they have collected. The option to unlock more icons is meant to encourage the player to complete all levels, to replay levels to find all coins, and to watch rewarded advertisement to gain additional coins. The shop window can be seen below in figure 23.

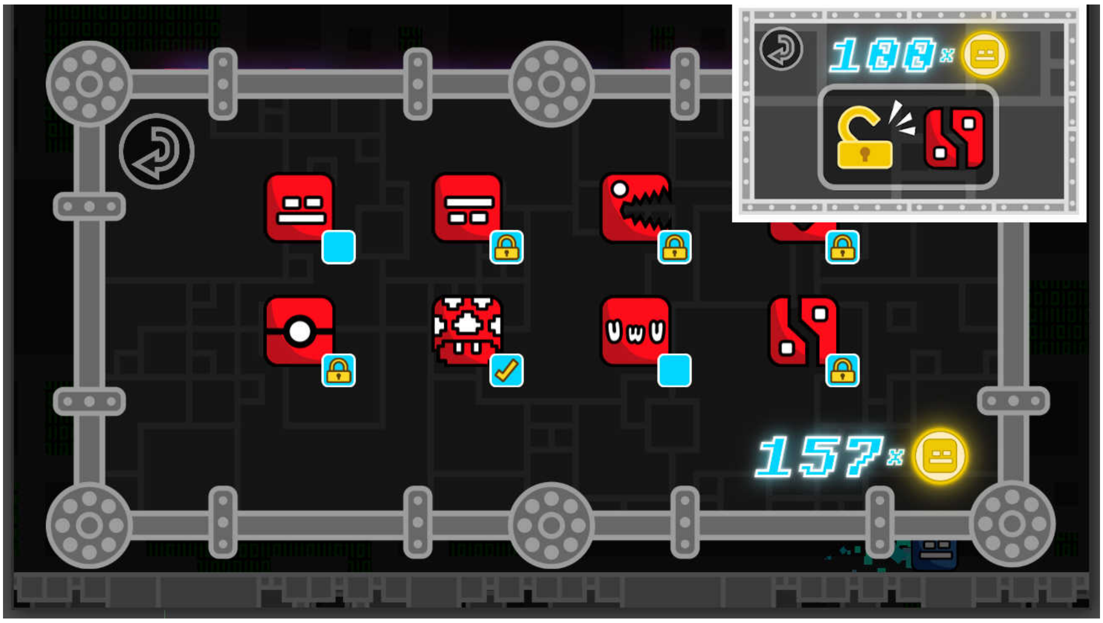


Figure 23. A screenshot of the shop window as seen when opened from main menu. Superimposed on top right is the modal window that pops up, centered and larger, when player attempts to unlock an icon. (Figure: CC BY 4 Markus Hirsimäki)

Choosing the level 01 from the main menu will take the player to a level that employs the mechanics they have learned from the tutorial. The same is true for level 02 with the distinction that level 02 can not be completed as it employs the infinite level generator seen in section 3.2.

The main menu option, Answer Survey, will display additional information, and an external link to Google Forms in which the survey was hosted. The additional information tells the player to only complete the survey after playing for fifteen minutes or after completing the level 01.

The survey was created using Google Forms and it consisted of three sections; a disclaimer section with consent form, background and demography section, engagement section and feedback section. The contents of the three sections are as follows; section one confirms that the player has played the game for 15 minutes of completed level 01. Additionally the section has the player agree to a privacy policy and declare if they have played earlier versions of the game. The section two collects background and demography information such as gender, age, and location. Section three uses the engagement measures described in the section 2.3.3 and appendix 5. Lastly, the section three collects more general feedback such as information about possible bugs the player might have encountered.

5. RESULTS

This section will first present the quantitative results and their related analysis on sources of error that were obtained via the Beta test survey. The quantitative results explore how the differences in advertising methods influenced the test groups in the terms of advertising intrusiveness and player engagement. These results are followed by qualitative results that focus on the feedback gained from the survey, such as technical problems and ideas for improvement.

5.1. Beta Test Quantitative Results

The results were acquired by having the test players answer a survey after they had played the game as described in the section 4.3.3. The players originated from two distinctly different groups; players invited from the game's Discord server, and players invited from the University of Oulu's email lists. The survey answers were collected via Google Forms.

Table 2. Demographics.

<table>
<tr><th colspan="6">Gender</th></tr>
<tr><td colspan="2">Male</td><td colspan="2">Female</td><td colspan="2">Other</td></tr>
<tr><td colspan="2">38 (74.5%)</td><td colspan="2">12 (23.5%)</td><td colspan="2">1 (2.0%)</td></tr>
<tr><th colspan="6">Age</th></tr>
<tr><td>13 or less</td><td>14-17</td><td>18-23</td><td>24-29</td><td>30-39</td><td>40-49</td></tr>
<tr><td>7 (13.7%)</td><td>14 (27.4%)</td><td>16 (31.4%)</td><td>10 (19.6%)</td><td>3 (5.9%)</td><td>1 (2.0%)</td></tr>
<tr><th colspan="6">Location</th></tr>
<tr><td>EU &amp; Russia</td><td>NA</td><td>Asia</td><td colspan="2">Lat. Am. & Carib.</td><td>Oceania</td></tr>
<tr><td>41 (80.4 %)</td><td>5 (9.8 %)</td><td>3 (5.8 %)</td><td colspan="2">1 (2.0 %)</td><td>1 (2.0 %)</td></tr>
<tr><th colspan="6">Recent mobile gaming experience</th></tr>
<tr><td colspan="2">15 or less</td><td colspan="2">16-30</td><td colspan="2">31 or more</td></tr>
<tr><td colspan="2">39 (76.5 %)</td><td colspan="2">7 (13.7%)</td><td colspan="2">5 (9.8%)</td></tr>
<tr><th colspan="6">Time spent playing Between Beats</th></tr>
<tr><td>15 or less</td><td>16-30</td><td>31-45</td><td>46-60</td><td colspan="2">61 or more</td></tr>
<tr><td>18 (35.3 %)</td><td>30 (58.8 %)</td><td>2 (3.9 %)</td><td>0 (0.0 %)</td><td colspan="2">2.0 (%)</td></tr>
</table>

The survey used three different types of inquiries; multiple choice questions, statements on a numeric scale of one to five, and open ended questions. The numeric Likert scale used was the same for every statement and ranged from one, strongly disagree, to five, strongly agree. Multiple choice questions were used for collecting demographics data which is presented in table 2. One player's reported red-green color blindness is excluded from the table.

5.1.1. Limitations On Quantitative Results

The data was first analyzed to gain insight on possible biases that could exist within. Specifically, one-way between groups ANOVA test was used to analyze if the players originating from Discord had different views towards the game than those who were recruited from the email lists. This analysis was done in relation to the cognitive engagement, as measured by the test presented in the appendix 5. The same analysis for bias was done in relation to advertising intrusiveness, as seen in section 5.1.2. The analyses of bias towards cognitive engagement and towards advertisement intrusiveness were also performed by splitting the player base based on the previous experience with the game, as opposed to their source of recruitment.

The reason these differences could cause bias in further analysis was that the members recruited from Discord and email lists were not equally distributed between the four groups used in the analysis. Similarly, players with previous experience were not equally distributed between the four groups. The Discord members are over-represented in G4 and under-represented in the other groups. Players that had no previous experience are under-represented in G1 and over-represented in the rest of the groups.

Table 3. Difference between players originating from Discord and email lists in the context of advertising intrusiveness.

Advertisement intrusiveness (4 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Discord	29	116	3.22	1.24 ± 0.11	**0.099**
Email	22	88	3.51	1.30 ± 0.14	

Table 4. Difference between players originating from Discord and email lists in the context of player engagement.

Cognitive engagement (11 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Discord	29	319	3.16	1.12 ± 0.063	0.75
Email	22	242	2.58	1.22 ± 0.079	
Affective engagement (10 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Discord	29	290	3.01	1.11 ± 0.065	**p<0.001**
Email	22	220	1.91	1.07 ± 0.072	
Behavioral engagement (8 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Discord	29	232	2.84	1.21 ± 0.079	**p<0.001**
Email	22	176	1.82	1.08 ± 0.081	
Combined (29 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Discord	29	841	3.02	1.15 ± 0.040	0.75
Email	22	638	2.14	1.18 ± 0.047	

The differences between these four groups are seen in figure 1. G4 is considered to be the baseline group that received no interstitial advertising and no rewarded advertising. This is in contrast to G3 that received both. G2 and G1 respectively received only rewarded and only interstitial advertising. Additionally, each one of these groups received banner advertising to further define a baseline for the deployment. It should be noted that many of the players received only a very limited amount of banner advertisements as the advertising network used did not provide an unlimited amount of banner advertisements to be used. This is opposite to rewarded and interstitial advertisements that were always available when requested.

The results of testing for the bias between Discord and email groups can be seen in the tables 3 and 4. These two tables are followed by tables 5 and 6 that explore the possible differences between players that have previous experience and those that do not have previous experience. All of the tables show a sample count that is obtained by multiplying the amount of questions for each category by the amount of players.

Tables 5 and 6 show the previous experience based on a yes-no statement; "I have played some other versions of Between Beats before this one" at the very beginning of the survey.

Table 5. Difference between players that had previous experience and players that had no previous experience with the game in the context of advertising intrusiveness. Higher values equal higher perceived intrusiveness.

Advertisement intrusiveness (4 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Has xp.	13	52	3.19	1.21 ± 0.17	0.32
No xp.	38	152	3.39	1.29 ± 0.11	

Table 6. Difference between players that had previous experience and players that had no previous experience with the game in the context of player engagement.

Cognitive engagement (11 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Has xp.	13	143	2.83	1.02 ± 0.085	0.84
No xp.	38	418	2.94	1.25 ± 0.062	
Affective engagement (10 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Has xp.	13	130	2.65	1.13 ± 0.099	0.21
No xp.	38	380	2.50	1.24 ± 0.064	
Behavioral engagement (8 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Has xp.	13	104	2.61	1.17 ± 0.11	0.56
No xp.	38	304	2.33	1.28 ± 0.074	
Combined (29 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
Has xp.	13	377	2.71	1.10 ± 0.057	0.81
No xp.	38	1102	2.62	1.29 ± 0.039	

Finally, the quantitative results have a possible source of error that is difficult to quantify. Table 7 shows the differences between group sizes and the differences between the amounts of advertisements watched on average for each group. The negative correlation between the different group sizes and the different counts for advertisements viewed raises into question if some participants assigned to groups with more advertisements decided to not answer the survey more often than their counterparts in other groups. The reason for G4 seemingly having a non-zero advertising count is explored in the next section.

Table 7. Table showing group names, group size, and the mean amount of full-screen advertisements watched by each member of the group.

	Rewarded advertising	**No rewarded advertising**
Interstitial advertising	G3, n=9 (17.6%), 7.06	G1, n=11 (21.6%), 2.14
No interstitial advertising	G2, n=12 (23.5%), 0.58	G4, n=19 (37.3%), 0.63

5.1.2. Results On Advertisement Intrusiveness

The intrusiveness of advertising was measured using four statements. These statements were as follows; "The in-game advertisements were distracting.", "The in-game advertisements were too frequent.", "The in-game advertisements were not too long.", and "The in-game advertisements were interesting." During analysis, the last two of these had their scale inverted.

In addition to these four statements, an open ended question was asked; "Estimate how many full-screen advertisements you saw". The answers to this question might be inflated as the participants could have confused banner advertisements with full-screen advertisements due to language problems or misunderstanding. The term *full-screen advertisement* was used to catch both interstitial advertisements and rewarded advertisements without revealing to the player what exactly was being studied.

This possible misunderstanding is demonstrated by three individuals in the G4 reporting 7, 3, and 2 advertisements as opposed to the expected zero as. For the purposes of analysis it is assumed this inflation remains constant between different samplings of the answers. Players that gave answers, such as "between three and five" had their answers replaced with the closed numeric equivalent, which would be four in this example. Table 9 shows the number of advertisements viewed in different groups.

The results on perceived intrusiveness of advertising can be seen in table 8 The low p-value (<0.00) indicates that the results are statistically significant. Table 8 is later considered in conjunction with table 9 when determining the results.

The two aforementioned tables allow for intriguing conclusions that agree with related literature presented in section 2.3.2. The number of advertisements viewed does not seem to strictly correlate with the amount of perceived intrusion; G3 has a vastly higher number of advertisements viewed than any other group while still having lower perceived intrusion than G1. The factor changing between G3 and G1 is that while both groups saw interstitial advertisements only G3 had rewarded advertisements.

Furthermore, G1 that had only interstitial advertising had the highest perceived intrusiveness whereas G2 that had only rewarded advertising had the lowest perceived intrusiveness. Together these findings suggest that adding rewarded advertising to situation that has only interstitial advertising will decrease the perceived intrusiveness while increasing the amount advertisements viewed.

Table 8. Differences between the groups for ad intrusiveness. As in other tables presented, G4 is the baseline with only banner advertising. Higher values equal higher perceived intrusiveness.

Advertisement intrusiveness (4 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	44	4.20	0.90 ± 0.14	**p<0.001**
G2	12	48	2.73	1.28 ± 0.19	
G3	9	36	3.722	1.14 ± 0.19	
G4	19	76	3.05	1.19 ± 0.14	

Table 9. Advertisements viewed per player in relation to the test groups G1 trough G4.

Estimate how many full-screen advertisements you saw.							
Group	Users	Min	Max	Mean	Mode	Median	SD
G1	11	0	4	2.14	1	2	1.24
G2	12	0	6	0.58	0	0	1.66
G3	9	0	20	7.06	none	5	6.00
G4	19	0	7	0.63	0	0	1.69

This result could be weakened or invalidated by the bias seen in different groups having different amounts of answers based on how many advertisements they saw; increase in advertising correlates with decrease in answers to the survey. This could mean that the findings are instead explained by change in the player base instead of change in conditions. More specifically, the change in the player base could be that players that are more resilient towards intrusiveness gave more answers.

5.1.3. Results On Player Engagement

Table 10 presents the results for data that has not been normalized in terms of the origin of the players. Player engagement was measured using the survey presented in section 2.3.3 and appendix 5. The bias that could ensue from not normalizing data could effectively make player engagement inflated in G4. The bias in table 10 and the conclusions that can be drawn from the table are elaborated on in the chapter 6.

The p-values for different engagement categories vary greatly. Of the three categories, only affective engagement has a low enough p-value to be considered statistically significant. The table does not present any clear trends in terms of player engagement for a particular category or for the combined question set. The greatest difference between the two means is between G1 in cognitive engagement and G2 in behavioral engagement but even this greatest difference, 0.9, is smaller than the smallest standard deviation seen in the table.

Based on this lack of trends, the thesis concludes that player engagement is not significantly affected depending on advertisements. This conclusion, however, ignores the presented bias. The table 11 presents the normalized data. Normalization was used instead of analyzing Discord and email groups separately due to the sample sizes becoming too small otherwise; smallest subset would have been only three samples as opposed to nine with normalization.

Table 10. Differences between groups in terms of player engagement without normalization.

Cognitive engagement (11 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	121	3.19	1.23 ± 0.11	0.76
G2	12	132	2.95	1.13 ± 0.099	
G3	9	99	2.68	1.22 ± 0.12	
G4	19	209	2.83	1.20 ± 0.083	
Affective engagement (10 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	110	2.58	1.21 ± 0.12	**0.018**
G2	12	120	2.82	1.20 ± 0.11	
G3	9	90	2.83	1.19 ± 0.13	
G4	19	190	2.41	1.22 ± 0.089	
Behavioral engagement (8 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	88	2.57	1.51 ± 0.16	0.40
G2	12	96	2.44	1.13 ± 0.12	
G3	9	72	2.40	1.34 ± 0.16	
G4	19	152	2.28	1.13 ± 0.092	
Combined (29 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	319	2.81	1.34 ± 0.075	0.75
G2	12	348	2.76	1.17 ± 0.063	
G3	9	261	2.50	1.25 ± 0.077	
G4	19	551	2.53	1.21 ± 0.052	

The normalization was performed separately for each of the three engagement categories. The reason for performing the normalization category-wise is that table 4 shows significant differences only for some of the categories. The same is true for table 6 to a lesser extent. In practice and from the standpoint of an individual player, this normalization method will, on average, lower the individual's answers in affective engagement and behavioral engagement categories if the player is from the Discord group.

The choice for lowering answers from the Discord group instead of increasing the answers of email group is due to an assumption that members of the Discord group are biased to give more positive answers due to their prolonged participation in the game's development.

To keep the normalization process simple and avoid over-correcting, answers are lowered in a linear manner. Other options were considered; e.g. correction could have been done by lowering averages of Discord group while increasing them for the email group. The most straight forward approach was chosen to avoid over-correction and to keep the effects of normalization easily understandable. While this approach likely is not perfect, it is reasonable to assume that it will lower the overall bias even if different type of unknown bias is introduced to the data set in the process.

The normalization process involved selecting a constant that was used to multiply each answer in one category if the answer was given by a Discord member. This constant was chosen so that within the category both the Discord group and the email group had the same average value. E.g. data set {3, 4, 5} would be multiplied element-wise by 0.625 to match the average of the following data set: {1, 2, 3, 4}. These multiplies for the three engagement categories, cognitive, affective, and behavioral, were as follows: 0.816830, 0.635685, and 0.639118. Values closer to one mean a smaller normalization effect. The resulting averages are equal with the accuracy of at least 5 decimal places.

After normalization the data seen in table 11 shows changes in p-values when comparing to the non-normalized data. The p-values generally increased, but the affective engagement category maintains a statistically significant p-value.

Table 11. Differences between groups in terms of player engagement with normalization.

Cognitive engagement (11 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	121	2.84	1.085 ± 0.099	0.75
G2	12	132	2.56	0.97 ± 0.085	
G3	9	99	2.33	1.064 ± 0.11	
G4	19	209	2.55	1.073 ± 0.075	
Affective engagement (10 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	110	1.95	0.84 ± 0.080	**0.045**
G2	12	120	2.049	0.86 ± 0.078	
G3	9	90	1.71	0.75 ± 0.079	
G4	19	190	1.91	0.95 ± 0.069	
Behavioral engagement (8 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	88	1.91	0.98 ± 0.10	0.66
G2	12	96	1.79	0.81 ± 0.083	
G3	9	72	1.73	0.92 ± 0.11	
G4	19	152	1.83	0.94 ± 0.076	
Combined (29 questions)					
Group	**Users**	**Sample count**	**Samples mean**	**Samples SD**	**p-value**
G1	11	319	2.28	1.072 ± 0.06	0.75
G2	12	348	2.17	0.94 ± 0.05	
G3	9	261	1.95	0.97 ± 0.06	
G4	19	551	2.13	1.050 ± 0.0447	

5.2. Beta Test Qualitative Results and Observations

In addition to numerical data, a variety of qualitative data was collected both via the survey and via general discussion in the game's Discord server. This section presents the most important notes from these findings by aggregating the answers from the survey and Discord server. Additionally, figure 24 is presented below as it shows information that is not strictly related to the analyses done in this thesis; the information was collected for gamedesign purposes and is of qualitative nature. Observations based on this material are described in the following, however this data did not go through a formal qualitative analysis, as these observations are considered complementary to the quantitative material.

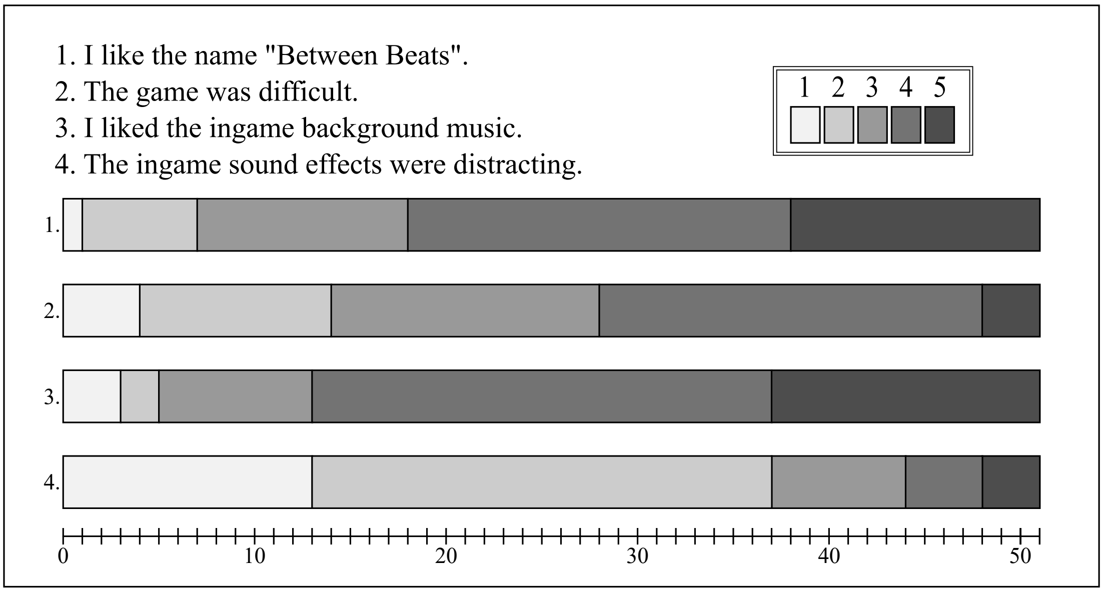


Figure 24. Auxiliary information collected in the survey. Scale of 1 to 5, strongly disagree to strongly agree. (Figure: CC BY 4 Markus Hirsimäki)

5.2.1. Gameplay Problems

Most of the findings related to gameplay problems were relatively minor, but some game-breaking bugs were revealed with the help of the Discord player base. The most important one of these bugs was reported by several players that were running the game on older Android devices. In essence, the players were able to press buttons in the main menu multiple times before the game would react in such a way that the buttons could no longer be pressed.

One player suffered from a problem where the game would only display four large white rectangles on the black background, similarly to piano keys. Additionally, some players running the game on old devices reported that the game's frame rate would drop over time if the infinite map was generated over long enough time periods without the character dying. Furthermore, some old devices suffered from a constant low frame rate.

The problem where frame rate would drop over time as infinite map generation progressed was known before the test but the fix was left out due to time constraints. The infinite map generation also suffered from a problem where the background texture would disappear during gameplay.

Finally, several players reported situations where they were unsure if they failed to select the correct color or if the game failed to register a switch in the player character's color. Despite extensive testing, this has not been replicated during development, and the issue is thus accounted to player error. While the source of the problem likely is a player error, the negative impact on player experience is still acknowledged.

In addition to problems that were clearly erroneous, the players also reported emergent problems; sometimes banner advertisements would cover interactables during the gameplay, which would leave the player with minimal a amount of reaction time. Several players also noted disliking how long some interstitial advertisements were. Interestingly, these notes did not apply to rewarded advertising despite them using the same source for loading adverts.

A different type of advertisement related problem reported by one player was general confusion as to why would a Beta version of the game even contain advertisements. This particular comment reinforces the assumption that despite active discussion the Discord community, as intended, were not aware of the focus of the study.

5.2.2. Player Suggestions

The Beta testing provided a variety of feedback relating to general improvements. Some of the suggestions were already under consideration, while some were novel. An example of an already considered suggestion was the addition of settings for individually choosing the music and sound effect volumes as some players considered them to be either too loud or quiet. A similar suggestion already under consideration was adding a confirmation box when closing the game from the main menu.

Other suggestions were something that were not considered before; some players suggested various Easter eggs that could be hidden in the game while other suggestions focused on quality of live improvements such as going to the next level without visiting main menu after one level was completed. The level-end screen was subject to multiple suggestions such as; adding the ability to drag and drop the animated coins that flood the screen and displaying coins gained from watching rewarded advertisements separately.

5.2.3. Positive Notes

The majority of the positive notes were in relation to three aspects of the game; the graphics, the animations, and the background music. In addition to these positive notes, one player reported disliking the graphical style of the game. Other positive notes included praise on the level design and how the games difficulty ramped up over the levels. Similarly to the graphical aspects, a minority of the players disliked the difficulty progression of the game, while some noted that despite disliking it they thought it was an integral part of the experience.

Several players noted that they were happy to see two variations of the player icon already unlocked at the very beginning of the game. Some players also described the gameplay as being smooth, which was one of the very early design goals. The ability to unlock new icons and seemingly gain progression that way was also described as being a positive experience with many players hoping for more levels and icons.

Within the Discord community, some players started competing amongst themselves on who could achieve the highest score; while the score meter did not gain any specific notes from the players it served an important purpose as it allowed for this sort of competition amongst the players.

5.2.4. Relationship To Other Games

The survey contained two open ended questions intended to gauge how the players associated the game with other titles they were aware of. The overwhelming majority of the answers noted a similarity to Geometry Dash with some answers containing names of multiple other games. Some players noted that the most important similarity to Geometry Dash was the visual style and specifically how the character looked like.

In the order of how commonly the games were mentioned, the most important titles were Chameleon Run, Super Meat Boy, and classic Sonic the Hedgehog games. Many titles were also mentioned only once such as The Impossible Game, OSU, Yoo Ninja, Gravity Guy, Hue and Spin Rhythm. In addition to naming specific titles the players were also asked to name genres. This proved to provide significantly less answers; only 5 categories were mentioned. These were rhythm game, arcade game, puzzle game, platformer, and color switching game.

5.3. Key Results

For clarity, this section summarizes the most important results and their limitations in the order they were presented. The survey involved three sections; demographics and background, player engagement as measured by the survey in the appendix 5, and general feedback along with statements on advertisement intrusiveness. The answers consisted of multiple choice questions, open ended questions, and sliding scale statements ranging from 1, strongly disagree, to 5 ,strongly agree.

The typical player who answered the survey played the game for 16-30 minutes and was a male aged between 18 and 23 living in Europe with little to no recent mobile gaming experience. In total 51 players answered the survey, 29 of whom were part of the game's Discord community, and of whom 22 were invited via University of Oulu's email lists.

The data was first inspected for possible sources of bias in terms of previous experience with the game and in the terms of the origin of the player by performing one-way between groups ANOVA tests. It was found that the previous experience had no significant influence on results, but participation in the games community resulted in players having more positive views towards the game in some aspects. Finally, it was found that the groups that had higher answer counts saw less advertisements, and that the average count reported for viewed full-screen advertisements in G4 was not zero despite the group having received no interstitial or rewarded advertisements.

The qualitative results obtained from the survey and the discussion on Discord revealed important technical problems, player experience problems, and suggestions for improving player experience. The qualitative results also revealed largely positive views towards the game's graphical style and audio. Lastly, the qualitative results highlighted other titles and genres the players associated the game with.

6. DISCUSSION

This chapter focuses on discussing the results presented in previous chapter and understanding how the related biases effect the results. The results were obtained by performing a survey study with constructive approach. The game, Between Beats, was used as a construct to study specific aspects of player experience. The academic key contributions are presented first. They are followed by answers to the research questions presented and analysis on the effects of different limitations the study has. The chapter concludes with a description on other gameplay-related findings.

6.1. Key Contributions

The most important conclusions can be drawn when considering tables 8, 9, and 11 together. The normalized results seen in table 11 behave as expected after normalization, with mean values being generally lower when compared to non-normalized data. The normalized data has a similar p-value for cognitive engagement when compared to the original data. Affective engagement and behavioral engagement, however, have higher p-values in the normalized data. Affective engagement remains as the only statistically significant category, while the variation between means also remains low.

Due to the lack of significant trends in table 11, which shows normalized results on player engagement, it can be concluded that engagement is not significantly affected depending on the advertisements used in the study; table has only one statistically significant p-value but even then the differences between means for different groups are insignificant when compared with their related standard deviations and the size of the scale used. In combination with the non-normalized results, the normalized results suggest that while changes in advertising do not cause significant changes in player engagement, they could cause minor changes in some aspects of player experience without having an effect on other aspects of the player experience.

Tables 8, 9 respectively show perceived advertisement intrusiveness and the average amount of advertisements viewed per group. G1 that had only interstitial advertising had the highest perceived intrusiveness, whereas G2 that had only rewarded advertising had the lowest perceived intrusiveness. G3 had the highest amount of watched advertisements by a large margin without having the highest perceived intrusiveness. Together these findings suggest that adding rewarded advertising to a situation that has only interstitial advertising will decrease the perceived intrusiveness while increasing the amount of advertisements viewed. Additionally, replacing interstitial advertising with rewarded advertising seems to lower perceived intrusiveness while also lowering the amount of advertisements viewed.

This result on intrusiveness could be weakened or invalidated by the bias seen in different groups having different amount of answers based on how many advertisements they saw; an increase in advertising correlates with the decrease in answers to the survey, as seen in table 7. This could mean that the findings are instead explained, or partially explained, by a change in player base instead of a change in the conditions. More specifically, the change in the player base could be that the players that are more resilient towards intrusiveness gave more answers. This possible change in player base in combination with the bias seen between Discord and email groups raises into question if small scale studies can produce meaningful results on this subject. More specifically, the question is if studies with relatively small sample sizes can avoid biases that arise when players are recruited from a specific group. Finally, classifying and quantifying the possible bias, introduced by some players opting not to answer the survey based on their group, would require a separate future study with a different setup. Similarly, a different study would be required to analyze if the results are to this type of study became biased due to some participants opting not to answer.

6.2. Answering the Research Questions

In this section, the answers to the following research questions are summarized along with other information related to them.

- How an increase in advertising influences player experience in mobile games?
- How can an increase in advertising be framed in such a way that it has a minimal negative impact on player experience in the context of mobile games?

It was found that perceived intrusiveness of advertising was lowered if rewarded advertising was added to a situation already containing interstitial advertising. Furthermore, adding rewarded advertising to a situation that had no interstitial advertising lowered perceived advertisement intrusiveness when baseline existed due to banner advertising. G2, which saw rewarded advertisements without interstitial advertisements had the lowest perceived advertisement intrusiveness. The perceived intrusion for G2 was lower than for G4, which saw no rewarded and no interstitial advertisements. It is important to note that each group had a baseline of banner advertising.

These results on perceived intrusiveness are at least partially weakened if the average amount of full-screen advertisements watched per group is taken into account, along with differences in group sizes. Assuming that the differences in group sizes, due to the increase in advertising decreasing the rate of answers, do not invalidate the results it can be said that G3 watched vastly more advertisements than the other groups, but it still did not have the highest perceived intrusiveness. Furthermore, rewarded advertising in combination with interstitial advertising yielded vastly more viewed advertisements than neither of the options alone.

The tests related to player engagement found no significant effect from advertising on the player engagement. The lack of significant findings remained the same after the data was normalized to account for Discord community members having more positive views towards the game.

In the context of mobile games, and assuming that differences differences in group sizes do not explain the results, it can then be concluded that: advertising and the amount of advertisements viewed, can be increased while simultaneously lowering perceived advertisement intrusion and while keeping player engagement constant if the increase in advertising is performed by adding rewarded advertisements to the game. These findings agree with the literature presented.

Additionally, the results raise a question if small scale studies on this subject can produce meaningful results as different sources for participants can introduce various biases. Furthermore, the bias arising from some participants refusing to answer for a reason or another could be a significant source of error that would require a different study setup to analyze. These two questions can likely be generalized to other similar studies. Furthermore, quantifying the possible errors from these would require separate studies.

The Beta testing also generated qualitative feedback on the game. The most important technical problems were related to old Android devices, infinite map generation, and the way how different scenes in the game are managed and transitioned between. Most of the technical problems found will require little effort to fix, but some of the problems are left outside of the scope of the game and considered to happen due to unsupported devices.

The technical problems were accompanied by a variety of suggestions for improving the player experience. Examples of such suggestions were: separate volume controls for effects and background music, ability to move to the next level after completing one level without visiting the main menu, and confirmation before closing the application.

Finally, the survey revealed that the majority of the players were pleased with the graphical style of the game, its animations, and its sounds. Important information about what other games the players associated with the game was also collected. The overwhelming majority of the players associated the game with Geometry Dash, but other titles were also revealed; Chameleon Run, Super Meat Boy, and old Sonic The Hedgehog games.

To summarize the answers to the research questions, it can be said that: an increase in advertising has minimal effect on player engagement while also having significant impact on perceived intrusiveness of advertising. The perceived intrusiveness is lowered when rewarded advertising is added, and the perceived intrusiveness is increased when interstitial advertising is added.

6.3. Limitations of Player Origin and Previous Experience

This section discusses four sources of error that were tested for using one-way between groups ANOVA. The sources of error are: effect of player origin on advertisement intrusiveness, effect of previous experience on advertisement intrusiveness, effect of player origin on engagement, and effect of previous experience on player engagement.

Advertisement intrusiveness was not strongly effected by the source of participants; the differences between Discord and email groups are small and the p-value seen in table 3 is high enough to not warrant further action. Additionally, the difference between the mean values for the groups is less than 0.3, which likely has a negligible effect in further analysis.

Furthermore, advertisement intrusiveness was not strongly effected by the players having previous experience with the game; group with previous experience and group with no previous experience had negligibly small difference between the mean values of the groups, 0.2. This is accompanied by a high p-value of 0.32 seen in table 5. Together these findings indicate that there is no reason to assume for the answers regarding advertisement intrusiveness to be biased in terms of the origin of the players nor the players' previous experience with the game.

The results, however, are very different in the terms of player engagement and origin of the player. The player engagement survey presents three engagement categories; cognitive, affective, and behavioral engagement. The results on the bias for all of these categories are presented separately in table 4 in terms of the origin of the player. Additionally, the table presents the information for the combined ANOVA test after the per-category ANOVA tests.

The low p-values for affective and behavioral engagement suggest that there exists a clear difference between players that originated from Discord and players that originated from email lists in terms of engagement. For two engagement types, the difference between means is significant, over 1.0. This difference is not applicable to all measurements, as seen in the cognitive engagement category that has a high p-value of 0.75. Together these findings point towards data normalization being needed between players originating from these two different sources when analyzing player engagement.

The effects on having previous experience with the game were analyzed, as seen in table 6. The p-values seen in the table are high, and the differences between the means of different groups are low. The highest difference in means seen in the table is smaller than the lowest standard deviation seen. Based on the findings seen in tables 3 trough 6 it is determined that the data should be normalized only for differences between the players origin only when analyzing player engagement.

6.4. Other Limitations

Figure 7 present one more source for bias which can not be easily accounted for. The different groups had vastly different answer counts, and the answer counts between the groups seem to negatively correlate with the amount of viewed advertisements. This raises into question if these differences are cause by the amount of advertisements viewed. The thesis hypothesizes that the players who were more easily annoyed by advertising ended up not answering the survey as the increase in advertising correlates with the decrease in answers. This bias likely partially explains the difference between groups sizes with the rest being caused by random chance. The data set was not corrected for this bias as it's effects are difficult to quantify; further studies focusing on this error are required to quantify it's effects on other studies.

Finally, it is important to note that the advertisement counts seen in table 9 are likely elevated due to language barriers and some participants mistakenly reporting banner advertisements as full-screen advertisements. For the purposes of analysis, it is assumed that these misunderstandings are balanced amongst the groups and that their effect on results is insignificant as the banner advertisements were used to establish a baseline.

A different explanation for the same error could be that some players answered incorrectly to statements that were used to determine their group. These statements were "I saw full-screen advertisements during gameplay." and "I was offered 50 in-game coins for watching an advertisement.". The matrix seen in figure 1 is built from the answers to these two statements. The thesis hypothesizes that the most likely explanation for this error is a combination of a language barrier and failure to report ones group correctly.

6.5. Other Gameplay Related Findings

Some usability issues arose during the Beta testing. These issues are presented here, along with their implications. These limitations were found partially due to the survey and partially due to discussion on the game's Discord server.

The most important one of these problems was caused by the game not disabling buttons in the main menu immediately after they were pressed. More specifically, buttons that would lead to transition from the menu to a in-game scene were not immediately disabled after pressing. This caused no problems on relatively new devices, but older devices allowed for the players to press the button leading to transition multiple times before it would take effect. This caused the game to load several instances of an in-game scene, breaking the games internal state machine.

One player running the game on particularly outdated devices was unable to access the content; only four large white rectangles were displayed on black background, similarly to piano keys, when the game was opened. This problem was left unsolved as the player's device was old enough to warrant deciding to leaving it unsupported. Other players running the game on older devices also reported that the game would slow down if the infinite mode was played long enough. Furthermore, a different problem relating to infinite mode was reported; the game's background would become gray if the player progressed far enough.

Both the frame rate problem and background problem relating to infinite mode were known before deployment. They were, however, left unfixed due to time constraints. The reasons for the problems are known: low frame rate results from the game not properly deleting unused objects that accumulate during infinite gameplay, and the missing background was the result of an oversight as the background texture had an unnecessarily low limit on how many times it could be repeated. All of these problems are left to be fixed in future versions of the game except for the one involving piano tile like appearance on an particularly old device.

7. SUMMARY

This thesis focused on exploring the research questions *How an increase in advertising influences the player experience in mobile games?* and *How can an increase in advertising be framed in such a way that it has a minimal negative impact on the player experience in the context of mobile games?* The questions were explored by deploying a mobile game, Between Beats, for Android users and collecting data with a survey. This thesis and Between Beats are grounded in prior work of individual games and genres that gave influence to the project. In addition, relevant literature was explored in the terms of rewarded video advertising and the context of advertising in mobile games.

The mobile game used for testing, Between Beats, was developed by me over the course of nine months during which this thesis was written. The thesis gives overview on the game's technical implementation while more information is given on level generation, animation technique used, and the connection of music and player movement. A description on how the project and it's complexity was managed is given, along with a view into the explorative and iterative design approach used. The thesis also discusses the effects of the overarching design principle used, *maximum impact with minimum effort*. The description on project management additionally contains a list of software tools used and example use cases.

Mixed data was collected in the form of an online survey targeting player engagement, advertisement intrusiveness, and general feedback. Statistical analysis was performed on player engagement and advertisement intrusiveness, along with analysis on sources of error in the data set. The thesis also presents brief descriptions on earlier, non-systematic player tests that were conducted before the large scale Beta testing using Between Beats. The test users were divided into four groups, all of which saw banner advertisements to establish a baseline. G4 saw no other advertisements besides the baseline, whereas G3 saw both rewarded and interstitial advertisements. G2 and G1 respectively saw only rewarded and interstitial advertising. The study found no significant differences in player engagement between the four groups. The groups, however, differed in their perceived advertisement intrusiveness.

It can be concluded that the number of advertisements and the amount of advertisements viewed can be increased while decreasing the perceived advertisement intrusiveness if the increase in advertising is performed by adding rewarded advertisements to the game while retaining interstitial advertisements. Furthermore, player engagement remained constant over these changes. The findings support the existing literature presented.

The results are limited by a possible source of error; the group sizes differed significantly, and these differences negatively correlate with the number of advertisements viewed. It is possible that the groups that had less members were smaller because more players in those groups decided not to answer the survey due to seeing more advertisements.

In addition to the previous source of error, it was found that the different player bases had small, statistically significant differences in how they answered to player engagement survey. This error was corrected for by normalizing the data set. This source of error, however, raises a further question, that is left unanswered. Can small scale studies on this subject produce meaningful results as different sources for participants can introduce various biases? Regarding this question it is concluded that analyzing the source of error requires additional studies with different set up. The same applies to the bias regarding differences in group sizes.

Finally, the survey collected valuable qualitative information on technical problems, suggestions for improving user experience, general feedback on the game, and information on what other games the participants associated the game with. This information will be used in developing Between Beats further.

8. REFERENCES

[1] Bang A.L., Krogh P., Ludvigsen M. & Markussen T. (2012) The role of hypothesis in constructive design research. Aalto University.

[2] Seif El-Nasr M., Desurvire H., Nacke L., Drachen A., Calvi L., Isbister K. & Bernhaupt R. (2012) Game user research. CHI'12 Extended Abstracts on Human Factors in Computing Systems , pp. 2679–2682.

[3] Steam (2021), Steam tags. https://store.steampowered.com/tag/browse/. [Online; accessed 19-February-2021].

[4] Corporation V. (2003), Steam. https://store.steampowered.com/. [Online; accessed 19-February-2021].

[5] Kiloo & Games S. (2012), Subway surfers. https://play.google.com/store/apps/details?id=com.kiloo.subwaysurf. [Online; accessed 19-February-2021. Developer: Kiloo and SYBO Games. Publisher: Kiloo Games].

[6] Studios I. (2012), Temple run. https://play.google.com/store/apps/details?id=com.imangi.templerun. [Online; accessed 19-February-2021. Developer: Imangi Studios. Publisher: Imangi Studios].

[7] Fitterer D. (2015), Audiosurf 2. https://store.steampowered.com/app/235800/Audiosurf_2/. [Online; accessed 19-February-2021].

[8] Cavanagh T. (2012), Super hexagon. https://store.steampowered.com/app/221640/Super_Hexagon/. [Online; accessed 19-February-2021].

[9] Herbert D. (2007), osu! https://osu.ppy.sh/home. [Online; accessed 19-February-2021].

[10] osu (2014), osu!academy episode 2 - standard gameplaysettings adjustments!! https://youtu.be/mswLEXK0eDk. [Online; accessed 19-February-2021].

[11] Reichelt D. (2015), Color switch. https://play.google.com/store/apps/details?id=com.colorswitch.switch2. [Online; accessed 21-February-2021. Developer: Reichelt, David. Publisher: Color Switch Phoenix LLC].

[12] Wissman B. (2018), Color switch: The mobile game as a multimedia powerhouse. https://www.forbes.com/sites/barrettwissman/2018/08/07/color-switch-the-mobile-game-as-a-multimedia-powerhouse/. [Online; accessed 21-February-2021].

[13] Takahashi D. (2018), After 40 mobile games, david reichelt hit gold with color switch. https://venturebeat.com/2018/03/31/after-40-mobile-games-david-reichelt-hit-gold-with-color-switch/. [Online; accessed 21-February-2021].

[14] Color Switch (2020), Color switch classic mode medium levels 33-72. https://www.youtube.com/watch?v=M6tXhw38pfM. [Online; accessed 22-February-2021].

[15] Fukasaku K. (2000), Battle royale (2000). https://www.imdb.com/title/tt0266308/. [Online; accessed 23-February-2021].

[16] Fillari A. (2019), Battle royale games explained: Fortnite, pubg, and what could be the next big hit. https://www.gamespot.com/articles/battle-royale-games-explained-fortnite-pubg-and-wh/1100-6459225/. [Online; accessed 23-February-2021].

[17] Wikipedia contributors (2021), Battle royale game — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Battle_royale_game&oldid=998768136. [Online; accessed 23-February-2021].

[18] Brace Yourself Games (2015), Crypt of the necrodancer. https://store.steampowered.com/app/247080/Crypt_of_the_NecroDancer/. [Online; accessed 15-February-2021]. Developer: Brace Yourself Games. Publisher: Brace Yourself Games and Klei Entertainment).

[19] Gilles Fauconnier and Mark Turner (1998) Conceptual integration networks. Cognitive Science 22, pp. 133–187.

[20] Nerlich B. (2003) Polysemy: flexible patterns of meaning in mind and language. Mouton de Gruyter.

[21] Zbikowski L. (2002) Conceptualizing music: cognitive structure, theory, and analysis. Oxford University Press.

[22] Michalko M. (2010) Thinkertoys - A Handbook of Creative-Thinking Techniques. Ten Speed Press, 2 ed.

[23] Gibbs, W. Raymond Jr. (2000) Making good psychology out of blending theory. Cognitive Linguistics Journal 11-3/4, pp. 348–357.

[24] Schell J. (2008) The Art of Game Design: A Book of Lenses. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[25] Keith C. (2010) Agile game development with Scrum. Pearson Education.

[26] Koutonen J. & Leppänen M. (2013) How are agile methods and practices deployed in video game development? a survey into finnish game studios. In: International Conference on Agile Software Development, Springer, pp. 135–149.

[27] Kent B., Mike B., Arie v.B., Alistair C., Ward C., Martin F., James G., Jim H., Andrew H., Ron J., Jon K., Brian M., Robert C.M., Steve M., Ken S., Jeff S. & Dave T. (2001), Agile manifesto. https://agilemanifesto.org/. [Online; accessed 12-May-2021].

[28] Raveh J. (2016), In-game advertising in free-to-play mobile: Getting the balance right. https://www.gamasutra.com/blogs/JonathanRaveh/20160317/268363/InGameAdvertisinginFreeToPlayMobileGettingtheBalanceRight.php. [Online; accessed 10-April-2021].

[29] Burns Z., Roseboom I. & Ross N. (2016) The sensitivity of retention to in-game advertisements: An exploratory analysis. AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment URL: https://aaai.org/ocs/index.php/AIIDE/AIIDE16/paper/view/14064.

[30] Kim S., Lee J., Hwang Y. & Jeong S.H. (2016) Effects of prominent in-game advertising in mobile media: Cognitive, affective, and behavioural outcomes and the moderating role of persuasion knowledge. International Journal of Mobile Communications 14, p. 203.

[31] Khai C., Sha Y. & Richard C. (2021), Understanding the effect of incentivized advertising on the conversion funnel. http://dx.doi.org/10.2139/ssrn.3714353.

[32] MoPub (2017), Rewarded video ads drive performance for social casino advertiser. https://www.aarki.com/blog/rewarded-video-ads-drive-performance-for-social-casino-advertiser. [Online; accessed 11-April-2021].

[33] AdColony (2016), Rewarded video & prettygreat. https://www.adcolony.com/wp-content/uploads/2017/02/AdColony-Case-Study-Rewarded-Video-PrettyGreat.pdf. [Online; accessed 11-April-2021].

[34] Smith R., Kelly B., Yeatman H., Moore C., Baur L., King L., Boyland E., Chapman K., Hughes C. & Bauman A. (2020) Advertising placement in digital game design influences children's choices of advertised snacks: A randomized trial. Journal of the Academy of Nutrition and Dietetics 120, pp. 404–413. URL: https://www.sciencedirect.com/science/article/pii/S2212267219310238.

[35] Abbasi A.Z., Umair Shah M., Rehman U., Hlavacs H., Hooi Ting D. & Nisar S. (2021) The role of personality factors influencing consumer video game engagement in young adults: A study on generic games. IEEE Access 9, pp. 17392–17410. URL: https://doi.org/10.1109/ACCESS.2021.3053207.

[36] Abbasi A.Z., Hlavacs H. & Hooi Ting D. (2017) Engagement in games: Developing an instrument to measure consumer videogame engagement and its validation. Int. J. Comput. Games Technol. 2017, pp. 7363925:1–7363925:10. URL: https://dx.doi.org/10.1155/2017/7363925.

[37] Zaheer H. & Mark D.G. (2009) The attitudes, feelings, and experiences of online gamers: A qualitative analysis. CyberPsychology & Behavior 12, p. 747–753.

[38] Hirsimäki M. (2021), Work in progress game preview. https://youtu.be/iXEZifi_uUQ. [Online; accessed 7-February-2021].

[39] Peterson M. (2011), Minecraft. https://www.minecraft.net/. [Online; accessed 19-February-2021. Developer: Mojang. Publishers: Mojang, Microsoft Studios, Sony Interactive Entertainment].

[40] Toy M., Wichman G., Ken A. & Lane J. (1985), Rogue. https://store.steampowered.com/app/1443430/Rogue/. [Online; accessed 7-February-2021. Developer: A.I. Design (Non-unix versions). Publisher: Epyx (Non-unix versions)].

[41] Warren T. (2020), Minecraft still incredibly popular as sales top 200 million and 126 million play monthly. https://www.theverge.com/2020/5/18/21262045/minecraft-sales-monthly-players-statistics-youtube. [Online; accessed 7-February-2021].

[42] Linietsky J., Manzur A. & the Godot community (2020), Tween - godot engine documentation. https://docs.godotengine.org/en/3.2/classes/class_tween.html. [Online; accessed 5-February-2021].

[43] Linietsky J. & Manzur A. (2020), Godot engine. https://godotengine.org/. [Version: v3.2.2.stable.official].

[44] Hirsimäki M. (2021), Compilation of animations. https://youtu.be/7EOAXyjxxVk. [Online; accessed 5-February-2021].

[45] Wikipedia contributors (2020), Space–time tradeoff — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Space%E2%80%93time_tradeoff&oldid=960226605. [Online; accessed 5-February-2021].

[46] *Android Developer* via Google Developers -platform (2021), Distribution dashboard. https://developer.android.com/about/dashboards/index.html. [Online; accessed 5-February-2021, Android is a trademark of Google LLC].

[47] Topala R. (2014), Geometry dash. https://play.google.com/store/apps/details?id=com.robtopx.geometryjump. [Online; accessed 17-February-2021. Publisher: RobTop Games].

[48] Linietsky J., Manzur A. & the Godot community (2020), Audiostreamplayer - godot engine documentation. https://docs.godotengine.org/fi/stable/classes/class_audiostreamplayer.html#class-audiostreamplayer-property-pitch-scale. [Online; accessed 17-February-2021].

[49] Wikipedia contributors (2020), Twelfth root of two — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Twelfth_root_of_two&oldid=994398149. [Online; accessed 17-February-2021].

[50] Wikipedia contributors (2020), Kanban (development) — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Kanban_(development)&oldid=1002634912. [Online; accessed 21-February-2021].

[51] Various Contributors (2020), Inkscape. https://inkscape.org/. [Version: 1.0].

[52] The Audacity Team (2018), Audacity. https://www.audacityteam.org/. [Version: 2.3.3].

[53] Finland Games Job Fair (2021), Finland games job fair. https://gamesjobfair.com/. [Online; accessed 14-April-2021. Contact; talents@gamesfactorytalents.com].

[54] Discord Incorporated (2021), Discord. https://discord.com/. [Online; accessed 14-April-2021].

[55] Oliver P. (2020), How to use social media for indie game marketing: A beginners guide. https://www.process.st/indie-game-marketing/. [Online; accessed 15-April-2021].

[56] DrMoriarty (2020), godot-applovin-max plugin. https://github.com/DrMoriarty/godot-applovin-max. [Online; accessed 16-April-2021].

[57] AppLovin (2020), Applovin max. https://www.applovin.com/max/. [Online; accessed 16-April-2021].

[58] Hirsimäki M. (2021), Level 01 completed & main menu, icon selection. https://youtu.be/4sYcud14B3s. [Online; accessed 14-April-2021].

9. APPENDICES

Appendix 1	Godot Engine licence
Appendix 2	Pitch Shift Workaround
Appendix 3	Most Popular Steam Tags
Appendix 4	Samples of Early Feedback
Appendix 5	Consumer videogame engagement scale

```
#############[ excerpt from gd script file ]#############
# excerpt from gdscript file                            #
#                                                       #
# the variable multi is float type variable describing  #
# the multiplier for tempo. e.g. 0.5 or 2.0             #
#                                                       #
# AudioServer.get_bus_effect(0, 0) refers to audio bus  #
# 0 which is the Master Bus, that has the effect        #
# AudioEffectPitchShift applied to it on position 0     #

my_audio_stream_player.pitch_scale = multi
AudioServer.get_bus_effect(0,0).pitch_scale = 1.0 / multi

#                                                       #
###########[ end excerpt from gd script file ]###########
```

This example assumes the reader to understand Godot Engine's basics as well as the concepts related to its audio server. The example is presented as the documentation surrounding these features is incomplete and acquiring this knowledge requires either rigorous testing and guesswork, or a teacher.

The example is significant as the project could not have proceeded forward as planned if this workaround had not been found. The Godot Discord community was integral in solving the problem.

To achieve a shift in tempo of music without affecting the pitch two separate changes need to be made. Firstly, the actual playback speed is increased by using the `pitch_scale` property of `AudioStreamPlayer`. This will affect the pitch along with playback speed; to counteract this `AudioEffectPitchShift` needs to be added to an audio bus. This allows counteracting the pitch shift produced by `pitch_scale` in the manner presented above.

List of the 350 most popular Steam tags as of February 19, 2021.

Indie, Action, Adventure, Casual, Simulation, Strategy, RPG, Early Access, Singleplayer, Free to Play, 2D, Violent, Sports, Massively Multiplayer, Atmospheric, Multiplayer, Puzzle, Racing, Gore, Story Rich, Nudity, Fantasy, Sexual Content, Anime, Great Soundtrack, Pixel Graphics, Funny, Colorful, 3D, First-Person, Sci-fi, Cute, Horror, Shooter, Arcade, Exploration, VR, Difficult, Retro, Family Friendly, Co-op, Platformer, Survival, Open World, Relaxing, Comedy, Female Protagonist, FPS, Third Person, Action-Adventure, Online Co-Op, Visual Novel, PvP, Realistic, Sandbox, Physics, Design & Illustration, Utilities, Stylized, Choices Matter, Turn-Based, Top-Down, Combat, Dark, Psychological Horror, Mystery, Space, Tactical, Character Customization, Replay Value, Local Multiplayer, Building, Cartoony, Point & Click, Multiple Endings, Management, Minimalist, Education, Old School, Controller, Action RPG, 2D Platformer, PvE, Mature, Futuristic, Crafting, Survival Horror, Side Scroller, Zombies, Classic, Puzzle Platformer, Local Co-Op, Magic, Procedural Generation, Shoot ’Em Up, Memes, Movie, Resource Management, Hand-drawn, Roguelike, War, Cartoon, Fast-Paced, Turn-Based Strategy, Medieval, Web Publishing, Linear, Hack and Slash, Turn-Based Combat, Short, Historical, Party-Based RPG, Roguelite, Dark Fantasy, Beautiful, JRPG, Animation & Modeling, Software, Dating Sim, Experimental, Romance, Post-apocalyptic, Walking Simulator, Stealth, Base Building, Drama, Bullet Hell, Fighting, Surreal, Logic, Choose Your Own Adventure, 3D Platformer, Turn-Based Tactics, Dungeon Crawler, RTS, Interactive Fiction, Hidden Object, Narration, Audio Production, Military, Third-Person Shooter, Music, RPGMaker, Score Attack, Isometric, Team-Based, Cyberpunk, Top-Down Shooter, Robots, Dark Humor, Driving, 4 Player Local, Competitive, Video Production, Text-Based, 1990’s, Aliens, Economy, 2.5D, Hentai, Card Game, Moddable, Flight, 1980s, Abstract, Perma Death, Tower Defense, Emotional, Soundtrack, LGBTQ+, Character Action Game, Cinematic, Action Roguelike, Investigation, Nature, City Builder, Conversation, Board Game, Beat ’em up, Detective, Immersive Sim, Tutorial, Thriller, Psychological, Arena Shooter, Inventory Management, Destruction, Real Time Tactics, Metroidvania, Addictive, Nonlinear, Strategy RPG, Demons, NSFW, Loot, World War II, Modern, Level Editor, Game Development, Psychedelic, MMORPG, Tabletop, Dystopian, Time Management, Software Training, Alternate History, Wargame, Crime, Clicker, Artificial Intelligence, Life Sim, Supernatural, Tactical RPG, Dark Comedy, Parkour, Lore-Rich, Blood, Twin Stick Shooter, Comic Book, Grand Strategy, Precision Platformer, Automobile Sim, 2D Fighter, Souls-like, Illuminati, Science, Open World Survival Craft, Co-op Campaign, CRPG, Runner, Episodic, Mythology, Philosophical, Lovecraftian, Epic, Split Screen, Space Sim, Real-Time, Rhythm, Grid-Based Movement, Cult Classic, Mouse only, Swordplay, Class-Based, Battle Royale, 3D Vision, Gun Customization, Photo Editing, Match 3, Dragons, eSports, Parody, Satire, Political, Steampunk, Bullet Time, Noir, Voxel, 6DOF, America, Deckbuilding, Cats, Vehicular Combat, Trading, MOBA, Mechs, Mining, Conspiracy, Tanks, Hunting, Remake, Time Manipulation, Collectathon, Capitalism, Gothic, 4X, Card Battler, Politics, Hacking, Real-Time with Pause, Pirates, Colony Sim, Ninja, Otome, Time Travel, Cold War, Kickstarter, 3D Fighter, Agriculture, Hex Grid, God Game, Underground, Trains, Quick-Time Events, Superhero, Martial Arts, Word Game, Mystery Dungeon, GameMaker, Dinosaurs, Idler, FMV, Dog, Experience, Farming Sim, Dynamic Narration, Fishing, Touch-Friendly, Automation, Assassin, Naval, Western, Programming, Mod, Trading Card Game, Spectacle fighter, Heist, Hero Shooter, Underwater, Combat Racing, Documentary, Minigames, Asynchronous Multiplayer, Immersive, Dungeons & Dragons, Archery, Vampire, Sequel, Diplomacy, Political Sim, Faith, Looter Shooter, Games Workshop, Solitaire, Narrative, Naval Combat, Sokoban, Party, Gaming, Sailing, Time Attack, Crowdfunded, Party Game, Foreign, Music-Based Procedural Generation, Snow, Football, Silent Protagonist, Soccer, World War I.

This appendix contains anonymized and translated samples of the early feedback received on the project. As the project did not use strict versioning in its early stages this list contains feedback from multiple different test versions in chronological order starting from first playable test that had twenty seconds worth of content.

The comments are translated maintaining their original style and square brackets are used to redact longer personal notes on the subject.

- funny. nice graphics. needs more variety. reminds me of geometry dash. might feel like "already experienced"
- good amount of challenge. Easy to get started
- Needs music and some effect for dying. Its hard to give more feedback yet as the game is short.
- [no feedback from multiple people due to using iPhones]
- The game is going in the right direction. Keep going.
- too fast. cant do it
- Good first impressions i like it
- [elaborate description and idea for infinite generation of map. this was later implemented]
- the colors are off character is blue and jumps violet
- game is too fast. textures are hard to discern
- The jumps pads are hard to make out. Move character closer to the edge of the screen.
- Game is too fast you have to choose both color and the right path in branches

Copy of the player engagement survey used is provided here for the convenience of the reader. Original work is by Amir Zaib Abbasi et. al. [35].

CONSUMER VIDEOGAME ENGAGEMENT SCALE
(Higher-Order/Third-Order Level Formative Construct)

COGNITIVE ENGAGEMENT (Second-Level Formative Construct)
Conscious Attention (First-order Level Reflective Construct)

1. Like to learn more about this video-game.
2. I notice information related to this video-game.
3. I pay a lot of attention to anything about this video-game.
4. I keep up with things related to this video-game.
5. Anything related to this video-game grabs my attention.
6. I concentrate on this video-game's story for a long time

Absorption

1. When I am playing this video-game, I forget everything else around me.
2. Time flies when I am playing this video-game.
3. When I am playing this video-game, I get carried away.
4. When I am playing this video-game, I feel immersed.
5. I feel happy when I am playing this video-game intensely.

AFFECTIVE ENGAGEMENT (Second-Level Formative Construct)
Dedication (First-order Level Reflective Construct)

1. This video-game inspires me.
2. I am enthusiastic about playing this video-game.
3. I am proud of playing this video-game.
4. I find this video-game full of meaning and purpose.
5. I am excited when playing this video-game.

Enthusiasm

1. I spend a lot of my discretionary time playing this video-game.
2. I am heavily into playing this video-game.
3. I am passionate about playing this video-game.
4. I enjoy spending time playing this video-game.
5. I try to fit playing this video-game into my schedule.

BEHAVIORAL ENGAGEMENT (Second-Level Formative Construct)
Social Connection (First-order Level Reflective Construct)

1. I love playing this video-game with my friends.
2. I enjoy playing this video-game more when I am with others.
3. Playing this video-game is more fun when other people around me play it too.

Interaction

1. In general, I like to get involved in the discussions about this video-game playing.
2. I am someone who enjoys playing this video-game with other like-minded video-game players.
3. I am someone who likes actively participating in the discussions about this video-game playing.
4. In general, I thoroughly enjoy exchanging ideas on this video- game with other video-game players.
5. I often participate in activities relating to this video-game.