Under Grove

Educating Users on Dark Design in Video Games Using Procedural Generation

by
Zack Turman

Dr Pete Bennett, Supervisor

A dissertation submitted in partial fulfillment
of the requirements for the
Degree of Master of Science in Computer Science

UNIVERSITY OF BRISTOL
Bristol, United Kingdom
September 27, 2021

Executive Summary

Video games can intentionally utilise dark game design patterns to manipulate players into detrimental behaviour. These design patterns come in a variety of forms that affect a player's money, time, social life, and psychological well-being. Such design patterns are associated with notorious video game consequences such as problematic video game playing and gambling. Reactive measures, such as regulations, are beginning to combat these practices by restricting player access to video games; however, few proactive measures are being taken to prevent the initial implementation of these questionable design patterns.

In this project, the educational video game *Under Grove* has been created to serve as a preventive measure against this prevalence of dark game design patterns. *Under Grove* is a procedurally generated adventure game where players are exposed to design patterns that impact player spending and play duration. The game also allows players to design levels using specific design elements in order to experiment and understand different dark game design patterns. The current prototype establishes a foundation for social networking and competition, which can be used to motivate players to engage in this educational material. By gaining awareness, players can become more resistant to these patterns, and game designers can limit their use in future development.

Players who played this game showed a non-statistically significant increase in familiarity with dark game design patterns. An interview with an expert game designer confirmed the prototype serves as a foundation for an educational game. In future iterations of the prototype, additional narrative, instructions, and motivational gameplay could be included to further improve *Under Grove*'s educational capabilities. The implementation of a database and user accounts will allow analysis of level design, player spending, and play duration in relation to dark game design patterns. With this current foundation, *Under Grove* possesses the ability to impact the efficacy of these dark patterns throughout the gaming industry.

The main contributions of this project were as follows:

- Developed a prototype game to educate both players and game designers on dark game design patterns
- Developed a prototype, procedurally generated, adventure video game
- Developed a prototype video game that includes configurable dark design patterns
- Conducted an experiment to determine if the game familiarises users with dark design

Author's Declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Taught Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, this work is my own work. Work done in collaboration with, or with the assistance of others, is indicated as such. I have identified all material in this dissertation which is not my own work through appropriate referencing and acknowledgement. Where I have quoted or otherwise incorporated material which is the work of others, I have included the source in the references. Any views expressed in the dissertation, other than referenced material, are those of the author.

SIGNED: [illegible] .... DATE: 27/09/21
(Signature of student)

Acknowledgments

This project fits within the scope of ethics application 0028, as reviewed by my supervisor Peter Bennett.

COVID19 Statement

Although I experienced some personal effects due to the COVID19 pandemic, such as impacts to motivation and engagement, the main issues stemmed from limitations to the project's evaluation. This project's evaluation was conducted remotely, which allowed recruitment of participants from around the world. However, it prohibited a controlled testing environment. Ideally, I would have preferred gathering participants and conducting experiments in person. By doing this, common questions could be clarified with the group as they appeared. Drawing from personal professional experience, software issues and feedback often go unreported. As a result, in-person testing may have revealed other faults in *Under Grove* that were not relayed over digital communication. Because the testing was remote, players were able to complete tasks at their leisure, which required me to keep track of tasks throughout the day. This aspect of testing limited which software changes I could implement by the following day. Participants who finished their tasks earlier were more likely to see their requested changes. In an in-person, controlled environment, everyone would complete tasks at the same time, and all feedback would be processed simultaneously. Overall, in-person testing would have created more robust and reliable data for this project's evaluation.

Contents

List of Figures

List of Tables

Listings

1. Introduction

1.1 Prevalence of Dark Game Design Patterns

As the video game industry continues to rapidly grow, ethical video game design becomes increasingly important. The video game industry in recent years has become an all-pervasive source of media. As a whole, the industry is projected to earn over $175 billion by the end of 2021 and projected to have 2.9 billion video game players [1]. In order to guide game design for this large audience, emerging philosophies for game design center on the player. These player-centered game experiences are ones that place the player's wants and desires from a video game at the forefront of a game's design and development [2]. From this definition, players often expect games to be fun [2, 3]. Based on these philosophies, the 2.9 billion players would always have an overall enjoyable gaming experience.

However, competing design patterns can directly oppose player-centered game design philosophies. These design choices, which intentionally manipulate the player for the game creator's benefit, have been termed dark game design patterns [4]. In general, the player is at risk of losing or spending something when exposed to these design patterns. These patterns can be detrimental to the player's experience and can be ethically questionable. As a result, players may spend unexpected time playing the game, feel required to spend unknown amounts of money, or be manipulated into having their friends join the game. These dark design patterns are prevalent throughout the gaming industry, and can be found in entire groups or types of video games [4, 5].

Solutions to combat the prevalence of dark game design patterns are needed in order to protect players from potentially negative effects. Regulation of video games typically target controversial video game use, which often results from the use of dark game design patterns. For example, the use of gambling-like loot crates, which are in-game containers that provide random rewards in exchange for real-life currency, have been recommended for government regulation [6, 7]. Other dark patterns that affect the amount of time spent playing a video game could be correlated with addictive or problematic video game playing [5]. The actual addictive nature of video games has been debated for decades, but, as of 2013, the DSM-5 recognises an internet gaming disorder diagnosis [8]. To combat problematic video game use, some countries have enacted policies that restrict how often people can play games [9]. These regulatory practices aim to protect players, however, additional proactive solutions could prevent the very existence of these design patterns at all.

1.2 Proactive Solution with *Under Grove*

This project offers a proactive solution to the prevalence of dark game design patterns through the use of a new educational video game prototype. This prototype is titled *Under Grove* and is a top-down, two-dimensional, adventure game. The game uses styling that is reminiscent of older games, such as *Final Fantasy* or *The Legend of Zelda*. The gameplay of *Under Grove* involves two different modes: Play Mode and Design Mode.

In Play Mode, players can explore a forest, beat enemies using real-time combat, and try to get the highest score possible. The forest is a labyrinthine array of procedurally generated rooms that is built at runtime using a custom algorithm. Each room has contents which are also created at runtime. While the player explores this forest, they must collect four special items coined *SpiriTokens* and present them to a mystical entity called the *SpiriTree*. Doing so allows the player to 'ascend' from the current level to the next and restart the collection process. In its current state, Play Mode allows the player to continue this cycle infinitely unless their character is defeated and loses all health.

In Design Mode, players can act as a game designer and configure settings that affect the behaviour and generation of each level. However, players are playing the role of a 'dark' designer because each setting is representative of a dark game design pattern feature. These settings are referred to as 'dark design elements' because multiple settings may not represent exactly one pattern. The Design Mode allows players to test these settings and learn how they affect Play Mode. Players also have the ability to copy an encoded string of their settings and share it with others to simulate a social environment.

1.3 Project Aims

In this project, I explore *Under Grove* as an educational video game with the potential to combat the prevalence of dark game design patterns. Educational video games are considered a form of edutainment, which utilises media to help teach material [10]. The aim of this video game is to educate two different audiences: game players and game designers. For game players, the goal is to improve recognition of certain design patterns. This knowledge can empower players with the choice of engaging with games that feature these patterns. For game designers, the goal of this game is to improve familiarisation with these patterns and their effects on players. With this knowledge, they can be better equipped to avoid these patterns in development.

The primary contributions of this project are the following:

- Developed a procedurally generated game
- Created configurable game settings to allow experimentation of dark design elements
- Created in-game educational material on dark game design patterns
- Gathered user feedback on both playing and designing levels
- Performed a longitudinal experiment to determine the game's teaching capabilities
- Collected player-created settings to determine trends given certain level-creation prompts

2. Background and Context

2.1 Making Games for the Player

Considering players are ultimately the consumers of video games, games are primarily made for the player. Such a product would ideally be developed based on the player's wants and needs. One game design philosophy, player-centered design, incorporates this sentiment by having game creators make games with the player's experience at the forefront [11, 12]. In general, player-centered design involves including the players at different development stages to make sure the game fits the players' expectations [11, 13]. This design philosophy exists within the broader human-centered design philosophy [12], and software that fails to be user-centered can produce inconvenient or even dangerous software [14].

However, limitations of a human-centered design approach have fostered the creation of alternative methods. In activity-centered design, the activity the user performs is considered the focus over the player. This philosophy attempts to prevent over-specification of software for certain user groups because it can hinder usability for others [15]. In player-decentered design, another game design philosophy, the player is removed as a central aspect of the game. This philosophy combats potential artistic limitations placed on a designer by the player-centered philosophy and allows the creation of more experimental games [16]. While these alternative design philosophies are not player-centered, they can still bring enjoyment to the player.

Rather than the chosen design philosophy, specific facets of a game instead determine a player's enjoyment. These facets can be grouped under a term called *playability* [13, 16, 12, 17]. This playability metric, similar to usability, involves a relationship between a game and its players, which primarily focuses on how enjoyable the game is [17]. However, playability is subjective and can vary between different people [17, 13].

Regardless of the subjective nature of playability, studies have attempted to demystify exactly what makes a game playable. Playability is understood to affect different properties of a person's gameplay experience. For example, one component of playability is satisfaction, which is derived from the pleasure experienced when playing a game. Another component is motivation, which comprises the driving forces that cause users to take in-game actions [13]. Playability heuristics have been created specifically to ensure games are playable [18]. These heuristics focus on different aspects of a game, like gameplay, story, mechanics, and usability. Some examples include prioritising the player's enjoyment, making challenges positive experiences, and providing feedback for player actions. Based on these definitions, playability directly contributes to a player's gaming experience.

The varying playability definitions can unexpectedly allow potentially harmful mechanics to exist within playable games. Playability is not solely defined by player enjoyment and sometimes relies on motivation and reward systems. In malicious circumstances, these facets can warp the gaming experience and manipulate players for the creator's benefit. Such game design can be classified as a dark game design pattern [4, 19]. Dark design patterns have been defined across software platforms and generally indicate specific mechanics and features with a negative impact on the user. However, the term is used differently across different fields [19]. For the purposes of this paper, the use of 'dark design pattern' will refer to the following game-specific definition:

> A dark game design pattern is a pattern used intentionally by a game creator to cause negative experiences for players which are against their best interests and likely to happen without their consent [4].

Dark game design patterns are classified into four different categories. This classification is subjective, but consistently requires that creators intentionally and non-consensually use these different patterns to create negative experiences. Temporal dark patterns are aspects of a game that consume an unexpected amount of the player's time. Monetary dark patterns are mechanics or features of a game that require the player to spend unexpected amounts of money. Social capital dark patterns are features and mechanics of a game that can harm a person's social reputation [4]. Psychological patterns use psychological-based strategies to encourage players to continuing playing a game even if they no longer want to [5].

As discussed in Sec. 1.1, effects on time and money found in these dark game design patterns can negatively affect players. However, dark game design patterns have been criticised as being subjective classifications without a clear ethical argument. The use of a mechanic or feature is not necessarily inherently 'dark' or unethical [20]. The following sections will focus on temporal and monetary dark patterns and how they are implemented. Additionally, normative ethical arguments on these patterns are presented to conclude how their use can be unethical.

2.2 I Have to Complete How Many Quests? Time Dilation in Dark Design Patterns

As described before, temporal dark design patterns involve game features or mechanics that are intentionally designed to make a player spend unexpected amounts of time playing a game. Notably, this definition includes features that require both more and less time than expected [4]. These design patterns can be considered unethical based on consequentialism and virtue ethics.

Temporal dark design patterns that require more time than expected are potentially linked with addiction and problematic video game playing. Whether video games are addicting has been a heavily debated topic. In the DSM-5, however, an internet gaming disorder became a recognised diagnosis [8, 21]. One theory posits video game mechanics or genres could contribute to this problematic video game playing [22, 23, 24]. For example, massive multiplayer online and first-person shooter games have been considered more addictive than other genres [24], which may suggest some aspect of these games is addictive. The main contributor, however. may not be due to particular features, but instead the amount of time spent playing the game [25]. As such, any temporal dark design patterns that inflate the amount of time players play a game could be directly encouraging problematic game consumption.

Design patterns that force the player to play the game less than expected also qualify as temporal dark patterns. Some instances of these patterns can cause the player to play the game more frequently and increase the likelihood of the game becoming a habit [5]. These habit-forming patterns, even though they force the player to spend less time playing the game, could lead to increased, prolonged playing and therefore contribute to problematic video game playing.

Temporal dark patterns used non-consensually raise ethical concerns. Manipulating the time players spend playing a game can pose an increased risk for problematic video game playing. In this case, temporal dark design patterns must be unethical considering consequentialism. The experience of time loss, or the experience of losing track of time, can be perceived either positively or negatively by players [26]. In both instances, inorganically changing the amount of time a game requires, in either direction, will harm some subset of player experiences. With consideration to virtue ethics, these patterns are unethical due to their negative impact on some subsets of players.

Some examples of temporal dark patterns include grinding, playing by appointment, and advertisements. Grinding can be defined as tedious actions repeated by the player and can be used to extend the game's duration [4, 5]. Grinding can be considered a temporal dark pattern because creators are requiring the player to repeatedly perform actions and expend unexpected time to progress in the game [4]. Playing by appointment is defined as a mechanic that requires the player to play the game at certain times and may punish players if they fail to do so [4, 5]. This mechanic is considered a temporal dark pattern because players are required to play at times specified by creators, which can result in the player spending more or less time playing the game. Advertisements can be considered a temporal dark pattern if they require the player to wait a certain amount of time before playing a game [5], which may unexpectedly extend players' gameplay duration.

2.3 Didn't I Already Buy the Game? Money Expenditures in Dark Design Patterns

Monetary dark design patterns involve features and mechanics in a game that can encourage people to spend money. Unlike temporal time patterns, spending less money than expected is not considered a monetary dark pattern [4]. However, like temporal dark patterns, these can be considered unethical based on consequentialism and virtue ethics.

Monetary dark patterns can be seen in a variety of games from both independent and AAA developers. One revenue model for the gaming industry comes from microtransactions, which are virtual items players can purchase with real-life currency. Once purchased, these items can not be converted to real-life currency. Although these are primarily used in free-to-play games, they can be seen in all areas of the gaming industry [27].

Monetary dark patterns often include these microtransactions. Some example patterns are loot boxes, pay to skip, and pay to win. Loot boxes are a type of microtransaction that players can purchase. They contain a random prize, with more valuable prizes appearing at lower frequencies [5]. These are considered monetary dark patterns primarily because the randomised rewards encourage players to purchase until they get a desired item [5]. In fact, loot boxes are notoriously compared to gambling [28, 29, 30, 24, 9, 23]. However, technically, loot boxes elude legal definitions of gambling because the contents cannot be converted into real-life currency. The purchased items, therefore, hold no monetary value [31].

Monetary dark patterns can also be used to give players an advantage against a game's challenge. Pay to skip is a type of microtransaction where players can spend real-life money in order to bypass an unwanted part of the game. Game creators can encourage this spending by gradually increasing the difficulty or tediousness of a game. Players are thereby encouraged to spend money to avoid such situations, like grinding [4]. Pay to win is a similar type of microtransaction; however, in this scenario players are purchasing advantages over other players in a multiplayer environment [4]. Pay to win can be combined with other dark patterns like depreciation, where previously purchased items gradually lose their effectiveness. These combinations encourage players to continuously purchase better items to maintain a competitive edge [5]. Pay to skip and pay to win are considered monetary dark patterns not only because the player is spending money, but also because the game is inorganically encouraging these purchases. These patterns are unethical based on virtue ethics because they deceive the player into spending money.

One shortsighted, motivating factor to implement monetary dark patterns involves a psychological dark pattern called invested or endowed value [5]. In this pattern, because players have already invested in a game, they are less likely to stop playing that game. In the long term, this pattern leads to larger temporal and monetary investments from players, which ultimately benefit the game's creators. However, contrary evidence in mobile games may suggest a fault in this strategy. In-game purchases do affect how long players continue to play a game. In fact, monetary investment in a game may lead to an initial period where users are more likely to continue playing the game. However, in the long term, players who made in-game purchases were less likely to continue playing [32]. Thus, the use of monetary dark patterns is not only harmful to players but potentially to their creators as well.

2.4 Defense Against the Dark Design Patterns

2.4.1 Regulations for Dark Design Patterns

The potential negative impacts of dark design patterns have prompted attempts to limit their influence and presence. One strategy to reduce dark game design patterns is regulation. Software is regulated at different levels, including government regulation and industry-based regulation. For instance, the Entertainment Software Association (ESA) is a regulatory body for video games in the United States [31], and Pan European Game Information (PEGI) is a regulatory body for video games in Europe [21]. The different attempts at regulation in the video game industry, however, have not wholly addressed dark design patterns, at least in the United States [31]. Generally, these regulatory bodies in Western society primarily focus on determining how age appropriate a game is [21].

In recent years, governments have begun an attempt to mitigate the use of dark design patterns across all software. In the United States, recent legislation is encouraging the transparency of influential design patterns and forbidding manipulative methods to obtain privacy information [19]. Transparency of influential software interfaces is also included in the right to explanation mandated by the European Union's General Data Protection Regulation [33]. One example that is regulated in both the EU and the US is the tracking of cookies on websites, which requires users to actively consent before a website can record personal data [34]. However, even with these regulations, companies have utilised dark design patterns to manipulate users into consenting [33, 34, 35]. In order to better regulate dark design patterns, regulation should also be mandated for the actual user interfaces [34].

Calls for dark game design pattern regulation have also started to appear. Currently, the World Health Organization and DSM-5 recognise a behavioural addiction called internet gaming disorder [36, 8, 21]. As a result, game companies have started including warning messages regarding video game addiction. Governments have started regulating the amount of time minors can play video games [9, 21]. Countries have also required video game service providers to prevent player connections during certain hours of the day [21]. The loot box monetary dark pattern has been a particular focus of criticism and legal debate [31], and calls have been made to regulate their use [6, 7]. In China and Belgium, government regulation for loot boxes has been enacted, ranging from making information transparent in the former to fully banning their use in the latter [7].

Although regulations seek to protect software users, they are not a complete solution. Considering some of the failings of regulating dark design patterns in other software, regulation may need to include specific mandates for the user interface. China's regulation on loot boxes requires disclosure of reward probabilities, but companies tend to minimally meet these mandates and display required information 'non-prominently' [6]. Like website cookie regulations, the end result is still manipulative of users. Outside of the government, the viability of industry-regulated bodies is not guaranteed either. In the United States, the ESA operates its own rating system through the Entertainment Software Rating Board, which means no independent third-party is responsible for deciding a game's rating. This relationship could be considered a conflict of interest [37] and weakens trustworthiness for self-imposed regulatory requirements.

2.4.2 Education to Combat Dark Design

Outside of regulation, education is another strategy to combat the prevalence of dark game design patterns [38]. In a previous study involving dark design patterns in websites, an educational tool improved participant awareness of manipulative design patterns [39]. With the ability to recognise dark design patterns, users become manipulation literate, meaning users can choose to engage in the manipulation. By exercising this choice, users can willingly provide consent [40, 39]. Educating players in dark game design patterns could similarly provide a means to give players choice. Consequently, those who choose not to engage can even impact the profitability of such games and motivate the gaming industry away from these unethical practices.

An educational game has the potential to impact the behaviour of game designers in addition to players. Software designers are aware of the existence of dark design patterns; however, they cannot always readily identify instances they have used them. Additionally, what knowledge they do have, is normally self-taught [38]. One method to fill this educational gap is by learning the theory of these patterns through traditional learning methods. Alternatively, designers could sharpen their knowledge using edutainment. In existing video game design courses, learning activities help students learn design theory more easily. Situations where students can be creative also stimulate additional drive to learn given material [41]. A game that is both interactive and requires creativity through experimentation with dark game design patterns could be an effective learning tool for designers. Increasing the knowledge of these patterns amongst designers could discourage their use throughout the gaming industry.

3. System Design

3.1 Dungeon Peddlers: The Origins of Under Grove

One strategy to educate people about dark game design patterns involves the ability to allow users to be creative and experiment with them. For this strategy, I initially pursued creating a level-editor game, titled *Dungeon Peddlers* that would allow players to take the role of a game designer. In level-editor games, users can create game levels for either themselves or others to play. This level-editor idea was inspired by *Super Mario Maker*, which is a game published by Nintendo that allows players to create their own *Super Mario* themed levels. However, instead of a 2D platformer, this game would use a top-down, 2D style. I chose this style because of a longtime love of *The Legend of Zelda* titles. These games feature gameplay where users navigate puzzle-based levels called *dungeons*. Dungeons are essentially a sequence of rooms players must navigate in order to reach some goal. Dungeons typically have enemies to defeat, puzzles to solve, and an over-arching theme.

To further bolster experimentation and mimic the gaming industry, *Dungeon Peddlers* would feature an in-game economy based around selling player-made dungeons. Designers would invest in their dungeons by spending in-game money to add elements to their levels. Designers would also be able to incorporate dark design elements, such as loot crates or advertisements, during this creation process. To recoup these investments and generate income, designers would then need players to pay an admission fee to play their levels. Players, in turn, are motivated to complete levels by receiving a reward upon level completion. These interactions would be based around an autonomous in-game bank that would support this monetary exchange.

A secondary objective for this game was to determine if the problematic use of dark game design patterns could be prevented. By logging which elements are used in a designer's level, I planned to calculate a dungeon's relative level of 'darkness'. Collecting this data would potentially allow identification of pattern trends based on popularity and profitability. To experiment with prevention, price-based restrictions and rewards would be applied to the use of dark design elements. These changes would, I theorised, determine methods to nudge designer's behaviour and reduce the prevalence of dark game design patterns.

Based on these design decisions, I created a low-fidelity digital prototype (Fig. 3.1) of the game's user interface and conducted user interviews for feedback. Low-fidelity digital prototypes can achieve similar feedback compared to traditional paper prototypes unless the digital prototype prevents demonstrating accurate behaviour [42]. Because this prototype simply involved the navigation of different game screens, the choice to use a digital prototype wireframe was sufficient.

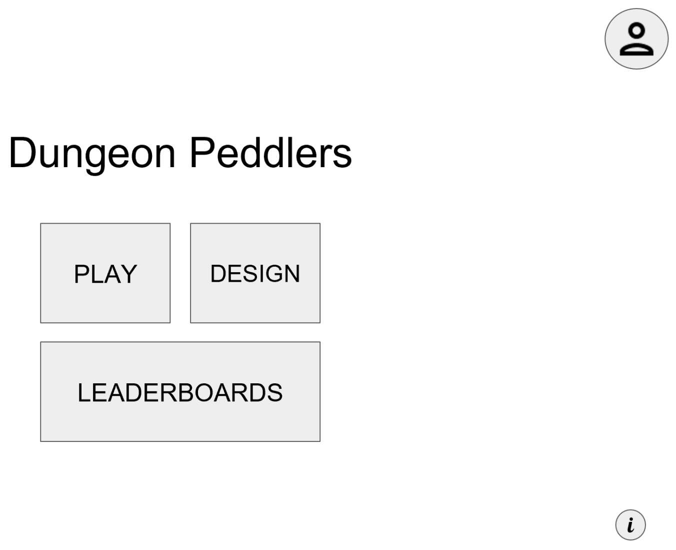


Figure 3.1: This image is the initial Home Screen wireframe for *Dungeon Peddlers* and represents the style of the wireframe presented to interviewed participants.

Seven participants were interviewed with ages ranging from 24 to 30. The participants varied in the frequency they played video games, but all participants were familiar with software development and design. The results of these interviews influenced *Under Grove*'s design and will be discussed where relevant.

After conducting these usability tests, the feasibility of this project was tested through an opportunistic evaluation of the level editor mechanics. Opportunistic evaluations are typically exploratory, and they allow designers get broad feedback, which can ultimately determine if an idea is viable [43]. One goal of this evaluation was to determine which types of elements players would add to their levels. Another goal was to understand which dark game design patterns people would intuitively incorporate into their dungeons.

Interviews were used for this evaluation, and participants completed a survey and a dungeon-designing exercise (Fig. 3.2). Each interview was completed individually for a sample size of $n = 4$. Participants ranged in age from 22 to 44 and had a variety of game-playing experience; two participants reported never playing video games. For the dungeon-designing task, participants were provided materials (a gridded sheet of paper and drawing utensils) and instructed to produce a dungeon that could generate the most in-game money. A dungeon was defined as meeting the following criteria:

1. A dungeon must contain at least one room.
2. A dungeon must have a starting location and a goal ending location for the player.

Outside of these two rules, participants were encouraged to be creative in their level-design and given full authority of their creations. In the event people were unsure of how to build a dungeon, a small number of prefabricated components were present (Fig. 3.3).

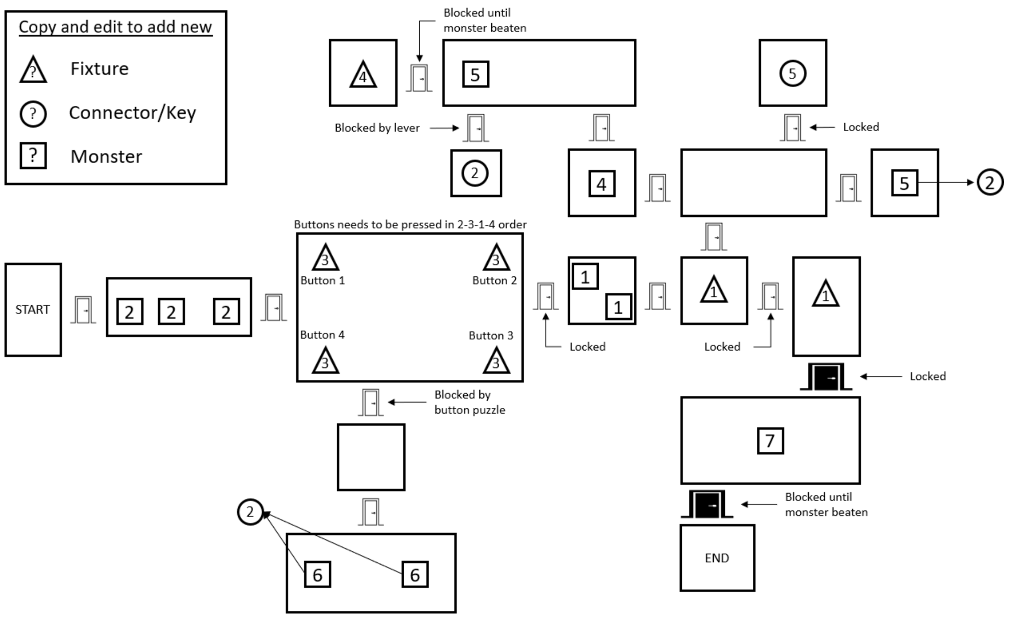


Figure 3.2: An example blueprint for a *Dungeon Peddlers* user-created level. Fixtures represent fixed objects in a room. Connectors represent keys.

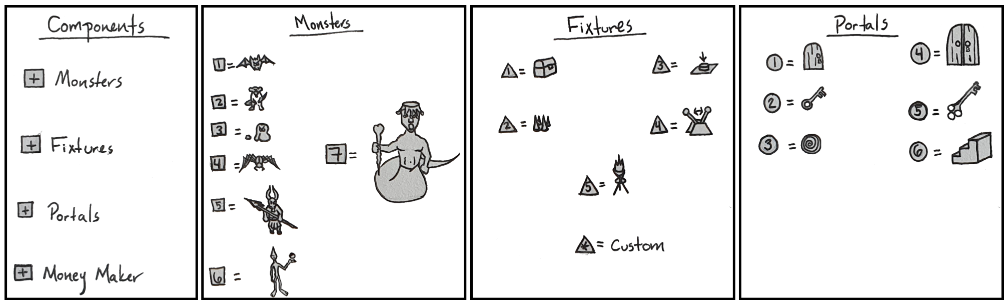


Figure 3.3: When designing levels, participants were given this set of prefabricated items in the event they needed inspiration for creating dungeons. These items were meant to replicate how a level-editor interface might operate.

The creations of all participants varied, with people inventing unique methods to generate money. One person created mechanics where a player could pay money to open chests with randomised rewards. This mechanism is reminiscent of the loot box mechanic. Another person created an entire market within their game to sell collected goods for increased abilities. This idea incorporated pay-to-skip mechanics to decrease the difficulty of the level.

The evaluation was successful in setting the direction for the game, but ultimately revealed several flaws with the level-editor premise. Overall, all players were able to create levels using the given materials, but most (75%) struggled to incorporate methods to generate in-game money without help from me. Without someone to moderate the process, these participants would have failed to accomplish the given task. Additionally, all participants took more than twenty minutes to quickly sketch a level. Creating a dungeons from scratch appeared as a complex task for players.

After reflecting on this outcome, I ultimately decided that several features of *Dungeon Peddlers* were not feasible for this project. Creating a level is a skill that would take a player time to learn. Furthermore, adding dark design elements to those levels is another skill the player would need to learn. In an experiment scenario, even if these ideas are well-explained to a participant, the task may prove too complex under certain time constraints. The secondary objective to analyse trends in dark game design patterns would require participants to create enough levels to build a sufficient data set, which could take an extended period of time. Due to these factors and the limited duration of this project, the level-editor and in-game economy features were removed following this evaluation.

Without a level-editor or in-game economy, the core ideas of *Dungeon Peddlers* were no more, and a new game emerged. In this new game, level generation and design would fall on me as the game creator. Removing this responsibility from the player allows players to better focus on the educational aspects. Because creativity can enhance learning game design, I maintained the option to allow players to experiment with dark design elements. Instead of building levels, however, players would have the option to configure the severity of dark design elements. With this change, the goal of the project shifted slightly. *Dungeon Peddlers* was intended to fulfill two roles: educating players on dark design patterns and determining methods to prevent their use in design. *Under Grove*, however, will solely focus on educating its users, both players and designers, on dark game design patterns.

Under Grove, like *Dungeon Peddlers*, would continue to be a top-down, 2D game. The new game would also maintain the original dungeon format because this idea received positive feedback during the opportunistic evaluation. Consequently, levels would be a sequence of rooms that a player could explore. They would begin in a starting room and need to navigate to an end room. Collectively, the rooms would combine like a blueprint with logical connections and no overlapping segments.

3.2 Generating Levels in Under Grove

Now that I was responsible for the level creation, I needed to determine the best strategy for creating the dungeons a player would navigate. Several options exist for creating video game content [44]. When a game runs, it produces content to the player based on the game's pre-programmed rules. This content can be created prior to the game's runtime through manual creation techniques or at runtime using automated techniques.

In manual generation, assets and components are created and placed by a person, such as a game designer. Manually created content is static. However, a game with manually created content is not necessarily the same for every player and can vary based on the player's interaction. For example, a manually generated game could have different story lines or dungeons. In *The Elder Scrolls V: Skyrim*, the player can join different factions in an in-game war. The player experiences different content depending on this decision. However, the content for each decision is always present within the game. Notably, manually created content is repeatable without relying on chance.

Automated content generation allows dynamic content creation and exists in two flavors. In random generation, a designer will create prefabricated assets, such as sprites, rooms, characters, etc. When the game runs, these assets are picked at random [44]. One example roguelike game, *Hades* features this type of content. The main character navigates from room to room in an attempt to escape the Hades underworld. Each time the character attempts an escape, he must again navigate out of Hades, but usually faces a different sequence of rooms. This sequence is generated at random on each playthrough from a set of pre-existing rooms.

In procedural generation, content is generated from building blocks using algorithmic rules. Procedural generation allows the game to create content own its own [44]. One example of procedurally generated content is the landscape of *Minecraft*. The cubes that form the world are not randomly placed, but follow a set of rules to create mountains, tunnels, and forests. Procedural generation does not require every aspect of a game be created at runtime. The building blocks and generation algorithms can incorporate manually and randomly generated content. In the *Minecraft* example, each building block must at some level be defined as a cube. These cubes also have pre-defined properties, such as color and behaviour.

Under Grove features a combination of manually, randomly, and procedurally generated content. All user interfaces and menus are manually generated to maintain consistent navigation outside of game play. Each dungeon of *Under Grove* is procedurally generated to create novel dungeons for each attempt at completing a level. Within each dungeon, enemies, and collectibles are randomly generated at the time of dungeon creation. Additionally, in-game advertisements and in-game store content is generated randomly each time the player encounters them. Dark design elements can also be randomly configured at the game's runtime.

3.3 The Fantasy of *Under Grove*

To create a more immersive environment for the gameplay, I added a theme and some basic story elements. I drew from my favourite genres, such as fantasy, science fiction, and horror. To help focus this theme, I asked fellow students which genres would be the most enjoyable. Overall, participants felt the gameplay itself was more important than the theme. With this feedback, I took artistic liberties and crafted a fantastical forest theme. This idea stemmed from character concept art I drew based on an animated tree stump named Twiggy Stumps (Fig. 3.4). Twiggy can navigate this fantastical forest; although, importantly, his movement is rigid to imitate the inflexibility of a tree stump. He has the magical ability to manipulate his fireflies to aid in his exploration. The stick he holds acts as a sword-like weapon that can be used to vanquish foes.

After conceptualising the main character, I was able to develop the other thematic elements with Twiggy at the core. Twiggy is animated by a tree deity of a dark forest, the SpiriTree. The SpriTree requires Twiggy to continuously collect SpiriTokens as special offerings. Each token would represent an element of a tree: a golden strand of acorns, a silver scroll of birch bark, a potion of deadly alder syrup, and a powder of invigorating sassafras. Collecting these SpiriTokens is the primary objective of the player. After collecting each SpiriToken and offering it to the SpiriTree, Twiggy can 'ascend' and the player can progress to the next level. In the subsequent levels, the cycle continues

Different enemies in the form of monsters can be found throughout each level. A subset of these monsters draw inspiration from parasitic organisms, symbolising a threat of corruption

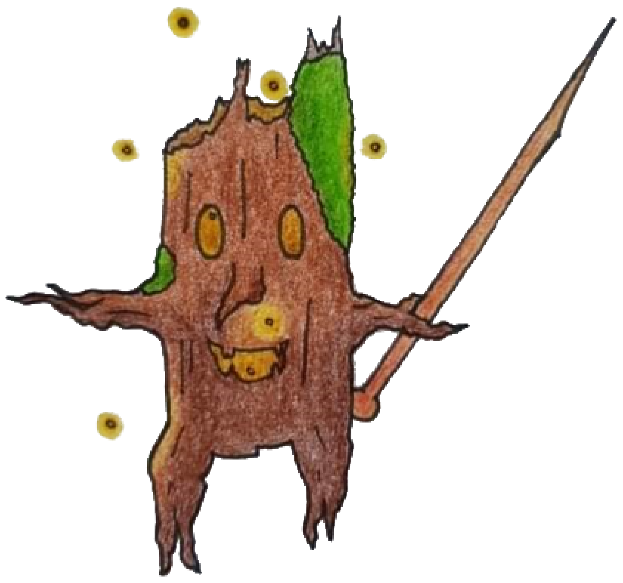


Figure 3.4: This image is the original concept art for Twiggy Stumps. His face is inspired by jack-o-lanterns that are illuminated with fireflies.

within the forest. One monster is based on sandalwood, which is a parasitic plant native to India [45]. Another monster is based on parasitic red algae, which can have parasitic relationships with other algae species [46]. The other subset of monsters, called Despirits, are monsters in direct opposition to the SpiriTree. They are stylistically similar to the SpiriTree. However, they serve as a symbol of growing corruption within the forest and are beings that failed to ascend through the SpiriTree.

Twiggy will struggle to navigate the labyrinthine forest not just because of the somewhat disorientating effect of the level, but also because of the atmosphere, which would be intentionally dark. Twiggy can summon additional fireflies to help him during his adventure. By summoning fireflies he can help light different areas of the forest and help himself identify when he has reached familiar ground.

3.4 Game Play of *Under Grove*

In most games, a user must provide input in order to play a game. When creating the game controls for *Under Grove*, I chose input that is consistent with the input of other games. One strategy was to draw from my past experience. I also referred to Unity forums and additional resources to learn what other people felt were common controls for a keyboard. Player controls can also be determined based on the dexterity and distance fingers need to move to reach certain controls [47]. Based on these sources, *Under Grove* utilises the keyboard controls in Table 3.1.

Game Controls

Key	**Function**
W or **Up**	Move character upwards
A or **Left**	Move character to the left
S of **Down**	Move character downwards
D or **Right**	Move character to the right
Q	Summon or dismiss fireflies
E	Interact with various objects, such as chests and doors
Space	Swing weapon to attack enemies
Esc or **P**	Pause the game, get help, or quit to the home screen
Left Mouse	Click buttons in the games UI

Table 3.1: This table shows all game controls used when playing *Under Grove*

Based on this control scheme, the player can place their fingers on the W, A, S, and D keys (WASD) with the left hand and the mouse with the right hand. After this initial hand placement, the user will not need to move their hands during gameplay with the exception of two keys. The pause functionality requires the player to move one hand to press either the Esc or P key, which will stop the game and open a new menu. As this key interrupts the game, it was intentionally moved out of close range from either hand. However, this key still maintains some flow. If the player uses the left hand to pause the game, the right hand can remain on the mouse and be ready to to interact with the subsequent menu.

The character movement functionality may also result in deviations from the intended hand position. If players opt for the arrow keys rather than WASD, they will likely have their right hand on the the arrow keys and their left hand ready to operate the Q, E, and Space keys. To users who are unfamiliar with the WASD controls, the arrow keys may be more intuitive for movement since they are used for navigation in other applications. Additionally, implementing the use of both WASD and the arrows keys is trivial. Therefore, the arrow keys are included as an optional method for character movement. However, because the player may need to move their hands more often when using the arrow keys, the game instructions specify that it is not recommended for the best gameplay experience.

3.5 Navigating the User Interface

Outside of *Under Grove*'s gameplay, the player must interact with various menus and windows in other areas of the game. Throughout the user interface, some consistent themes were maintained. All buttons are given the same color scheme, animations, and sound effects. All buttons are brown, with a green outline. When a player hovers overs over a button with a mouse, the button wiggles to provide feedback to the user. When the player clicks a button, the button shrinks in size to imitate the effect of pressing the button. For each of these actions, a quick, small sound also plays to provide additional feedback.

Buttons throughout the game possess different shapes and sizes (Fig. 3.5), but these aspects are used to help signify the button's functionality. Smaller square buttons are used if a button's functionality can be symbolised using an icon. Buttons that close windows or screens are represented by square buttons with an *x* icon. A button used to copy information is designed as a square button with a clipboard icon. Circular buttons with *i* icons are used for the game's info buttons. Buttons on the home screen are large squares, a design choice that received positive feedback in wireframe interviews for *Dungeon Peddlers*. Either the word *play* or *design* is used to indicate different gameplay modes. All other buttons in the game are rectangular and use a word or phrase to describe their functionality. These buttons try to maintain a 500 x 150 pixel ratio where possible.

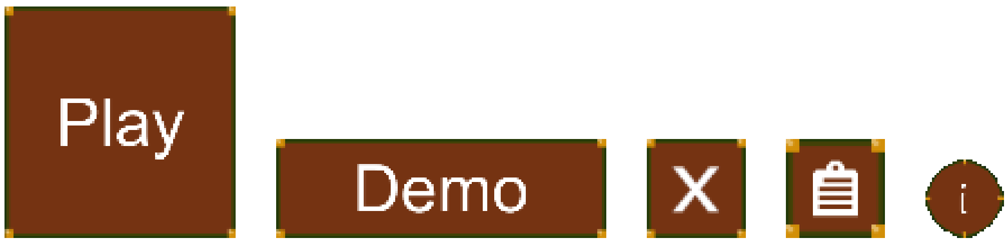


Figure 3.5: This image exhibits all the different buttons that appear in *Under Grove*.

All text throughout the game is presented using the same font and color to preserve consistency. The text used in the game features a white text with a surrounding yellow glow, which thematically connects to the fireflies used throughout the game. If possible, text is displayed on top of a background window to prevent background images from reducing readability. All of these backgrounds use a darker hue, such as dark greens and black (Fig. 3.6). The size of the font was not consistent throughout the game. However, in general, larger text sizes were used for titles and headings, while smaller text sizes were used for body text and notations.

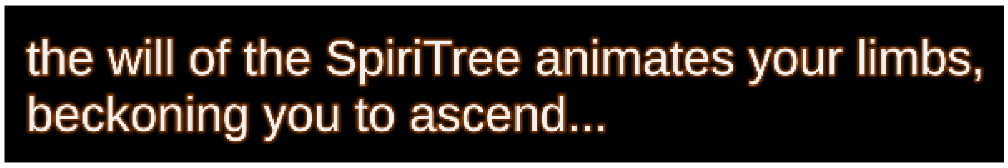

the will of the SpiriTree animates your limbs, beckoning you to ascend...

Figure 3.6: This is an example of the text used throughout the game. The darker background combined with the use of the text glow improve readability of the text.

The home screen is where the player starts their interaction with the game (Fig. 3.7). This screen is based on the home screen for *Dungeon Peddlers* (Fig. 3.1). On this screen, there are three buttons. The Play and Design buttons are the primary buttons on this screen. The third button on this page is an info button. When clicked, a popup window appears that provides a brief description of the project. Because this information requires clicking a button, it is optional information for player.

Play Mode

To bolster the intended creativity and exploration, *Under Grove* originally had little instruction or story. This method has been used in other games, such as those from the *Dark Souls* series. In these games, the player is given some initial story and instructions, but outside of this information, the player controls how much of the story they uncover through discovery. However, during initial, informal player feedback, several players mentioned not

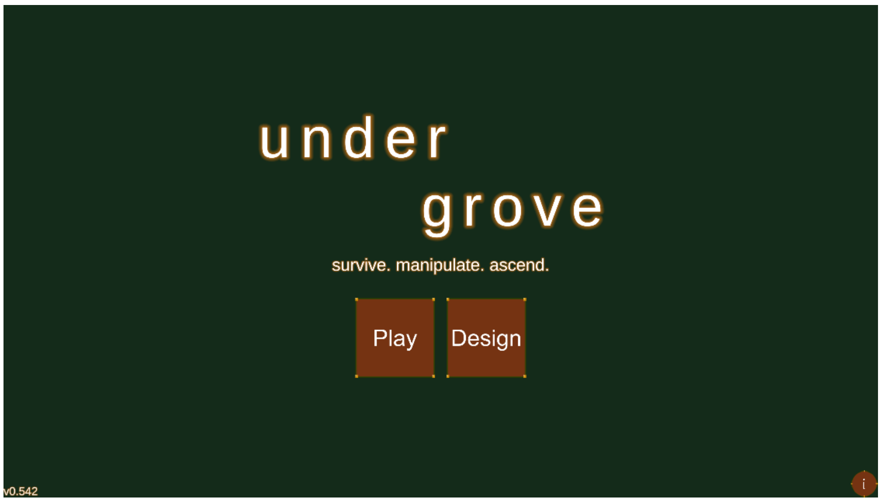


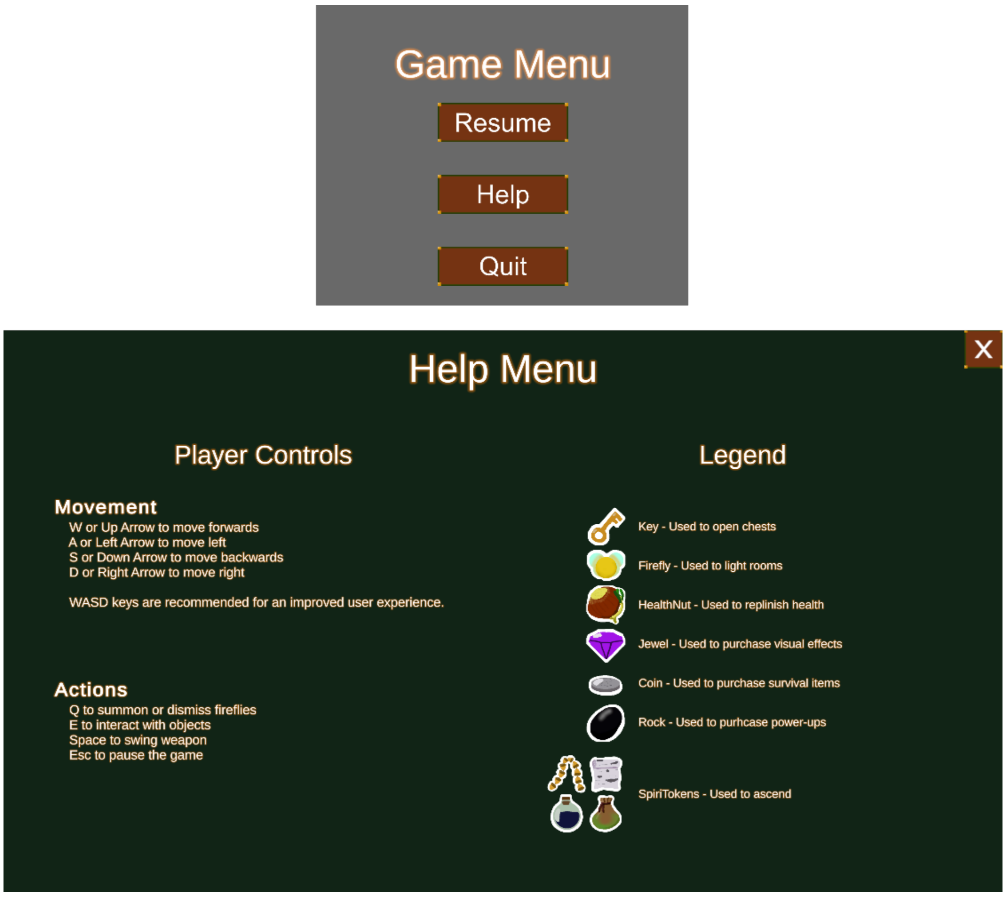
Figure 3.7: This image shows the prototype home screen for *Under Grove*. The Play and Design buttons are placed side-by-side to indicate both options are equally important. The info button has been placed in the bottom right to indicate lesser importance.

understanding how to play the game or wanting some story to motivate them into playing the game. Players went as far as saying they would only play the game given more instruction. I subsequently added an introductory story and instructions to fulfill these desires.

After clicking the Play button, players will be taken through a series of screens. The first screen is an introductory screen that briefly presents the character's background and provides the player's goal of collecting SpiriTokens. These story elements are still somewhat vague to encourage the player's imagination and exploration. Players are informed via a text-based instruction that they must press the E key to progress to the next screen. This action serves as a brief tutorial for object interaction within the game, which requires the same command. In the next screen, players are presented with the controls for the game. Players must click an Okay button to progress to the actual game. Using a mouse click is a more active movement than pressing the E key, which could be easily repeated from the previous command. This change discourages simply skipping the instructions.

Once playing the game, players will primarily interact with UI elements when pausing the game. When paused, all gameplay freezes, and the player is presented with a popup window (Fig. 3.8). Players can exit the pause menu and resume play, get help, or quit to the home screen. If players click the Help button, a subsequent popup appears. Here, players can get more info about the game. Based on informal player feedback, this window contains a game-item legend in addition to the player controls. This information is not present on the initial instructions to prevent spoiling this information for players who would rather discover this information through gameplay. For example, the function of SpiriTokens and the different currencies is meant to be discovered through gameplay, but is spoiled by this legend.

Players can keep track of their metrics and their character's status via UI elements on the head's up display (HUD). The HUD is an overlay where player's can see their current health, collectibles, collected SpiriTokens, credit balance, and score. The HUD went through several iterations before settling on the final design (Fig. 3.9). Originally, all information was



Help Menu

Player Controls

Movement
W or Up Arrow to move forwards
A or Left Arrow to move left
S or Down Arrow to move backwards
D or Right Arrow to move right

WASD keys are recommended for an improved user experience.

Actions
Q to summon or dismiss fireflies
E to interact with objects
Space to swing weapon
Esc to pause the game

Legend

Key - Used to open chests
Firefly - Used to light rooms
HealthNut - Used to replinish health
Jewel - Used to purchase visual effects
Coin - Used to purchase survival items
Rock - Used to purhcase power-ups
SpiriTokens - Used to ascend

Figure 3.8: This image presents the initial pause menu when a player enters a pause command. If clicking the Help button, a subsequent window with controls and a legend overlays the window.

present in the bottom left corner of the screen. However, the majority of this information was moved to the upper left corner. The health, collectibles, and SpiriTokens are important for the current level and can immediately affect player actions. The player credit balance and score information, however, persist into the Game Over screen. Therefore, these two groups were split visually.

In the top row of the HUD is the player's health, which is represented with HealthNut icons. This style is inspired by the HUD from *The Legend of Zelda* series. Below the health are icons with counters for keys, fireflies, and currencies, respectively. In the bottom row, a player can see the SpiriTokens they have currently collected. I could have provided a black silhouette to indicate which SpiriTokens the player still needs to collect. However, I wanted to maintain surprise for which SpiriTokens were needed, so this feature was not implemented. The credits are symbolic of a player's real-life currency, and players use it to make in-game purchases. The credits are always present, but are mainly utilised when dark design elements are enabled. The player score updates based on actions taken by the player, such as defeating enemies, collecting items, and advancing levels.

Under Grove also features an in-game store. Once a player completes a level, they are

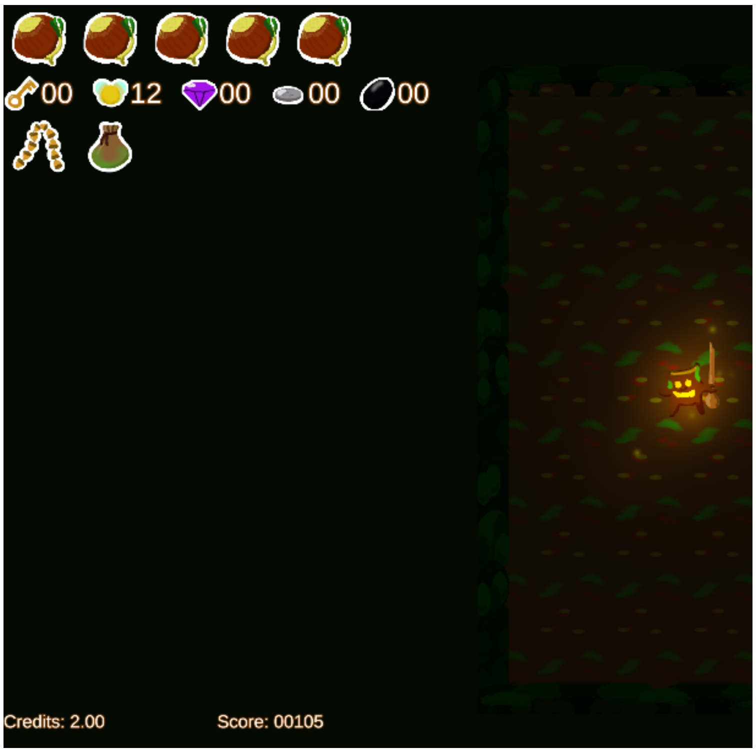


Figure 3.9: This image contains the HUD and all tracked metrics. These icons are outlined in white to improve visibility, and they each update dynamically based on the player's status.

presented with a new window for The Shoppe Ascension and can purchase items for Twiggy. These items are purchased using one of three currencies players collect by defeating enemies and opening chests. Jewels are used by players to purchase different firefly colours. Coins are used to purchase consumable items, such as fireflies and HealthNuts. Rocks can be used to improve player abilities, such as speed and attack range.

The store is organised so players can easily purchase desired items or proceed to the next level (Fig. 3.10). Each column within the window is labeled to provide context for players who have not already viewed the currencies' purpose in the pause menu's legend. Within each column players can find two randomised items, each with a descriptive name and icon to indicate its effect. When players have finished with the store they can click a close button to proceed to the next level. The close button is placed in the upper-right corner to maintain consistency with other close buttons.

Players can reference their current balance intuitively while navigating the store. The player's currency amounts are displayed at the top of the window. These currencies are arranged in the same order as the shop's columns to provide players with easier referencing. The player's balance is dynamic and updates with each purchase. If an item in the store at any point costs more than the player's current balance, the item's Buy button will disabled.

Players encounter the Game Over screen if Twiggy loses all health (Fig. A.1). When encountering this screen a sound is played to indicate the transition. Players can either click a Revive or Quit button. Beneath the Revive button, a player can see how many credits it

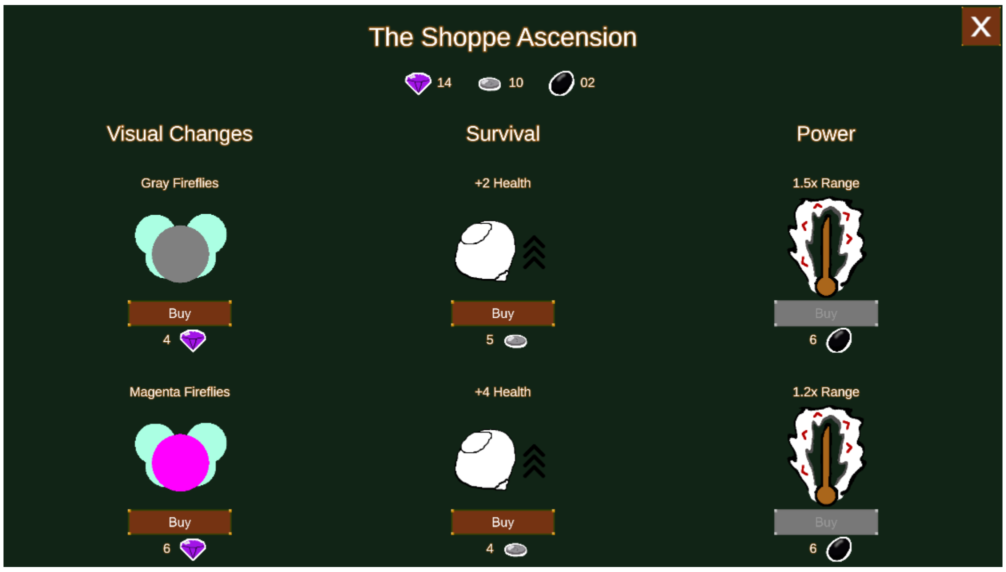


Figure 3.10: The in-game store is divided into three columns based on the purchase's in-game effect and required currency. Buy buttons are disabled if the player cannot afford an item.

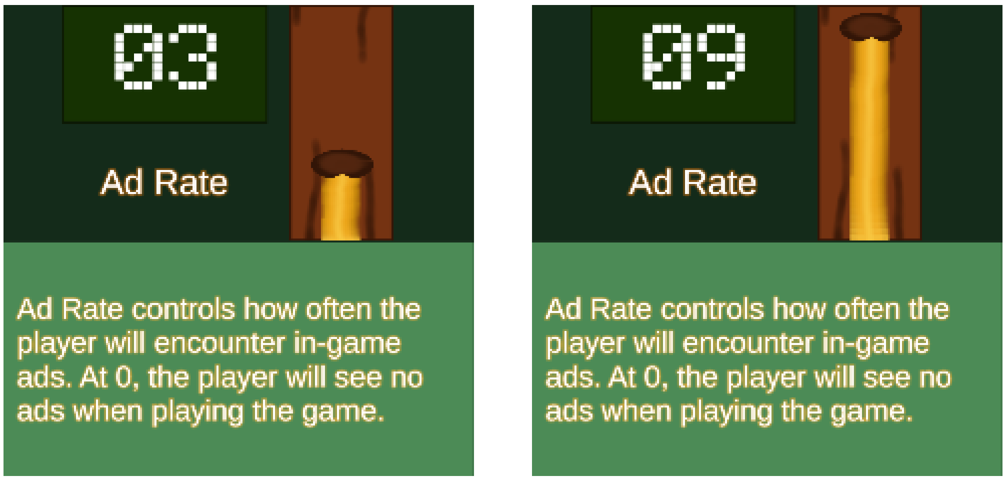
costs to revive their character. If clicked, the player's credits are updated and they begin a new level without losing their current credit balance and score. If players click the Quit button, they reset all player metrics and return to the Home screen. The Game Over screen also has the player's current score, number of completed levels, and current credit balance present for referencing.

Design Mode

If players click the Design button on the home screen, users enter the game's Design Mode. In this mode, players can add dark design elements to gameplay. This screen (Fig. A.2) is broken into two main sections, which represent both temporal and monetary dark game design patterns [4]. Settings are also grouped horizontally based on similar themes. For example, all settings in the first row involve in-game ads. Each dark design element is set using a slider (Fig. 3.11), which allows users to pick discrete values by clicking and dragging the mouse.

Informal player feedback shaped how information was presented in Design Mode. Players found the sliders alone did not provide enough context to elucidate what these settings controlled. To remedy this, I included an info button that opens a popup window and explains dark game design patterns, temporal dark patterns, and monetary dark patterns. I also added setting tooltips that describes the behaviour of each dark design element. This tooltip also describes relationships with other settings and default behaviour.

The Design Mode also features a Demo Mode to allow players to test their settings. Players can click the Demo button on the bottom of the screen and enter directly into Play Mode without the introduction or instruction screens. They can quickly exit Demo Mode using a special close button that appears in the HUD only when demoing.


03

Ad Rate

Ad Rate controls how often the player will encounter in-game ads. At 0, the player will see no ads when playing the game.

09

Ad Rate

Ad Rate controls how often the player will encounter in-game ads. At 0, the player will see no ads when playing the game.

Figure 3.11: Dark design elements are configured using sliders, which have a minimum value of 0 and a maximum value of 10. The current value of the slider and name of the setting are displayed to the left. The tooltips appear on mouse hover, and disappear when the mouse is no longer over the slider.

Players requested the ability to randomise and share settings during informal feedback. To accommodate this request, I added an Automate section. Here, users can input or copy strings that represent a configuration of settings. An Automate button is used to decode given strings (Fig. A.5). If a string is invalid, a warning icon appears next to the field, and the button does nothing. If no string is given, the button randomises all settings. The current settings display in a dynamically generated string, and a copy button can be used to copy it to the user's clipboard. This section allow users to share or save settings.

3.6 Dark Design Elements

All dark design elements found in Design Mode are based on temporal and monetary dark patterns. The temporal dark design elements were based on grinding [4] and advertisements [5]. Monetary dark design elements were based on pay-to-skip, pay-to-win, and gambling mechanics [4, 5].

I used repeatable tasks in *Under Grove* as the foundation for grinding mechanics. I incorporated a dark design element for each of the currencies that controls their relative value (Jewel Inflation, Coin Inflation, and Rock Inflation). Items in The Shoppe Ascension are purchased using in-game currency. To gather this currency, players must open chests and defeat monsters. As these settings increase, players must spend more time collecting currency in order to make purchases. I also incorporated a Search Difficulty dark design element to control the difficulty of finding SpiriTokens. Similarly, this setting increases the time players must spend to complete levels.

To simulate advertisements and pay-to-skip patterns, I created four advertisement settings. Ad Rate controls the chance at which ads appear when the players enters a new room. This frequency varies from 0% to 100%. Additionally, Ad Length controls how longs the player must wait before they can close the ad, ranging from zero to five seconds. Ad Free

Cost, a pay-to-skip pattern, allows the player to spend credits to remove ads for the current playthrough. Ad Purchase Cost, another pay-to-skip pattern, controls how many credits players spend on ads to get player bonuses.

The design of *Under Grove*'s ads is inspired by stereotypical mobile-game ads (Fig. 3.12). A randomised ad image is located in the center of the ad window (Table 3.2). These ads can be clicked to open a popup and purchase character bonuses. Originally, the ad image was static, but informal player feedback found purchasing behaviour counterintuitive. To improve user response, I added animations for when a player hovers over the image and when they click the image. After clicking the ad, a window appears, confirming the player wants to spend credits to purchase the ad. If players click the Yes button the purchase will take effect and credits will be deducted from the player's balance. After purchasing an ad, the confirmation window closes and the purchasing behaviour is disabled. If the player clicks the No button, the confirmation window simply closes.

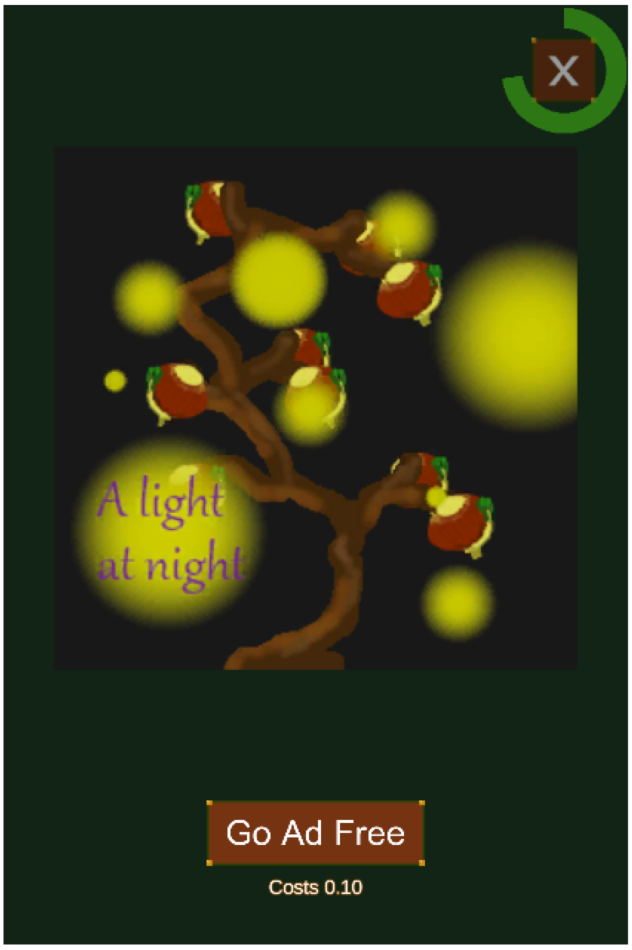


Figure 3.12: This image shows an example in-game ad from *Under Grove*.

In-game Ad Types

Ad Name	**Effect**
20-20-20 Currency	Player automatically receives 20 of each jewels, coins, and rocks
Spirit Fireflies	Player's firefly colour changes to cyan
Survival Package	Player receives max health and 20 fireflies

Table 3.2: This table provides descriptions for each of the available advertisements in *Under Grove*

In the upper-right corner, a special wait-to-close button is present (Fig. 3.13). At the bottom of the ad, players can click the Go Ad Free button to remove ads for the duration of the game. The Ad Free Cost is displayed to make the cost transparent. However, players showed confusion towards this text, initially believing it indicated the cost to purchase the ad. To address this problem, additional space was added between the ad and ad-free sections. To better indicate the ad-free cost, the cost could be incorporated into the button text in future versions.

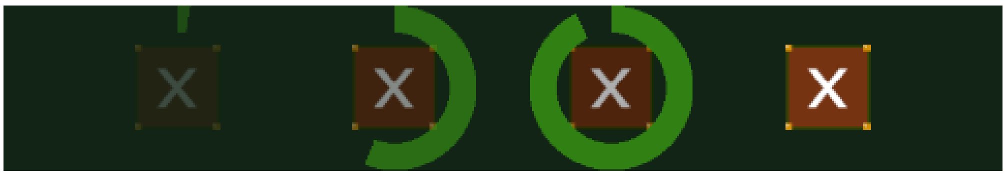


Figure 3.13: If Ad Length is configured, the close button is initially disabled. An annulus appears around the button in a clockwise matter to indicate the wait time before the button will become enabled. This image shows the progression of this animation from left to right.

Pay-to-skip dark design elements relate to temporal and psychological patterns within the game. In-game advertisements are one way to skip grinding. For example, purchasing the '20-20-20 Currency' ad would allow players to easily buy many items from The Shoppe Ascension. Another grinding activity in *Under Grove* is collecting SpiriTokens. The Ascension Cost dark design element allows players to bypass SpiriToken collection and immediately complete a level. If the player dies, the Revive Cost setting controls the cost to continue playing without the player's status resetting. Because players have invested time before death, they may be more likely to pay this fee rather than lose everything. This dark design element is driven by the invested or endowed value pattern [5].

The gambling pattern was achieved in the game through the use of *Under Grove*'s loot chests. Loot chests vary from regular chests in two ways. First, they cannot have SpiriTokens, which prevents needing to pay credits to complete a level. Second, the held collectible quantity is multiplied by a randomised number to increase its value relative to regular chests. However, by design, loot chests can occasionally generate an unexpectedly low quantity of items. The Loot Chest Rate dark design element controls the chance a chest will become a loot chest, and the Loot Chest Cost element controls how many credits are required to open the chest.

3.7 Artwork, Sound and Lighting

The artwork, sound, and lighting for the game are meant to create a stylistic and fantastical environment for the player. In order to achieve this mood, I followed a set of basic rules. All art should be a similar style. Sound should never be too loud. Sound effects should only be used to provide feedback to the player or create background music. Lighting should be used to indicate important features in the game, but should remain muted to establish a dark atmosphere. Finally, transitions between windows, screens, and rooms should be smooth to maintain player immersion.

In order to create a consistent art style throughout the game, I personally created all game art, drawing inspiration from other top-down, 2D games, primarily *The Legend of Zelda*. I maintain a similar perspective for all sprites (Fig. 3.14). Rooms maintain a consistent style by constructing them from the same core components: floors, walls, and doors (Fig. A.3). The SpiriTree room is coloured cyan to demarcate it from the others (Fig. A.4).

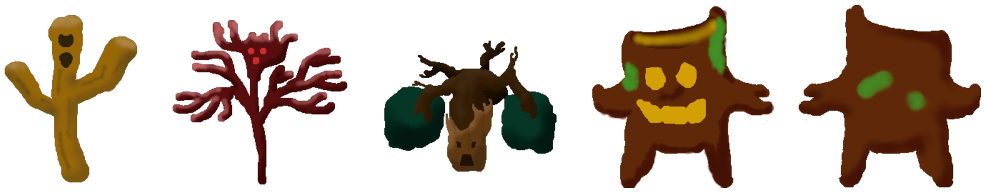


Figure 3.14: All monsters face forwards. Twiggy has both forward- and backward-facing sprites. All sprites have a similar perspective.

Sound throughout the game is meant to enhance the player experience and maintain stylistic consistency. Due to my inexperience making sound effects, *Under Grove*'s sounds are downloaded from the Unity Asset store. Player actions can provide sound-based feedback in the form of short chimes. Buttons emit chimes on mouse hover and click. Chimes also play when players collect items, open chests, and take damage from monsters. Sound effects also appear in the form of jingles when players access The Shoppe Ascension or Game Over. The final use of sound is background music, which loops repeatedly in the background. All sound volume is balanced to avoid overpowering noise.

Lighting was used within Play Mode to provide atmosphere and direct player actions. The scene lighting in Play Mode is low to create a dark atmosphere. Although rooms are visible, the primary light source is the player's fireflies, which surround Twiggy's body to allow easy identification of his sprite. Fireflies used as navigation markers also act as area lighting. Finally, to address feedback of poor door visibility, fireflies automatically appear at doors and act as a light source to indicate their location.

When changing from one view to another, transitions are smooth to maintain an immersive experience. For each of these transitions, the game uses a fading effect. Popup windows also open and close with a fading effect. When transitioning between different screens, background music is also faded. These transitions prevent any jarring changes for the player.

4. System Implementation

4.1 Project Software and Tools

Game Engine

The first major technical decision for developing *Under Grove* was choosing which game engine to use as a foundation. Although the exact definition of a game engine can vary, it can be defined as a set of abstracted and modular frameworks that run a game's content. A game engine may contain frameworks to handle a game's physics, lighting, or sound [48]. Because *Under Grove* will be distributed to players through a website, the selected game engine should be capable of publishing a game in WebGL format, which is a JavaScript API that can render graphics and games on a website [49].

I considered Unity, Babylon.js, and Three.js as possible engines for this project. In the end, Unity was chosen for two reasons. Unity allows distribution to multiple platforms, which allowed flexibility if publishing the game to a website failed. Unity is also a popular engine that has detailed framework documentation, step-by-step tutorials, and community forums.

Game Distribution

I explored two methods for distributing the game to players. I tried to distribute the initial prototype by building the Unity project as a Windows Standalone application. If delays prevented the completion of a hosting website, a standalone application would be used instead for user evaluation. As the author of the game, I was able to open and play the standalone application. However, when sending the game file to another computer it was marked as unsafe. After some investigation, I discovered this warning indicates the compiled, executable files are not code signed, and thus, not trusted applications. On Windows computers, a person can continue to play the game after bypassing the warning. However, when testing the viability of a MacOS Standalone application, users were unable to run an unsigned application. Code signing requires a prohibitively expensive signing certificate, so pursuing standalone applications would limit the study population to Windows users. With this understanding, web distribution became vital to the evaluation.

After some initial investigation, I discovered a quick solution for distributing the game over the web. Public GitHub repositories have the ability to host a web page. When Unity builds a WebGL application, it produces an index.html file that can be used to host such a web page. These files were pushed to a GitHub Repository and hosted at Under Grove. One main benefit of hosting the game using this method is that updates only need to be distributed to the host. One challenge, however, is website caching, which stores information in a user's browser to make loading faster. Caching can prevent users from seeing these updates immediately. Even when preventing caching through Unity's build settings, players may need to take the extra step of hard refreshing the page to ignore cached data. This additional step did not, however, outweigh the limitations of standalone applications.

Art and Sound Tools

To produce the game's sprite art, I used GIMP, which is a free image-editing software. After producing the sprites, they were imported into Unity. Certain sprites were sliced using Unity's Sprite Editor. Slicing a sprite creates portions of an image that do not scale when it is resized. For example, all rectangular buttons, regardless of size, utilise one base sprite with sliced borders.

I used Audacity, which is a free audio-editing software, to edit sounds and music files. These edits primarily involved normalising volume sounds to prevent overly loud effects. However, additional editing was completed for Play Mode's background music. The original music included extraneous bird noises in the opening that prevented looping. The song also had drastically different volumes throughout. Sections of this song were trimmed and the volume was normalised to improve the immersive qualities of the song.

4.2 Data Model

Whenever a player starts a new level, a dungeon is generated in the background. This dungeon data model is used to generate the player's current room. The data model does not rely on the Unity framework, and can act as a standalone C# project. All classes within the data model implement unit testing to ensure proper class behaviour.

Dungeons and Rooms

In *Under Grove*, each level is called a 'dungeon'. Dungeons are composed of rooms, and each room is composed of room units. A dungeon is also an internal data structure with two main forms (Fig. 4.1). The Room Dictionary is useful for iterating over rooms and retrieving rooms from a given id. The Dungeon Matrix is used for iterating over the entire dungeon and easily checking neighbouring units. The Dungeon Matrix allows generating rooms while guaranteeing they will not overlap geometrically.

The units within a room are stored in a list. A room contains definitions for its contents to allow different handling during rendering:

- Has the SpiriTree
- Has monsters
- Has a chest
- Has a dungeon object.

.

A room unit is the most basic component of a room and is represented as a square. Each side corresponds to a CardinalDirection object. Cardinal directions can have a value of either

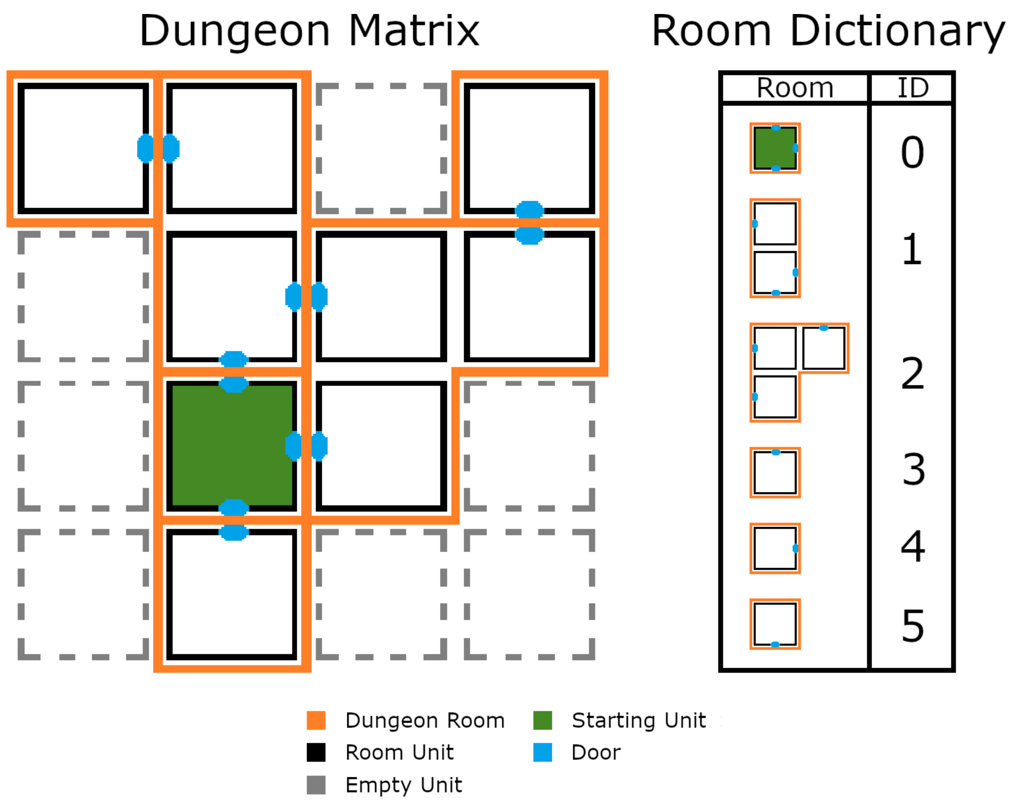


Figure 4.1: In the Dungeon Matrix, a dungeon is defined as an array of room units. The Dungeon Matrix has a constant size that is greater than the maximum possible size of generated dungeon. In the Room Dictionary, dungeon rooms are paired with their assigned ids.

north, west, east, or south. These directions also have methods to help determine properties based on the direction. For example, a north cardinal direction can generate northward dungeon coordinates from a given Dungeon Matrix coordinate. The CardinalDirection class also overloads operators. The opposite direction is generated with the ! operator. The ++ operator allows looping through all directions to check all sides of a unit.

The room unit also acts as a data structure. A unit holds a reference to its parent room and surrounding walls. Walls are stored in a dictionary with a cardinal direction key. Each unit also contains its Dungeon Matrix coordinates so neighbouring units can be checked regardless of their parent room. The unit also contains several properties to help unit rendering:

- Has the SpiriTree
- Has a dungeon object
- Has fireflies

Dungeon walls are a data structure that act as room borders. Walls contain references to the parent unit and room. They can also contain a reference to a dungeon door. Doors are used to connect rooms. They contain pointers to both parent and connecting units. Doors allow retrieving information about a connecting room when a player exits the current room.

Dungeon Generation

A dungeon is generated using a procedural generation algorithm. All dungeons start with one starting room. This room is always composed of a single room unit. The first step in this propagation is to add doors to surrounding walls of the starting room. After this, the same steps are repeated until the dungeon is fully generated:

1. For each door, generate a dungeon room
2. For each new room:
   (a) Generate room units
   (b) Add walls to the new room
   (c) Try to add doors to each valid wall in the new room

Walls in the current room are only invalid for door generation if the neighbouring unit's wall does not have a door. In other words, a door must eventually connect with another door. When new rooms are created, doors normally lead to null room units that are subsequently generated. Because room generation is completed for each door, all existing doors will eventually connect to a non-null room unit.

When adding new doors to walls, a custom Branching Probability Equation (BPE) is evaluated (Eq. 4.1). The BPE is used to determine if a room will have new doors. The probability (p) is 100% for the dungeon's very first door, which guarantees at least one door will be created in the starting room. Every subsequent door in the dungeon has a decreased chance of generating based on two factors: how many doors already exist in the room (x) and the current room id (n). As a result, as room id increases, door generation becomes less likely. This equation is designed to generate dungeons with approximately 11 rooms.

$$p = \frac{\lfloor 4^{-x} \times 100 \rfloor}{100} \left(1 - \frac{\lfloor (\frac{\arctan(25(n-10))}{\pi} + 0.499) \times 100 \rfloor}{100} \right) \tag{4.1}$$

When generating a new room, a starting unit is created in the direction away from the connecting door. In order to add variety to the shapes of the rooms, I created an algorithm to propagate a room from this starting unit (Fig. 4.2). To determine if a unit will propagate in a specific direction, I created a Room Propagation Equation (RPE, Eq. 4.2). Units are propagated in a variable number of rounds called generations (x). For each round, every vacant neighbour is evaluated for propagation. The chance (p) for propagation decreases as the number of generations increases. In the final prototype, based on user feedback, rooms propagate for two generations.

$$p = \frac{1}{2x} \tag{4.2}$$

After the rooms of a dungeon are generated, they are populated with various dungeon elements. The starting room, however, will never have any of these special features. When populating the rooms, the order in which features are added is important because features can be dependent on other features. The SpiriTree is the first object that is populated, and can be placed in any room aside from the starting room. Once added, this SpiriTree room is tagged and cannot be populated with any other feature.

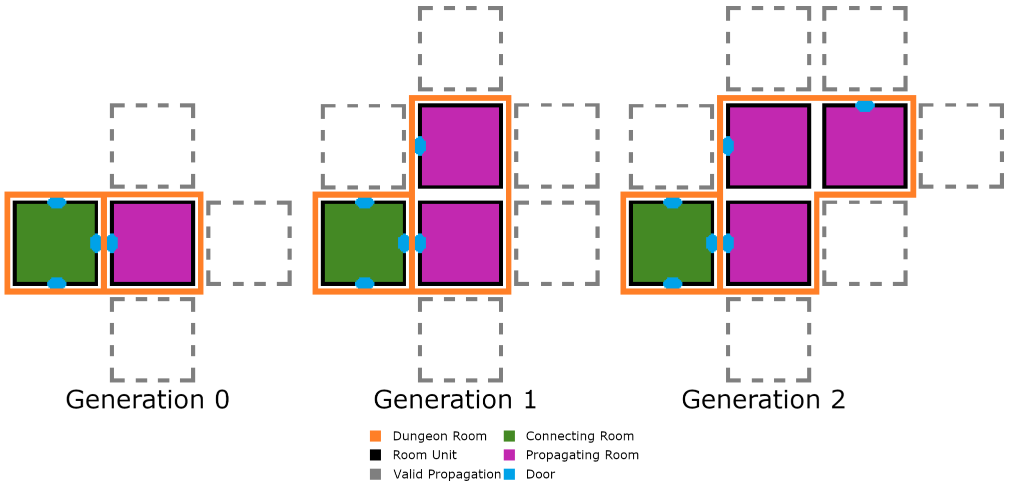


Figure 4.2: When a new room is first generated, it consists of one room unit. A room then undergoes propagation for a designated number of generations. In each generation, the neighbouring vacant room units of each existing room unit is evaluated for propagation. If a vacant unit passes the propagation checks, it is added to the room. This image shows an example propagation, from left to right, for two generations.

The next features to populate are monsters. Monsters are enemies which the player can attack and destroy. Interacting with the monsters can also reduce the player's health. Monsters are added to rooms based on probability. In these monster rooms, each unit is evaluated and will have monsters based on random chance. When monsters are added to a room unit, they are added as a monster unit, which contains definitions for its monsters. During this stage of population, the size (small, medium, or large) and combat type (melee or ranged) of each monster is determined randomly. Monster units are assigned before any other objects are added for extensibility reasons. For example, some overworld objects might need to be placed in non-monster rooms.

Within the dungeon, players are able to interact with dungeon objects, which are overworld objects found in the center of a room unit. All dungeon objects belong to the IDungeonObject interface and are added to rooms via an IObjectGenerator interface. Currently, the types of objects available in *Under Grove* include SpiriTokens, keys, and chests. In future versions of the prototype, I plan to add landmarks, such as trees and boulders, to improve dungeon navigation. Only one SpiriToken and chest can exist within a room as an overworld object. To maintain this rule, rooms are marked with special tags if a SpiriToken or chest is added to any child unit.

Once all objects have been added to the rooms, monsters creation can continue by arranging each monster unit. During arrangement, the quantity of monsters in the unit is negatively correlated with monster size. The monster quantity also helps determine how monsters will be arranged within the unit (Table 4.1). Because dungeon objects are always found in the center of a room unit, monster units composed of a single monster, are placed dependent on the existence of an IDungeonObject.

Monster Arrangement

Quantity	Arrangement	Conditions
4	Square	All cardinal direction orientations of a room unit
3	Triangle	Combination of three cardinal direction orientations of a room unit
2	Diagonal	Combination of two cardinal direction orientations of a room unit
1	Center	If there is no IDungeonObject in the room unit
1	Side	If there is an IDungeonObject in the room unit

Table 4.1: This table explains the different arrangements of monsters within a room unit.

Collectibles, which are items contained by either monsters or dungeon objects, are generated next. These items are defined as jewels, coins, rocks, fireflies, HealthNuts, and SpiriTokens. Currently, collectibles appear in the game when a chest object is opened with a key or a monster with a held item is defeated. Because SpiriTokens are needed to complete the level, these are the first collectibles to be generated. They are preferentially added to random chests based on probability. Any remaining SpiriTokens are then added to random monsters as held items. In the unlikely event remaining SpiriTokens could not be given to monsters, they are added as dungeon objects in any available room units. After adding the SpiriTokens, all remaining chests are filled with a randomised item, and a specified percentage of monsters are given a randomised held item.

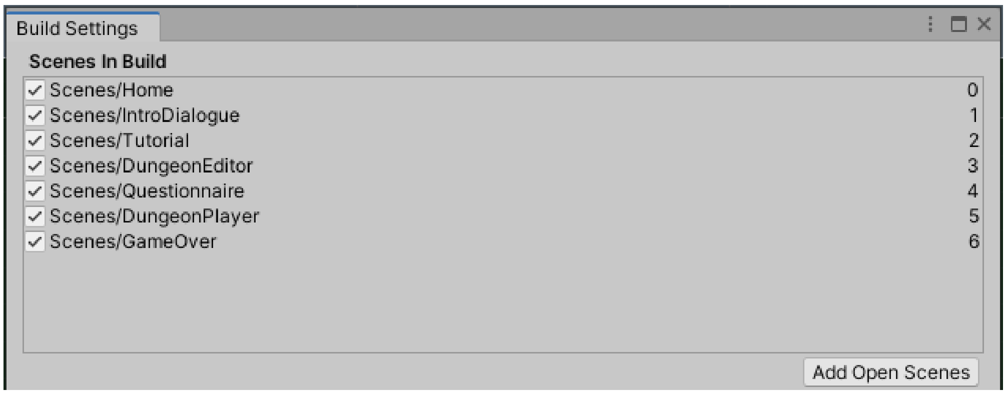
4.3 Game Behaviour Scripting

Unlike the data model, game behaviours and rendering utilise the Unity game engine frameworks.

Game Scenes and Transition

In Unity, a game is broken into different scenes. Previously, in Sec. 3.5, I referred to these different scenes as screens. For the intents and purposes of this game, each scene represents a different screen, and these terms are synonymous in this implementation. Windows, however, do not comprise an entire scene, but only function as part of one. When running the game, a set of scenes indices are loaded (Fig. 4.3). The Home scene launches automatically because it is marked as index 0. After launching the Home scene, the remaining scenes are launched programmatically based on the actions of the user. Typically, scenes are loaded when a player clicks a designated button, such as the Play or Design button in the Home scene. However, when playing the game in the DungeonPlayer scene, the game can change scenes automatically, such as when the player's character runs out of health.

In order to transition from one scene to another, several actions are taken. In each scene, one game object contains a full-screen overlay. This overlay is typically black, and transparency is used to fade the object when entering or exiting the scene. Animating the transparency provides a non-programmatic method to create a smooth visual transition. Audio is also transitioned by fading music during scene changes. This transition cannot be animated,



Scenes In Build

✓	Scenes/Home	0
✓	Scenes/IntroDialogue	1
✓	Scenes/Tutorial	2
✓	Scenes/DungeonEditor	3
✓	Scenes/Questionnaire	4
✓	Scenes/DungeonPlayer	5
✓	Scenes/GameOver	6

Figure 4.3: This image shows all scenes used in the *Under Grove* prototype. The scene with index 0 is the first scene to load when the game starts.

but is instead accomplished programmatically using a Coroutine, which allows programmatic changes to occur over multiple game frames.

Room Fabrication

Room fabrication is a process by which rooms are created visually in the DungeonPlayer scene based on the dungeon data model. The fabrication process also involves any activities that occur while the room is loading, such as loading ad windows or The Shoppe Ascension. Only one room is presented to the player at a time, even though the entire dungeon already exists in the background. With this approach, only the player's current room needs to be fabricated. The process begins by rendering the starting room of the dungeon with the player's character at its centre.

To accomplish fabrication, a collection of game objects are used within Unity. Generally, these objects are called 'fabricators', and they are responsible for rendering different game objects in the scene. A Store Fabricator object is used to generate the store window when the player completes a level. If ads are enabled, an Ad Fabricator is used to generate ads when the player travels between rooms. Within the Room Fabricator, other helper fabricators are used, such as the Monster Fabricator, Dungeon Object Fabricator, and Unit Fabricator.

Once a room is fabricated, game components control the behaviour of the new game objects. These game components can include scripts that allow temporary data storage and reactions to the environment or player. A room is fabricated based on the room units it contains. For each unit in a room, game objects exist for the walls, doors, and floor. Each of these objects contains the sprites that visually make up the unit. The floor object uses a script component to reference the corresponding room unit in the data model, which allows referencing any elements present in a room unit. The door objects similarly hold a reference to the corresponding data model's door. When player's interact with a door in a room, the connecting room is loaded from this reference.

After the components of a room are rendered, dungeon objects and monsters are fabricated. All dungeon objects are placed in the center of the corresponding floor unit. Because all objects implement the IDungeonObject interface, they are handled identically (Lst. 4.1).

Monsters are then generated for each of the corresponding room units. After the arrangement process, monsters contain a cardinal direction that indicates their unit position. Dungeon objects and monsters all hold a reference to their data model equivalent, which allows tracking of their status, such as a monster's health.

```
public void FabricateObject(RoomUnit unitToMake, GameObject parentUnit)
{
    IDungeonObject unitObject = unitToMake.UnitObject;
    GameObject tokenToInstantiate = unitObject.GetPrefab();
    GameObject createdToken = Instantiate(tokenToInstantiate, parentUnit.transform.position,
     Quaternion.identity);
    createdToken.transform.parent = parentUnit.transform;
    IObjectBehaviour objectBehaviour = (IObjectBehaviour)createdToken.GetComponent(
    unitObject.GetBehaviour());
    if (objectBehaviour != null)
    {
        objectBehaviour.ModelObject = unitObject;
        objectBehaviour.ParentUnit = unitToMake;
    }
}
```

Listing 4.1: This code is responsible for dungeon object fabrication.

The SpiriTree room requires special handling to differentiate it from the other rooms in a level. Doors leading to this room are indicated by cyan fireflies rather than the normal amber ones. This change can be accomplished because door game objects have a script component that references the data model's doors and connecting rooms. If the connecting room contains the SpiriTree, the firefly colour is programmatically changed to cyan. Furthermore, the actual SpiriTree room has different colouration than the other rooms. This is accomplished via a Material, which is a Unity game component that can control a game object's appearance. A Material is applied to the floor programmatically if the room is contains the SpiriTree.

Player Status and Behaviour

Within the DungeonPlayer scene, one game object represents the player's character. The sprites and animations for the character are linked to this object. In addition, all scripting to control the character's behaviour can be found on this object. These behaviours include player movement, player status, player interaction, and weapon behaviour.

Player movement is controlled via a custom script on the player object. The player is able to move the character in any direction on the $x - y$ plane. This is accomplished by using Unity's built in input scripting API and a GetAxis method. This method allows the system to receive player input from either the WASD keys or the arrow keys [50]. GetAxis returns a positive or negative value for either the vertical or horizontal axis when a corresponding key is pressed. If this value is non-zero, the player object's velocity is changed to move in designated directions. This strategy alone would cause the player to move faster diagonally when both a horizontal and vertical input occurs. However, to maintain consistent player speed, the velocity is divided by $\sqrt{2}$ in this scenario.

By default, the player character is facing the front and can only attack to the right of the character. Based on user feedback, players wanted the character to be able to attack from both directions. To add this functionality, the input scripting API was utilised again. If the horizontal input axis is negative, this implies the character is moving to the left, and the player's velocity will be negative. In this scenario, the character's sprite will change to face the back of the room, and his weapon will switch to the left side. Thus, when players are moving to the left, they will be able to attack on the left side.

Two separate scripts control the behaviour of the character's attacks. The first script handles the rendering of the weapon and its strength. The second script controls the character's actual attack. When the player presses the Space key, an animation plays and the weapon swings. The method to damage monsters during the swing could be accomplished through two different options. The first is utilising a BoxCollider component on the weapon. A BoxCollider is used to detect contact with other objects. This method poses some problems, however. The BoxCollider would exist on the weapon when not attacking, and extra handling would be needed to prevent collision detection in this default state. Instead, another method has been implemented that instantiates a new BoxCollider when the character swings the weapon. This collider only exists during the swing and can detect all possible collisions. The size of the collider is also easily configured and serves as the mechanism for the Attack Range perk in The Shoppe Ascension. After detecting collisions during a swing, monsters can be identified and react to being hit.

One script controls the players abilities to add fireflies to room units. The current room unit is detected using BoxColliders on both the player and the floor. When the player presses the Q key during this collision, the script updates a Boolean in the data model's room unit reference. When this field is true, the script renders a firefly game object.

A handful of other scripts handle the remaining player behaviour. One main script controls the player's interaction during object-player collisions. This script is responsible for allowing the character to interact with objects using the E key and updating the player status. Interactions drive game dialogue via an IInteractable interface, which is implemented by all interactable game elements. During an IInteractable collision, an interface method creates a text box with corresponding dialogue and displays it to the screen. The player status is also handled through collisions. Collectibles and keys are updated when the player collides with these objects, and the player's health decreases when they collide with monsters.

The player status also helps maintain the player's credits and score; however, these elements are stored in a static class to achieve persistence. If the player loses all of their health, they will be sent to the Game Over scene, which destroys the player's game object and any stats they had at that point in time. Using a static class is a simple way to maintain these data across scenes. Alternatively, Unity has a scripting API that prevents destruction of a game object when a new scene is loaded [51]. This API was not used because some player stats, like health, keys, and currency should reset upon character death.

Design Mode

Under Grove's Design Mode allows players to adjust the levels of dark design elements within the game. The values for these settings are stored in a static DungeonSettings class to maintain values during scene transitions. This class also contains methods to encode and decode these settings using strings. As discussed in Sec. 3.5, each dark design element is represented using a slider UI element. A Slider Value Interface game object acts as a bridge between the slider values and DungeonSettings by keeping all values synced. In the Slider

Value Interface, each dark design element is paired to a slider and DungeonSetting field. If either updates during a game frame, then the corresponding values are updated.

Design Mode allows players to use an Automate button to configure all dark design elements at once (Fig. A.5). An encoded string can be used as input for the Automate button. This string consists of a character for each dark design element, with each possible value mapping to exactly one character (Table A.1). If no encoded input string is provided, the Automate button calls a DungeonSettings method that randomises each element value. If an encoded string is present, a DungeonSettings method runs during each frame of the game to evaluate its validity. If the string is too short or contains invalid characters, a warning icon is enabled, and the Automate button does nothing. Otherwise, the settings are interpreted, and dark design elements are updated to their corresponding values.

The Design Mode also has a two other functionalities. The player can test their current settings by entering Demo Mode. Demo Mode is the same as Play Mode outside a few differences. In Demo Mode, the player is taken directly to the DungeonPlayer scene, bypassing both the intro and instruction scenes. Additionally, within the player, there is an extra close button. When this button is clicked, the Design Mode scene is reloaded. Finally, in order to test settings, the dark design element values are maintained when transitioning to and from Demo Mode.

To leave Design Mode, players can click a close button and reload the Home scene. When this button is clicked, all dark design elements are reset to zero. Resetting these values required additional consideration. The Slider Value Interface detects changes during each frame the game is running. Loading a new scene uses a Coroutine to ensure certain methods are completed before others. If the dark design elements were not reset after loading the Home Scene, the Slider Value Interface would revert settings to the previous slider values with its syncing behaviour. As a result, if users re-entered Design Mode, their old settings would remain. In order to handle this situation, a new Coroutine for scene loading was implemented where DungeonSettings are cleared only after the Home scene is loaded.

Universal Elements

Throughout the game, fireflies are used as visual indicators. In several circumstances, these fireflies possess colours that vary from the default amber. This change can happen for the fireflies surrounding the players character, the fireflies the player places in each room unit, and the fireflies used as door indicators. In order to achieve this effect, a script was created to alter the colour of the fireflies based on certain parameters. The player's status stores the colour that will be used for the fireflies around the player and those placed in room units. Door fireflies use a default colour unless the door leads to the SpiriTree. In this situation, the colour is changed to cyan.

In order to change the colour of the fireflies, two steps need to be taken. The first is to change the base colour of the corresponding Particle System, which is a Unity component that emits the firefly particles with a variety of effects. The second change involves the blinking of the fireflies. In order for the fireflies to blink, a colour transition gradient is needed to alternate between one colour and transparency. In the Unity editor, this is created within the GUI. However, to change this dynamically, the gradient must be created programmatically [52]

(Lst. A.1). The designated colour must be assigned as a colour key. Any blinking behaviour must be assigned using alpha keys, which signify the transparency.

Dark design elements are also implemented differently throughout the game. The value of each element is used as a variable in an equation that changes the behaviour of the corresponding dark design element. For example, when the game creates chests, each has a chance of becoming a loot chest. This chance (p) is based on an equation (Eq. 4.3) that factors in the corresponding dungeon setting (x). These equations are incorporated either in the data model (e.g. Loot Chest Frequency, Search Difficulty) or during fabrication (e.g. Ad Rate and Ad Length).

$$p = x \times 0.08 \tag{4.3}$$

5. Evaluation

The main goal of this project is to determine how effective *Under Grove* is as an educational tool. To support this goal, one primary hypothesis was evaluated: Players will be able to recall more dark game design patterns and feel more familiar with these patterns after playing *Under Grove*.

In addition, other educational effects were explored through two secondary hypotheses:

1. Players will feel more familiar with game design after playing *Under Grove*.
2. Game designers will have a better understanding of dark game design patterns after playing *Under Grove*.

5.1 Longitudinal Experiment

Experiment Methods

To test my hypotheses, I used a longitudinal experiment to allow participants to become familiar with *Under Grove* over time. During this experiment participants would be required to complete daily tasks to gradually acclimatise participants with the prototype's functionality. This process is important because player unfamiliarity could lead to a lack of understanding of *Under Grove*'s educational features and become a confounding variable when testing all hypotheses. Prior to the experiment, I recruited fellow coursemates and acquaintances to form a study sample of size $n = 10$.

For the first task of the experiment, I gathered baseline information about the experiment group using a required pre-experiment survey. During this survey, I asked questions to learn how familiar the participants were with video games. This set of questions allowed me to determine if video game experience impacted the results. I also gathered baseline data for participant familiarity with dark game design patterns and game design in general. To quantify this information, I used Likert-type questions. I also asked participants to produce any known examples of dark game design patterns. Finally I asked participants how they would design games given certain priorities, such as enjoyment and profitability.

After participants completed this survey, they began the tasked portion of the experiment. During the first two days, tasks focused on the game's Play Mode instead of the Design Mode. The goal for these days was to increase player familiarity with gameplay mechanics and identify any major game issues. On days one and two, participants were tasked with trying to achieving a personal highest score. During day two, dark game design patterns were randomised into the game to allow participants to experience the different dark design elements. Following these two tasks, participants were required to report their final score to me.

During the final two tasks, participants focused on Design Mode. The goals for these two days involved getting participants to apply different dark design elements to games to achieve different goals. On day three, participants were asked to create a level that would keep players playing the longest. The primary goal of this task was to encourage users to learn about temporal dark design elements. On day four, participants were asked to create a level that would encourage players to spend the most credits. The primary goal of this task was to educate participants on monetary dark patterns. For these tasks, players needed to send me the encoded setting string for their custom level.

Following the experiment, all participants were asked to complete a post-experiment survey. In order to categorise participants based on gaming habits, certain questions from the pre-experiment survey were repeated. These questions included video game play frequency and preferred video game genres. In addition to these questions, I once again asked participants if they felt familiar with dark game design patterns and game design in Likert-type question format. I also asked each participant to produce examples of dark game design patterns.

A game designer was also interviewed to gain expert insight into the current prototype. At the time of the interview, the designer had six years of experience, some of which includes educational video game design. During this interview, the designer was given tasks so they could familiarise themselves with *Under Grove*. The data from these tasks was compared to the player tasks. The designer also provided insights into improving the educational components of the game and overall gameplay experience.

Experiment Results

Based on the pre-experiment survey, approximately 40% of participants responded that they play games infrequently, or less than monthly. 20% of participants reported that they have never played video games. 70% of participants have played free-to-play games prior to the experiments. The participants represented the following age categories: 18-24 ($n = 2$), 25-34 ($n = 4$), 35-44 ($n = 1$), and 45-54 ($n = 3$). Overall, the study population had a diverse range of age and gameplay experience.

Participants were asked to provide preliminary information regarding video games use. When participants were asked what they like about video games, 40% reported the ability to escape, 20% reported video games are relaxing, and 20% reported enjoyment. When asked what they don't like about video games, participant responses had no clear trends. However, participants mentioned ads (20%) and addiction (20%) as detractors. When deciding to play a game, participants choose whether to buy a game based on either friend or critic recommendations (40%), price (30%), or story line (20%). When deciding to purchase a game, participant responses had fewer trends, but mentioned payment methods (30%), such as subscription-based costs, and critical reception of the game (20%). A subset of participants do not purchase games (20%). Participants were also asked if they have made in-game purchases before, and 70% responded yes. Following this survey, one participant was removed from the study sample because they did not complete the subsequent daily tasks.

Participant self-reported familiarity with dark game design patterns and game design was recorded pre- and post-experiment. Because this data was recorded using a Likert-type question, pre- and post-values were mapped to integers ranging from one to five, with one mapping to 'Strongly Disagree' and five mapping to 'Strongly Agree'. When analysing the participant responses, these were not normalised, so a non-parametric statistical test was needed to see if familiarity had a significant increase.

Using a Mann-Whitney U Test to compare values, dark game design pattern familiarity had a non-significant increase ($p = 0.60$, Fig. 5.1). Data from participants could be broken into two groups based on frequency of gaming: plays games monthly of more often (frequent) and plays games less often than monthly (infrequent). When comparing these groups, frequent players saw a greater increase in dark design familiarity. Similarly, a Mann-Whitney U Test showed a non-significant increase in game-design familiarity ($p = 0.52$). Based on these data, null hypotheses that these increases were due to randomness for the primary hypothesis and secondary hypothesis 1 could not be disproven.

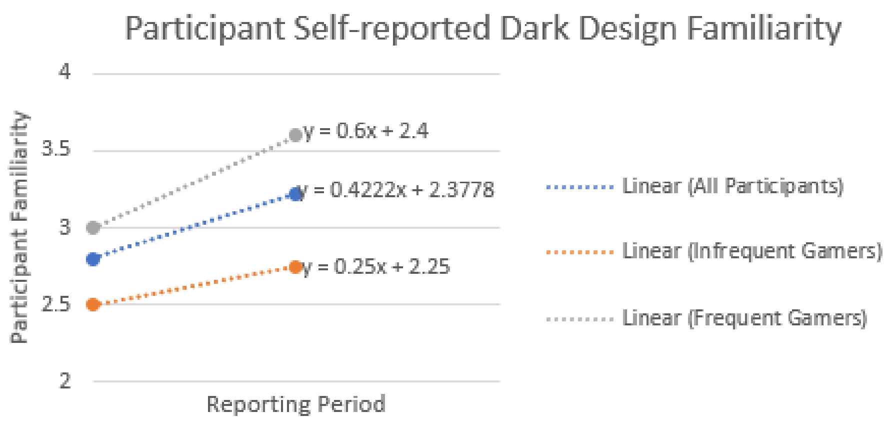
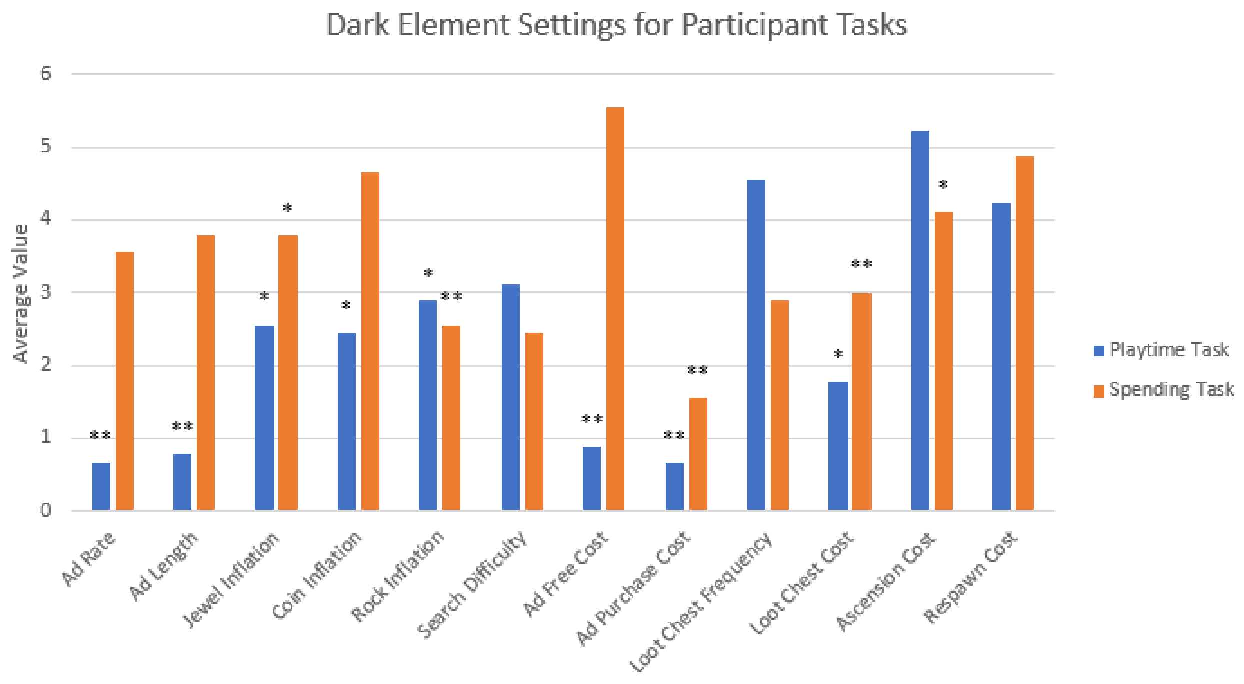


Figure 5.1: This figure displays the increase in dark design familiarity for participants over the longitudinal experiment.

During the experiment participants were asked to complete two design tasks. In the first task, participants configured dark design elements to keep players playing the longest. In the second task, participants were asked to design a level they thought would make participants spend the most credits. Because each setting can have a value between 0 and 10, truly random settings would have an average of $m = 5$. To determine if the submitted settings differed from a random mean significantly, a single sample t-test was performed for each setting for each task. However, re-evaluating the data revealed the setting data is not guaranteed to be normalised. This significance was re-assessed with the non-parametric single sign test (Fig. 5.2).

For the play duration task, Ad Rate, Ad Length, Ad Free Cost, and Ad Purchase Cost varied significantly from an expected mean $m = 5$ ($p \leq 0.01$). Jewel Inflation, Coin Inflation, Rock Inflation, and Loot Chest Cost also varied from the expected mean ($p \leq 0.05$). When



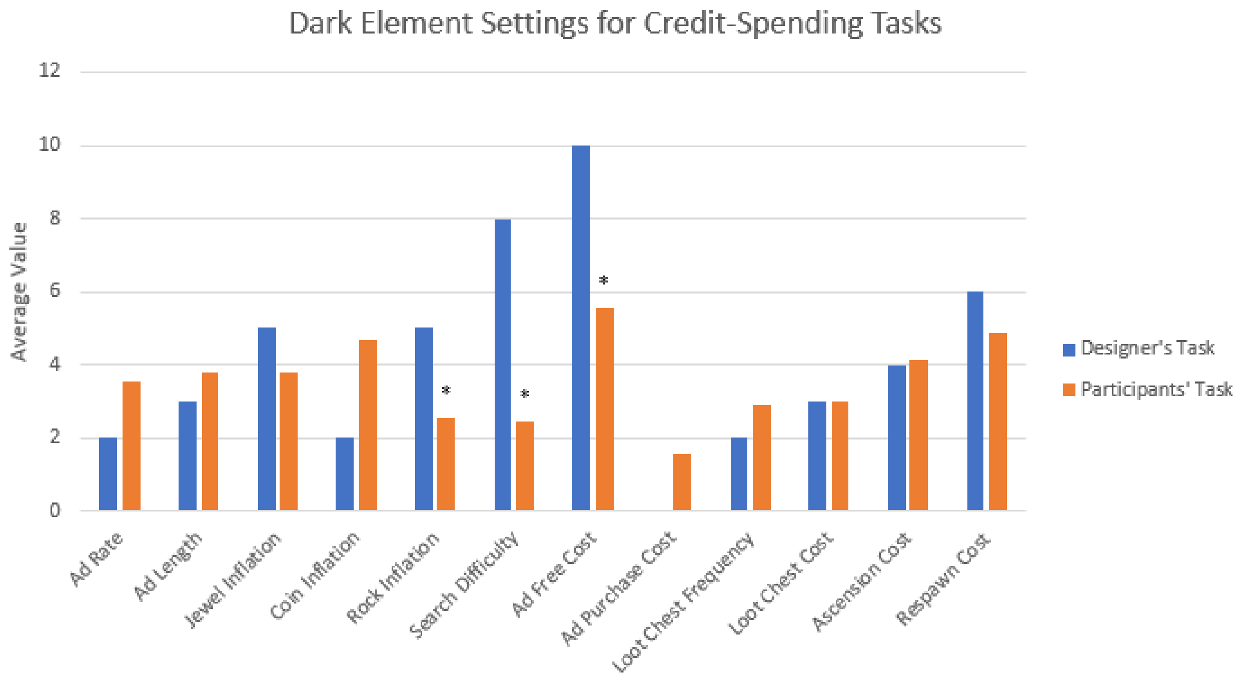


Figure 5.2: This figure displays the different settings for each design task of the longitudinal study. If values differ significantly from a truly random mean, * and ** represent $p \leq 0.05$ and $p \leq 0.01$, respectively.

asked for their reasoning for choosing these settings, participants mentioned increasing player rewards ($n = 6$) and limiting ads ($n = 6$).

For the credit-spending task, Rock Inflation, Ad Purchase Cost, and Loot Chest Cost varied significantly from the expected mean $m = 5$ ($p \leq 0.01$). Jewel Inflation and Ascension Cost also varied significantly from this mean ($p \leq 0.05$). When asked for their reasoning for choosing these settings, participants mentioned making the game harder ($n = 3$), including longer ads ($n = 4$), and loot chests ($n = 2$). All participants mentioned some form of in-game purchase ($n = 9$).

User feedback was assessed during the post-experiment survey. Satisfaction scores for *Under Grove* were compared to an expected randomised mean $m = 2.5$ using a single sign test for the five-point Likert-type questions (Fig. 5.3). Satisfaction for Play Mode overall, visuals and style, user interface, and gameplay varied significantly from and were greater than the expected mean ($p \leq 0.05$). Participants were also asked about *Under Grove*'s Design Mode and Play Mode. For Play Mode, players wanted improved navigation or a map ($n = 4$) and improved enemy behaviour ($n = 2$). Players enjoyed Play Mode's chests ($n = 3$), exploring ($n = 2$), and the game's atmosphere and theme ($n = 4$). For Design Mode, players wanted additional demonstrations or tutorials for configuring settings ($n = 2$). Players enjoyed Design Mode's customisation ($n = 4$) and demo capabilities ($n = 2$).

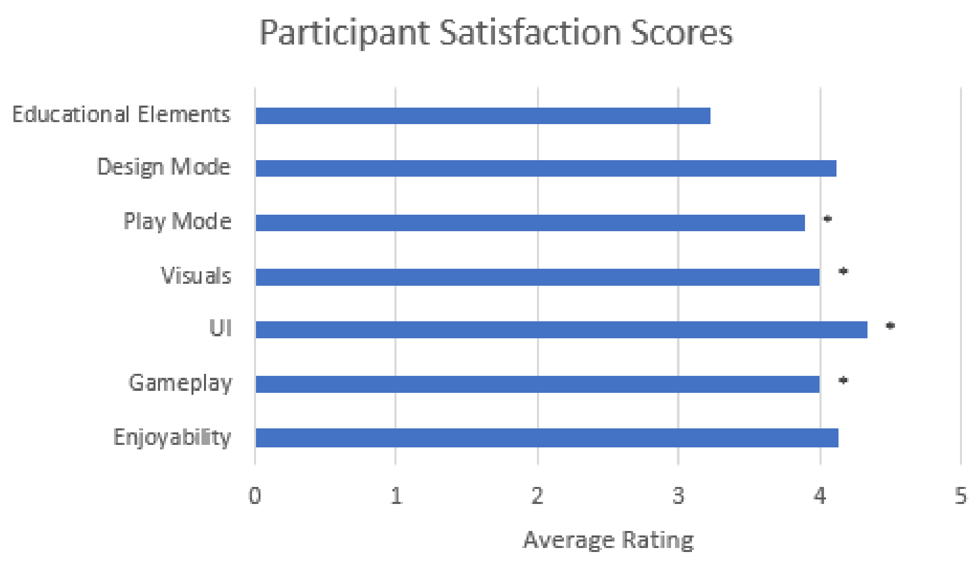


Figure 5.3: This image displays the average participant satisfaction scores from the post-experiment survey. If values differed significantly from a truly random mean, * represents $p \leq 0.05$.

During a separate evaluation process that involved interviewing an experienced game designer, additional feedback was obtained. The designer suggested adding clearer instructions and motivations for gameplay. They also mentioned incorporating a goal or time to give players a stopping point. To make a more effective educational tool, they also suggested re-framing the current toolkit within a more narrative-based environment to keep players engaged while they learn. Not enough data was collected to quantitatively validate secondary hypothesis 2.

The designer completed a level-design task where they designed a level that would cause players to spend the most credits (Fig. 5.2). To decide these settings, the designer targeted actions the user would be most likely to take. To encourage attacking monsters, Search Difficulty was increased. In-game transactions were configured so that rarer and beneficial items cost the most (e.g. Ascension Cost). They chose to keep the Ad Free Cost high because it was a permanent, one-time purchase for each playthrough. The designer ranked jewels the most valuable currency because subsets of players love visual changes. The designer felt rocks were the second most valuable currency because they valued the associated perks over coin-based ones. For this reasoning, the designer increased the Jewel Inflation and Rock Inflation elements. The designer's settings were compared to the participants' credit-spending task. For this comparison, the designer's settings were considered the expected mean, and the average participant score was compared to these using a single sample sign test. Only Ad Free Cost ($p \leq 0.05$), Rock Inflation ($p \leq 0.05$), and Search Difficulty ($p \leq 0.05$) differed significantly between the designer and other participants.

Experiment Discussion

Based on the collected data, none of the primary or secondary hypotheses could be supported. However, participants did articulate the use of dark game design patterns in reasoning for design tasks. In debriefing conversations, participants also revealed they did not realise each element represented a dark game design pattern. Educational elements were rated the poorest in the satisfaction scores, and the game designer discussed the need to further engage players in the educational material. Considering this information, players may have learned about dark game design patterns, but may not have recognised they had. Additionally, players only played *Under Grove* for a minimum of 15 minutes each day. If players had more experience with *Under Grove* it is possible they would have learned more about dark game design patterns. Regardless, the educational content should be made clearer in future iterations of *Under Grove*. Finally, the small sample size prevented discovering any significant trends, especially amongst subgroups. Gathering a higher sample size could further identify trends and determine if factors like gaming frequency change the effectiveness of the game's educational capabilities.

Participants revealed which settings they perceive as contributing to increased play duration. Players mention reward systems as a means of getting players to play the game longer. By artificially increasing rewards, participants believed they could inflate play time. This finding in general aligns with the temporal dark pattern definition. Interestingly, however, participants believed ads would lead to less play time overall, even though advertisements can be used as a temporal dark pattern [5]. This finding suggests that the advertisements, which can benefit creators by increasing the amount of time players spend on the game, could actually have the opposite effect on players. Players also minimised the cost to purchase ads and the cost to remove ads entirely, which might suggest players believe they will play a game longer if microtransaction prices are lower.

To increase player's credit expenditures, participants chose lower Ad Purchase Cost, Loot Chest Cost, and Ascension Cost settings on average. None of these settings differed significantly from the designer's settings for the same task. One major difference between players and the designer was the Ad Free Cost. When interviewing the designer, they specifically mentioned maximising the Ad Free Cost solely because it was assumed to be a permanent, one-time purchase. While some players had a similar strategy, others felt that prioritising costs for visuals would make more money in the long term. One player specifically stated that if Ad Free Cost were too high, they would choose not to play the game. This may indicate that implementing this type of microtransaction may unexpectedly harm the designer rather than benefit them with certain player bases.

A subset ($n = 4$) of participants also utilised the ad-free microtransaction to encourage users to spend credits. However, rather than increase the Ad Free Purchase Cost, they increased the Ad Length. Each of these participants believed that doing so would manipulate the player into spending money. One participant specifically said, “I made the ads really annoying and apparent.” Overall, participants employed many strategies for manipulating players into spending credits. Participant use of Ad Length specifically aligns with the typical use of monetary dark patterns.

Throughout the experiment, players made interesting comments regarding their gaming behaviour. During day two, when dark design patterns were enabled randomly, several players commented how they refused to purchase items within the game. Although the credits were meant to represent real money, players unexpectedly treated it this way, which is normally difficult to achieve in a laboratory setting [53]. Four people notably discussed attempting to beat the current high score when it was available, and two people actively competed between each other. The game is intended to be social, but these features have not yet been implemented. This competitiveness between participants shows promise in moving forward with social features in the next versions of the prototype.

Overall, based on these data and designer feedback, *Under Grove* has established a potential foundation as an educational tool for dark game design patterns. However, educational elements need to be further researched and improved, potentially with the help of education experts. Improving gameplay mechanics to motivate players to learn, and mechanics to indicate a designated gameplay duration need to be implemented as well. A larger sample size in future studies can help determine for which players this tool will be most useful, such as differentiating between players with high or low gaming experience. If the latter, the game could potentially be tailored for younger audiences.

5.2 User Validation Testing

Starting the week prior to the experiment, I completed two sprints for the prototype. During the first sprint I worked with individuals to gather informal feedback for *Under Grove*. This initial feedback was used to discover any major issues or missing functionality prior to the experiment. The second sprint ran during the experiment. I collected user feedback from participants and added daily improvements to the game. This feedback was provided on an ad-hoc basis and was used to create user stories (Table 5.1).

User Stories
Players cannot copy and paste in the Design mode.
Players can exit ads without waiting.
Players cannot pause the game without exiting full-screen mode.
Players do not spend money when purchasing loot crates.
Players can purchase Revives even when they don’t have enough credits.
Players want the character to be able to attack from both directions.

Table 5.1: This table exhibits example users stories addressed during Under Grove’s prototype sprints.

After creating the user stories, they were prioritised in three ways: how fast the task could be completed, how severely it impacted the playability of the game, and how many people reported it. Most of the provided feedback involved minor changes to the game. In the hosted version of the game, for example, the Esc key both paused the game and exited full-screen mode. A solution involved enabling the P key to pause the game. Other changes were more complex. Several players commented that the game was too dark and doors were difficult to see. The brightness of a game can vary on different computers, but brightness settings have not been implemented in the current prototype. I intended the lighting of the game to be dark, so rather than change the overall lighting for everyone, I added a firefly effect around the doors to better indicate their location. I also added different colour fireflies to indicate a room connected to the SpiriTree room. Several participants provided positive feedback on this change.

One major problem was reported during day three of the experiment. While players could randomise the dark design elements and customise them, they could not copy and paste the encoded setting string from the game. Although I had tested the functionality within Unity, I did not expect the functionality to vary once hosted on the web. Upon investigation, this issue was due to browser security that prevents using Unity’s clipboard API. In order to re-implement the copy functionality, a JavaScript plugin needed to be included within my source code. Because this issue was due to browser security, most participants encountered the error. As a result, participants manually typed the codes they submitted to me on day three, which led to one known instance of a mis-typed string. Fortunately, I was able to clarify the submission with the participant. Interestingly, this problem was only reported once, which suggests other unreported issues may have been encountered during the experiment.

6. Future Work

6.1 Additional Game Features

6.1.1 Improved Procedural Generation Algorithm

While a new procedural generation algorithm was created for this project, similar algorithms have been created in previous work, albeit with slightly different functions. Functionality from these algorithms could be applied to *Under Grove* to increase variety within the game, which addresses critique received during user validation testing. In one example algorithm, which also drew inspiration from *The Legend of Zelda* series, locked doors and keys are incorporated as an additional puzzle element [54]. Although the puzzles were largely linear in this algorithm, it could be improved and incorporated into the branching paths created by *Under Grove*'s procedural generation algorithm.

A similar procedural generation algorithm for dungeon-style levels incorporates a third dimension by adding multiple floors [55]. This feature would add further complexity to the levels, but an in-game explanation may be required for this feature to be believable in *Under Grove*'s forest setting. To implement this feature a new ladder dungeon object could be added during data model generation. The Dungeon Matrix structure can be utilised to align each floor of the dungeon.

6.1.2 Database and Server Connection

Connecting *Under Grove* to a database and tracking data would benefit the player experience and overall purpose of the game. This game is currently a standalone game that is hosted on the web. As is, the game does not save progress, keep track of user created levels, or store player metrics. Implementing a database and user accounts could allow tracking these data, which would allow further analysis of player responses to dark design elements. Average level playtime and credits spending, for example, could be tracked alongside dark design element values. A database could also identify which elements designers are more inclined to use and if certain conditions can influence their use. These data could be used to help influence the regulation of dark game design patterns within the gaming industry.

6.1.3 Social Networking Capabilities

Adding social networking capabilities to *Under Grove* would allow enhanced abilities to learn about dark game design patterns. During this project's experiment, the competitive aspect of achieving a high score helped motivate participants to play the game longer. Design Mode is the most intuitive step for implementing these social networking capabilities. Currently, in order to share level configurations, users have to copy an encoded setting string and send this to friends using an external application. In future versions, players would share their custom levels within the game. They would be able to compete to create the best levels or compete to attain high scores for given levels. Overall, this feature would encourage further awareness of dark game design patterns.

6.1.4 Social-Capital and Psychological Dark Patterns

Outside of temporal and monetary dark patterns, other dark game design patterns are not currently explored in *Under Grove*. For example, this prototype does not feature dark design elements for psychological or social-capital dark patterns [4, 5]. Psychological dark design patterns focus on psychological methods to keep players playing a game [5]. Social-capital dark patterns involve multiple players and how they interact together within a game [4]. Dark design elements for these patterns could be implemented to create a more well-rounded educational tool.

6.2 Extended Game Applications

6.2.1 How Does the Credit Amount Impact Player Purchases?

During the experiment performed for this project, players were limited to two credits when playing the game. Interestingly, even though this was not real currency, multiple players actively avoided making any in-game purchases, even when tempted by features such as long ad length.

As an extension of this experiment, one could explore the possibility that the amount of perceived currency a player has affects how likely they are to make in-game purchases. This study could be accomplished by giving different default credit balances to different study groups. A study on *Fortnite* found that income was not a significant factor in whether players purchased items [56]; however, these results required self-reported answers by study participants. *Under Grove* presents an opportunity to explore this relationship directly if tracking player purchases. Ensuring participants treat credits like real money can be difficult in a laboratory setting [53], but methods from gambling studies can potentially be used to accomplish this task [57].

6.2.2 How Does Credit Amount Impact Designer Decisions?

During the experiment in this project, users in Design mode always had two credits. Similarly to Sec. 6.2.1, one future experiment could explore ways in which the credit amount affects a designer's choice to include dark design elements. This study could determine whether a credit limit encourages designers to use fewer dark design elements compared to those with no limit. If successful, inducing an imaginary budget in game testing could be used as a strategy to reduce the use of dark game design patterns by designers.

6.2.3 How do Dark Patterns Affect Play Duration?

Under Grove can be used to analyse the effects of dark game design patterns on play duration. Previous studies have shown that players are more likely to uninstall a game in the long term if they have made an in-game purchase [32]. Considering the objective of monetary dark patterns is to influence users into spending money, the resulting transactions could impact the amount of time a player will invest in a game. *Under Grove* can be used to explore the relationship between monetary dark patterns and playtime by logging the length of player sessions. This information could ultimately influence game developers to avoid these patterns if they are also harmful to their goals. Temporal dark patterns, on the other hand, could be analysed to see which have the most impact on playtime. This information could be used to categorise the severity of these patterns and influence regulations or rating systems.

6.2.4 Investigate Educational Impact on Game Designers

In this study, an experiment was performed to determine if players can learn about dark game design patterns by playing *Under Grove*. A designer was also interviewed in a separate portion of the game's evaluation (Sec. 5.1). However, time constraints prevented recruiting enough designers for quantitative data analysis. In a future study, a larger sample of designers could be used to determine if playing *Under Grove* educates designers on dark game design patterns and discourages their use.

7. Conclusion

7.1 Reflecting on *Under Grove*

This project examined a new, educational game prototype titled *Under Grove* and its capability of educating players on dark game design patterns. The game seeks to protect players from the effects of these design patterns, while simultaneously discouraging game designers from employing them. To achieve this goal, dark design elements are incorporated into an adventure game using a combination of procedural and random generation. Dark design elements serve as the primary source of education in the game, and players can configure them to varying degrees. These configurations allow players to experiment with these patterns and understand their effect on gameplay. The settings can also be turned on at random to allow unique and unexpected experiences.

User feedback drove development and enhancements for this prototype. Overall, all users generally reacted positively to the game and expressed feedback using both informal and formal methods. Both forms of feedback generally produced similar commentary. When provided, this feedback was used to prioritised continuous updates to the prototype. Due to time constraints on the project, however, constructive feedback for extensive features was not implemented. Such features included incorporating artificial intelligence for monsters, adding a game ending, and expanding game variety.

User feedback indicated that educational aspects, compared to entertainment ones, were less successful. While playing the game, players noted dark game design patterns; however, participants did not feel significantly more familiar with these patterns after playing the game. Thus, the hypothesis that playing *Under Grove* would improve players' familiarity with dark design patterns could not be supported. The game features information and explanations for dark game design patterns, but this educational material is likely not intuitively presented to players. In subsequent versions of the prototype, these aspects require additional research and development, potentially guided by education experts.

7.2 Study Limitations

Overall, the experiment performed as part of this project had several weaknesses that potentially reduced the validity of its results.

The study sample and duration inhibited the ability to observe trends in the collected data. The sample featured a somewhat diverse population, but the sample was relatively small. As a result, trends for different subsets, such as frequent and infrequent gaming, could not be analysed for statistical significance. Additionally, players may not have been able to fully familiarise themselves with the game before completing any required tasks. Each day, participants were only required to play the game for approximately fifteen minutes. If participants only played this minimum amount, their experience could have impacted the data for each task. To address these issues, the experiment should be conducted for a longer duration with a larger, but similarly diverse, sample.

Survey formatting posed another limitation on data collection. In both the pre- and post-experiment surveys, the quantitative basis for determining if users learned about dark game design patterns relied on two questions. In the first, users were asked how familiar they were with these design patterns based on a five-point Likert-type rating scale. However, participants had the option to choose a midpoint, 'Neither Agree Nor Disagree,' which prevented separating data into two distinct categories: agree and disagree. Removing the midpoint could potentially better identify changes in familiarity [58]. In the second question, participants were asked to produce up to three examples of dark game design patterns. However, this question was optional in the event participants could not recall any examples. The majority of participants skipped this question, but potentially because it was optional rather than a lack of knowledge. Instead of this recall format, a quiz might better judge the improvement in dark game design pattern recognition.

As mentioned in Sec. 7.1, participants found the educational aspects to be the least satisfying of all measured metrics during the post-experiment survey. After debriefing the experiment with participants, I learned that users did not always explore all information available in the game. For instance, when playing in Design Mode, an info button exists to explain how the settings correspond to dark game design patterns. However, participants did not readily click this button, so they did not all make the connection between the game's settings and dark design patterns. A small change by making this information window appear by default could drastically impact future studies with this video game. This change would match the instruction presentation already present for Play Mode.

7.3 Ethical Considerations

While the use of dark design patterns in software already involves ethical discussion, this section will focus on ethical considerations for the content and applications of *Under Grove*. This ethical consideration will consider arguments from the perspective of normative ethics. Overall, I argue that *Under Grove* is ethical because its purpose to educate players and designers on dark design patterns ultimately benefits all parties. Some aspects of the game, however, could improve the ethicality of the prototype.

Although *Under Grove* intentionally exposes users to dark design elements, this use case is ethical. Each of these elements exhibits a different implementation of a dark game design pattern; however, one key attribute of both temporal and monetary dark patterns is that players unknowingly spend time or money [4]. In *Under Grove*'s case, players are informed upfront about the settings' effects on users. As a result, the dark design elements within *Under Grove* are not technically dark game design patterns, and their use is ethical considering deontological ethics. Additionally, the purpose of the exposure to these patterns is to educate users, which ultimately aims to protect them. Therefore, this use of dark design patterns is also ethical based on both virtue ethics and consequentialism.

Under Grove itself is not free from dark game design patterns, even though unintentional. For example, the current prototype has no ending, and the game continues infinitely. This game feature can be classified as the infinite treadmill pattern, which can potentially contribute to problematic video game playing [5]. With a consequentialism view, this aspect of the game is potentially unethical. Thematically, a game where a fantastical being must ascend infinitely to reach some form of enlightenment arguably has artistic meaning. However, this choice potentially is at the expense of its players. In future versions of the prototype, an ending will be implemented to eliminate this ethically questionable design choice.

The current *Under Grove* prototype was designed to avoid the need to store personal data. The game does not feature any user accounts or methods to track user data. However, in future versions, the game will store anonymous data to track how players respond to the various dark design elements. The purpose for this tracking is to discover circumstances where dark game design patterns are less effective on users. This tracking will also be used to determine strategies for discouraging designers from implementing these patterns, such as the idea described in Sec. 6.2.2. In both instances, data will be used to improve player experiences throughout the gaming industry. One may consider this use case ethical considering virtue ethics; however, potential use cases raise ethical concerns with regards to consequentialism. Data tracking could alternatively be used to determine which patterns are most effective on users. With personal accounts, this efficacy could even be identified at the individual level. In a worst-case scenario, these changes could allow tailoring of dark game design patterns to specific people.

Considering these unethical use cases, even if unrealistic, extra care should be taken in future versions of the prototype to ensure it continues to protect players rather than expose them. To prevent other parties from recreating this game and utilising its features for nefarious purposes, one might file either a copyright or a patent for *Under Grove*. However, software patents are currently a contentious topic [59], and the ability to do so varies across the world. In Europe, attaining a software patent for *Under Grove* would not be possible because it does not meet the requirements of an invention [60]. Although the software would be eligible for copyright, copyright may not necessarily protect another party from creating similar functionality [61] . Considering these obstacles in legal protection, preventing unethical applications of *Under Grove* would require alternative means, such as actively advocating against any malicious competitors.

7.4 Contributions and Final Remarks

This study contributes a new procedural generation algorithm. This algorithm generates multi-room levels based on two main equations (Eqs. 4.1 and 4.2). One helps determine the size and shape of a room, and the other helps determine if a room will extend into another room. After all rooms are generated, they are populated with various objects in a sequential order. Because the data model makes use of interfaces and is independent of any game assets, it can serve as the foundation for a variety of different games.

In this study, I used this data model for the prototype game *Under Grove*, which also features procedurally generated dark design elements. The prototype shows promise in educating users on dark game design patterns. However, this study's evaluation could not support its hypothesis that playing the prototype will increase players' familiarity with these patterns. Based on user feedback and user responses, players had some understanding of these patterns, but did not fully connect these ideas with the term *dark game design pattern*. Additional problems surrounding the data collection likely impacted the results from this study. Thus, in order to determine if the hypothesis can be supported, the study should be repeated with improved educational components, a larger study sample, and improved data collection methods.

The prevalence of dark game design patterns in the games industry presents an ethical debate for the future of video game development. *Under Grove* proposes a foundation for educational methods that can be used to combat these patterns by enlightening both players and designers on their use. Future enhancements and experiments can help further solidify the game as a powerful educational tool, potentially influencing formal education for game designers. If *Under Grove* is able to gain any popularity, it may even be able to increase notoriety for dark game design patterns and provide a way to influence regulations surrounding their use. In these ways, *Under Grove* serves as a proactive solution for the prevalence of dark game design patterns.

A. Supplemental Figures

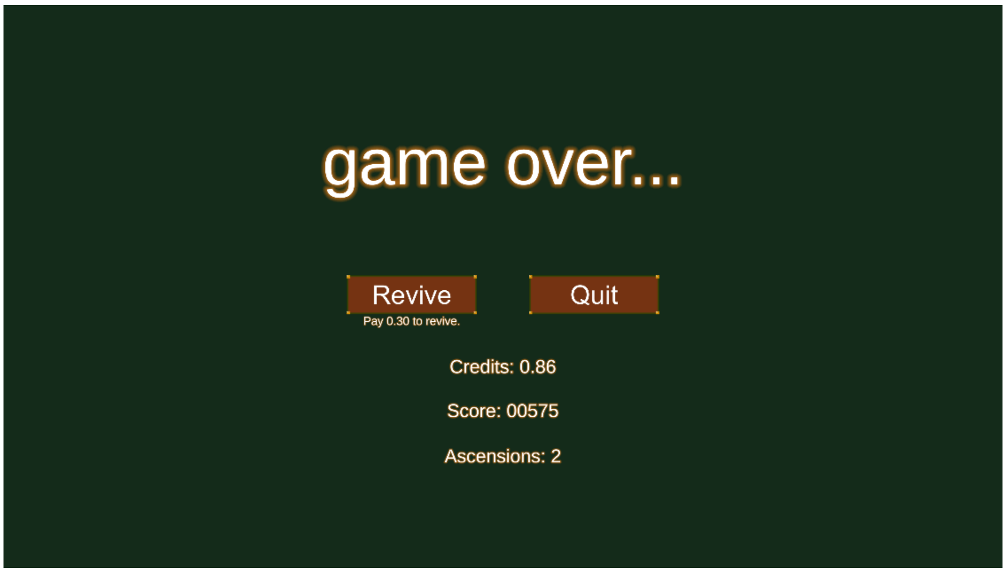


Figure A.1: The Game Over scene allows players to spend credits to continue playing the game or return to the game's Home scene.

Setting Value	Encoded Character
0	Q
1	C
2	x
3	Y
4	E
5	!
6	V
7	z
8	w
9	?
10	M

Table A.1: This table shows the relationship between dark design element values and encoded characters.

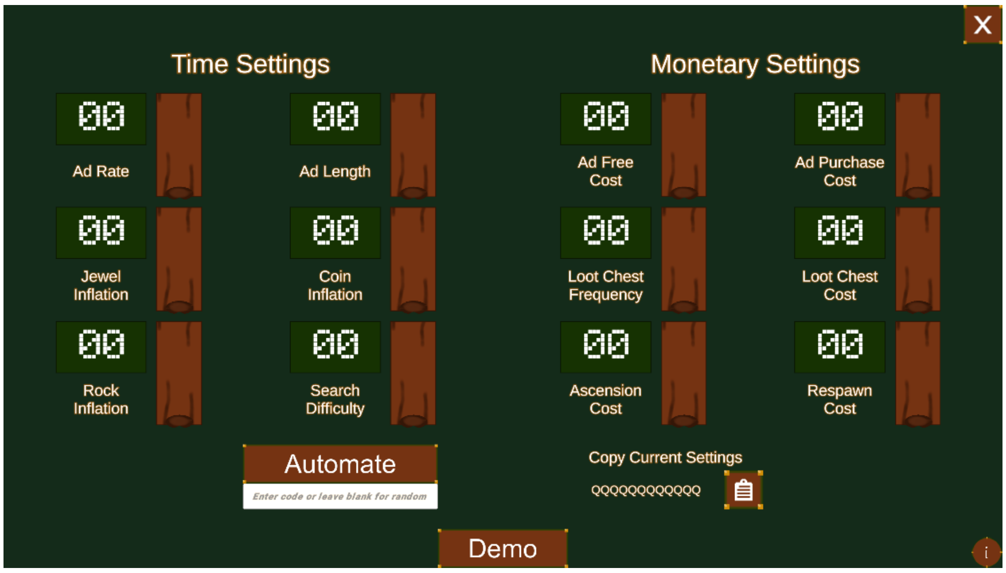


Figure A.2: In Design Mode, players can configure dark design elements. Players can also demo this behaviour or share encoded settings with friends.

```
private void updateColor(Color colorToSet)
{
    particleColour = colorToSet;
    ParticleSystem particleComponent = GetComponent<ParticleSystem>();
    var main = particleComponent.main;
    main.startColor = particleColour;
    var lifetimeColor = particleComponent.colorOverLifetime;
    lifetimeColor.enabled = true;
    lifetimeColor.color = getLifetimeGradient();
}

private Gradient getLifetimeGradient()
{
    Gradient gradient = new Gradient();

    GradientColorKey[] colorKeys = new GradientColorKey[2];
    colorKeys[0].color = particleColour;
    colorKeys[0].time = 0.0f;
    colorKeys[1].color = particleColour;
    colorKeys[1].time = 1.0f;

    GradientAlphaKey[] alphaKeys = new GradientAlphaKey[8];
    alphaKeys[0].time = 0.0f;
    alphaKeys[0].alpha = 0.0f;
    alphaKeys[1].time = 0.1f;
    alphaKeys[1].alpha = 0.0f;
    alphaKeys[2].time = 0.25f;
    alphaKeys[2].alpha = 1.0f;
    alphaKeys[3].time = 0.4f;
    alphaKeys[3].alpha = 0.0f;
    alphaKeys[4].time = 0.6f;
    alphaKeys[4].alpha = 0.0f;
    alphaKeys[5].time = 0.75f;
    alphaKeys[5].alpha = 1.0f;
    alphaKeys[6].time = 0.9f;
    alphaKeys[6].alpha = 0.0f;
    alphaKeys[7].time = 1.0f;
    alphaKeys[7].alpha = 0.0f;

    gradient.SetKeys(colorKeys, alphaKeys);
    return gradient;
}
```

Listing A.1: This code shows how fireflies are given different colours based on a given colour.

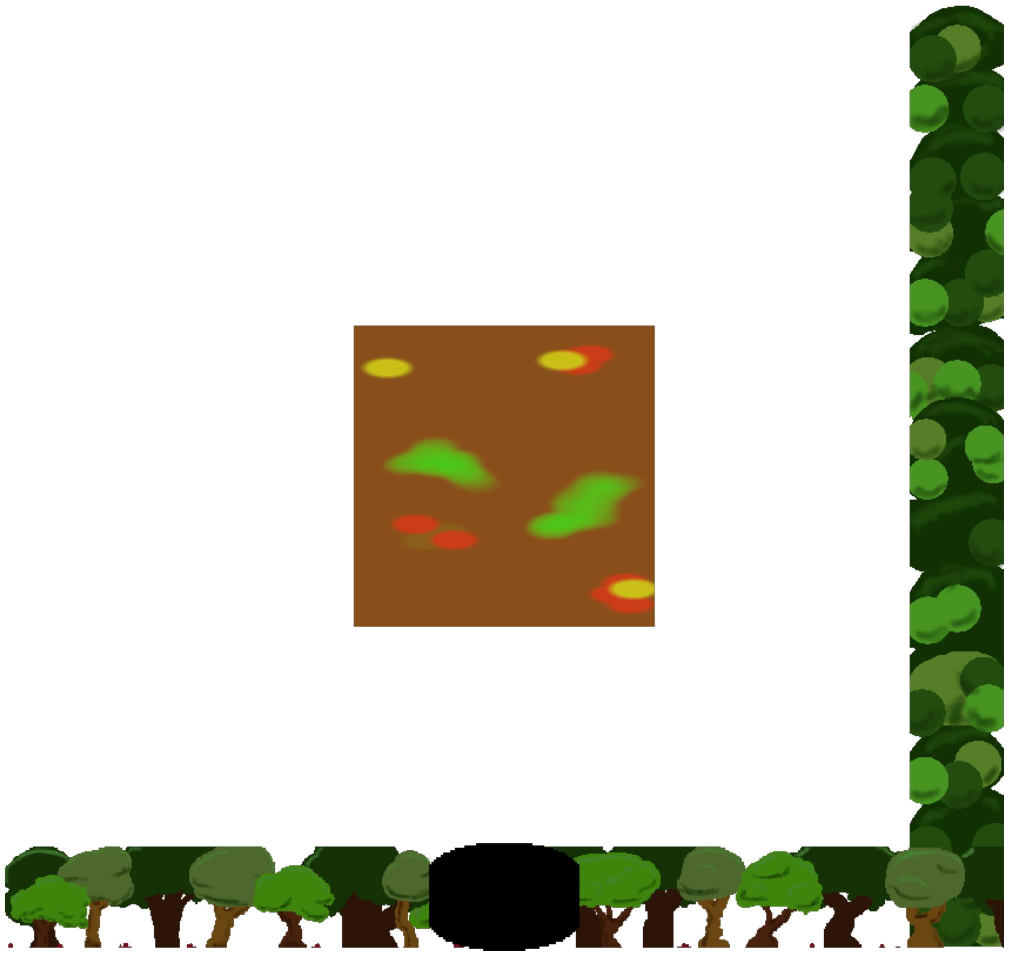


Figure A.3: Rooms are composed using floor, wall, and door sprites. Floor sprites are repeated over a designated size using a tiling effect. Horizontal and vertical walls each have their own sprite. Door sprites are black and represent a gap in a wall and are placed in the center of wall sprites.

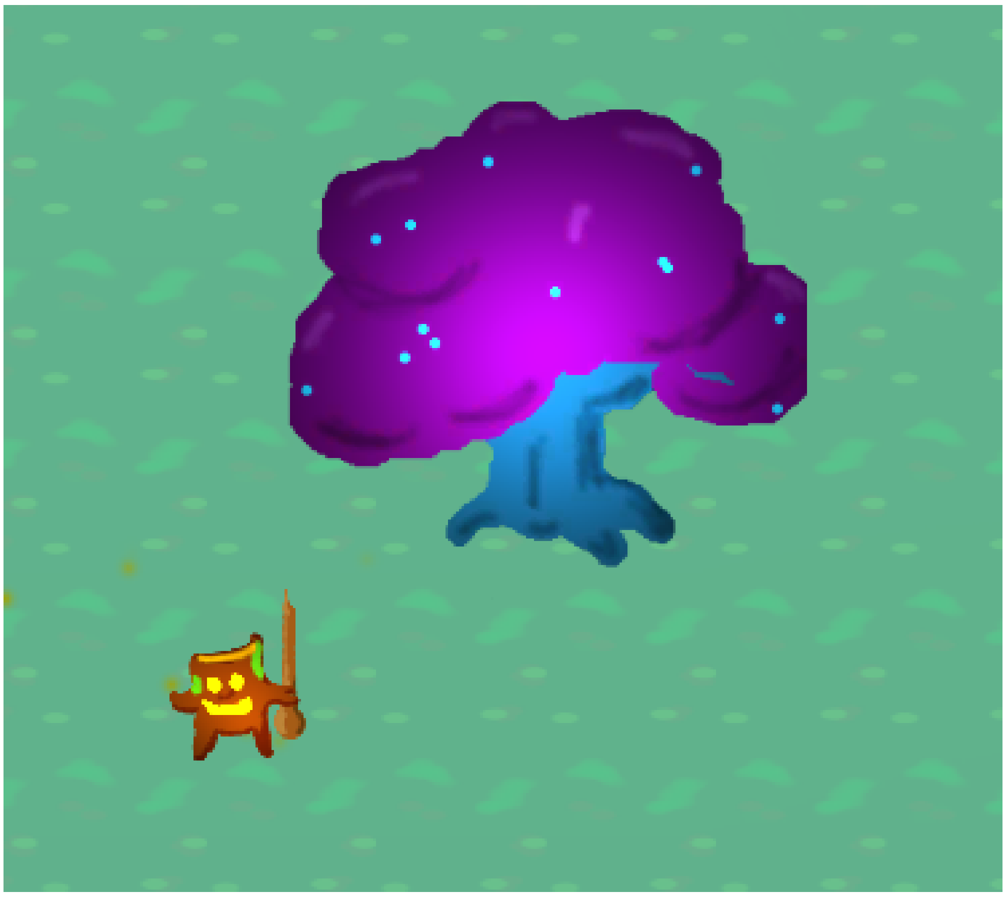
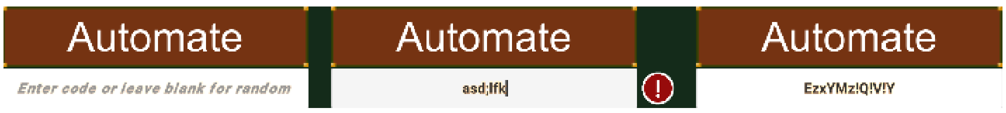


Figure A.4: The SpiriTree room is coloured cyan to identify this room as having a different function than all other rooms. This room can have no features, such as monsters or objects, other than the SpiriTree.



Enter code or leave blank for random

asd;lfk

EzxYMz!Q!V!Y

Figure A.5: The Automate section allows users to set all dark design elements at once. If empty, the button will randomly set all elements (left). If users input an incorrect setting string, an warning icon will appear (middle). If a valid string is entered, the string will be parsed, and each element will be set to the corresponding value (right).

B. Surveys

B.1 *Dungeon Peddlers* Wireframe Interview

Hello,

Today I'll be reviewing part of a prototype for a game that is currently in development.

For research purposes, do I have your consent to collect some basic information from you including birthdate, gender, and information on your gaming habits before we being?

Interviewee ID: Interviewee ID
Birthdate: Click or tap to enter a date.

Brief gaming habit survey:

1. How often do you play games?
   Choose an item.
2. What genres of games do you enjoy?
   Click or tap here to enter text.
3. What aspects of games do you enjoy?
   Click or tap here to enter text.
4. What aspects of games do you dislike?
   Click or tap here to enter text.
5. Have you played a level-editor game before?
   Choose an item.

If yes:

6. What did enjoy about these games?
   Click or tap here to enter text.
7. What did you dislike about these games?
   Click or tap here to enter text.
8. Have you ever made in-game purchases?
   Choose an item.

Before we begin, I'll provide a description of the game for context. This prototype is for a level-editor game where users will be able to create their own custom levels or play custom levels created by other users. The dichotomy between different types of users is intended to foster a supply and demand relationship between designers and players.

All levels will be in the form of 'dungeons', which are top-down and two-dimensional. One can think of dungeons as something similar to blueprints. The player will have a bird's eye view of a character as they navigate through the various rooms of the level. The gameplay will consist of adventure, combat, and puzzle elements. For each level, a player must reach a goal area in order to complete the dungeon and claim their reward.

In addition to designing and completing levels, players will also compete to accumulate the most in-game money, which will be tracked and viewable in a community leaderboard. Designers can earn money from players who pay an admission fee to attempt a level. They can also generate income by employing various in-dungeon mechanisms. For example, designers can implement treasure chests that players pay to open with in-game currency for the chance to make the level easier. Players, on the other hand, can earn money by completing a level. Earned rewards can be used to play more levels or improve the equipment of the player's character, thereby making it easier to complete more difficult, profitable levels.

Do you have any questions about what the game is or how it works?

Click or tap here to enter text.

In this version of the prototype, we'll be reviewing the basic UI and usability for the main game screens. During this portion of the review, I'll provide relevant context to pages and action if necessary, but otherwise you will be able to guide the navigation of the prototype. While reviewing the prototype, please provide any feedback that comes to mind. I will ask questions along the way to get feedback on certain elements.

Do you have any questions before we begin?

Click or tap here to enter text.

Login Screen

Initial feedback

Click or tap here to enter text.

1. Do you feel any information is missing from this screen?

Click or tap here to enter text.

Login information

2. How do you feel about the use of usernames over emails?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

Registration information

3. Would be willing to provide this information to register an account for a game?

Choose an item.

Feedback

Click or tap here to enter text.

Additional feedback

Click or tap here to enter text.

Home Screen

Initial feedback

Click or tap here to enter text.

4. How do you feel about the layout of the buttons?

Click or tap here to enter text.

Account button

Menu items --> under construction

5. Would you expect to see other options here?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

Information button

Feedback

Click or tap here to enter text.

Additional feedback

Click or tap here to enter text.

Leaderboards

6. What type of leaderboards would you expect to see for this type of game?

Click or tap here to enter text.

7. What time periods for rankings would you like to see for this type of game?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

Play Mode Home Screen

Initial feedback

Click or tap here to enter text.

Store

8. Do you think you would be able to find and purchase desired goods?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

Selected level

Feedback

Click or tap here to enter text.

Account button

9. Would you expect to see anything different in this menu here?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

Information button

Feedback

Click or tap here to enter text.

10. How do you feel about the use of popup windows for these buttons? How would you feel if these windows had their own dedicated screens?

Click or tap here to enter text.

11. Which areas of this screen are clear/easy-to-understand?

Click or tap here to enter text.

12. Which areas of this screen are unclear/confusing?

Click or tap here to enter text.

Later feedback

Click or tap here to enter text.

Level Search Screen

13. Would you expect to see any other information in your search?

Click or tap here to enter text.

14. Is the ability to use advanced search features, such as finding levels by certain users, important to you?

Choose an item.

Feedback

Click or tap here to enter text.

Design Mode Home Screen

Initial feedback

Click or tap here to enter text.

New Dungeon --> under construction

Information button

Feedback

Click or tap here to enter text.

Account button

15. Would you expect to see anything different in this menu compared to the home screen?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

Level details editor

16. Would having the ability to change the admission price of a level improve your gameplay experience?

Choose an item.

Feedback

Click or tap here to enter text.

17. Which areas of this screen are clear/easy-to-understand?

Click or tap here to enter text.

18. Which areas of this screen are unclear/confusing?

Click or tap here to enter text.

19. Would it be important to you to have full editing capabilities of levels after they have been created?

Choose an item.

Additional feedback

Click or tap here to enter text.

Overall

20. Which section of the game was the clearest?

Click or tap here to enter text.

21. Which section the game was the least clear?

Click or tap here to enter text.

Feedback

Click or tap here to enter text.

B.2 *Dungeon Peddlers* Opportunistic Evaluation Interview

Hello,

Today I'll be reviewing part of a prototype for a game that is currently in development.

For research purposes, do I have your consent to collect some basic information from you including birthdate, gender, and information on your gaming habits before we being?

Interviewee ID: Interviewee ID
Gender: Click or tap here to enter text.

Brief gaming habit survey:

1. How often do you play games?
   Choose an item.
2. What genres of games do you enjoy?
   Click or tap here to enter text.
3. What aspects of games do you enjoy?
   Click or tap here to enter text.
0. What aspects of games do you dislike?
   Click or tap here to enter text.
0. Have you played a level-editor game before?
   Choose an item.

If yes:

1. What did enjoy about these games?
   Click or tap here to enter text.
0. What did you dislike about these games?
   Click or tap here to enter text.
1. Have you ever made in-game purchases?
   Choose an item.

Before we begin, I'll provide a description of the game for context. This prototype is for a level-editor game where users will be able to create their own custom levels or play custom levels created by other users. The dichotomy between different types of users is intended to foster a supply and demand relationship between designers and players.

All levels will be in the form of 'dungeons', which are top-down and two-dimensional. One can think of dungeons as something similar to blueprints. The player will have a bird's eye view of a character as they navigate through the various rooms of the level. The gameplay will consist of adventure, combat, and puzzle elements. For each level, a player must reach a goal area in order to complete the dungeon and claim their reward.

In addition to designing and completing levels, players will also compete to accumulate the most in-game money, which will be tracked and viewable in a community leaderboard. Designers can earn money from players who pay an admission fee to attempt a level. They can also generate income by employing various in-dungeon mechanisms. For example, designers can implement treasure chests that players pay to open with in-game currency for the chance to make the level easier. Players, on the other hand, can earn money by completing a level. Earned rewards can be used to play more levels or improve the equipment of the player's character, thereby making it easier to complete more difficult, profitable levels.

Do you have any questions about what the game is or how it works?

Click or tap here to enter text.

In this version of the prototype, we'll be reviewing the dungeon editor aspect of the game. I am providing a gridded sheet of paper and a pencil on which you can design and draw your own dungeon. You will need to satisfy the following rules:

- The dungeon must consist of at least one room
- The player’s start position must be marked with the word *start*
- The dungeon end point must be marked with the word *end*

Outside of these rules, you can customise your dungeon however you want. In the planned version of the game, designers can include elements such as enemies, puzzles, etc. If you need ideas for what to add, the paper on the right is a preliminary representation of a menu for adding dungeon components. If using these, you can use the corresponding number and symbol in your drawing.

When designing your dungeon, consider the goal of the game, which is to make money. One way to earn money is from the admission fee of a level, but designers can incorporate other strategies. If you’re not sure how to add these features to your dungeon, I can give you examples.

As you work on your drawing, you can think aloud and let me know your process.

Do you have any questions before we begin this part of the interview?

Click or tap here to enter text.

Notes from the design session:

Click or tap here to enter text.

Exploratory Questions

1. Do you feel any information is missing from this screen?

Click or tap here to enter text.

2. Is the ability to change the shape of the room important?

Choose an item.

3. If playing the game, would you use a feature that generates a random dungeon within certain constraints?

Choose an item.

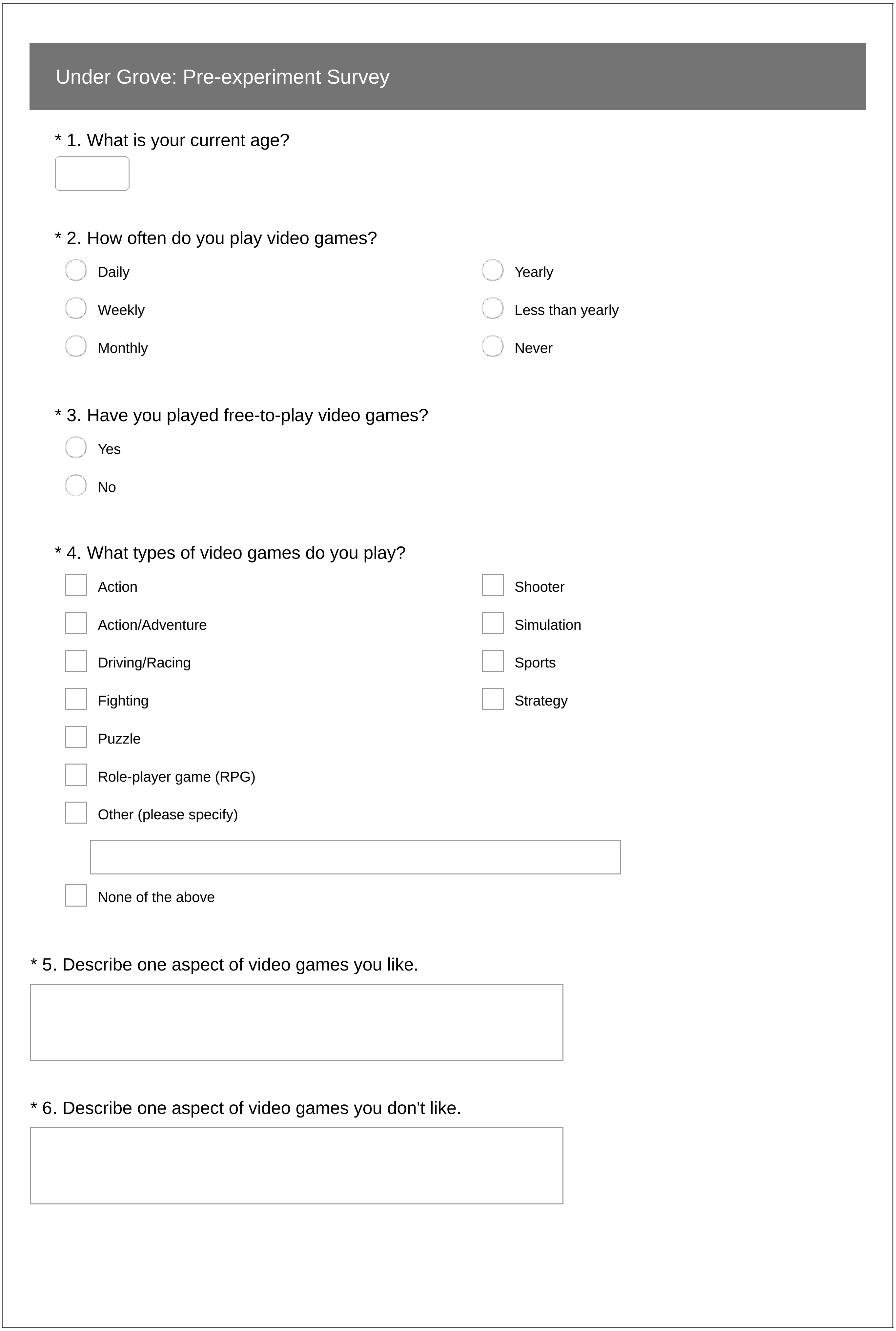
4. Would you rather the limit to dungeon size be limited to a certain number of components or be based solely on the designers budget?

Choose an item.

5. If you were designing a dungeon, would the ability to change the admission fee of a custom dungeon be important?

Choose an item.

B.3 Longitudinal Pre-experiment Survey



Under Grove: Pre-experiment Survey

* 1. What is your current age?

* 2. How often do you play video games?

Daily	Yearly
Weekly	Less than yearly
Monthly	Never

* 3. Have you played free-to-play video games?

- Yes
- No

* 4. What types of video games do you play?

Action	Shooter
Action/Adventure	Simulation
Driving/Racing	Sports
Fighting	Strategy
Puzzle	
Role-player game (RPG)	
Other (please specify)	
None of the above	

* 5. Describe one aspect of video games you like.

* 6. Describe one aspect of video games you don't like.

* 7. Describe one factor that helps you decide whether you play a game or not.

* 8. Describe one factor that helps you decide whether you purchase a game or not.

* 9. Which types of in-game purchases have you made with real money?

- [ ] Content which adds new story elements or levels
- [ ] Content which adds aesthetic/visual changes
- [ ] In-game currency
- [ ] In-game bonuses or perks to make the game easier
- [ ] Other (please specify)
- [ ] None of the above
- [ ] An ad-free version of a game
- [ ] Option to bypass difficult areas or aspects of a game

* 10. How much do you agree with the following statement: I am familiar with dark-design patterns in video games?

Strongly Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Strongly Agree
○	○	○	○	○

11. Provide up to three examples of dark design patterns in video games. Leave blank if unsure.

Example 1

Example 2

Example 3

* 12. How much do you agree with the following statement: I am familiar with designing video games?

Strongly Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Strongly Agree
○	○	○	○	○

* 13. If designing a video game, what methods would you use to make the game enjoyable?

* 14. If designing a video game, what methods would you use to make the game profitable?

B.4 Longitudinal Post-experiment Survey

Under Grove: Post-experiment Survey

* 1. What is your current age?

* 2. How often do you play video games?

Daily	Yearly
Weekly	Less than yearly
Monthly	Never

* 3. Have you played free-to-play video games?

- Yes
- No

* 4. What types of video games do you play?

Action	Shooter
Action/Adventure	Simulation
Driving/Racing	Sports
Fighting	Strategy
Puzzle	
Role-player game (RPG)	
Other (please specify)	
None of the above	

* 5. How much do you agree with the following statement: I am familiar with dark-design patterns in video games?

Strongly Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Strongly Agree

6. Provide up to three examples of dark design patterns in video games. Leave blank if unsure.

Example 1

Example 2

Example 3

* 7. How much do you agree with the following statement: I am familiar with designing video games?

Strongly Disagree	Somewhat Disagree	Neither Agree nor Disagree	Somewhat Agree	Strongly Agree
○	○	○	○	○

* 8. On day 3, you were asked to create a level that would keep players playing the longest. What was your rationale for your level design?

* 9. On day 4, you were asked to create a level that would make players spend the most money. What was your rationale for your level design?

* 10. While playing Under Grove in Play Mode (not in Design Mode), did you make any in-game purchases?

○ Yes

○ No

* 11. Please rate how satisfied or dissatisfied you were with each of the following aspects in Under Grove.

	Very Dissatisfied	Somewhat Dissatisfied	Neither Satisfied nor Dissatisfied	Somewhat Satisfied	Very Satisfied
Overall Enjoyability	○	○	○	○	○
Gameplay	○	○	○	○	○
User Interface (buttons and menus)	○	○	○	○	○
Visual Appeal/Style	○	○	○	○	○
Play Mode Overall	○	○	○	○	○
Design Mode Overall	○	○	○	○	○
Educational Elements	○	○	○	○	○

* 12. Please describe one aspect of Under Grove's Play Mode that you would change.

* 13. Please describe one aspect of Under Grove's Play Mode that you enjoyed.

* 14. Please describe one aspect of Under Grove's Design Mode that you would change.

* 15. Please describe one aspect of Under Grove's Design Mode that you enjoyed.

Bibliography

[1] T. Wijman, *"global games market to generate* 175.8 *billion in 2021; despite a slight decline, the market is on track to surpass* 200 *billion in 2023"*, 2021.

[2] B. Brathwaite and I. Schreiber, *Challenges for game designers.* Course Technology/Cengage Learning Boston, Massachusetts, 2009.

[3] S. N. Baharom, W. H. Tan, M. Z. Idris, *et al.*, "Emotional design for games: A framework for player-centric approach in the game design process," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 9, no. 10, pp. 387–398, 2014.

[4] J. P. Zagal, S. Björk, and C. Lewis, "Dark patterns in the design of games," 2013.

[5] DarkPattern.games, *Dark gaming patterns.* [Online]. Available: https://www.darkpattern.games/patterns.php (visited on 09/01/2021).

[6] L. Y. Xiao, "A primer on the legal regulation of loot boxes: History, business, psychology, law and regulation," 2019.

[7] L. Y. Xiao, L. L. Henderson, Y. Yang, and P. W. Newall, *Gaming the system: Legally required loot box probability disclosures in video games in china are implemented sub-optimally*, 2021.

[8] O. Király, M. D. Griffiths, and Z. Demetrovics, "Internet gaming disorder and the dsm-5: Conceptualization, debates, and controversies," *Current Addiction Reports*, vol. 2, no. 3, pp. 254–262, 2015.

[9] A. J. Van Rooij, G.-J. Meerkerk, T. M. Schoenmakers, M. Griffiths, and D. Van de Mheen, *Video game addiction and social responsibility*, 2010.

[10] K. Rapeepisarn, K. W. Wong, C. C. Fung, and A. Depickere, "Similarities and differences between" learn through play" and" edutainment"," 2006.

[11] J. Sykes and M. Federoff, "Player-centred game design," in *CHI '06 Extended Abstracts on Human Factors in Computing Systems.* New York, NY, USA: Association for Computing Machinery, 2006, pp. 1731–1734, ISBN: 1595932984. [Online]. Available: https://doi.org/10.1145/1125451.1125774.

[12] C. Köknar, "Who is at the center?: Designing playful experiences by using player-centered approach," in *International Conference on Human-Computer Interaction*, Springer, 2019, pp. 11–21.

[13] J. L. G. Sánchez, N. P. Zea, and F. L. Gutiérrez, "From usability to playability: Introduction to player-centred video game development process," in *International Conference on Human Centered Design*, Springer, 2009, pp. 65–74.

[14] N. Sebe, "Human-centered computing," in *Handbook of ambient intelligence and smart environments*, Springer, 2010, pp. 349–370.

[15] D. A. Norman, "Human-centered design considered harmful," *interactions*, vol. 12, no. 4, pp. 14–19, 2005.

[16] E. Morrell, "Player decentered design," M.S. thesis, 2020.

[17] J. Kücklich and M. C. Fellow, "Play and playability as key concepts in new media studies," *STeM Centre, Dublin City University*, 2004.

[18] H. Desurvire, M. Caplan, and J. A. Toth, "Using heuristics to evaluate the playability of games," in *CHI'04 extended abstracts on Human factors in computing systems*, 2004, pp. 1509–1512.

[19] A. Mathur, M. Kshirsagar, and J. Mayer, "What makes a dark pattern... dark? design attributes, normative considerations, and measurement methods," in *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 2021, pp. 1–18.

[20] C. S. Deterding, J. Stenros, and M. Montola, "Against" dark game design patterns"," in *DiGRA'20-Abstract Proceedings of the 2020 DiGRA International Conference*, York, 2020.

[21] O. Király, M. D. Griffiths, D. L. King, H.-K. Lee, S.-Y. Lee, F. Bányai, Á. Zsila, Z. K. Takacs, and Z. Demetrovics, "Policy responses to problematic video game use: A systematic review of current measures and future possibilities," *Journal of Behavioral Addictions*, vol. 7, no. 3, pp. 503–517, 2018.

[22] S.-L. Chang, C.-Y. Chen, *et al.*, "An exploration of the tendency to online game addiction due to user's liking of design features," *Asian journal of health and information sciences*, vol. 3, no. 1-4, pp. 38–51, 2008.

[23] D. L. King, P. H. Delfabbro, and M. D. Griffiths, "The role of structural characteristics in problematic video game play: An empirical study," *International Journal of Mental Health and Addiction*, vol. 9, no. 3, pp. 320–333, 2011.

[24] L. Elliott, G. Ream, E. McGinsky, and E. Dunlap, "The contribution of game genre and other use patterns to problem video game play among adult video gamers," *International journal of mental health and addiction*, vol. 10, no. 6, pp. 948–969, 2012.

[25] G. J. Entwistle, A. Blaszczynski, and S. M. Gainsbury, "Are video games intrinsically addictive? an international online survey," *Computers in Human Behavior*, vol. 112, p. 106 464, 2020.

[26] R. T. Wood, M. D. Griffiths, and A. Parke, "Experiences of time loss among videogame players: An empirical study," *Cyberpsychology & behavior*, vol. 10, no. 1, pp. 38–44, 2007.

[27] C. Osathanunkul, "A classification of business models in video game industry," *International Journal of Management Cases*, vol. 17, no. 1, pp. 35–44, 2015.

[28] P. Bach and C. Jordon, "At a crossroads: Video game addiction," *XRDS*, vol. 13, no. 2, p. 2, Dec. 2006, ISSN: 1528-4972. DOI: 10.1145/1217728.1217730. [Online]. Available: https://doi.org/10.1145/1217728.1217730.

[29] E. J. Jeong, D. J. Kim, and D. M. Lee, "Game addiction from psychosocial health perspective," in *Proceedings of the 17th International Conference on Electronic Commerce 2015*, ser. ICEC '15, Seoul, Republic of Korea: Association for Computing Machinery, 2015, ISBN: 9781450334617. DOI: 10.1145/2781562.2781587. [Online]. Available: https://doi.org/10.1145/2781562.2781587.

[30] K. A. Harrigan, K. Collins, M. J. Dixon, and J. Fugelsang, "Addictive gameplay: What casual game designers can learn from slot machine research," in *Proceedings of the International Academic Conference on the Future of Game Design and Technology*, ser. Futureplay '10, Vancouver, British Columbia, Canada: Association for Computing Machinery, 2010, pp. 127–133, ISBN: 9781450302357. DOI: 10.1145/1920778.1920796. [Online]. Available: https://doi.org/10.1145/1920778.1920796.

[31] S. A. Goodstein, "When the cat's away: Techlash, loot boxes, and regulating" dark patterns" in the video game industry's monetization strategies," *U. Colo. L. Rev.*, vol. 92, p. 285, 2021.

[32] K. Boghe, L. Herrewijn, F. De Grove, K. Van Gaeveren, and L. De Marez, "Exploring the effect of in-game purchases on mobile game use with smartphone trace data," *Media and Communication*, vol. 8, no. 3, pp. 219–230, 2020.

[33] M. Chromik, M. Eiband, S. T. Völkel, and D. Buschek, "Dark patterns of explainability, transparency, and user control for intelligent systems.," in *IUI workshops*, vol. 2327, 2019.

[34] T. H. Soe, O. E. Nordberg, F. Guribye, and M. Slavkovik, “Circumvention by design - dark patterns in cookie consent for online news outlets,” in *Proceedings of the 11th Nordic Conference on Human-Computer Interaction: Shaping Experiences, Shaping Society*, ser. NordiCHI ’20, Tallinn, Estonia: Association for Computing Machinery, 2020, ISBN: 9781450375795. DOI: 10.1145/3419249.3420132. [Online]. Available: https://doi.org/10.1145/3419249.3420132.

[35] M. Nouwens, I. Liccardi, M. Veale, D. Karger, and L. Kagal, “Dark patterns after the gdpr: Scraping consent pop-ups and demonstrating their influence,” in *Proceedings of the 2020 CHI conference on human factors in computing systems*, 2020, pp. 1–13.

[36] R. M. Gil and J. Arnedo-Moreno, “Designing an activity to help reflect on ”healthy engagement vs video game addiction”,” in *Eighth International Conference on Technological Ecosystems for Enhancing Multiculturality*, ser. TEEM’20, Salamanca, Spain: Association for Computing Machinery, 2020, pp. 682–687, ISBN: 9781450388504. DOI: 10.1145/3434780.3436607. [Online]. Available: https://doi.org/10.1145/3434780.3436607.

[37] J. Williams, “Faulting san andreas: The call to arms for sensible regulation of violent video games,” *Hastings Comm. & Ent. LJ*, vol. 29, p. 121, 2006.

[38] E. Eskelinen, “Dark design patterns: What are the next steps towards more ethically designed digital products and services?,” 2021.

[39] J. Magnusson, “Improving dark pattern literacy of end users,”

[40] C. Lewis, “Motivational design patterns,” Ph.D. dissertation, UC Santa Cruz, 2013.

[41] A. Sullivan and G. Smith, “Lessons in teaching game design,” in *Proceedings of the 6th International Conference on Foundations of Digital Games*, 2011, pp. 307–309.

[42] R. Sefelin, M. Tscheligi, and V. Giller, “Paper prototyping - what is it good for? a comparison of paper- and computer-based low-fidelity prototyping,” in *CHI ’03 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA ’03, Ft. Lauderdale, Florida, USA: Association for Computing Machinery, 2003, pp. 778–779, ISBN: 1581136374. DOI: 10.1145/765891.765986. [Online]. Available: https://doi.org/10.1145/765891.765986.

[43] H. Sharp, Y. Rogers, and J. Preece, *Interaction Design: Beyond Human Computer Interaction*. Hoboken, NJ, USA: John Wiley amp; Sons, Inc., 2007, ISBN: 0470018666.

[44] J. Bycer, *Procedural vs. randomly generated content in game design*, Aug. 2015. [Online]. Available: https://www.gamedeveloper.com/design/procedural-vs-randomly-generated-content-in-game-design.

[45] S. N. Rai, “Status and cultivation of sandalwood in india,” in *In: Hamilton, Lawrence; Conrad, C. Eugene, technical coordinators. Proceedings of the Symposium on Sandalwood in the Pacific; April 9-11, 1990; Honolulu, Hawaii. Gen. Tech. Rep. PSW-GTR-122. Berkeley, CA: Pacific Southwest Research Station, Forest Service, US Department of Agriculture: p. 66-71*, vol. 122, 1990.

[46] N. A. Blouin and C. E. Lane, “Red algal parasites: Models for a life history evolution that leaves photosynthesis behind again and again,” *Bioessays*, vol. 34, no. 3, pp. 226–235, 2012.

[47] A. Dots, *Designing game controls*, Mar. 2017. [Online]. Available: https://www.gamedeveloper.com/disciplines/designing-game-controls.

[48] A. Thorn, *Game engine design and implementation*. Jones & Bartlett Publishers, 2011.

[49] L. Caballero, *An introduction to webgl - part 1*, Oct. 2011. [Online]. Available: https://dev.opera.com/articles/introduction-to-webgl-part-1/.

[50] Unity Technologies, *Input*. [Online]. Available: https://docs.unity3d.com/ScriptReference/Input.html (visited on 09/08/2021).

[51] ——, *Object.dontdestroyonload*. [Online]. Available: https://docs.unity3d.com/ScriptReference/Object.DontDestroyOnLoad.html (visited on 09/08/2021).

[52] ——, *Gradient*. [Online]. Available: https://docs.unity3d.com/ScriptReference/Gradient.html (visited on 09/08/2021).

[53] M. Dixon, P. Ghezzi, C. Lyons, and G. Wilson, *Gambling: Behavior theory, research, and application*. New Harbinger Publications, 2006.

[54] A. Gellel and P. Sweetser, “A hybrid approach to procedural generation of roguelike video game levels,” in *International Conference on the Foundations of Digital Games*, 2020, pp. 1–10.

[55] W. Forsyth, “Globalized random procedural content for dungeon generation,” *Journal of Computing Sciences in Colleges*, vol. 32, no. 2, pp. 192–201, 2016.

[56] D. L. King, A. M. Russell, P. H. Delfabbro, and D. Polisena, “Fortnite microtransaction spending was associated with peers’ purchasing behaviors but not gaming disorder symptoms,” *Addictive behaviors*, vol. 104, p. 106 311, 2020.

[57] A. E. Brandt and J. Martin, “Simulating personal wealth in the laboratory,” *The Journal of general psychology*, vol. 142, no. 3, pp. 167–181, 2015.

[58] R. Garland, “The mid-point on a rating scale: Is it desirable,” *Marketing bulletin*, vol. 2, no. 1, pp. 66–70, 1991.

[59] D. Yang, “Software protection: Copyrightability vs patentability?,” 2012.

[60] European Patent Office, *Article 52 – patentable inventions*. [Online]. Available: https://www.epo.org/law-practice/legal-texts/html/epc/2020/e/ar52.html (visited on 09/10/2021).

[61] G. Fu, “Google v. oracle: Weighing fair use factors in software copyright infringement cases,” *BC INTELL. PROP. & TECH. F.*, vol. 2020, pp. 1–4, 2020.