2018

Plymouth University

School of Computing, Electronics and Mathematics

PRCO304

Final Stage Computing Project

BSc (Hons) Computing & Games Development

B & F HATE MAZES
(S) BOGDAN STOICA

Acknowledgements

Abstract

Table of Contents

Introduction

Background

Project Background

This project does not have a client at the time of its development. The purpose is to present a game demo based on a maze generation tool. My personal interest and fascination with mazes and Labyrinths is the primary inspiration that has led to this tools development. Although the number of available hours and resources would not allow for the development of a project ready for public release, the MVP for this module have been set to allow for a god understanding of the tools features and game design potential. Creating a releasable game on the mobile market was the initial goal as previously mentioned and led to the initial research phase that helped in designing this demo using the generation tool.

There are many maze enthusiasts as seen by the primary documentation resource used in the development ("Mazes for Programmers") and consequently many already existing games present on the market. B & F hate mazes is a game whose focus is maintaining a claustrophobic atmosphere and spatial confusion, commonly encountered by people attempting to solve Labyrinths, while allowing for infinite replayability. These main features set this prototype aside from current maze games. More information regarding competition can be viewed in the *Market analysis* section of this report.

Project Aim

The aim of this project is to successfully use the generation tool to develop a prototype of a game which can be later released on both the Play Store and AppStore. Another aim is to modify the tool in such a manner that it can be used as a library for other platforms and projects.

As previously mentioned there are many people passionate about mazes and many users on the mobile platform that also enjoy the challenge of a maze. On the Android platform just by searching maze you are presented with many games and apps where only the first 10 accumulated over 25 million downloads. These numbers help justify the idea of building a good quality product for passionate users. The demographic for such a project however cannot be set since the low entry difficulty present in such games together with non-violent imagery can allow for various users of all age groups to participate.

Objectives

- Research market and design a game with a maze puzzle as its core mechanic.
- Create a maze generation tool that allows randomization and design customization.
- Develop a demo based on the developed tool.
- Acquire user feedback of the gameplay and overall game feel concerning overall atmosphere and controls.
- Polish the title accordingly to the provided user feedback.

Minimum Requirements

- Develop a maze generation tool
- Demo game on PC using the generation tool
- Create a new maze for each new game

Research

Programming for Mazes has been the primary research resource used in the development of the project. There are many different types of mazes and rules in creating them from a mathematical point of view. When developing the tool that is set as the core of the project there have been various factors such as efficiency, game design and maze features that have been taken into consideration during both the development and research phases of the project. More information showcasing the reasoning behind the choices made in this project will be available in the research segment of the Project.

Mazes vs Labyrinths

Although the initial inspiration for the project had risen from my knowledge of labyrinths, it slowly transitioned into the appreciation of the Mazes throughout closely studying its properties and features. These two series of pathways although similar in nature and commonly mistaken for each other are very different in some key features that make them unique. Both mazes and labyrinths are series of pathways as previously mentioned, however the difference between them is as follows: A labyrinth has a single route containing twists and turns whereas a maze has many branches and offers various path choices while also presenting the user with dead ends. A second key difference between these similar concepts is the difference in difficulty and creation mindset. Because a labyrinth is built with a set start and ending point separated by a single path, its difficulty is significantly lower than the one of a maze. Labyrinths can be long and can require time to complete based on its size but its unicursal design makes it more of an attraction and symbolic construction rather than a puzzle. In some cultures, labyrinths symbolize the path to God. Mazes however are designed and with a different purpose. They are created as puzzles of various difficulties in many environments including research and game development. Most of the experiments involving mazes are testing the special awareness of the test subject. Because of their core structure, mazes can have various shapes and sizes and allow for more than one creation methods to be used.

Core Concepts

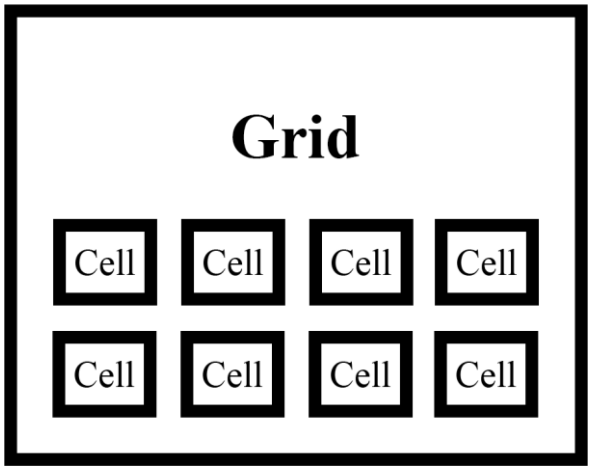
To be able to generate a maze there are a few core functionality traits required. As previously stated a maze is a collection of paths which typically has a starting and ending point. Each maze can be simplified, from a programming point of view, to two simple elements: Cell and Grid. A Cell represents a room in the maze while the Grid is the container of all the cells (See Fig. 1). These traits help the developer keep track of the connections and attributes of the maze allowing for a larger customization possibility.

Each Cell has the following attributes:

- It knows all its neighbours (Northern, Southern, Western and Eastern Cells);
- It knows which cells it is connected to;
- It can link or unlink itself from another Cell;

Each Grid has the following attributes:

- It contains all the Cells of a maze;
- It knows the total number of Cells in each maze;
- It knows the position of each Cell inside the maze;



(*Fig. 1*)

Procedural Generation

Although the tool used for maze generation allows for the same starting and end points to be chosen for each iteration, this would constitute in a poor design choice. The primary reason being the opportunity for the user to slowly learn the position of the exit therefore removing some of the difficulty. Creating a new map for each game with a different starting and end point offers a Stochastic Procedurally Generated Content. This will ensure infinite content for the user with the only difference being the type of maze he encounters based on the used algorithm. All the algorithms used for the assignment version of this project are of a Constructive type. More information regarding how the algorithms function will be discussed later in the paper.

Maze Randomization

There are various types of ways to build mazes, and currently no ideal algorithm to create them. However, each algorithm has unique attributes which can vary from ease of implementation to efficiency. All methods researched for the development of this project use randomization techniques. The purpose of the tool is to allow the possibility for more than one method to be used as well as edit existing mazes before rendering.

Some traits are present across all maze algorithms chosen for this project while others are unique to allow for a more varied experience for the user. The first and most important trait present across all used mazes is the concept of a perfect maze. What this concept means is that a path can be created between any to cells of the maze. Having this characteristic allows for the player to freely navigate through the levels.

A second influential trait is having a biased algorithm for some mazes. A biased algorithm allows for a maze to have similar characteristics that can be identified and learned by the user to allow him to learn how to navigate through a maze. An example of such an algorithm is Binary which will be discussed in the Binary Tree section.

A third characteristic is to have a non-biased algorithm. This trait is as the name suggests the opposite of having it biased therefore not allowing for newly generated mazes to form a repetitive pattern making the solving of the maze more difficult for the player.

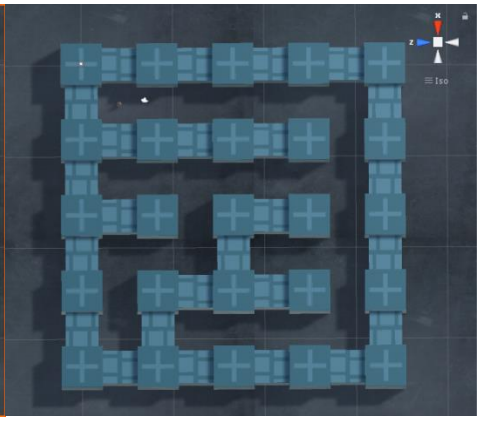
This segment will be briefly presenting some characteristics and observations for each algorithm used as part of the maze generation process, as well as highlight some future potential implementations.

Binary Tree

Binary Tree is possibly one of the simplest available algorithms for generating mazes. The core concept behind creating a maze using Binary is similar to using it in data structures s that have hierarchically arranged values. In this case the Northern and Eastern Neighbour of each Cell represents the two available choices for the maze creator.
This algorithm has been used for testing the wall generating class because there are some general characteristics present in Binary Mazes. The characteristic that distinguishes Binary Mazes from other Mazes is the uninterrupted corridors alongside the top corridor and far right corridor (if you are using North and West as Binary elements).

(*Fig. 2*)

(*Fig. 3*)



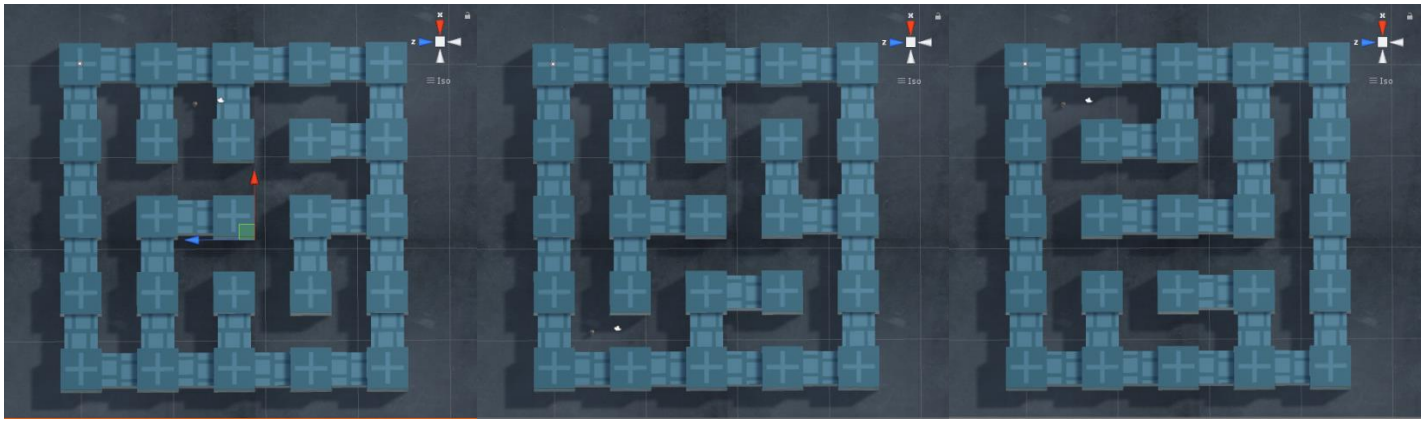
(*Fig. 4*)

Figures two, three and four are four by four sized examples of mazes created using the Binary Tree Algorithm with the Build Maze Class. Although not difficult to implement, there are some characteristics that have a strong impact on the puzzle element of the game. The first one is the tendency for all paths to lead to the top right cell of the maze and the

second one is the absence of dead-ends pointing North and East. Algorithms that are “straight-forward” such as Binary can also called biases. These therefore become easier to learn and solve because the user is able to easily orientate and reach each end much quicker. This can help in creating levels of different difficulty by just choosing which type of algorithm you want to use for your maze.

Aldous-Broder

Aldous-Broder has been independently developed by David Aldous and Andrei Broder and is almost as simple to implement as Binary [7]. Compared to Binary, Aldous-Broder navigates through the Cells in a more chaotic manner. It randomly walks from cell to cell connecting the current cell with a random unlinked neighbour. Although this method allows for “better” Levels to be generated due to the random elements of algorithm not making it biased, the random walking also requires a long time for such a maze to be generated making it extremely inefficient.



(*Fig. 5*) (*Fig. 6*) (*Fig. 7*)

Figures five, six and seven are four by four sized examples of generated mazes using Aldous-Broder. The lack of a pattern in the mazes highlights the primary difference between this algorithm and a biased one such as Binary. This algorithm allows for generated levels to be more challenging for the user because he will not be able to learn a repetitive pattern that will work for all cases in which he can easily orientate himself.

Market Research

B & F Hate Mazes is at its core a puzzle game focusing on solving mazes. There is an already existing and competitive market for such games which vary from top down-viewed style to first person. Because a final release of this project is aimed for the mobile market (Android First), the research done focuses on targeting that market even if the version prepared for this module will run on PC only. The transition between platforms will be smoother because of the Tool used to develop the title. More information about the tools will can be read in the Tools Chapter of this paper. Solving Mazes as a core gameplay feature is not appealing for the average. This can be evidenced by the low number of games and game purchases for existing titles available on the Android Market. Popular Maze Games such as “Mazes & More” have managed to reach a significant user-base with ten million downloads which is high number but titles such as “Candy Crush” or “Subway Surfers” have reached over 500 million and 1 billion downloads respectively. New releases with less time on the Market constantly reach millions of downloads. Although there are other factors required when taking into consideration the popularity of a game or in this case a genre such as quality and marketing strategies, a safe assumption can be made regarding the overall popularity of Maze focused titles. There is a specific demographic of users which can be targeted with a good quality game focused on solving mazes. For marketing research and comparison five titles from the Play Store have been chosen that use mazes as their core gameplay. All the titles can be seen in the table below with some information about their current state on the Mobile Game Market for Android devices (all information presented has been recorded in April 2018).

Title Name	Downloads	Style	Rating
Castle Maze [8]	1.000	First Person	4.6/5
Maze World 3D [9]	1.000.000	First Person	4.1/5
Labyrinth Maze	5.000.000	Third Person	4.1/5
Mazes & More	10.000.000	Top-Down	4.6/5
Maze.io	1.000.000	Third Person/Top-Down	4.1/5

All five titles mentioned in the above table will be divided into three different categories to be analysed and compared from a game design point of view.
The first category consists of "*Castle Maze*" (*Fig. 8*) and "*Maze World 3D*" (*Fig. 9*). These two titles are both first-person games which require the user to reach a goal in various already created levels. Some key features that appear in both games is the inability of the user to see beyond a small number of cells (larger number if he is facing a long straight corridor) and both games assist the user with a mini-map. While the first-person view provides a challenging and entertaining gameplay experience for the user, some of the difficulty is removed via the mini-map that can help players reduce the number of corridors they visit by knowing whether the next few cells surrounding them represent dead ends or not.

(*Fig. 8*)


(*Fig. 9*)

The second category is formed of "Labyrinth Maze" (Fig 10.) and "Maze.io" (Fig 11.). Both titles share some key features and strongly differentiate the overall game feel from the first category. They are similar to the first category regarding using already created mazes.
The first major difference is the Camera angle and view over the grid. The camera is placed to allow for the player to see not only his current cell but a large portion or in some cases the whole grid. This can affect the whole decision-making process of the user because he is no longer required in many cases to do any path testing to be able to reach the end-game. Because most or in some cases all cells are visible along with their links, the player just needs to visually create his path and follow it. Such a game design is more appealing to users who want the challenging confusion presented with solving mazes removed but makes each level easier and faster to finish.

The final category also includes the most popular and successful genre of maze-games. Top-down view maze games such as "Mazes & More" (Fig. 12) have more common attributes with the second category. In this case all titles use pre-created levels and allow for the user to see the Grid. Being automated is the crucial element that separates this game from the previously mentioned ones. The unit controlled by the player is moving until it reaches a junction of paths where the user needs to swipe in which direction the controlled unit moves. The process is repeated until the player reaches the exit.

Version 1.0 of B & F Hate Mazes will focus on combining the lack of vision presented by the first category with a similar third person view comparable to the second category. This game design can provide a both challenging experience for users interested in solving such puzzles as they are required to remember previously explored segments of the maze to be able to reach the exit. An extra optional functionality is added for the less "hardcore" users and that is the ability to drop a limited quantity resource on the ground to mark the cells they have visited. This option can assist players with orientating themselves and therefore finish the level much quicker.

Method of Approach

The project has been developed and managed using a mixture of Scrums and Kanban. Weekly scrums have been set for which the author has been working as a manager, lead artist, lead programmer and game designer. At the end of each week an evaluation was made analysing the time required to finish certain tasks and establish and divide future work accordingly. Work what was not finished or extended would be pushed for next Week's panel and in some cases divided into smaller segments. This agile methodology has been chosen because it was already practiced by the author in other previous projects therefore saving time from learning a new management method and using it for the development of

the game. Other tools for highlighting and outlining overall requirements and the minimum viable product have also been used and will be further presented in the management section.

A different methodology that could have been used is test-driven development. This method requires the author to turn all the requirements of this project into specific test cases and then build the software that passes these tests. This method is extremely efficient concerning debugging code and having a clear outline of all the requirements. It can also smooth the flow of the development and help create higher quality code. The primary downside of this method is the time required to include all the test functions which do not add anything to the actual development of the game. This process can take a long time, especially since the author is the only developer. All tests can also be less efficient because of the number of people working on the project. In some cases, the developer creating the tests might not be able to observe that certain parameters must be checked therefore allowing for blind spots in the code [10].

Tools

Several tools have been used for the development of this project. The primary tool was however Unity 3D together with Microsoft Visual Studio. Although other similar game engines such as CryEngine, Frostbite or Unreal are great options, I have chosen Unity for several reasons which I will be listed further in this paper.

Adobe Photoshop CC 2018 is one of the software tools used for editing and creating various textures for the project as well as some UI Elements.

3DS Max has also been used to create some small assets for the game to avoid using extra unnecessary resources from the Unity Asset store.

GitHub is the resource used for version control for both the game as well as the documentation provided such as the weekly highlight reports and pictures used in the final paper.

Software

There are two general types of software that I have used for the development of my project. The first one is the software used to develop the game and the second one consists of various software tools used for its design elements (sound and visual assets).

Unity 3D

As previously mentioned, Unity 3D is the primary tool used for the development of this project. Unity 3D is a multiplatform game engine created in 2005 by Unity Technologies. Unity supports both 2D and 3D games development with drag and drop functionality and uses scripting features with the following languages: C#, JavaScript and Boo. There are various reasons for using Unity as the primary tool compared to other available engines. Unity uses a subscription based licencing agreement which starts from free use and extends to an enterprise level with a negotiable monthly fee.

Users using the Personal version (free) do not have access to some features such as Performance Reporting, Premium Support and Access to the Source Code. Moreover, the revenue is also capped at 71.000 British Pounds and using the Unity Multiplayer networking tool caps the number of concurrent users at 20. [6]

All these mentioned limitations are however surpassed by the large number of features it provides the user with. For example, the Unity Asset Store is an online shop where developers can publish or purchase various tools and assets created by other users. Everyone can access it and prices vary from free to over 2500 pounds. This feature allows for users with little experience or skills in a development area such as 3D/2D modelling or even programming to use some of the available assets for the development of their projects. Design assets purchased from the Asset Store have also been used for the development of this project, details will be discussed later in this paper. Another great feature of Unity 3D is the combination between its user-friendly interface and active community that provide useful and up to date documentation and information which is helpful for both novices as well as more experienced users. One last important feature that this game engine offers is its multiplatform capability. The current iteration of the project is available on Windows only however, because the software offers flexibility in building on different platforms, it creates the opportunity in releasing it on other devices such as mobile phones (both Android and IOS) as well as consoles. [3]

All the previously mentioned features and the already accumulated experience from working with both Unity and C# in various modules throughout the first and second university years have convinced me that this tool can provide me with all the necessary resources for the development of this project. Although learning a new engine could have been a

valuable skill to obtain while completing this module, the time and resources required to acquire a similar or the same result would have been longer therefore reducing the chances of delivering a finalized product.

CRYENGINE

CryEngine is a game engine designed by the German game developer Crytek and is primarily used for 3D computer and video games as well as high end 3D technologies. [7] Its initial release was on the second of May 2002. The CryEngine software development kit is using a “sandbox” editing concept. It emphasises large terrains and freestyle mission programming for video games compared to other engines.

GBP 8.71/Month Subscription based

Frostbite

Unreal Engine

Management

Project management required the usage of various tools to help the author both visualise and follow the progress of the game. Using these tools have saved a lot of time for the actual project and allowed for a more linear development of the game and this paper. One tool was especially useful since it provided a platform to perform both the Scrums and Kanban methodologies simultaneously.

Trello is the primary tool used by the author to outline and follow the progress and requirements of this project. This tool is a free web-based project management application available on both desktop as well as mobile platforms. It uses editable boards containing lists where tasks can be outlined and divided into various technologies. Trello can also support the upload of files that do not exceed 25 MB for the free version making it an excellent source of storing frequently used documents such as pictures or papers.

A user can freely customize and adjust his board to match the project and he can also add other users that can either view or edit the board based on the rights assigned by the creator of the board.

For this project the author has divided the lists into two major categories. The first category consists of lists that vary from Index to documentation. These lists do not indicate the progress and work on the project but rather are used to store essential information and provide quick access to it. This board pattern can be universally used for any other projects since it does not focus on specific characteristics of the project but it can be altered to match its requirements in needed.

Documentation is the first list included in this category. It is used to store important references that the author has collected for research purposes. Some examples include links to papers related to Procedural Generation. (Fig. 13)

Second list is called Report information. This list has not been used as frequently as the rest because most of the information required for the report has been used from external sources. Most of the information provided in this list is focused on the research segment of the report focusing on Procedural Generation. (Fig. 14)

The third list used is Documents. As the name implies this list is used to store documents included throughout the development of the project. Most documents present in this list are Mind Maps and important documents. (Fig. 15)

Useful Links only contains the link to the SPMS page since most useful links are mostly related to the paper research therefore are included in the Documentation list. Future links might include Google Forms containing user feedback or even release / download links. (Fig. 16)

The last List coming under the first category is the Index. This can also be a transitional list to the project development category that follows. Index contains five cards. The first card which is not labelled by any colour briefly describes the general purpose of the colours in the following cards. Each of the next card is labelled with a colour and a description for its connotation. (Fig. 17) All lists following this one will be marked with one of the four colours accordingly to each weekly sprint. (Fig. 18)

All following lists are included in the second category which is the weekly sprint section. A list with the week count and dates is created for each week of the project development. The lists contain cards highlighting all the tasks divided for

the respective sprint session. At the beginning and end of the week all cards on the list are labelled with a colour matching their respective state. Tasks that need to be worked on are marked with yellow and in case they need to be pushed forward they are labelled with the orange colour. No requirements are removed from the lists but instead these are marked with the colour red. The reasoning behind this approach is that all requirements can be still used for either later development or just as a reminder of which features not to include. All cards should have the yellow colour removed by the end of the project and the respective colour assigned based on the state of the requirement. If all cards are marked with one single colour by the end of the project, then the development of the project has undergone smoothly and without any delays. Having cards marked with two colours will always have one orange colour by the end of the project. All requirements that have two colours can be of two types, the first one is for tasks that have been delayed but resolved, therefore being marked with orange and green. The second type of double coloured requirements are the ones that have been delayed and removed or abandoned. These will be indicated by the colour orange and red. This is the worst-case scenario since it generally implies that a requirement has been not met due to delays.

Other tools used for creating various management files include Coggle for creating Mind Maps for both the report and the game helping outline all their respective features. This tool allows to both create new Mind Maps as well as save existing ones and later edit them in the browser. As previously mentioned there have been two Mind Maps Created. The first one highlighting possible requirements for the game development (Fig. 19) and the second one having more of a guide role for writing this paper (Fig. 20).

Stages

The number of stages initially mentioned in the PID has remained the same throughout this module, however, the outcomes have changed. The division of work has been between three primary categories: Research, Development and Report. Another major change is the choice of developing the Maze Generation Tool after finishing the research phase, therefore delaying the game design development.

Stage 1: Initiation

At the beginning of the project the management tools have been setup. This includes the Trello Board as well as the User Story Map highlighting the Minimum Viable Product (Fig 21. Note: The initial MVP does not match the MVP for this assignment but the one for the Game). During this stage a blank project in Unity and GitHub Repo have also been setup with some small scripting done as part of a refresh on how to use the tools. The author has also began creating a Game Design Document for the Game to make the outcome easier to present to the supervisor.

After setting up all the tools, the first sprint requirements have been added into the Trello board.

Stage 2: Research

This stage mostly included reading “Mazes for Programmers” [7] which has been the primary research resource used since it covers the basic and advanced information required to create the core tool used for the game development.

Another research topic includes Unity C# scripting web pages which were mostly used as reminders of various features that the tool offers. Some sources used for this phase include Unity’s tutorial page that has always provided the must up to date information as well as going through Lynda courses teaching Unity (See Research Links).

Martin Beck has also provided the author with research papers providing useful information about procedural generation from both a programming as well as game design point of view. Although this research has been done before choosing the project to allow for the creation of a more achievable outcome, the literature had also been reviewed during this stage.

Paper experimentation was one exercise required to understand the core concepts behind building mazes. This process helped the author properly understand and evaluate the amount of time he would require for the implementation the algorithms as well as choosing which of them to use for the first version of the Game. For more research information revisit Research section. For more implementation information visit Stage: 4 Generation Tool. During this stage the GDD document had been updated a second time matching some feedback provided by the project supervisor, however the author decided to abandon finishing this document because of the time required to maintain the document with most changes done to the game design over the course of its research.

Stage 3: Game Design

The Game design stage is the first stage where the development process had also begun. All the market research had also been done during this stage. Most design choices were focused on overall game feel as well as which Levels out of the initially desired ones could be included in the version released for the module. The initial GDD had not specified what types of mazes can be used because it had been written before the maze creation research period. Another conflicting design choice was the position of the camera in accordance to the Grid. This decision was crucial because it was one of the initial game design suggestions as far as its Unique Selling Points. This decision had also been influenced by the design of the Maze. The walls thickness and shape were also essential elements in providing an optimal experience for the user that does provide an enjoyable puzzle experience while also creating environmental confusion (another USP for the Game). This was also the stage where the decision to create version 1.0 of the game on only the Windows platform had been made.

Stage 4: Generation Tool

Stage 5: Core Functionality

Stage 6: User Testing

Stage 7: Game Polish

Stage 8: Final Evaluation

Stage 9: Report

End Project Report

Deliverables

Project Post Mortem

Conclusions

References

[1] Buck, J., 2015. *Mazes for Programmers: Code Your Own Twisty Little Passages*. Pragmatic Bookshelf.

[3] Unity. 2018. *Unity - Products*. [ONLINE] Available at: https://unity3d.com/unity. [Accessed 25 April 2018].

[4] Unity. 2018. *Unity - Multiplatform - Publish your game to over 25 platforms*. [ONLINE] Available at: https://unity3d.com/unity/features/multiplatform. [Accessed 25 April 2018].

[5] Dice Insights. 2018. *How Unity3D Became a Game-Development Beast*. [ONLINE] Available at: https://insights.dice.com/2013/06/03/how-unity3d-become-a-game-development-beast/. [Accessed 25 April 2018].

[6] Unity Technologies Blog. 2018. *New Unity products and prices launching soon – Unity Blog*. [ONLINE] Available at: https://blogs.unity3d.com/2016/05/31/new-products-and-prices/. [Accessed 25 April 2018].

[7] Crytek GmbH: Crytek announces its Game Engine CryENGINE. 2018. *Crytek GmbH: Crytek announces its Game Engine CryENGINE*. [ONLINE] Available at: https://web.archive.org/web/20081115062536/http://www.crytek.com/news/news/?tx_ttnews%5Bpointer%5D=17&tx_ttnews%5Btt_news%5D=76&tx_ttnews%5BbackPid%5D=9&cHash=e4dea47a80. [Accessed 26 April 2018].

[7] Buck, J., 2015. *Mazes for Programmers: Code Your Own Twisty Little Passages*. Pragmatic Bookshelf, Part I.4.

[8] Castle Maze â€“ Apps on Google Play. 2018. *Castle Maze â€“ Apps on Google Play*. [ONLINE] Available at: https://play.google.com/store/apps/details?id=tms.gdx.maze&hl=en_GB. [Accessed 23 April 2018].

[9] Maze World 3D - Apps on Google Play. 2018. *Maze World 3D - Apps on Google Play*. [ONLINE] Available at: https://play.google.com/store/apps/details?id=ru.empiregames.maze3d&hl=en. [Accessed 24 April 2018].

[10] Problems with TDD. 2018. *Problems with TDD*. [ONLINE] Available at: http://dalkescientific.com/writings/diary/archive/2009/12/29/problems_with_tdd.html. [Accessed 01 May 2018].