National College of Ireland
BSc in Computing
2015/2016

Conor Clarkson
x12494292
x12494292@student.ncirl.ie

Sample Space

Technical Report

Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name:
Student ID:
Supervisor:

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature:_____ Date:_____

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

Complete the sections above and attach it to the front of one of the copies of your assignment,

Table of Contents

1 Introduction

The purpose of this document is to set out the planning process for the development of an interactive videogame using the Unity Game development engine that musically reacts to how the player plays the game.

The intended customers for this product are both videogame enthusiasts and music enthusiasts alike. The game also has potential in musical theory in allowing the user to get a better understanding of musical theory and musical keys.

1.1 Background

This project idea is the result of my experience in a multitude of videogames across all genres, as well as my lesser experience in music and song writing. I have always had an interest in the use of music in all forms of media, particularly films and videogames. I've found that an effective use of music and sound design can have a bigger effect on a scene or gameplay moment than almost any other aspect of a scene. Be it used to establish atmosphere or mood, or ramp up tension and action to reach a chilling or bombastic crescendo, sound design and music are one of the most effective methods of reaching a player/viewer.

In the past I have seen this effectively used in combination with an element unique to the medium of videogames. Interactivity. This connection first occurred to me whilst playing "Superbrothers: Sword & Sworcery" by Capybara Games, in a small moment that allowed players to freely play musical notes in accompaniment of an intimate music track by the games composer, Jim Guthrie. This created a connection with the game world itself, that the player was an active part of this moment and could change it in their own creative way. From that point on I had a fascination with sound design in videogames and how it could be used to create new, affecting moments that can only be achieved through that medium.

1.2 Aims

The objective of this 4th year project will be the development of an interactive videogame using the Unity Game development engine. The game will be a 2d platformer that tasks players with traversing a variety of worlds and levels, each with their own theme and atmosphere. The player will be able to control the game character through movement along a 2 dimensional plane with basic movement and jumping, as well as using environmental triggers to alter the geometry of the game environment to traverse and defeat AI enemies.

The novelty of the game will lie in its use of musical theory, rhythm and time signatures. As the player traverses the level, a basic melody or musical track will be playing and will serve as a crucial aspect of gameplay. With each use of one of the players moves and environmental trigger a sound is generated in the same Musical key as the backing track. This will create a fluid blend of gameplay and music that allows players to shape the sound of the level through the way they play.

The style of music will also be decided based on the players own preference, through the use of a player profile system that generates a musical genre dependent on the users answers to specific questions during the creation of the profile. This will require the development of primary gameplay mechanics, as well as custom assets and music for each particular level.

1.3 Technologies

Unity game development engine was used for the primary development of my videogame. After much deliberation between Unity and Unreal engine in regards to which software to use in my development, Unity seemed the more appropriate choice due to its preference for Low-fi, 2d game development. I also have past experience in Unity game development engine from previous college assigned projects.

Depending on project progress, *Unreal engine* may be used for creation of 3d backgrounds and environments, as it is renowned for its ease of use in regards to

3d modelling. *Cubase* and *Audacity* will be used in the recording and editing of the games music and sound effects. Cubase will be used to record audio from an external audio interface, while Audacity will be used to quickly edit the recorded audio for use in the game.

2 System

2.1 Requirements

2.1.1 Functional requirements

- Allow user to control a character in a 2d space, with necessary colliders and triggers in place to provide a structure level design.
- Responsive controls that allow user to effectively maneuver around the game environment and combat enemies. Can be achieved through intuitive control design and use of a tight control scheme.
- Effective sound design in regards to the action notes and they're implementation. Each action note should play the instant the player interacts with an object, with little to no delay. Each note will need to be sound balanced for multiple interactions in a row to avoid audio clipping or messy sounds.

2.2 Use Case

2.2.1 Requirement 1: Profile Creation/User Login

2.2.1.1 Description & Priority

Creation of a User profile at beginning of game

2.2.1.2 Use Case

Description

Use case describes the process involved creating a user profile upon starting the game and completing

Flow Description

Precondition

User has not launched application already

Activation

Player launches game from desktop, having not created a profile before

Main flow

1. Player selects "New Game" button prompt on main menu
2. Player is prompted to enter profile name, Player submits name

3. Player is prompted with multiple choice questions to assign music preference, submits choices
4. Player is assigned musical genre and is presented with 3 open save slots, player selects slot and is prompted with a confirmation pop up
5. Save is created on that slot and user is brought to load screen, where the new save slot is located
6. Player selects save slot and begins game

Alternate flow

A1: Pre-existing save file in all slots

1. Player selects “New Game” button prompt on main menu
2. Player is prompted to enter profile name, Player submits name
3. Player is prompted with multiple choice questions to assign music preference, submits choices
4. Player is assigned musical genre and is presented with 3 occupied save slots
5. Player selects “Overwrite” button prompt and then selects an occupied profile
6. Player confirms via pop up prompt and saves over existing save
7. Save is created on that slot and user is brought to load screen, where the new save slot is located
8. Player selects save slot and begins game

Termination

Process is terminated once user successfully loads their save game

Post condition

Game has begun and correct player profile is loaded

2.2.2 Requirement 2: Level Completion/Leaderboard Logging

2.2.2.1 Description & Priority

Player completes a level and their score is logged in the database for that specific level.

2.2.2.2 Use Case

Description

Use case describes the process of the player character completing a level after having collected a number of collectibles. They’re score is then calculated and logged in the database for the level

Flow Description

Precondition

User is in the middle of playing a level

Activation

Use case is active when player begins the level and starts collecting collectibles and reaches the end.

Main flow

1. Player starts level, collecting collectibles as they traverse the level
2. Player enters the exit level trigger, telling the system that the level is over
3. Player is presented with a pop up displaying their total collected collectibles, time taken to complete level and final assigned score
4. Player selects “continue button prompt” and their score is entered into the database
5. Player is taken to the leaderboard display screen where the 10 highest ranked scores are generated
6. Player selects continue and resumes level

Termination

Once leaderboard is generated and player hits “resume” button prompt

Post condition

Game generates next level in the level queue

2.2.3 Resource Utilization requirements

In order to keep file size to a minimum and performance at optimal level (see performance requirements), resource/assets need to be re-used and recalled frequently. By storing important assets that appear frequently (projectiles/prefabs) as prefabs, Unity can call the same asset multiple times.

2.2.4 Accessibility requirements

It is important for videogames to be as accessible to ensure user engagement. With this in mind, the primary control of the game has been designed to be as accessible as possible with the “pickup and play” mentality in mind. In conjunction with this, the structure of the opening level itself has been designed to teach the player the mechanics of game movement without intrusive tutorial/instruction prompts

Figure 1. Controller Support & Mapping

2.2.5 Performance/Response Time requirement

In order for the gameplay to be as accessible and enjoyable as possible, the gameplay must maintain a minimum framerate of 30fps (frames per second). This will be achieve by effective use of efficient gameplay design, removing redundant game objects that aren’t crucial or necessary, and correct use of efficient basic collider design. As the game is majority in 2d, this shall not be a difficult undertaking.

2.2.6 Extendibility requirements

An advantage of having a level based structure is the openness to expand the game with additional levels and content, allowing for an easily extended product. The opportunity for additional levels, SFX and music is an important requirement for the games original design.

2.3 Architecture

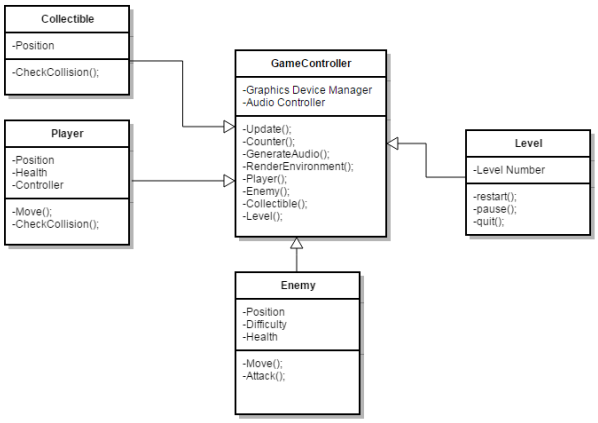


Figure 2 Basic Class Diagram to clarify architectural structure of the game system

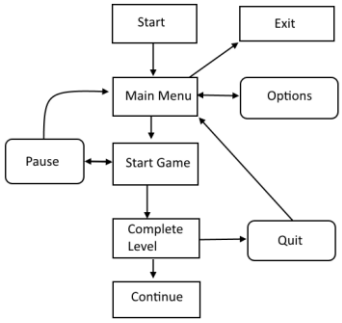


Figure 3 Workflow of the games basic menu system

2.4 Implementation

Implementation of the project has been relatively straight forward, with the help of the Unity community forum, most minor issues were able to be resolved. Some issues required a more methodical and researched approach however. One such issue was the development of the AI enemy that the player encounters in certain areas. This enemy is designed to pursue the enemy through the varied and complex structure of the level design. This was developed

Through the use of the A* Pathfinding Project package, I was able to implement a number of features to assist in the decision making of the AI agent. By adding a Grid Graph, the space which an AI can and cannot move is quickly decided. The Grid graph envolopes an area with a grid of nodes that dictate whether an area of

the map is “walkable” or “unwalkable”, achieved by working with the existing colliders of the level geometry and mapping around the spaces that are free to move in.

The grid detects existing colliders marked with the tag “obstacle”.

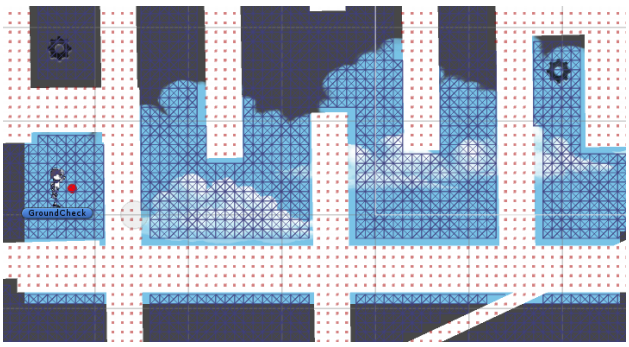


Figure 4 Mapped grid of movement from Grid Graph. Blue nodes indicate walkable. Red nodes indicate unwalkable.

On play, the Grid Graph scans the grid and calculates the optimal path distance to the target (Player) and plots the string of waypoints in the grid of nodes that the AI will follow. The AI keeps track of the distance between itself and the current waypoint on the path. Once distance to the current waypoint has reached required proximity, the AI moves to the next waypoint in the path. The AI scans the graph every second to respond to changes in the player’s position or the environment itself, ensuring quick reaction time to the active play environment.

Figure 5 Plotted path of AI agent to player destination

If the current waypoint is greater or equal to the total number of waypoints in the waypoint array, the path is ended. The AI calculates the direction to the next waypoint by subtracting its current position from the path of destination waypoint, this calculation works in tandem with the Force Mode method that controls how force is applied to the AI’s collider, allowing for movement through the environment.

For the system to interact with the locally stored database for leaderboard functionality, the game must directly communicate with the database using SQL commands. Firstly, the system must retrieve the information from the database using the *HighScoreManager.cs* script file. This file is responsible for most of the functions related to the database. This manager must call the GetScores function to obtain the information contained within the database. This is done by communicating SQL code to the database, the code in the GetScores function is selecting all entries in the "HighScores" database with the same level number as the variable "levelNo" which is called from the scene manager. This obtains all the high scores for the current level of play.

```
public void GetScores(int levelNo)
{
    highScores.Clear();
    //connection to database
    using (IDbConnection dbConnection = new SqliteConnection(connectionString))
    {
        dbConnection.Open();

        //command
        using (IDbCommand dbCmd=dbConnection.CreateCommand())
        {
            string sqlQuery = String.Format("SELECT*FROM HighScores WHERE LevelNo=\"{0}\"", levelNo);

            dbCmd.CommandText = sqlQuery;

            using (IDataReader reader = dbCmd.ExecuteReader())
            {
                while (reader.Read())
                {
                    highScores.Add(new HighScore(reader.GetInt32(0), reader.GetInt32(2), reader.GetString(1),
                    reader.GetDateTime(3), reader.GetInt32(4),reader.GetString(5)));

                }
                dbConnection.Close();
                reader.Close();
            }
        }
        highScores.Sort();
    }
```

Figure 6 GetScores() function

Now the player's current score needs to be inserted into the database. This is done in a similar manner using the *InsertScore* function. This function also directly communicates with the database, but instead of reading from it, it inserts the player's information directly into database via SQL commands. By allowing a variable to be placed within the SQL command itself, we are able to read or write any information from the game environment we need in to the database for storage.

```
public void InsertScore(string name, int newScore, int levelno, string character)
{
    //connection to database
    using (IDbConnection dbConnection = new SqliteConnection(connectionString))
    {
        dbConnection.Open();

        //command
        using (IDbCommand dbCmd = dbConnection.CreateCommand())
        {
            string sqlQuery = String.Format("INSERT INTO HighScores (Name,Score,LevelNo,Char) VALUES(\"{0}\",\"{1}\",\"{2}\",\"{3}\");", name,newScore,levelno,character);

            dbCmd.CommandText = sqlQuery;
            //when executing commands into database
            dbCmd.ExecuteScalar();
            dbConnection.Close();


        }
```

Figure 7 InsertScore() function

Unlike the high score data, the user profile data does not require large amounts of multiple datatypes. I came to the conclusion that the entire user profile can be managed with two string variables: the user's name, and they're selected musical style. By accessing these variables, we can easily place the users name on a leaderboard correctly as well as tailor the audio generator to the select musical style. Because there are only two variables which will be unique, we don't have to use a database. Instead, I used Unity's *PlayerPref* function, which stores small datatypes locally in its own manager. This is especially effective for single variables which don't require a database storage. The users Name and music type are set using the PlayerPrefs.SetString("StringName",String) command, storing the data in the Unity manager locally.

```
public void Create1() //saveSLOT1
{

    PlayerPrefs.SetString("Name1", Name);
    PlayerPrefs.SetString("Option1", Option);
    //slot1text.text = PlayerPrefs.GetString("Name1");
    SaveUI.SetActive(false);
    LoadUI.SetActive(true);

}
```

Figure 8 Setting the user info with PlayerPref

The most important benefit of using PlayerPrefs in Unity is the ability to access stored information from any scene in your project. Because the player profile is created and set in the MainMenu scene, I would not be able to access the stored info were it not for PlayerPrefs. Now we can simply use the PlayerPrefs.GetString() method in the Start() function of any script in the project and we can access the stored variables. I use this to get the final name and final musical type that will be used for as long as the player remains on that player profile.

```
void Start () {
    Name.text = PlayerPrefs.GetString("FinalName");
    if (Name == null)
    {
        Name.text = "";
    }
    Option.text = PlayerPrefs.GetString("FinalOption");
    if (Option == null)
    {
        Option.text = "";
    }
}
```

Figure 9 Getting stored variables using PlayerPrefs

2.5 Testing & Evaluation

Unit Testing

During the development of this project I undertook a number of isolated unit testing on individual pieces of code. This was an incremental process over the course of the project development. With the use of Unity's Debug Log function, it was possible to easily test and assess each function and process safely, without jeopardising the other processes. The user login/profile creation was developed in a completely separate project for controlled development and testing, without risk of compromising other processes. This was also done for the advanced AI enemies that appear on certain levels, due to their demanding

Audience testing

As well as isolated testing of my own, I tasked a number of user tests among a wide range of people, ranging in experience in regards to Music and Videogames. I collected the contact details of possible test users and emailed them a build of the game with a list of instructions and questions to effectively test the system. A number of planned tests were unable to be executed as a result of the restrictions that Gmail hold on sending .exe programme files, thus preventing exclusively Gmail users from participating.

Planned user tests	User tests executed	Use case 1 Pass	Use case 1 Fail	Use case 2 Pass	Use case 2 Fail
13	10	8	2	7	3

Functionality Test

Of the successfully assessed test group, each user was given a list of tasks to be completed within the system. Both overall tasks were based on the critical use cases detailed previously (see Use Case 1+2).

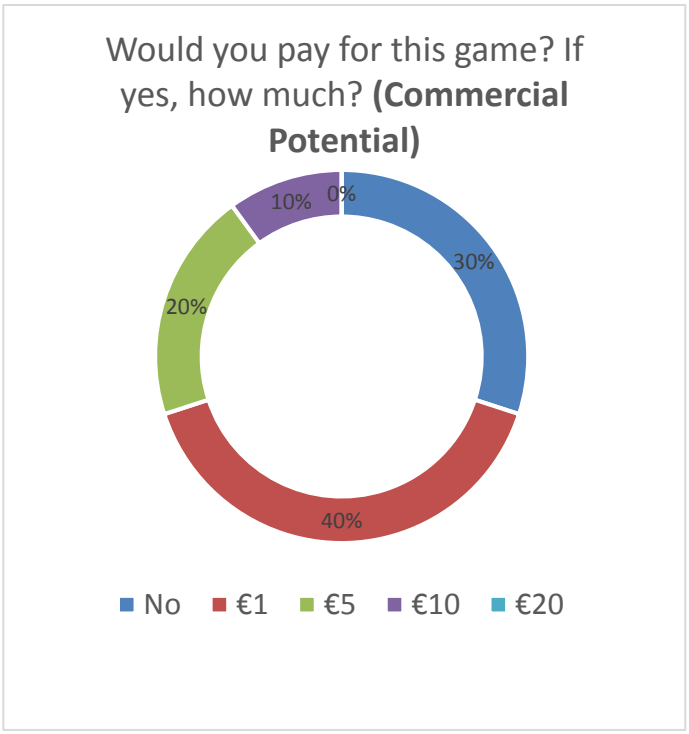
This was the logical choice to make use of the limited testing time available. As shown in the chart below, we can see a higher success rate in Use Case 1 than in Use Case 2. This is most likely a result of Use Case 2 requiring the user to complete an entire level via actual gameplay, a very skill dependent task.

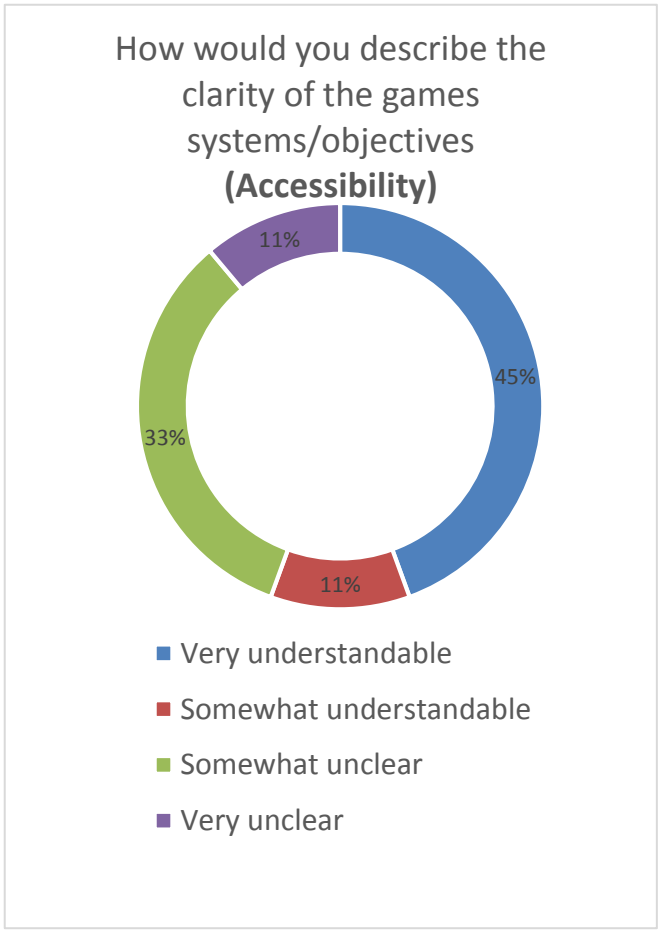
The success rate of both tasks however can be seen as being very high, thus fulfilling the successful accessibility requirement, as well as functionality.

Despite the high success rate, the results of Use Case 1 were of slight concern, as it is a solely UI based task. The straightforward nature of this task should ideally have a 100% success rate. This will factor into any future UI design decisions that I will undertake if required.

Subjective testing

The same test group was sampled after the functionality test to obtain data from an opinion survey regarding their play session. The users were asked 3 multiple choice questions that are linked directly with this projects non-functional requirements.





2.6 Graphical User Interface (GUI) Layout

This is a representation of the interface that exists in the game during gameplay. UI has purposefully been kept as minimalistic as possible to allow users to clearly understand the goals and feedback from the game itself. The players number of collectibles (represented as notes here) are displayed in the top right corner of the screen, as well as the players current score in the top right of the screen.

This is a representation of the opening menu that will be the first thing the user sees on launch of the game. The user is presented with 4 options at launch. The “New Game” button will take the player to the player profile creation menu, where they will follow a number of questions that will decide they’re musical genre. The “Load” button will lead

3 Conclusions

Upon reflection, the development of this project has been a huge growth period for myself as a software developer. The task of developing multiple different systems from AI, to physics, to databases all within the context of a single project has been a huge undertaking, yet a rewarding one, The decision to create almost all the systems from scratch was probably an inefficient method from a time management perspective, but it advanced my own learning exponentially, having become technically proficient at C#, with no previous experience of it in the past. The experimental nature of working with Unity and discovering its features has been a particular highlight for me, inspiring future project ideas that I will undertake once this project is completed. A huge issue I've had over the course of this project was planning. Because of my relative inexperience with Unity, I was not fully aware of what was and was not possible. This lead to multiple changes in design for the project, even in the final months. The design and requirements of the finished project were not as refined and polished as I'd have hoped and this was a direct result of a lack of attention on my part during the planning stages of the project. Overall this project has opened my eyes to the broad nature of software and game development, be it programming, art or music, there is no end to the skills that can be developed during the creation of a video game, or any piece of software.

4 Further development or research

With more resources, a system that rewards players for playing in time with the background music may be implemented. This mechanic would track the BPM (Beats per minute) of the backing track and will register if the player completes actions in rhythm with the track, most likely through use of a typical 4/4 drum beat that can be most recognized by the game system. If the player completes a number of interactions in a row in time with the drum beat, they will be rewarded with a score boost.

5 References

"Brackeys | Become A Developer". Brackeys. N.p., 2016. Web. 10 Apr. 2016.

"Unity Community". Forum.unity3d.com. N.p., 2016. Web. 3 Feb. 2016.

Technologies, Unity. "Unity - Manual: Unity Manual". Docs.unity3d.com. N.p., 2016. Web. 1 Feb. 2016.

Raaschou, Gucio. "Guciodevs - Unity3d Tutorials". Guciodevs.com. N.p., 2016. Web. 3 Jan. 2016.

"Musictheory.Net - Lessons". musictheory.net. N.p., 2016. Web. 10 Nov. 2015.

Zanfir, Alex. "5 Tips Every Unity Developer Should Know". Pluralsight.com. N.p., 2016. Web. 2 Feb. 2016.

"Game Development Stack Exchange". Gamedev.stackexchange.com. N.p., 2016. Web. 3 Nov. 2015.

"Game Engine Development Game Development Tutorials By Envato Tuts+". Game Development Envato Tuts+. N.p., 2015. Web. 10 May 2016.

"Stack Overflow". Stackoverflow.com. Web. 9 Nov. 2015.

6 Appendix

6.1 Project Proposal

1. Objectives

The objective of this 4th year project will be the development of an interactive videogame using the Unity Game development engine. The game will be a 2d platformer with relatively basic “beat-em-up” that tasks players with traversing a variety of worlds and levels, each with their own theme and atmosphere. The player will be able to control the game character through movement along a 2 dimensional plane with basic movement and jumping, as well as attacking enemy actors through the use of 3 basic attacks (High attack, medium attack, low attack)

The novelty of the game will lie in its use of musical theory, rhythm and time signatures. As the player traverses the level, a basic melody or musical track will be playing and will serve as a crucial aspect of gameplay. With each use of one of the 3 attacks, the player will create a specific musical note, unique to each attach, in the same Musical key as the backing track. This will create a fluid blend of gameplay and music that allows players to shape the sound of the level through the way they play. Depending on time restraints and future feasibility studies, I may implement a mechanic that rewards players with stronger attacks if they attack in time with the time signature of the song, though that has yet to be decided.

This will require the development of primary gameplay mechanics, as well as custom assets and music for each particular level.

2. Background

This project idea is the result of my experience in a multitude of videogames across all genres, as well as my lesser experience in music and song writing. I have always had an interest in the use of music in all forms of media, particularly films and videogames. I’ve found that an effective use of music and sound design can have a bigger effect on a scene or gameplay moment than almost any other aspect of a scene. Be it used to establish atmosphere or mood, or ramp up tension and

action to reach a chilling or bombastic crescendo, sound design and music are one of the most effective methods of reaching a player/viewer.

In the past I have seen this effectively used in combination with an element unique to the medium of videogames. Interactivity. This connection first occurred to me whilst playing “Superbrothers: Sword & Sworcery” by Capybara Games, in a small moment that allowed players to freely play musical notes in accompaniment of an intimate music track by the games composer, Jim Guthrie. This created a connection with the game world itself, that the player was an active part of this moment and could change it in their own creative way. From that point on I had a fascination with sound design in videogames and how it could be used to create new, affecting moments that can only be achieved through that medium.

3. Technical Approach

I will be using the Unity game development engine for the development of my videogame. After much deliberation between Unity and Unreal engine in regards to which software to use in my development, Unity seemed the more appropriate choice due to its preference for Low-fi, 2d game development. I also have past experience in Unity game development engine from previous college assigned projects.

Depending on the feasibility of these ideas, I may be using Unreal engine for creation of 3d backgrounds and environments, as it is renowned for its ease of use in regards to 3d modelling. Support and tutoring in the use of Unreal engine is also provided in one of my current course modules so assistance is readily available to me from my lecturers.

I will be developing music and audio specifically for each level of the game, using both Audacity and Cubase music software to develop.

More sample data and game design theory will be gathered from in depth research into videogame development of similar premise as my game. Other games that I have discovered that feature similar theory towards music (in some form or

another) include: *Sound Shapes, Hotline Miami, Hohokum, Vib-Ribbon, Superbrothers: Sword & Sworcery* and *FEZ,* among many others.

4. Special resources required

This project will require a number of basic resources that are readily available at the college or freely available through the internet, on top of more niche hardware/software that I will procure myself.

Software:

- Unity
- Visual Studio
- Cubase
- Paint.NET/Photoshop
- Audacity
- Unreal

Hardware:

- Windows PC
- Playstation DS4 controller
- TASCAM US-144 audio interface

6. Technical Details

This project will require the use of C# coding language for Unity through Visual studio, as well as JavaScript for certain specific functionality and features in the Unity engine itself. Past experience in Unity, C# and JavaScript will be useful during primary development.

The 2d character models will be created in Paint.NET and Photoshop, primarily with Paint.NET as it is open source and freely available to the public. Unless time restraints deem otherwise, the background and overall levels will be modelled in 3d through the Unreal Engine 3d modelling feature.

The custom audio and music for the game will be developed in Audacity and Cubase audio software, used to edit audio recorded in-house through a TASCAM US-144 audio/MIDI interface and my own music/instrument set-up.

7. Evaluation

Feasibility studies will be completed during initial development to determine possible implementable gameplay mechanics, such as the rewarding of players by playing in time with song time signature and 3d environments/levels.

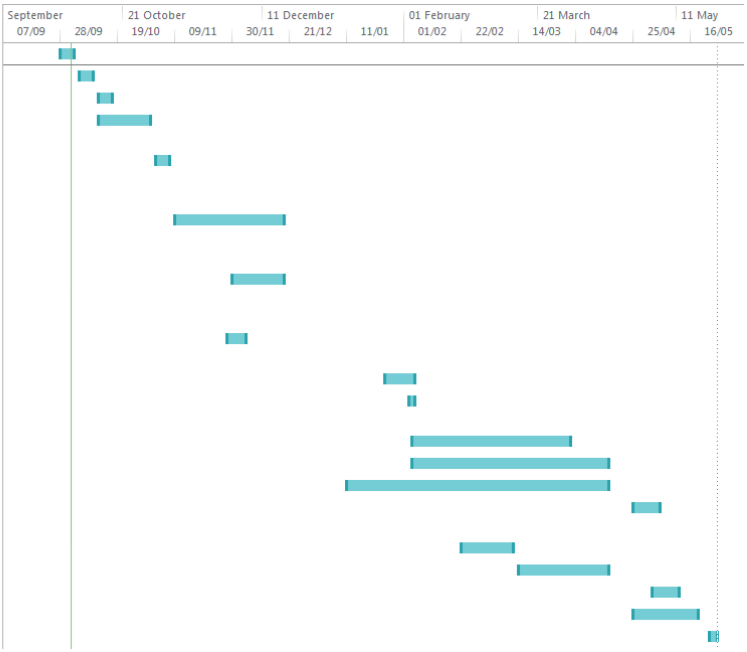
Evaluation will be completed through rigorous playtesting by myself over the course of initial development of basic gameplay mechanics, as well as public playtesting once a prototype is completed. This doubles as bug-testing as well as general enjoyment-factor. A wide test audience will be sampled, including fellow students, musicians, and both gamers and non-gamers. This has been scheduled to allow adequate time for bug fixing and gameplay tweaks before the final upload/presentation.

6.2 Project Plan

The following is my overall plan/timeframe for the completion of my project. Primarily I have assigned the first semester of the project class to the design and implementation of primary gameplay mechanics and functionality. Second semester has been scheduled with the completion of non-functional aspects such as art, animation, environments and more advanced music and audio. Level design has been scheduled for second semester but may be moved to Semester 1 if I finish implementing the main gameplay mechanics ahead of schedule.

Gantt chart

Task Name	Duration	Start	Finish
Project proposal	5 days	Mon 28/09/15	Fri 02/10/15
Feasibility Study	5 days	Mon 05/10/15	Fri 09/10/15
Software Testing	5 days	Mon 12/10/15	Fri 16/10/15
Initial Ground Design	15 days	Mon 12/10/15	Fri 30/10/15
Requirement Specification Document	5 days	Mon 02/11/15	Fri 06/11/15
Primary gameplay/game engine development	30 days	Mon 09/11/15	Fri 18/12/15
Additional/Tertiary Functionality Development	15 days	Mon 30/11/15	Fri 18/12/15
Project Analysis + Design	6 days	Sat 28/11/15	Fri 04/12/15
Gameplay Finalisation	9 days	Mon 25/01/16	Thu 04/02/16
Midpoint+ Prototype Presentation	2 days	Wed 03/02/16	Thu 04/02/16
Asset+Art Design	42 days	Thu 04/02/16	Fri 01/04/16
Level Creation	52 days	Thu 04/02/16	Fri 15/04/16
Sound Design	70 days	Mon 11/01/16	Fri 15/04/16
Final Documentation + Code	8 days	Mon 25/04/16	Wed 04/05/16
Playtesting	15 days	Mon 22/02/16	Fri 11/03/16
Bug resolution + fixing	25 days	Mon 14/03/16	Fri 15/04/16
Final polish + bugfixing	8 days	Mon 02/05/16	Wed 11/05/16
Project presentation	18 days	Mon 25/04/16	Wed 18/05/16
Project showcase	3 days	Mon 23/05/16	Wed 25/05/16



6.3 Monthly Journals

Month: September

My Achievements

Gained initial insight into project direction by attending the Project classes and speaking with lecturers, as well as other module lectures that have a relevance to my specific area of project work (Gaming and Multimedia). Through these interactions I was able to decide on the final idea for my final year project and complete the “Project proposal” documentation upload that details the rough plan and idea for the project.

I was also able to decide upon the technology I will use for my project, specifically the decision between using the Unity or Unreal game development engines. As my project will primarily be 2D, I decided on Unity, due to its more 2D-centric design approaches.

My Reflection

So far I'm optimistic as to the feasibility of my project, I've learned from past projects about the dangers of an over-ambitious idea and have decided on a relatively simple premise this time. I have also got one or two smaller mechanical ideas that might be implemented into the game to improve its complexity and fun-factor if I finish my initial requirements earlier than expected. These extra mechanics were decided on through discussion with my close friends and colleagues, emphasising the importance of an outside perspective in a solo project like this. I intend to continue discussion and collaboration with outside sources for this project to help keep an outside perspective on what I'm developing, thus avoiding the pitfall I have seen in other creators' work in not being able to see the flaws of your solo project because of how close you are to it. This will be a consistent goal throughout the development of the game.

Intended Goals/Changes

- To get a better handle on the Unity game development engine
- Create a basic mock up 2d platformer in Unity
- Better organise my time schedule and weekly goals

Month: October

My Achievements

Created a base level game environment for testing and development. Developed a 2D character controller using placeholder sprites to better understand the Unity animator. Developed a simple jumping mechanic and movement script.

My Reflection

I'm finding that this project is proving much more complex than I originally intended already. Due to the open nature of game development on Unity, it is very difficult to find the necessary help online in regards to implementing specific game mechanics, though there are forums to consult so the community can be of great help.

Intended Goals/Changes

- Begin development on combat system, basic attacks and collider creation
- Further develop movement and platforming sections

Supervisor Meetings

Items discussed: Explanation of basic project premise. Initial constructive feedback on project idea. Discussed using alternate hardware controller for game controller. Got a better understanding of the projects that my fellow students aimed to complete for their final year project.

Month: November

My Achievements

Developed basic delayed collider creation for the games combat system. Currently has no animation attached to it, only using placeholder visuals to show they are functioning as intended. This should provide a useful base to build the combat mechanics on.

My Reflection

Due to other module projects and submissions this semester, I feel I have neglected the development of the project this month. The initial stress of final year results had led to me prioritising more urgent project deadlines than this module, even resulting in missing this month's Supervisor meeting. The gradual development nature of this project had resulted in me taking it for granted and I understand now that I will need to work harder on the projects development. I hope to prioritise project development over the course of the Christmas break once other modules projects are completed.

Intended Goals/Changes

- Have functioning prototype completed over the Christmas break
- Focus development on combat mechanics and movement

Month: December

My Achievements

Redesigned combat mechanics to be solely based around the dash system, placing an emphasis on speed and momentum, as well as combo attacks. Expanded test area level to give the player a better understanding of the mechanics and how they can be used to manoeuvre the game environment.

My Reflection

I am understanding more and more as the project progresses the workload required to deliver a fully functioning and polished game. It is progressively becoming more difficult to resolve technical issues and bugs as the game mechanics become more complex. Despite the plethora of support forum assistance available, it still can only do so much due each game being so unique and each bug or mechanic so specific. This has led to a slower development process than I had intended but it is progressing nonetheless.

Intended Goals/Changes

- Implementation of a collectible system to give a goal to players
- Begin development of audio feedback system

Month: January

My Achievements

Implemented collectible item system that allows users to traverse the level and collect gears, further work needs to be done (end game state once all gears are collected), but it at least gives players an objective while the AI enemies are in development. Also began development of audio feedback system, with two of the

basic attacks being assigned correct sounds. Further development is required to implement the note modifier for variety.

My Reflection

During my research into the use of audio mechanics in Unity, I discovered a feature that allows the frequencies of the sound spectrum to be detected in a piece of music or audio and thus manipulated by Unity itself. This opens up more possibilities for future game ideas that can further implement sound in videogames. Time permitting, I may even implement this feature into this current project itself, using audio to manipulate objects around a stage.

Intended Goals/Changes

- Final implementation of collectible system (end game state once all gears collected)
- Further development of Audio feedback system
- Final development on game prototype for mid-point presentation

Month: February

My Achievements

Implemented leaderboard system into the project using SQLite. Allows ranking of users rank, score and level, stored locally in a database. Completed testing on leaderboard functionality in final build version, resolving any issues that were encountered.

My Reflection

As development continues I am consistently discovering new techniques and resolutions to previous problems I had in past Unity projects. This is both a source of satisfaction and frustration as I'd like to work on other side projects during the development of this main project, yet the time restrictions don't allow this freedom. It can mostly be seen as a positive at is a sign of my own personal growth as a software developer.

Intended Goals/Changes

- Further development of scoring system, how it is calculated and how it is inserted into leaderboard
- Further development of Audio feedback system
- Polishing existing mechanics for final upload

Month: March

My Achievements

Finalised scoring system and how it is calculated and inserted into the leaderboard database. Began development of User login and profile system, this was developed in an isolated unit testing environment due to the overall size of my main project. Created an isolated side-project for the development/testing of login/profile system before being inserted into the final project

Intended Goals/Changes

- Completion of login system
- Further development of Audio feedback system
- Polishing existing mechanics for final upload

Month: April

My Achievements

Finalised login system, began researching possible testers for audience testing.

Intended Goals/Changes

- Final polish for final code upload
- Completion of User
- Fine tuning of audio levels