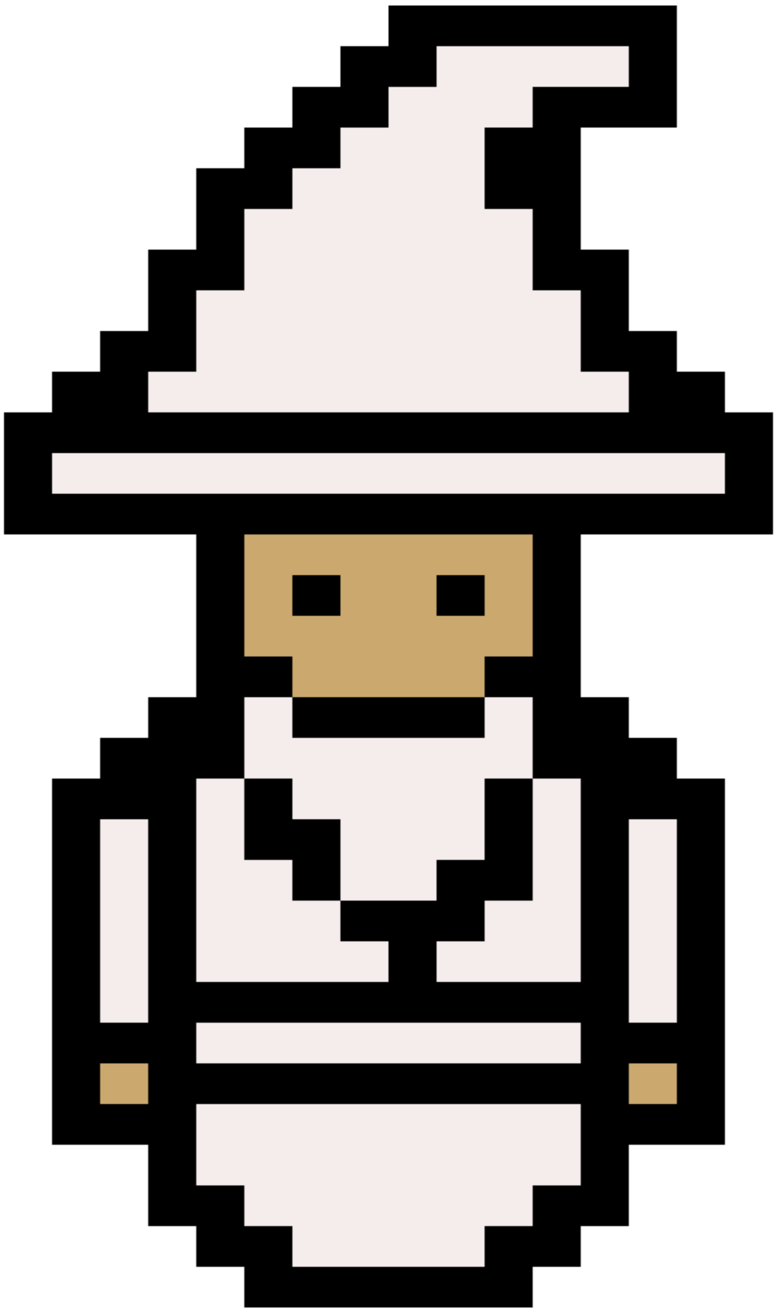
[Games Development Project]

Project Report, Game Design Document

and Technical Report

20/09/2017

Spelling Test



Contents

1. Proposed Plan

1.1 Team Roles

The team for this project will consist of:

- Yu-Ching Ho
- Nathanael Omnes
- David Byron

Yu-Ching was happy to take on the role of Project Leader although both David and Nathanael were heavily involved in the decision making. We decided to keep the team roles flexible so that each member would be able to contribute to all aspects of the project and learn new skills.

In the end, Yu-Ching and Nathanael did most of the work on the prototype, while David was mainly responsible for the documentation and presentation.

All members had some involvement in each aspect though, with David playing an important part in shaping the ideas of the game and Yu-Ching and Nathanael helping to edit and streamline the documentation and presentation.

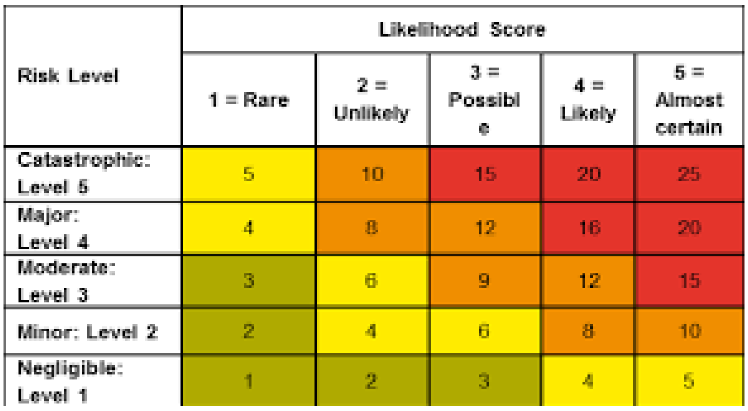
Trimester Two
As the group discussed the game, we realised that our project was too ambitious. The game was broken down into cut-scenes and enemy wizards, with Nathanael taking care of the actions, animation, and coding of the enemy AI, while Yu-Ching took care of the sprites, cut-scenes, puzzles design and coding and linking the scenes together. David was given the task of creating the tutorial, and after not doing the task for several weeks, the task was handed to Yu-Ching.

1.2 Risk Analysis

Risks	Probability (⅕)	Severity (⅕)	Risk level
A late change of game engine	1	4	4
Lateness / non-participation	2	3	6
Bad team communication	2	3	6
Missing work	3	3	9

Risks	Probability (⅕)	Severity(⅕)	Risk level
Visual attractiveness	2	1	2
Too difficult / easy	3	2	6

Mechanics not enjoyable	2	3	6
Not understandable	2	4	8
Too many bugs	3	3	9

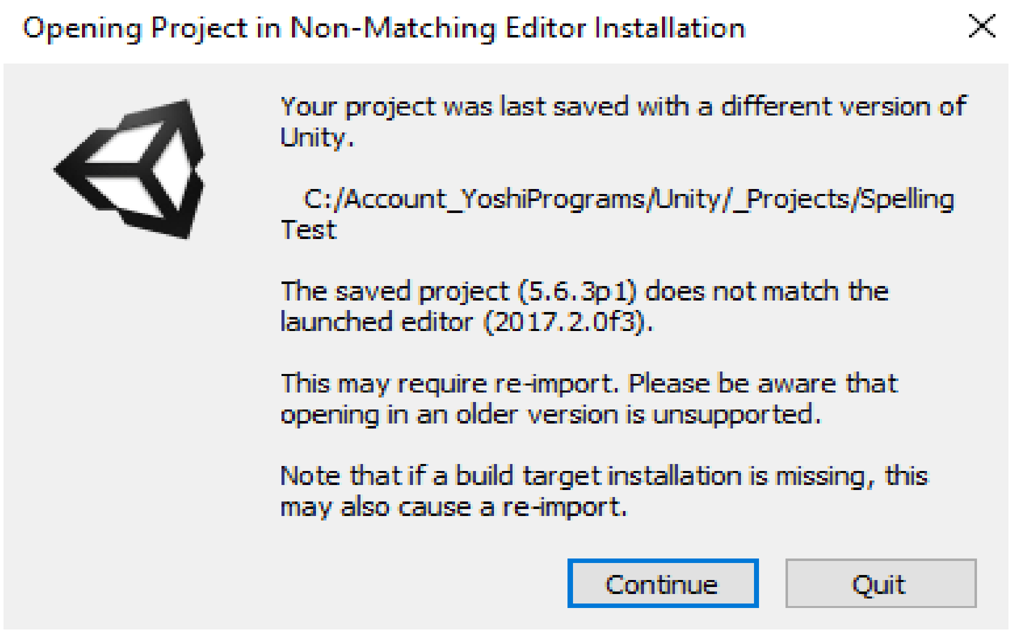


Risk Level	Likelihood Score				
	1 = Rare	2 = Unlikely	3 = Possibl e	4 = Likely	5 = Almost certain
Catastrophic: Level 5	5	10	15	20	25
Major: Level 4	4	8	12	16	20
Moderate: Level 3	3	6	9	12	15
Minor: Level 2	2	4	6	8	10
Negligible: Level 1	1	2	3	4	5

As it happens, and as discussed in the presentation, the biggest problem for the project was having to switch from Unity to GameMaker quite late on as the team failed to resolve some issues with Unity.

These can be summarised as follows:

- Old folders were re-appearing
- A notification saying that when importing GitHub version control, it was a different version of Unity even though we all had the latest version of Unity installed.
- Sprites appeared in massively different locations
    - Nathanael's local project showed animations appear correctly
    - When rest of group installed the project from GitHub and even a zipped Google drive folder, the sprites showed up all over the place.



Opening Project in Non-Matching Editor Installation

Your project was last saved with a different version of Unity.

C:/Account_YoshiPrograms/Unity/_Projects/Spelling Test

The saved project (5.6.3p1) does not match the launched editor (2017.2.0f3).

This may require re-import. Please be aware that opening in an older version is unsupported.

Note that if a build target installation is missing, this may also cause a re-import.

Trimester Two

Risks	Probability (⅕)	Severity (⅕)	Risk level
A late change of game engine	5	1	5
Lateness / non-participation	4	3	12
Bad team communication	1	3	3
Missing work	3	3	9

Risks	Probability (⅕)	Severity(⅕)	Risk level
Visual attractiveness	2	1	2
Too difficult / easy	3	2	6
Mechanics not enjoyable	2	3	6
Not understandable	2	4	8
Too many bugs	1	3	3

The game engine change was discussed over Christmas Break and ran into no further issues when restarting, unlike last semester when we had to quickly switch to build the prototype. There were no any major hiccups except for having the scale down the game, and more so when a team-member left.

2. Game Overview

Spelling Test will be a mixed-genre game, meshing together of top-down RPGs and twitch-based roguelikes and arcade shooters. We wanted to bring something fresh to the table and Spelling Test will bring about a unique feeling.

Trimester Two
As said above, the group discussed the project illustrated here was too ambitious so we scaled down the game, and had to do so more when David did not do the work assigned. Before, we wanted a Pokémon-like world which the player can explore and hop back to old locations to figure out puzzles that were previously inaccessible and also wanted different combat styles for each Elemental Master.

But as the group talked, we realised that the progression of Pokémon was linear and not choosing a choice in the game will affect the storyline. Also, with our concept of the Grand

Master having learnt all the spells of the previous Elemental Masters, it was not viable to merge them together and since we were scaling down – the solution was to make unique attacks for the two Elemental Masters and combining them into the final boss battle of the Grand Master, and having the storyline progress linearly with the entrance to a Master guarded by a unique puzzle. We kept the number of Elemental Masters low as it reduced stress on having to think of a unique ability of each one, something we agreed on to make more Masters if we had the time. We also fixed our Unity problem which was a relief since we could then do more with the game engine, as GameMaker 8.1 **Lite** was very restrictive.

2.1 Game Concept

As mentioned briefly above, the main concept of the game is to take the best parts of the RPG and Roguelike genres and integrate them to make a fun and functional game.

Spelling Test will have two main modes of play:
The World Map will be similar to that of classic old top-down RPGS (think Pokémon and the earlier Final Fantasy). In this mode, the player will be able to move around the world investigating points of interest. This could be in the form of NPCs to talk to, items to collect and many other things.

The Battle Map(s) will be a set of uniquely created levels with different mechanics that the player has to defeat to progress in the adventure and become stronger. These maps will be skill-based boss battles that will all be unique and require different strategies to defeat. The Binding of Isaac was one of the main influences in this aspect of the game.

2.2 Concept Justification

Deciding on our concept took quite a bit of discussion. Our group members play a variety of different games and there was no obvious consensus on making a game in any particular genre that we all loved.

The idea of mixing genres began to emerge as we talked instead about some of our favourite games and what we liked about them. RPGs, Roguelikes and challenging boss battles were the three aspects that linked our passions in the end and so, Spelling Test, a game that combines all of these elements began to evolve.

2.3 Target Market

We believe Spelling Test to be well suited to attract a variety of different player types. Both the top-down RPG and Roguelike are very popular genres currently and also have a sizable nostalgia factor for many players.

Spelling Test has the potential to pull in a solid player base from both these genres and also from players who like the idea of trying something new and unique. While we’re a long way from it, the concept of a game that does both Pokémon and Binding of Isaac well (both juggernaut franchises) should be very attractive indeed.

We hope our game will also be attractive (aesthetically and mechanically) to all age groups. We plan to use a simple colourful art style with cheerful music and sound to be as accessible as possible. This style is attractive to both a younger audience and a nostalgic older audience.

2.4 Research of Competition

Throughout this document we've mentioned already a couple of games who were major influences for us. Pokémon and The Binding of Isaac were the key building blocks for the two major parts of our game: Our interactive world map and our battle maps.

There were other games that also got us thinking too, however. The main one worth mentioning is Undertale. Although Nathanael is the only one who has played it, all of us are familiar with it and admire its style and unique gameplay. We hope to pay homage in some way to it in Spelling Test.

As mentioned previously, both the RPG and Roguelike genres are very popular at this time and there are many quality examples in recent years as well as gems from the past and so we think our game, if developed to a high standard and marketed properly has potential for a great start

As we develop Spelling Test, it will be important for us to look at the successful games of both genres and analyse both the design of the games themselves and the marketing and community side of things.

To our knowledge, no one has mixed these two genres before in the same way we have, at least not to any level of success. It's exciting to think that we could be pioneers.

2.5 Game Controls

We will be keeping the game controls very simple and accessible and will try to follow some genre conventions. As our game has two major gameplay modes, the control schemes for each will be different, but should follow a similar basic guideline.

Our control scheme has not been finalised and is even simpler in our prototype, so this section will likely be subject to change with the final submission next year.

World Map
Movement on the world map will be controlled with the traditional WASD keys. The world map will be shown from a top-down perspective and so the WASD keys will correspond to the cardinal directions (North, South etc.)

The player will also have an interactive button (E) which will be used for a variety of context-sensitive actions such as picking up objects and talking to NPCs.

The player will also be able to access various other screens from here which have not yet been fully designed or the controls mapped out.

Some likely examples would be:

- **Q** – Quest screen, brings up a screen that tells the player his next objective/visit location and details of any side quests he has started.
- **I** – Inventory screen - shows items/magic the player has collected.
- **P** – Stats screen - records time played, enemies defeated etc.

Battle Map
Movement on the battle map will also be controlled with the WASD keys, although this time the directions will correspond to left, right, up and down in the traditional side-scrolling fashion. As demonstrated in the prototype, the player will occupy the bottom portion of the battle map and the enemy will occupy the top portion.

The player will use SPACE to fire his currently selected spell and can hold SPACE to release a more powerful charged version of the spell.

The player will be able to switch between spells as he collects more throughout the game. These will likely be able to be toggled using the NUMBER keys (1-5).

The player will also be able to access the same menus as he can from the world map using the same keys which will pause the current fight.

There will likely be additional controls available on the Battle Maps depending on the ideas we come up with during development so as mentioned, this section is very much a work in progress.

2.6 Game Mechanics

2.6.1 Game Flow

The main flow of the gameplay, broadly speaking, will be the main character exploring the world and gathering the power necessary to 'win'. The details of the story will be discussed in that section, but some information might be repeated here to give a better context to the gameplay descriptions below.

Our main character is a young Wizard apprentice who has been outcast from the Wizard's Academy by his master who deems him weak and lacking the potential to become a true Wizard. Our hero must travel the world, find out information about the great Sites of Elemental Power and defeat the Guardians there to gain the strength to one day return to the Academy and prove his master wrong…

In gameplay terms, the player character has two main repeating tasks:

1. Discover information about the location of a Site of Elemental Power
2. Visit the Site and defeat the Guardian there to learn his secrets and add to his repertoire of spells

This gameplay loop will form the majority of the game, although we hope to add enough diversity to the questing and the combat mechanics to keep things interesting. The game will culminate in a final showdown against the Master Wizard where the player must use all his knowledge and gained spells to defeat him in a final boss battle.

2.6.2 Gameplay

The game flow summary above hopefully gave a rough guide to Spelling Test. We'll talk a bit more about the gameplay in detail in this section. As well as the main *Pokémon*-style world map where the player will spend most of his time, there will also be an overarching 'hub' map where the player can fast travel to locations he has discovered or already visited.

In terms of the gameplay loop mentioned in the last section, the player will visit a location (a town for example), discover information about a nearby Site of Power (a Volcano for example) and that Site will then become available to fast travel to on the hub map. The player will then be able to explore the area as normal in the main map and on finding the Guardian, will move to the designed battle map for that area.

The main driving mechanic behind the game will be adding new Spells to the player's collection. The player will start with a basic projectile spell and will gain new ones as the world is explored and the Site Guardian Wizards are defeated. The player may also be able to gain spells from possible side quests and exploration, but the main idea to visit a number of unique element-based locations (levels) and gain spells based on that element. For example:

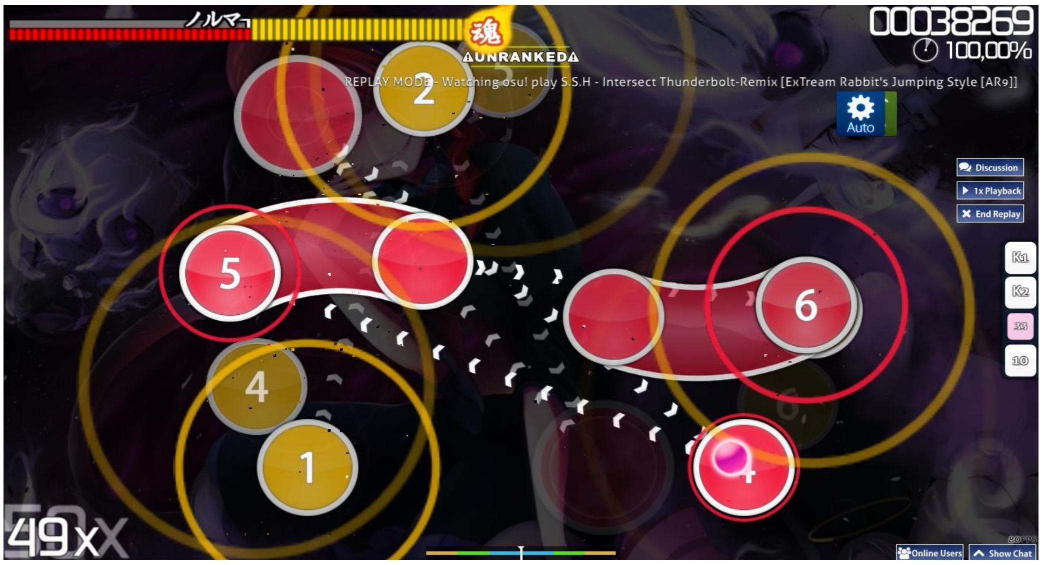
- Volcano (Fire)
- Underwater city (Water)
- Mountain top (Lightning)
- Cave (Earth)
- Desert (Wind)

If possible, we hope to make the spell system quite detailed with different types of attacks and counters possible based on the element. The attacks might each be quite different (Fireballs, Water Sprays, Spinning Wind etc.) and could react differently depending on the spells the opponent uses.

This system is very ambitious however and we may have to scale it back to a simple counters system, e.g. water attacks vs the Fire Wizard do more damage. If we cannot implement the full system we still hope to create more than just simple attack spells and also have utility spells like Shields and Heals available to add an extra layer of fun and complexity. I'll share here some early round-table design work to give an idea of the kind of thinking that went into our proposed system:

- As you learn a type of magic, get a set amount of charges per match.
- Fire - Throw a fireball making ricochets on platforms, after X ricochets, the fireball disappears (keyboard)
- Lightning -
  - Enemy puts up a shield cover, shield cover has gaps in it, click gaps to damage
  - Electrical charges in air and have to click to focus them into final spell
  - Aim and focus a big lightning bolt from sky.
  - Will have a big warning indicator and enemy will try to dodge. Have to control with left and right (that speeds up as you press the key so it's hard to control) to hit the enemy.
- Earth -
  - Building blocks (Tetris?)
  - Summon an earth golem (golem can be controlled?)
  - Shoot out flurry of shrapnel/stones

- Water - set slide pattern which you have to keep to (mouse) like *Osu!*


00038269
100,00%
UNRANKED
REPLAY MODE - Watching osu! play S.S.H - Intersect Thunderbolt-Remix [ExTream Rabbit's Jumping Style [AR9]]
Auto
Discussion
1x Playback
End Replay
49x
Online Users
Show Chat

- Obvious counters -
  - Water deals double damage to fire
    - When using a counter, enrages the Guardian
      - Attacks more often
      - Hits harder
      - After being hit, the Guardian will recognise his weakness and try to evade more to not get hit
- Instead of lots of different elements, can focus on a couple and add more different interactions with them?
- Further spelling test pun - use letters as a mechanic for creating spells?

2.6.3 Summary

To summarize this section, Spelling Test will be a mechanically ambitious game. Implementing a detailed magic system and brainstorming many different challenging boss fights will be a real challenge for the team, but something we foresee as being a lot of fun. Undoubtedly some of these ambitious plans will have to be refined, but with a bit of work and research we hope we can do the original idea justice.

2.7 Story and Theme

2.7.1 Theme

Spelling Test's theme is one of a simple, colourful and cheerful adventure in a pleasant land. Dark Lords and evil-filled lands have been overdone, so our tale is more one of determination and proving an old teacher wrong rather than saving the world.

2.7.2 Story and Narrative

Spelling Test is set in a high-fantasy world of magic and wonder. The main protagonist and player character is the Apprentice Wizard. The Apprentice Wizard can be male or female and is named by the player at the start of the game, but for now let's call him Gavin.

Young Gavin is kicked out of the great Wizard Academy by the Grand Master there, having been told he does not have the skill to ever make it as a full Wizard and should take up an honest profession like farming instead. Despondent, young Gavin trudges off unsure what to do. By chance he meets a mysterious stranger that leads him to a conclave of Outcast Wizards who live in a forest nearby. The Wizard Academy is very strict and tends to cull students on a regular basis without giving them a second chance.

Most of the Outcasts in the forest are indeed people with only basic magic ability and no potential to do more than the simplest things and Gavin prepares to accept that he is the same and will live with them now. The village Elder however, after consulting the Mists of Magic, senses a great hidden well of power within Gavin, undetectable to most Wizards. In fact, he has only sensed this odd hidden power once before in his long life...in the current Grand Master of the Wizard Academy, his once friend!

He eagerly tells young Gavin of the Elemental Sites of Power scattered throughout the world and protected by powerful Guardians. Once, long ago, his friend Elrik, the current Grand Master went on a great journey to visit and defeat these Guardians and claim their power for themselves. This is a very rare occurrence as only 7 Wizards in the 400 years since Magic was discovered have ever defeated all of the Ancient Guardians and declared themselves a Grand Master of Magic.

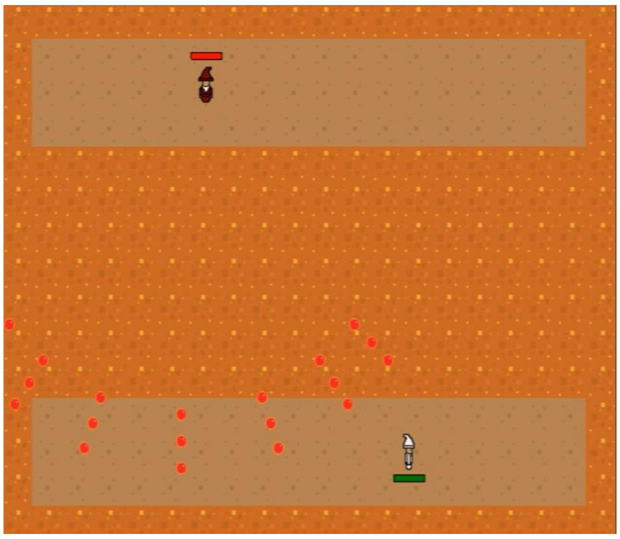
It is time for young Gavin to begin his journey to become the 8th Grand master. He must seek out new friends, new lands and new challenges and become strong enough to defeat each of the Guardians. Only then can he prove his former Master wrong and perhaps take his place as a kind and just Grand Master of the Academy.

2.8 Level Design

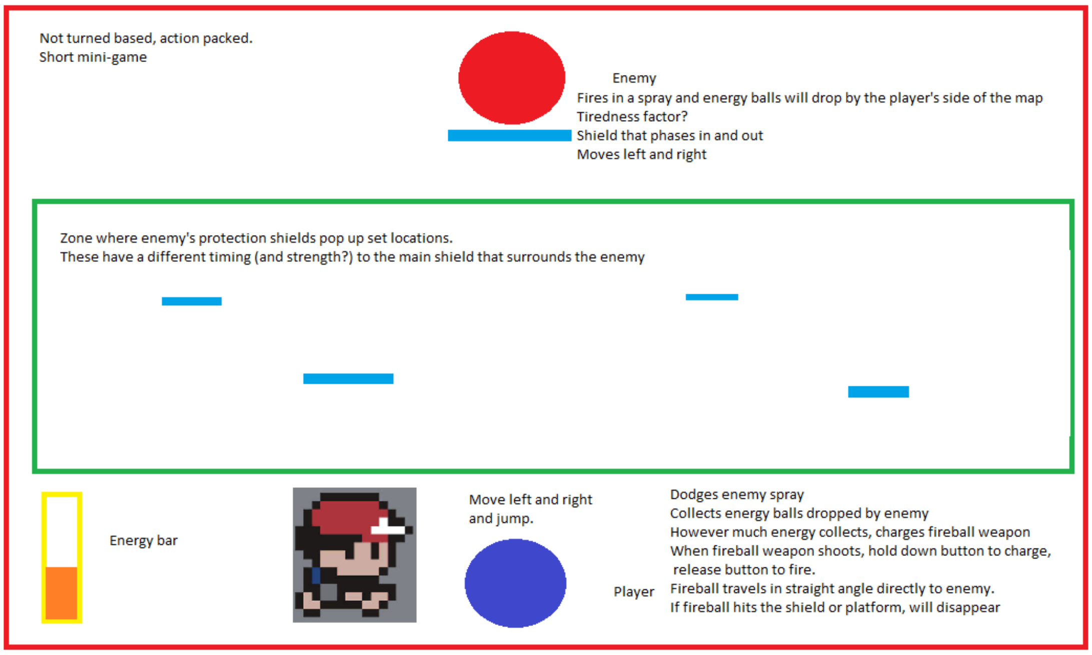
Designing many different unique Battle Maps based on their elemental location will be where most of our level design work will lie. Our designs will start simple since our main character has only a basic starter spell, but we can steadily become more ambitious as the story progresses and the character gets different spells representing the different elements.

Speaking of elements, it will also be important to take into account the current location when working out our boss mechanics. A map based in a volcano will be very different than one in the frozen north and allows different opportunities for obstacles, traps and other aspects of the level.

I’ll post here a screenshot of our simple prototype design to give a rough idea of what a boss battle will look like. We hope of course to add a lot more detail, variety of spells, enemy attacks and background objects to make the levels as visually appealing as they are mechanically challenging.



We think it’s also worth sharing an early image of a level design as well as some rough notes we made below it:


Not turned based, action packed.
Short mini-game
Enemy
Fires in a spray and energy balls will drop by the player's side of the map
Tiredness factor?
Shield that phases in and out
Moves left and right
Zone where enemy's protection shields pop up set locations.
These have a different timing (and strength?) to the main shield that surrounds the enemy
Energy bar
Move left and right and jump.
Player
Dodges enemy spray
Collects energy balls dropped by enemy
However much energy collects, charges fireball weapon
When fireball weapon shoots, hold down button to charge, release button to fire.
Fireball travels in straight angle directly to enemy.
If fireball hits the shield or platform, will disappear

- Not turn-based, action-packed. Short mini-game. Enemy fires in a spray and energy balls will drop by the player’s side of the map. Tiredness factor?
- Zone where enemy's protection shields pop up at set locations. These have a different timing (and strength?) to the main shield that surrounds the enemy.
- Move left and right, dodges enemy spray and jump. Collects energy balls dropped by enemy.
- However much energy collects, charges fireball weapon. Energy bar.
- When fireball weapon shoots, hold down button to charge, release button to fire. Fireball travels in straight line directly to enemy. If fireball hits the shield or platform, will disappear.

3. Technical Design

3.1 Engine and Software

3.1.1 Unity

We planned to make the prototype in Unity and still hope to make our full game using this software so we feel it is worth discussing here.

We chose this engine for our game as it is, in many ways, the industry standard for 2D games in this current time. It had a fairly simple and intuitive interface once you get going with is and is a popular choice with smaller indie developers such as ourselves as there are a wealth of tutorials and support available from both the software developers themselves and the larger community.

Not only does it suit our game well, it suits us well as aspiring professionals to begin to familiarise ourselves with one of the largest and most well-known game engines out there. This experience will undoubtedly serve us well in the future in both our honours year and in future employment.

3.1.2 Visual Studio

To develop the game, scripts are necessary to create artificial intelligences but also to enable the player to attack and manage some of the collisions. Unity has a specific library available on Visual Studio permitting to easily manipulate the game objects.

3.1.3 GameMaker 8.1 Lite

As mentioned earlier, the team had some major issues syncing our work on Unity and had to change plans late in the day and develop our prototype on GameMaker instead.
We all had some experience is working with this software before so the switch was not overly difficult except the limited amount of time we had left.

GameMaker is very capable software in its own right though and more intuitive to get to grips with than most. We have not yet decided if it will be better to continue using GameMaker next term, but if we do use it, we will be creating the game entirely from GameMaker script to give us finer control and test our skills a bit more.

3.1.4 Photoshop

The team will be using these image manipulation programs depending on personal preference for asset creation and modification. Each member has some experience with either or both, so we should be able to divide this sort of work well.

3.1.5 Audacity

We will be using Audacity to create and edit simple sound effects and basic music. David has some experience in it, but the whole team would like to familiarise themselves a bit more with it as it is an excellent piece of free audio software.

We may also be using more complex professional audio software (Pro tools, Ableton Live 9 and Logic Pro X) in conjunction with a friend of Nathanael's who is studying Music Technology. If we do manage to team up in this way to create some unique high quality sound and music, we will update this section with the software used.

3.2 Platforms and Hardware Specifications

3.2.1 Platform

Spelling Test could suit a variety of platforms. We will be developing it primarily with PC in mind, but it could easily port over to Mobile or perhaps even the console market (although this latter would be very difficult and unlikely without expanding the team).

The game would translate well to controller play on PCs and is simple enough to work well on Mobiles or handhelds too. We could also work on Mac and Linux ports if there was a demand for it and the initial PC release proved successful.

3.2.2 Specifications

Spelling Test will be a very small, compact game that should not be a strain on any but the oldest of PCs. With a simple pixel art style, chiptune music and sound and not an overly large amount of levels, we're confident that very few people will have issues running it. As an example, Undertale, a game similar in style but much larger and more complex has the following recommended requirements:

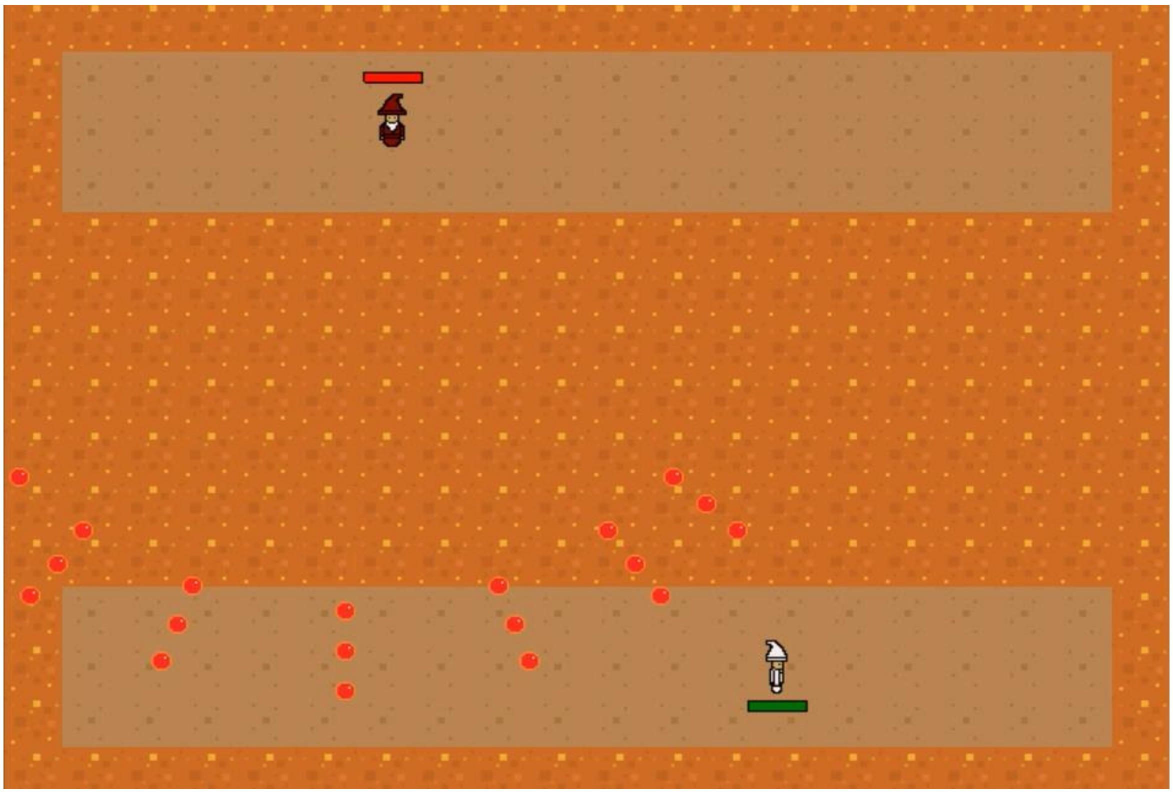
- OS: Windows XP, Vista, 7, or 10.
- Processor: 2GHz+
- Memory: 3 GB RAM.
- Graphics: 512MB.
- Storage: 200MB available space

We think this is a solid guideline for the running requirements of Spelling Test, although it would lightly run on weaker systems than that.

3.3 Visual Design

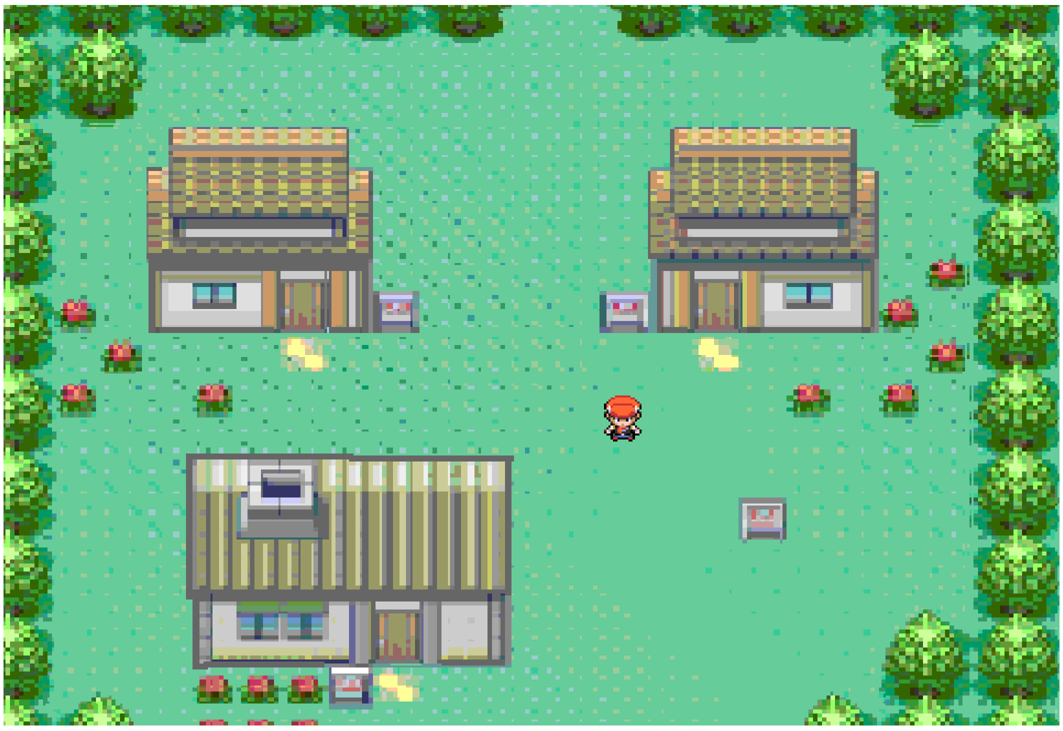
As mentioned above, Spelling Test will be using a simple pixel-art style with basic sprite characters, backgrounds and other assets. It will very much embrace a colourful, cheerful aesthetic, attractive to most audiences.

There will be some difference in visual styles between our World Map and Battle Map views to pay homage to their respective genres and the previously mentioned games that inspired us. The basic visual design of our sample battle map (each one will be different!) is shown in the Level Design section, but it's worth posting here for reference:



Bear in mind that this was a fairly rushed design in GameMaker based on running low on time. Our fully developed battle maps, regardless of the engine we decide to use, will be much more detailed both in visual quality and in the number of objects on screen.

We have not yet developed our world map view as we decided getting the battle maps right was most important as they are very much the meat of the game. It is likely we will go for something in the *Pokémon* style like the example below:



3.4 Audio Design

For now, our plan is to use free stock music and sound effects to populate the audio landscape of our game but that could change if we are able to collaborate with Nathanael's Music Tech friend.

Like the visual style, we want to keep the audio simple and cheery with 16-bit chiptunes and simple 90's console era sound effects. We could potentially have some basic voice acting the in the game for a bit of fun, but unless we could find talented students, we fear that this could reduce the enjoyability of the game!

One of our early game ideas, based on Nathanael's friend's assistance, was a game heavily based on audio cues. It could be possible to make a very unique and interesting boss fight using this idea and it's something the team are currently considering the viability of.

3.5 Scripting

Trimester Two
Scripting was used to create the player's and the elemental wizards' different attacks, movements systems, update the animations but also manage the collisions between the different objects composing the game. A commented part of the code is available in the Appendix part to show how the scripts are build and used, the game works object oriented programming.

4. Version Control

We decided to use GitHub as our main method of version control. Nathanael had the most experience using it and was able to provide tutorials to help his group mates familiarise themselves a bit more with it.

It should prove very useful for both this module and for work in the future, similar to Unity. Our repository can be found here: https://github.com/Dyanek/SpellingTest

Trimester Two
The project was mainly built by Nathanael and Yu-Ching, in a new GitHub repository: https://github.com/Dyanek/UnitySpellingTest; David did not do much work as evidenced inside the Insights Contributors. Trello and WhatsApp were mainly used for tasks for the week and communicating when each part was doing and done, as to not clash with other team-members.

5. Testing

As is fairly common in student environments, we decided to use the Agile Development methodology as our guideline for both developing and testing Spelling Test. Our weekly lab meetings and our early implementation of GitHub meant that we were able to look over and modify each other's work as necessary and provide support where needed. We did discover the Unity issue fairly late on and tried to address it over a couple of weeks before making the switch to GameMaker, but this could have ended up worse if our team communication had been poor.

There was not a lot to test in our simple GameMaker prototype, but we did show it to a handful of people and each play-tested it ourselves too. Obviously testing will be a much bigger part next trimester and we are currently working on a bit of more of a detailed development cycle to make sure everything runs as smoothly and efficiently as possible.

Trimester Two

The game was alpha tested by the team after every GitHub push to make sure everything was working. We ran into two bugs that were kept in as they do not affect the gameplay. The first bug is in the Fire Puzzle Scene where the player does initially collide with the puzzle wall, but after a second, will pass through the wall; this bug was only seen once and could not be replicated since. The second *bug* is the Fire and Grand Master shooting out the spray, the particles are not all correctly aligned.

For beta testing, the game was handed to our friends and the feedback was that it was a bit too easy to defeat the Masters. Thus we played with the health system and decided on having it so the player could continue as much as the player liked, with the option to quit after each death.

The bugs were not noticed by the testers as the second bug was more alignment, and after players bounced off the puzzle wall the first time, they did not try it again and focused more on completing the maze.

6. Marketing

For a game as small as Spelling Test, it would likely be best to release the game for free on a website designed to showcase smaller games, such as a flash games site. Similarly, releasing the game as a free app on mobile will allow the game to be seen by a wide audience and give us some immediate recognition and feedback.

It might be possible to earn some small amount of advertisement revenue in this way which we could possible use to continue to develop the game to a higher standard and maybe an eventual cheap (£1 or less) release on a platform like Steam.

If we are enjoying continuing to develop and our initial release is successful, we can spread the word through our contacts on LinkedIn and the UWS Game Dev Society. It might be possible to get some wider recognition if we consider our game good enough by posting on indie game development sites and perhaps attempting to contact YouTubers/streamers who specialise in showing off indie games. Definitely would not hurt to fire off as many free demo copies as we can to the wider internet.

A more immediate and tangible goal is market research amongst the students of UWS. As mentioned in the Testing section, sending out surveys and demos could help us streamline our game to what the people want and also has the advantage of doubling as testing as people give us feedback on early versions.

7. Minutes of Meetings

We have created separate documents containing the details of each week's meeting which we will also submit but have decided to record the information in this document:
Colours key:
Green: On schedule, no problems encountered

Amber: Some problems or issues have been encountered and dealt with. The plan may need to be updated.
Red: Project stalled. Immediate action is required to get it back on track.

7.1 Week01 – 06/09/2017 – 1st Semester

Week01 was admin week, so we began at Week02.

7.2 Week02 – 13/09/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

13/09/16, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
First meeting:

- Introduced ourselves and set up group WhatsApp
- Filled in Team Skills Inventory form
- Brainstormed 3 potential ideas

Work planned before next meeting

- Flesh out ideas a bit further
- Create shared Google Drive

Any Other Business

- Look into GitHub or alternative

7.3 Week03 – 20/09/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

20/09/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- 3 ideas fleshed out
- Final idea chosen (using elements of each)
- Proposed plan and basic overview completed

Work planned before next meeting
- Become familiar with GitHub
- Continue adding to Game Overview doc
- Consider template for Game Design doc

Any Other Business
- Install / update Unity

7.4 Week04 – 27/09/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

27/09/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Game Design Document template found
- Draft Gantt completed

Work planned before next meeting
- Continue skeleton of the document (headings and bullet points)
- Flesh out details where possible (many areas still in flux)

Any Other Business
- Install / update Unity

7.5 Week05 – 02/10/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

02/10/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Game Design Document continued

Work planned before next meeting
- Familiarising with GitHub
- Adding more fine detail to Game Design Document
- Discuss story in group chat?

Any Other Business
- Work through some Unity tutorials
  (Ready to begin basic prototype design work next week)

7.6 Week06 – 09/10/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

09/10/17, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
David was ill

Project Status:
Green

Work completed since previous meeting:
- Game Design Document continued
- Trello created

Work planned before next meeting
- Familiarising with GitHub
- Adding more fine detail to Game Design Document
- Familiarising with Unity

Any Other Business

7.7 Week07 – 16/10/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

16/10/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Game Design Document continued

Work planned before next meeting
- Adding more fine detail to Game Design Document
- Familiarising with Unity

Any Other Business

7.8 Week08 – 23/10/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

23/10/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Basic mechanics created on Unity

Work planned before next meeting
- Begin developing the 1st phase

Any Other Business

7.9 Week09 – 30/10/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

30/10/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Unity project pushed onto GitHub

Work planned before next meeting
- Begin developing the 1st phase

Any Other Business

7.10 Week10 – 06/11/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

06/11/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Tutorials of Unity

Work planned before next meeting
- Begin developing the 1st phase

Any Other Business

7.11 Week11 – 13/11/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

13/11/17, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Animations created and added to the game
- Fight zone defined

Work planned before next meeting
- Add AI and fireball sprites

Any Other Business

7.12 Week12 – 20/11/2017

COMP09097 Games Development Project

Progress Meeting - Games Development Project

20/11/17, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
David was ill

Project Status:
Amber

Work completed since previous meeting:

- Game engine changed to GameMaker 8.1 Lite
- Creating a part of the prototype
- Beginning the PowerPoint presentation

Work planned before next meeting

- Continue prototype and PowerPoint

Any Other Business

7.13 Week13 – 10/01/2017 – 2nd Semester

COMP09097 Games Development Project

Progress Meeting - Games Development Project

10/01/18, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- After finalisation of prototype on GameMaker 8.1 Lite, the team talked about the best game engine that can be used.

Work planned before next meeting
- Try and solve Unity problem
- Determine best game engine to use with the data available

Any Other Business

7.14 Week14 – 17/01/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

17/01/18, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Updated Unity to correct current version
- Added some basic functions to GitHub

Work planned before next meeting
- Complete tutorial

Any Other Business

7.15 Week15 – 24/01/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

24/01/18, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Tutorial progressed
- GitHub issues resolved
- Trello updated

Work planned before next meeting
- David finishes tutorial
- Yu-Ching designs over-world map
- Nathanael works on enemy AI

Any Other Business

7.16 Week16 – 31/01/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

31/01/18, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:

- New background designed (can be used as template for other levels)
- First pass basic Fire AI
- Movement animation smoothing

Work planned before next meeting

- Complete Fire AI
- Design Fire Maze
- Complete Tutorial (speech, collision)

Any Other Business

7.17 Week17 – 07/02/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

07/02/18, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
David was ill

Project Status:
Green

Work completed since previous meeting:
- Advanced the Fire AI
- Redefined scene's edges for the player

Work planned before next meeting
- Finish the Fire AI
- Design Fire Maze
- Complete tutorial (speech, collision)
- Add life bars

Any Other Business

7.18 Week18 – 14/02/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

14/02/18, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
David was ill

Project Status:
Green

Work completed since previous meeting:
- Scaled down game
- Removed over-world map

Work planned before next meeting
- Design Water map
- Design Grand Master map
- Complete tutorial (speech, collision)
- Add life bars

Any Other Business

7.19 Week19 – 21/02/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

21/02/18, 10am, UWS Paisley Campus E113

Those attending:
David Byron
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Created assets for the water battle
- Complete the Fire puzzle
- Finished the Fire AI
- Added menu

Work planned before next meeting
- Create the Water AI
- Create the Grand Master assets
- Complete tutorial (speech, collision)
- Add life bars

Any Other Business

7.20 Week20 – 28/02/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

28/02/18, 10am, UWS Paisley Campus E113

Those attending:

Apologies for absence:
Snowed in

Project Status:
Green

Work completed since previous meeting:
- Tutorial reassigned
- Life bars reassigned
- Water AI created

Work planned before next meeting
- Create the Grand Master AI
- Create the cut-scenes

Any Other Business

7.21 Week21 – 07/03/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

07/03/18, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
David was unsure about continuing

Project Status:
Green

Work completed since previous meeting:
- Water and Grand Master Battles done
- Grand Master assets created
- Cut-scenes added
- Life system and bars added
- Water Puzzle done

Work planned before next meeting
- Add player attacks
- Add audio

Any Other Business

7.22 Week22 – 14/03/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

14/03/18, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

Work completed since previous meeting:
- Player attack
- Water Puzzle refined

Work planned before next meeting
- Add player attack sounds
- Add audio
- HUD menu

Any Other Business

7.23 Week23 – 21/03/2018

COMP09097 Games Development Project

Progress Meeting - Games Development Project

21/03/18, 10am, UWS Paisley Campus E113

Those attending:
Yu-Ching Ho
Nathanael Omnes

Apologies for absence:
N/A

Project Status:
Green

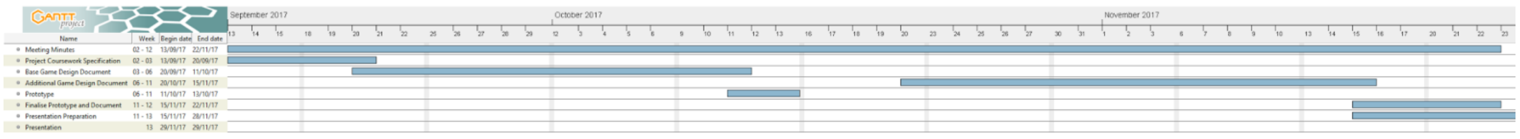
Work completed since previous meeting:
- HUD Menu
- Player attack sounds

Work planned before next meeting
- Finish off player attack sounds
- Prepare for presentation
- Video

Any Other Business

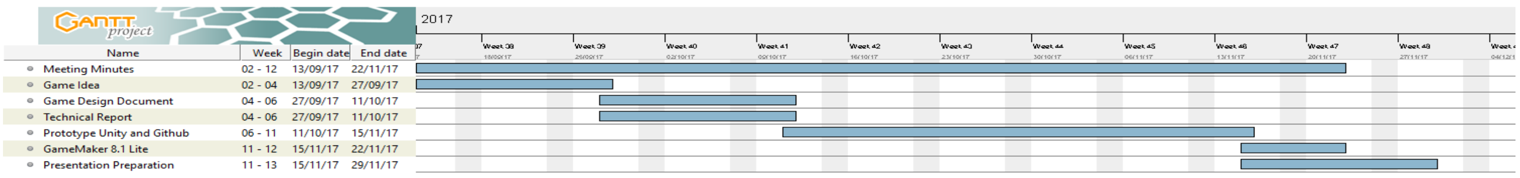
8. Appendix

Trimester One Draft Gantt Chart



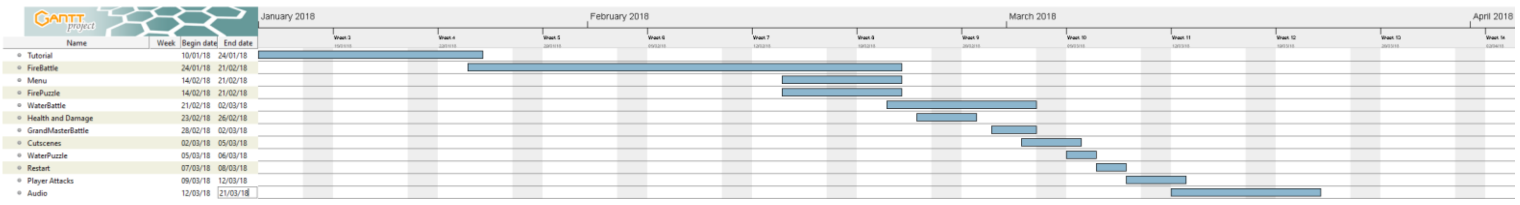
Name	Week	Begin date	End date
Meeting Minutes	02 - 12	13/09/17	22/11/17
Project Coursework Specification	02 - 03	13/09/17	20/09/17
Base Game Design Document	03 - 06	20/09/17	11/10/17
Additional Game Design Document	06 - 11	20/10/17	15/11/17
Prototype	06 - 11	11/10/17	13/10/17
Finalise Prototype and Document	11 - 12	15/11/17	22/11/17
Presentation Preparation	11 - 13	15/11/17	28/11/17
Presentation	13	29/11/17	29/11/17

Trimester One Actual Gantt Chart



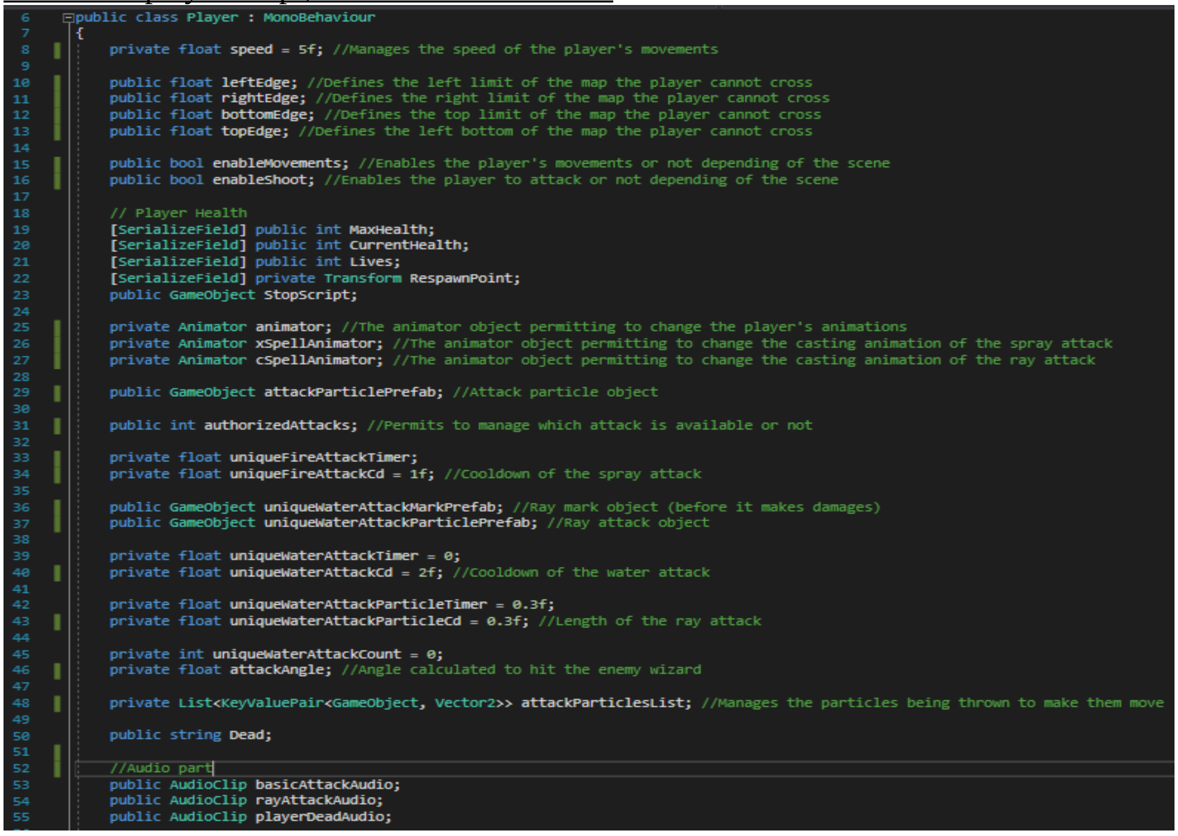
Name	Week	Begin date	End date
Meeting Minutes	02 - 12	13/09/17	22/11/17
Game Idea	02 - 04	13/09/17	27/09/17
Game Design Document	04 - 06	27/09/17	11/10/17
Technical Report	04 - 06	27/09/17	11/10/17
Prototype Unity and Github	06 - 11	11/10/17	15/11/17
GameMaker 8.1 Lite	11 - 12	15/11/17	22/11/17
Presentation Preparation	11 - 13	15/11/17	29/11/17

Trimester Two Actual Gantt Chart



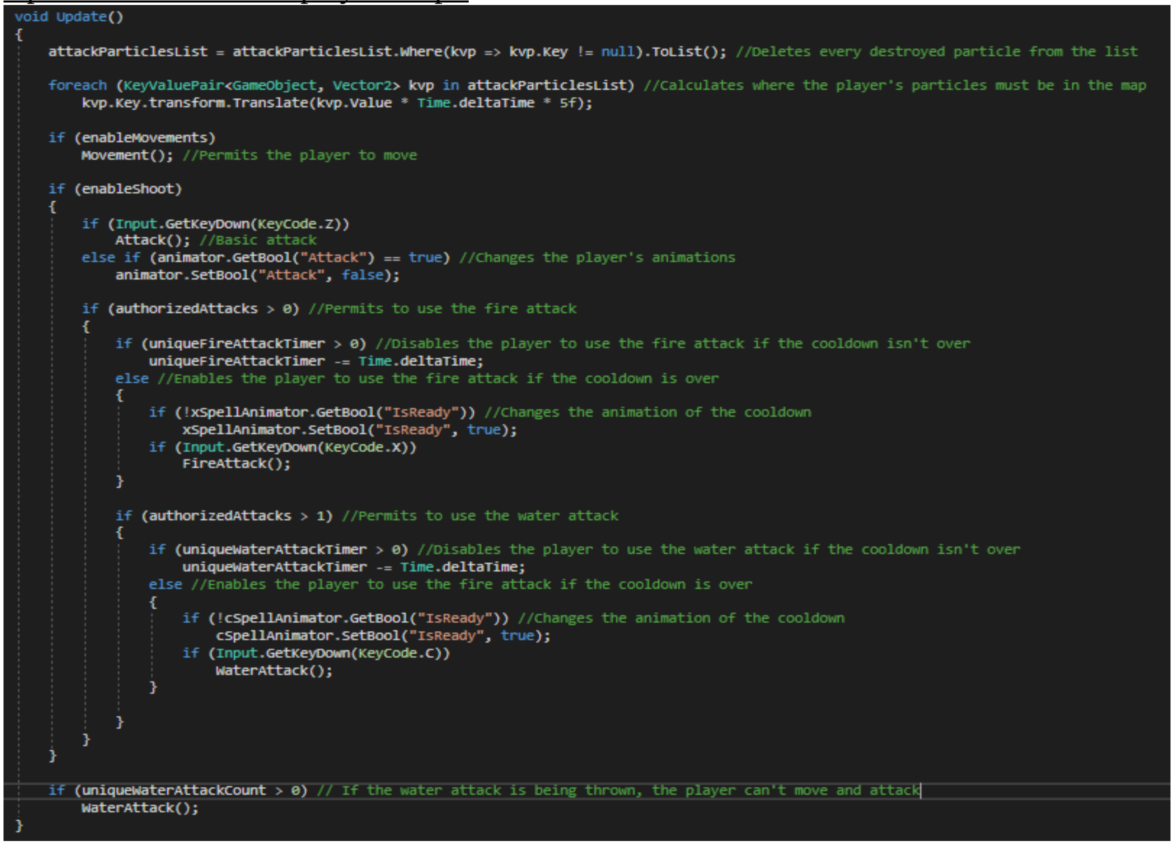
Name	Week	Begin date	End date
Tutorial		10/01/18	24/01/18
FireBattle		24/01/18	21/02/18
Menu		14/02/18	21/02/18
FirePuzzle		14/02/18	21/02/18
WaterBattle		21/02/18	02/03/18
Health and Damage		23/02/18	26/02/18
GrandMasterBattle		28/02/18	02/03/18
Cutscenes		02/03/18	05/03/18
WaterPuzzle		05/03/18	06/03/18
Restart		07/03/18	08/03/18
Player Attacks		09/03/18	12/03/18
Audio		12/03/18	21/03/18

Part of the player script, variables initialisation:



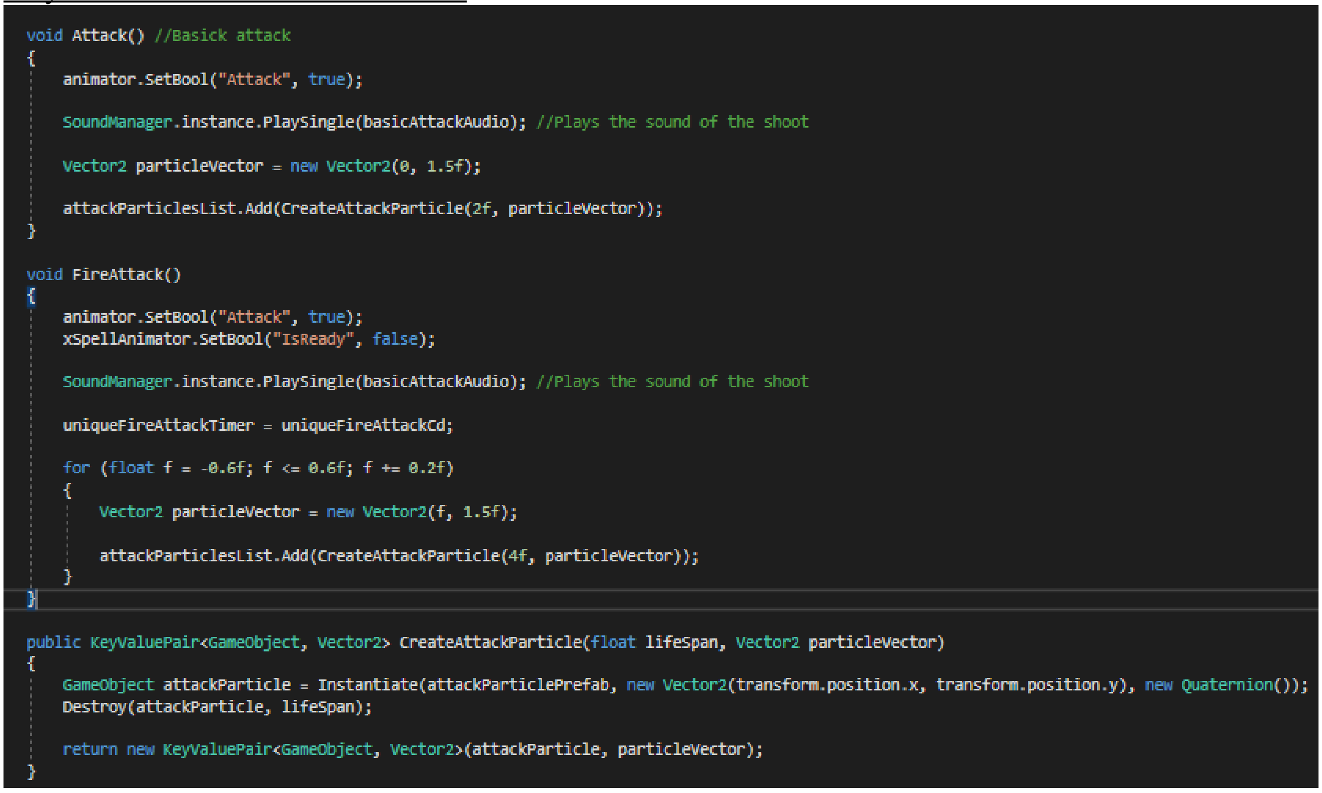
```
public class Player : MonoBehaviour
{
    private float speed = 5f; //Manages the speed of the player's movements

    public float leftEdge; //Defines the left limit of the map the player cannot cross
    public float rightEdge; //Defines the right limit of the map the player cannot cross
    public float bottomEdge; //Defines the top limit of the map the player cannot cross
    public float topEdge; //Defines the left bottom of the map the player cannot cross

    public bool enableMovements; //Enables the player's movements or not depending of the scene
    public bool enableShoot; //Enables the player to attack or not depending of the scene

    // Player Health
    [SerializeField] public int MaxHealth;
    [SerializeField] public int CurrentHealth;
    [SerializeField] public int Lives;
    [SerializeField] private Transform RespawnPoint;
    public GameObject StopScript;

    private Animator animator; //The animator object permitting to change the player's animations
    private Animator xSpellAnimator; //The animator object permitting to change the casting animation of the spray attack
    private Animator cSpellAnimator; //The animator object permitting to change the casting animation of the ray attack

    public GameObject attackParticlePrefab; //Attack particle object

    public int authorizedAttacks; //Permits to manage which attack is available or not

    private float uniqueFireAttackTimer;
    private float uniqueFireAttackCd = 1f; //Cooldown of the spray attack

    public GameObject uniqueWaterAttackMarkPrefab; //Ray mark object (before it makes damages)
    public GameObject uniqueWaterAttackParticlePrefab; //Ray attack object

    private float uniqueWaterAttackTimer = 0;
    private float uniqueWaterAttackCd = 2f; //Cooldown of the water attack

    private float uniqueWaterAttackParticleTimer = 0.3f;
    private float uniqueWaterAttackParticleCd = 0.3f; //Length of the ray attack

    private int uniqueWaterAttackCount = 0;
    private float attackAngle; //Angle calculated to hit the enemy wizard

    private List<KeyValuePair<GameObject, Vector2>> attackParticlesList; //Manages the particles being thrown to make them move

    public string Dead;

    //Audio part
    public AudioClip basicAttackAudio;
    public AudioClip rayAttackAudio;
    public AudioClip playerDeadAudio;
```

Update function of the player script:



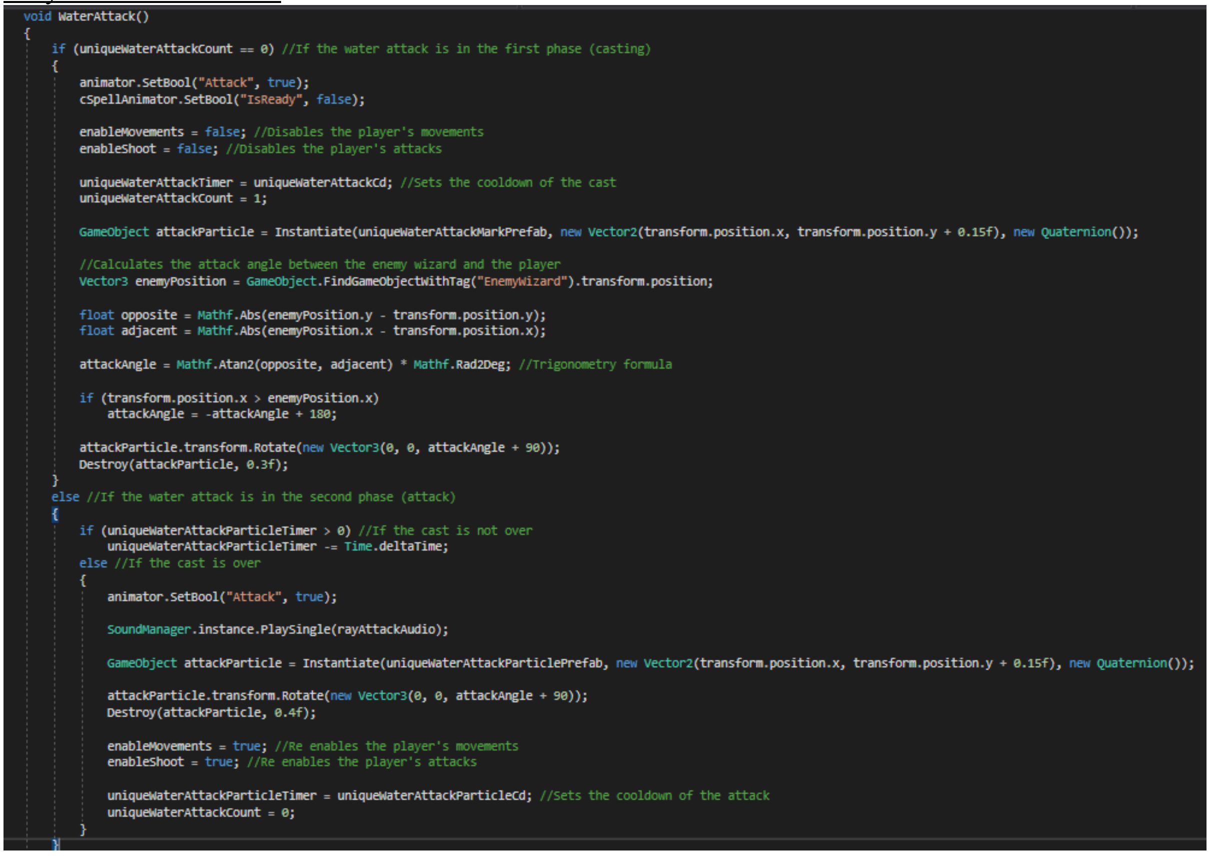
```
void Update()
{
    attackParticlesList = attackParticlesList.Where(kvp => kvp.Key != null).ToList(); //Deletes every destroyed particle from the list

    foreach (KeyValuePair<GameObject, Vector2> kvp in attackParticlesList) //Calculates where the player's particles must be in the map
        kvp.Key.transform.Translate(kvp.Value * Time.deltaTime * 5f);

    if (enableMovements)
        Movement(); //Permits the player to move

    if (enableShoot)
    {
        if (Input.GetKeyDown(KeyCode.Z))
            Attack(); //Basic attack
        else if (animator.GetBool("Attack") == true) //Changes the player's animations
            animator.SetBool("Attack", false);

        if (authorizedAttacks > 0) //Permits to use the fire attack
        {
            if (uniqueFireAttackTimer > 0) //Disables the player to use the fire attack if the cooldown isn't over
                uniqueFireAttackTimer -= Time.deltaTime;
            else //Enables the player to use the fire attack if the cooldown is over
            {
                if (!xSpellAnimator.GetBool("IsReady")) //Changes the animation of the cooldown
                    xSpellAnimator.SetBool("IsReady", true);
                if (Input.GetKeyDown(KeyCode.X))
                    FireAttack();
            }

            if (authorizedAttacks > 1) //Permits to use the water attack
            {
                if (uniqueWaterAttackTimer > 0) //Disables the player to use the water attack if the cooldown isn't over
                    uniqueWaterAttackTimer -= Time.deltaTime;
                else //Enables the player to use the fire attack if the cooldown is over
                {
                    if (!cSpellAnimator.GetBool("IsReady")) //Changes the animation of the cooldown
                        cSpellAnimator.SetBool("IsReady", true);
                    if (Input.GetKeyDown(KeyCode.C))
                        WaterAttack();
                }

            }
        }
    }

    if (uniqueWaterAttackCount > 0) // If the water attack is being thrown, the player can't move and attack
        WaterAttack();
}
```

Player’s basic attack and fire attack:



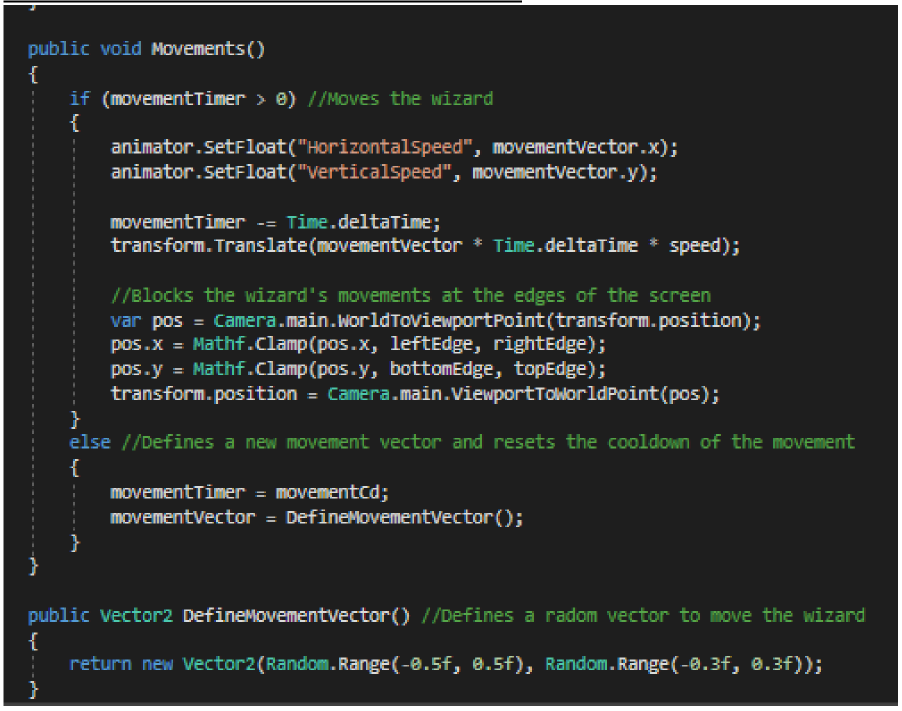
```
void Attack() //Basick attack
{
    animator.SetBool("Attack", true);

    SoundManager.instance.PlaySingle(basicAttackAudio); //Plays the sound of the shoot

    Vector2 particleVector = new Vector2(0, 1.5f);

    attackParticlesList.Add(CreateAttackParticle(2f, particleVector));
}

void FireAttack()
{
    animator.SetBool("Attack", true);
    xSpellAnimator.SetBool("IsReady", false);

    SoundManager.instance.PlaySingle(basicAttackAudio); //Plays the sound of the shoot

    uniqueFireAttackTimer = uniqueFireAttackCd;

    for (float f = -0.6f; f <= 0.6f; f += 0.2f)
    {
        Vector2 particleVector = new Vector2(f, 1.5f);

        attackParticlesList.Add(CreateAttackParticle(4f, particleVector));
    }
}

public KeyValuePair<GameObject, Vector2> CreateAttackParticle(float lifeSpan, Vector2 particleVector)
{
    GameObject attackParticle = Instantiate(attackParticlePrefab, new Vector2(transform.position.x, transform.position.y), new Quaternion());
    Destroy(attackParticle, lifeSpan);

    return new KeyValuePair<GameObject, Vector2>(attackParticle, particleVector);
}
```

Player’s water attack:



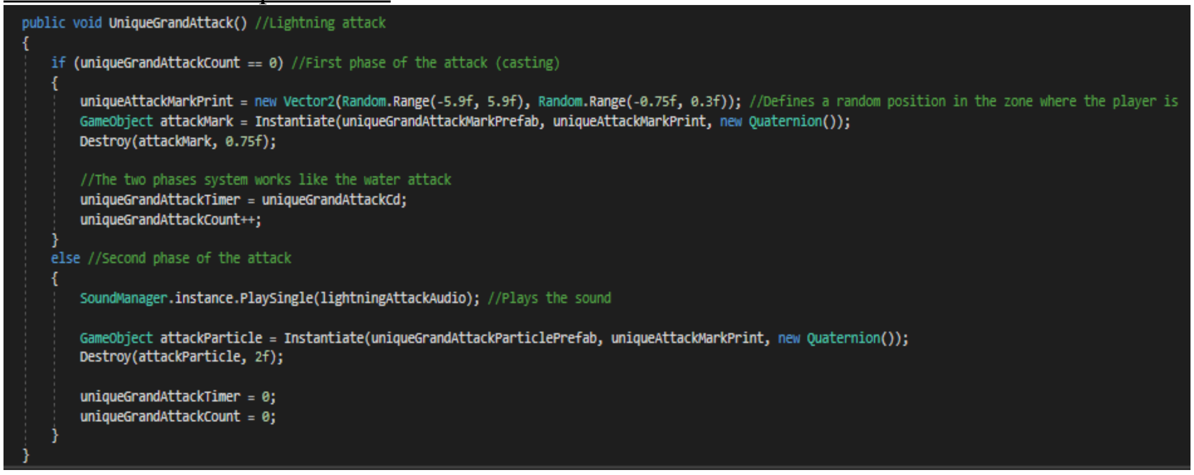
```
void WaterAttack()
{
    if (uniqueWaterAttackCount == 0) //If the water attack is in the first phase (casting)
    {
        animator.SetBool("Attack", true);
        cSpellAnimator.SetBool("IsReady", false);

        enableMovements = false; //Disables the player's movements
        enableShoot = false; //Disables the player's attacks

        uniqueWaterAttackTimer = uniqueWaterAttackCd; //Sets the cooldown of the cast
        uniqueWaterAttackCount = 1;

        GameObject attackParticle = Instantiate(uniqueWaterAttackMarkPrefab, new Vector2(transform.position.x, transform.position.y + 0.15f), new Quaternion());

        //Calculates the attack angle between the enemy wizard and the player
        Vector3 enemyPosition = GameObject.FindGameObjectWithTag("EnemyWizard").transform.position;

        float opposite = Mathf.Abs(enemyPosition.y - transform.position.y);
        float adjacent = Mathf.Abs(enemyPosition.x - transform.position.x);

        attackAngle = Mathf.Atan2(opposite, adjacent) * Mathf.Rad2Deg; //Trigonometry formula

        if (transform.position.x > enemyPosition.x)
            attackAngle = -attackAngle + 180;

        attackParticle.transform.Rotate(new Vector3(0, 0, attackAngle + 90));
        Destroy(attackParticle, 0.3f);
    }
    else //If the water attack is in the second phase (attack)
    {
        if (uniqueWaterAttackParticleTimer > 0) //If the cast is not over
            uniqueWaterAttackParticleTimer -= Time.deltaTime;
        else //If the cast is over
        {
            animator.SetBool("Attack", true);

            SoundManager.instance.PlaySingle(rayAttackAudio);

            GameObject attackParticle = Instantiate(uniqueWaterAttackParticlePrefab, new Vector2(transform.position.x, transform.position.y + 0.15f), new Quaternion());

            attackParticle.transform.Rotate(new Vector3(0, 0, attackAngle + 90));
            Destroy(attackParticle, 0.4f);

            enableMovements = true; //Re enables the player's movements
            enableShoot = true; //Re enables the player's attacks

            uniqueWaterAttackParticleTimer = uniqueWaterAttackParticleCd; //Sets the cooldown of the attack
            uniqueWaterAttackCount = 0;
        }
    }
```

Elemental wizards’ random movements:



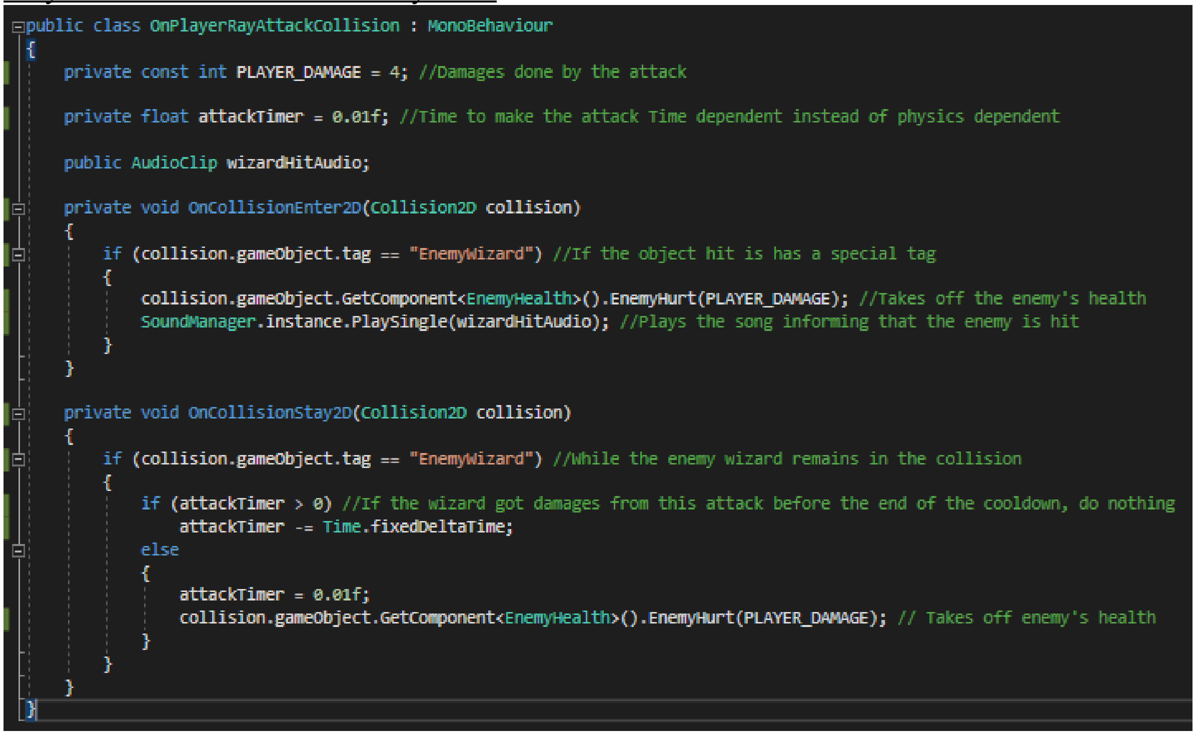
```
public void Movements()
{
    if (movementTimer > 0) //Moves the wizard
    {
        animator.SetFloat("HorizontalSpeed", movementVector.x);
        animator.SetFloat("VerticalSpeed", movementVector.y);

        movementTimer -= Time.deltaTime;
        transform.Translate(movementVector * Time.deltaTime * speed);

        //Blocks the wizard's movements at the edges of the screen
        var pos = Camera.main.WorldToViewportPoint(transform.position);
        pos.x = Mathf.Clamp(pos.x, leftEdge, rightEdge);
        pos.y = Mathf.Clamp(pos.y, bottomEdge, topEdge);
        transform.position = Camera.main.ViewportToWorldPoint(pos);
    }
    else //Defines a new movement vector and resets the cooldown of the movement
    {
        movementTimer = movementCd;
        movementVector = DefineMovementVector();
    }
}

public Vector2 DefineMovementVector() //Defines a radom vector to move the wizard
{
    return new Vector2(Random.Range(-0.5f, 0.5f), Random.Range(-0.3f, 0.3f));
}
```

Grand wizard’s unique attack:



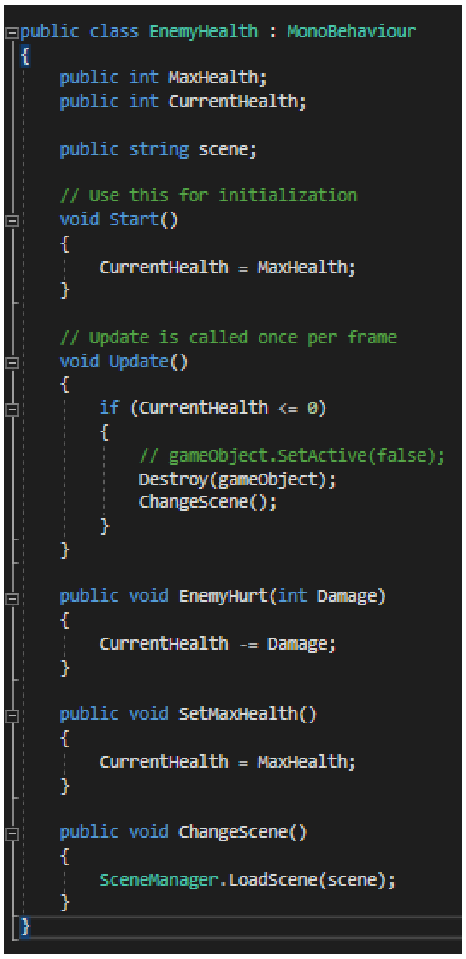
```
public void UniqueGrandAttack() //Lightning attack
{
    if (uniqueGrandAttackCount == 0) //First phase of the attack (casting)
    {
        uniqueAttackMarkPrint = new Vector2(Random.Range(-5.9f, 5.9f), Random.Range(-0.75f, 0.3f)); //Defines a random position in the zone where the player is
        GameObject attackMark = Instantiate(uniqueGrandAttackMarkPrefab, uniqueAttackMarkPrint, new Quaternion());
        Destroy(attackMark, 0.75f);

        //The two phases system works like the water attack
        uniqueGrandAttackTimer = uniqueGrandAttackCd;
        uniqueGrandAttackCount++;
    }
    else //Second phase of the attack
    {
        SoundManager.instance.PlaySingle(lightningAttackAudio); //Plays the sound

        GameObject attackParticle = Instantiate(uniqueGrandAttackParticlePrefab, uniqueAttackMarkPrint, new Quaternion());
        Destroy(attackParticle, 2f);

        uniqueGrandAttackTimer = 0;
        uniqueGrandAttackCount = 0;
    }
}
```

Player's water attack collision system:



```
public class OnPlayerRayAttackCollision : MonoBehaviour
{
    private const int PLAYER_DAMAGE = 4; //Damages done by the attack

    private float attackTimer = 0.01f; //Time to make the attack Time dependent instead of physics dependent

    public AudioClip wizardHitAudio;

    private void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "EnemyWizard") //If the object hit is has a special tag
        {
            collision.gameObject.GetComponent<EnemyHealth>().EnemyHurt(PLAYER_DAMAGE); //Takes off the enemy's health
            SoundManager.instance.PlaySingle(wizardHitAudio); //Plays the song informing that the enemy is hit
        }
    }

    private void OnCollisionStay2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "EnemyWizard") //While the enemy wizard remains in the collision
        {
            if (attackTimer > 0) //If the wizard got damages from this attack before the end of the cooldown, do nothing
                attackTimer -= Time.fixedDeltaTime;
            else
            {
                attackTimer = 0.01f;
                collision.gameObject.GetComponent<EnemyHealth>().EnemyHurt(PLAYER_DAMAGE); // Takes off enemy's health
            }
        }
    }
}
```

Update of the enemy's health:



```
public class EnemyHealth : MonoBehaviour
{
    public int MaxHealth;
    public int CurrentHealth;

    public string scene;

    // Use this for initialization
    void Start()
    {
        CurrentHealth = MaxHealth;
    }

    // Update is called once per frame
    void Update()
    {
        if (CurrentHealth <= 0)
        {
            // gameObject.SetActive(false);
            Destroy(gameObject);
            ChangeScene();
        }
    }

    public void EnemyHurt(int Damage)
    {
        CurrentHealth -= Damage;
    }

    public void SetMaxHealth()
    {
        CurrentHealth = MaxHealth;
    }

    public void ChangeScene()
    {
        SceneManager.LoadScene(scene);
    }
}
```