Haunted Mansion

PROJECT EVALUATION – BY ANDREW LETAILLEUR (b00334945)

Contents

[**Assignment Outline** 2](#_Toc499190303)

[Strengths & weaknesses 2](#_Toc499190304)

[Pros of development output 2](#_Toc499190305)

[Cons of development output 2](#_Toc499190306)

[Unforeseen Events 2](#_Toc499190307)

[Problem solving summary 3](#_Toc499190308)

[Development problems & Solutions 3](#_Toc499190309)

[Improvements 4](#_Toc499190310)

[**EVIDENCE OF WORKING WITH OTHERS** 4](#_Toc499190311)

[SELF EVALUATION 4](#_Toc499190312)

[Contribution 4](#_Toc499190313)

[Effectiveness/Efficiency of contribution 4](#_Toc499190314)

[Strengths & Weaknesses of my own contribution 5](#_Toc499190315)

[Virtues 5](#_Toc499190316)

[Faults 5](#_Toc499190317)

[TEAM EVALUATION 5](#_Toc499190318)

[Ability of Collaboration during Planning & Development 5](#_Toc499190319)

[Individual Team Member’s Contribution & Effectiveness 6](#_Toc499190320)

[**Conclusion** 6](#_Toc499190321)

[References 7](#_Toc499190322)

Haunted Mansion Group Project:

Evidence of Problem Solving

# Assignment Outline

I was tasked as part of a small group, to develop a prototype of a game based of a crafted game design document. And once we’ve done developing the prototype and documentation (both game design and technical documentation, blueprint wise) within twelve weeks, to present it to the class in a presentation. In our group’s case, we decided to build a prototype with the focus being based on a horror theme, where the protagonist has to survive (but not win/beat) enemies that can only harm him, not be harmed themselves.

# Strengths & weaknesses

## Pros of development output

Easier means of communication as a two-person team, on top of having at least a decent split of artistic and technical skill on both our ends, meant that we didn’t had to get too far ahead in contrast to a solo developer, and can hire more talented people as the need arises. All we needed to do, was have a discussion on what tasks or issues needed to be done or resolved, and we more or less were able to get our wishes more or less followed.

## Cons of development output

Lack of manpower and full technical experience on both our ends to cover for any issues either of us faced, served to limit the scope of the project demo, and overall lead to the requirement of cutting a few features due to having to meet those time constraints. This also meant we didn’t had a cover system in place, in the sense of having to justify each of our assigned roles, and not end up hijacking the project by taking on too much responsibility, which can also lead to even more undue stress.

# Unforeseen Events

An initial version of the project becoming corrupted beyond repair, due to a corrupt storage device. Thankfully, progress lost due to data corruption, was averted due to having recently set up GitHub to act as a long-term storage, and back up facility to prevent a similar sort of incident from happening again. A similar incident also happened, when we encountered a surprise bug where I was unable to reasonably update the development project within Unity, to GitHub. This eventually lead to the emergency requirement of putting the most recent version of the project, into a separate zip folder, before moving a copy of that zip folder to the GitHub drop box function, as a labelled version.

One other case of data corruption happened, and that was due to an accidental deletion of a variable leading to a grand mess of missing cameras, which ended up with me having to nuke or delete the version I was working on. Thankfully, having an adequate back up strategy with hard drives, served to limit the damage done by that, to just the work I was doing at the time.

Lastly, end week time constraints, served as a rushed reminder that we had limited time. And due to having assets acquired last minute, ultimately left me with no time to fully implement the full range of sprite sheets, as much as unable to fully implement audio or music cues for the demo itself.

# Problem solving summary

## Development problems & Solutions

I made a long list of bugs and problems that happened during development, not all of which were reasonably resolved on time, or features that were able to be included within the time constraints. This list (or board), can be found in the link provided below. (Minus more informative commentary providing context behind the cards, due to issues with changing privacy settings of boards at the time)

<https://trello.com/b/QyClJDsv/program-dev-blog-copy-ail>

A few key highlights on issues encountered during development of the project in coding, were as follows;

* The clock HUD initially being at an odd angle, either not following the camera properly, or not angled right. This was eventually resolved through utilising the ‘ViewportToWorldPoint’ function within unity’s C# code.
* Fixed visuals on 3D Assets not loading over walls properly. Which was fixed by reversing the positioning of the 3D was to the opposite z value in positioning, to help with a sense of ‘depth’.
* Level swapping trigger boxes not registering if a player’s there, which was fixed by ensuing proper use of Collider2d tags.
* HP Slider not updating properly, which was resolved by having the variables properly linked / reference to the player’s variables, stat wise. Which was later amended to being linked to a persistent ‘stat’ variable, that will persist until the game is over.
* Damage that flashes red, flashing by too fast, and too faintly. This was eventually resolved, when it was discovered the alpha transparency was too light for the ‘damage’ icon, and was thus amended accordingly to make the screen flash much redder upon damage.
* Non-Responsive GUI, upon reaching a ‘game over’ screen, which was caused by camera clashing with destroyed player variable (which included a health slider/canvas), lead to a technical clash due to event checker’s being also deleted by technicality. This was resolved by placing the variables into a more separate/manageable asset, though this came at the expense of placing a player on every single map instead, placement wise.
* Ghosts stuck spawning at the centre of the stage, due to a technical glitch when it came to using perspective view on the camera, in contrast to orthodontic view. This was resolved by tweaking the variables to store and acquire an orthodontic scale on positioning, and updated to keep the ghosts spawning at each individual edge instead.
* Ghosts not spawning at all four sides, due to underestimating the precise value of how random numbers work in C#. This was resolved by switching from a 2-5 check, to a 1-5 check, say.
* New player asset’s not rotating great with the previous rotation upon movement code. Which was resolved by disabling the rotation function within the PlayerCode.

Sadly, for many bugs and problems that where resolved, there existed a couple that I wasn’t able to resolve in time. In order, those issues consist of the following.

* Ghosts over HUD issue
* KeyCode input suddenly not working. Which forced me to go under more general keycodes, such as ‘axis’ and ‘spacebar’, instead of individual keys, such as ‘E’ and ‘R’.
* Persistent testing of lighting, to ensue that there existed adequate lighting by the final build.
* Sprite Animation woes, where I had to include a last minute functionality on four separate facings, and transition between them, and a ‘standing’ wait. Due to time constraints, I wasn’t able to fully implement facing on a 2d scope, thus the facing aspect had to be scrapped.

In addition, there where additional problems encountered outside programming, that is still relevant to the development of this project, in order;

* Inability to manually update my version of the project to GitHub conventionally. It got so bad, I had to link in a more zipped version of my (then) current progress, to ensure there was at least some safety under my belt in proper testing on time.

# Improvements

Decent contact over time, especially during homework, improved skillset in programming, and more confidence in being able to include lighting effects in my future projects in the Unity engine. On top of gaining improved experience in manually debugging issues, through the use of online help forums.

EVIDENCE OF WORKING WITH OTHERS

## **SELF & TEAM EVALUATION**

# SELF EVALUATION

Here’s a Self-Evaluation on all the work I’ve done for the team, and the group project.

## Contribution

To start with, I helped provide the initial template of the documentation, along with talk on discussion with initial planning, before I started focus on the programming and development aspect of the project itself. While I was doing this, my partner; b00328251 focused on the documentation. This gave me enough free reign and focus to fully program the assets for the base game, on top of creating at least a basic template of level layout, along with set up the GUI elements along with the menu to gameplay interface/loop. Afterwards, it was a case of adding input to the PowerPoint presentation, on top of providing aid and double checking for last minute required features or variables within the project itself, among giving input on paperwork if need be.

## Effectiveness/Efficiency of contribution

On a pure motivation scale alone, I was driven to do the best I can for this project within a more limited timescale than I was used to. Though this passion alone lead to the feeling that I did quite a bit (if not too much!) In making the final product of the game polished to at least a properly playable standard, the limited timescale ultimately served to limit the amount of time I was freely able to add finishing touches to the project, especially with a couple last features and fine tuning I felt would be out of reach, at times. Outside of programming however, I was just as helpful for the initial layout on the documentation and other paperwork, but due to a combination of assigned duties, and my own quirks of perfectionism and bad grammar on initial drafts acting as a curse that’d slow the development time of the project even more; it ultimately led to me entrusting that burden more to b00328251.

My only fear on my overall efficiency on my own contribution, is a possible inefficiency of having and maintaining proper logs, or documentation of my activities as such. But in practice, at the time I felt that this might have been remedied by leaving log notes on a Trello board instead, which served to act as a genuine Log on my activities. By proxy, another thing I forgot to do in hindsight, is properly comment my code, which might be due to how by passion, I was more focused on the developing end, than the commenting end all the time. Which may have lead to occasional gaps in commentary.

## Strengths & Weaknesses of my own contribution

### Virtues

Passion for developing the game itself, which ultimately served to give motivation and confidence that I can finish my assigned tasks, without experiencing doubt or negativity on my own ability. Time Management, in a general sense of not assigning too much work at once, and in the general eye on the deadline to further motivate on building as polished a demo as I can reasonably develop in a limited time. Problem solving skills as a programmer, which served to help me tremendously in debugging the game, developing the code, and imagining a means of getting the assets to work as I felt a game should work, and lead to occasional bursts of innovation. An eye for detail which led to me being able to provide a framework for the documentation, and fine tune any visual faults in debugging to the best of my limited timeframe. And lastly, a constant vigilance when it came to back ups of the project itself; not only on my laptop and GitHub, but also on my External Hard Drive after dealing with the personal loss of critical data due to a corrupted drive.

### Faults

Perfectionism, which on one end can help drive to make a singular asset as perfect as it reasonably can, yet if left unchecked can lead to a massive drain on time, which can serve to impose additional limitations in time if left unchecked. Forgetfulness, where at times my very passion lead to me spending more time on the project than I was supposed to, or indulge in my own personal vice of perfectionism, time management wise, on top of possibly having a lack of commentary in some areas, or redundant code in other areas, when it comes to my own code. And lastly, my inability to at times manage distractions, which can easily shave off hours’ worth of possible work, if left unchecked.

# TEAM EVALUATION

Here’s an Evaluation on the rest of my team, and their overall contribution to the group project.

## Ability of Collaboration during Planning & Development

As a team, we had regular meetings during labs, after university, and through skype, and generally maintained contact to lightly check on each other’s progress, and note when any of us where struggling, or had issues to contend with. On top of that, we also had a set up GitHub project, in order to access, and update any files that get manually edited by either of us. Though this later on didn’t come without its own due issues, of course. Throughout the development of the project, I’ve at times lost sight on the progress of the documentation itself due to being more concerned on the ‘moment’ of the development, in fixing problems and developing assets I come across as spiritually envisioned on what a game would require. This lead to at times, forgetting the finer details on contributing to the planning and documentation, past looking at the tasks I felt where needed to fulfil in order to meet the overall aims of the scope under the ‘Kanban Board’ format, from the games design documentation. This lead to my partner, b00328251, to work on the finer points of the documentation itself, while I focused on the actual development of the project, occasionally pulling requests on when I’d need a certain asset to include, sprite/background wise, and such. That said, in spite of having a small team (or duo), we were more or less able to efficiently collaborate to a reasonable degree, in at least ensuring that all the minimal work was at least done to some degree.

## Individual Team Member’s Contribution & Effectiveness

I only had one team member, and student identity number is b00328251.

B00328251 provided me with the assets, and helped with fine tuning the documentation, which could have otherwise sucked in far too much of my valuable time/focus, that could have been better spent developing the demo itself, properly. On top of help pitch and refine the general story concept of the game itself. However, there was times in which I had to wait for the required assets, and due to time constraints when I finally acquired those assets, I could not fully implement the sprite sheet of all character’s fully as it was visually intended for. Above all else, if the end documentation is good, then as a general rule I feel that b00328251 helped a lot, and done more than enough when it came to making sure everything was laid out as planned in the technical scope.

Conclusion

The first thought I have in mind, was there was in a way, a few features I wasn’t able to fully implement due to shorter time constraints and delayed acquiring of assets, a couple more features I didn’t even expect to implement until I stumbled upon a means of adding them by pure coincidence, to reasonable degrees of success. And a few bugs and kinks I’d still want to debug, but didn’t had the time to actually resolve by the time documentation requirements, and practice for presentation crept in as a higher priority. And that’s on the development end. Paperwork however, is another issue, that I mostly depended on my partner to properly handle, and outside of a Kanban board, wasn’t able to double check as often as I’d have liked, in case there was possible issues with the project. And due to eventual issues with GitHub when it came to ensuing version updates (in hindsight), vice versa when it came to my partner being unable to access the most up to date version of the project I was working on.

Overall, I felt that I polished the skill I had with developing games under the Unity Engine, and learned a massive amount of ‘to do’s when it comes to version management, along with contingencies to be made in case I’m unable to properly update a version of the project asset’s themselves, due to size issues and such. And that lastly, that if in the future I’m to make a certain board accessible to the public, Trello wise. I should ideally ensue it’s public from the start, or have time in place to figure out a means to make a Kanban board public later, if at all possible. These experiences, and more, will lay a foundation, for a more accessible, and backed up projects for future references.

# References

**Unity forums,** A common source for questions and answers, frequently asked by programmers, on things relevant to coding. Typically, a search engine result on a programming issue with ‘unity’, will lead to this site. - <https://forum.unity.com>

**Stackoverflow**, Another common source for general questions and answers, on the nature of programming, from JavaScript to C#. A reliable source for solutions to frequent problems, due to the universal nature of code, on certain languages. - <https://stackoverflow.com/>

<https://www.raywenderlich.com/61532/unity-2d-tutorial-getting-started> - even though it is outdated, the tutorial still had a few key notes on references that aided immensely when it came to reminding myself on hitboxes, rigid bodies, and a few other quirks with 2d assets.

**GitHub** – a repository site, for the storing, sharing, and downloading of various program projects. - <https://github.com>

<https://github.com/AndrewLetailleur/GameDesign_GroupProject_Horror> – A public link to the project itself, containing the final program, along with all relevant documentation.

**Trello**, a Kanban board service, that allows users to make boards, that can have various labels attached to them, which helps simplify task management. - <https://trello.com/>

<https://trello.com/b/QyClJDsv/program-dev-blog-copy-ail> - Public copy of board, without any developer notes attached.)

<https://trello.com/invite/b/1LkJxykh/0223004e94248e51c70baa24182fe6d0/program-dev-blog-board> - Experimental link, may be able to join with an account, and view the left over comments from there.