

Computer Games Development

Project Report

Year IV

[Seán Whelan]

[C00250016]

[26-04-2023]

[Declaration form to be attached]

**Contents**

[Acknowledgements 2](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714373)

[Project Abstract 2](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714374)

[Project Introduction and/or Research Question 2](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714375)

[Literature Review 2](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714376)

[Evaluation and Discussion 3](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714377)

Project Milestones

Major Technical Achievements

Project Review

[Conclusions 3](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714378)

Future Work

[References 4](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714379)

[Appendices 4](file:///C:\Users\gameuser\Desktop\y4MainProject\No_Code_Game_Editor_SlayerMaker\Documentation\Project-Report-NoCodeGameEditor.docx#_Toc54714380)

# Acknowledgements

I would like to thank the following people who assisted in completing this project including;

Martin Harrigan, who was my project supervisor throughout the development cycle.

I would also like to thank ICME for use of ….

Use this template when writing your research report. As a rule of thumb, the report should be of the order of 10 pages (about 250 words/page).

# Project Abstract

The main goal of my project is to create a no-code game editor which allows users without any coding experience to create a 2d, top-down, “shootemup” game. In the landscape of today’s gaming industry allowing users to create their own game modes and environments seems essential to the continued popularity of a game. With the industry shifting further every week towards a game’s as a “live service” model, the expectation of players is a consistent amount of new content. Very few developers can create content which they can put their name behind fast enough to meet this demand. This is where a game editor comes into play. Outsource the creation and distribution of new content to the players themselves. This has proven to be a great addition to many of the major games which have tried it.

# Project Introduction and Research Question

In the last couple of years most games which have had continued success and a thriving player base beyond the initial launch phase of the game have had this to some capacity. Roblox, Fortnite, Minecraft and GTA 5 are the best examples of this. Halo Infinite for example, launched without this and while it saw a massive player base initially, this dwindled off as 343 Industries were unable to add new content fast enough. Now they have released their in-game editor “Forge Mode” and the player base is beginning to return.

In my opinion, having some form of level editor in a lot of games, is near essential in order to sustain a large player base that keeps coming back for more. Furthermore, how your players are able to distribute these levels to each other is vital.

No game has gone further in this regard than Fortnite Battle Royale, and low and behold consistently one of the most played games in world since its release 5 years ago. Not just does Fortnite have a game editor called “Creative” mode which allows you to create your own maps and minigames, but it also has a way to get your game viewed by the masses, distributed to them and played. Lastly, Fortnite incorporates referral codes called “creator codes” which allow the builders of these game modes to get paid. A player who plays their game, will see their creator code and opt to put in in their item shop, then when they buy something from the shop, a portion (5%) of that money goes to the creator. This is genius as it incentivises creating content more than even the enjoyment of being creative, but also the excitement of others enjoying your game and lastly the bonus of a pay check for doing a good job.

While its good for a game editor to have options, as it allows the player/creator to be as creative as they care to be, it’s also essential for an editor to have clear barriers to what can’t be done and a direction/ use case which the editor is for.

In this instance the use case of my game editor is specifically to create 2d, top-down, “shootemups”. This ensures that people who come looking to create that, won’t be overwhelmed by a million things that don’t relate to what they are specifically trying to create. More options are good to a point, where it crosses over and makes you less likely to create.

The 2 main focuses of my editor are a smooth level designer and an accessible UI. The level designer allows you to create the world you want (within reason) that your game takes place in. An easy to use user interface is essential to the editor as players will only stay creating if their user experience is enjoyable.

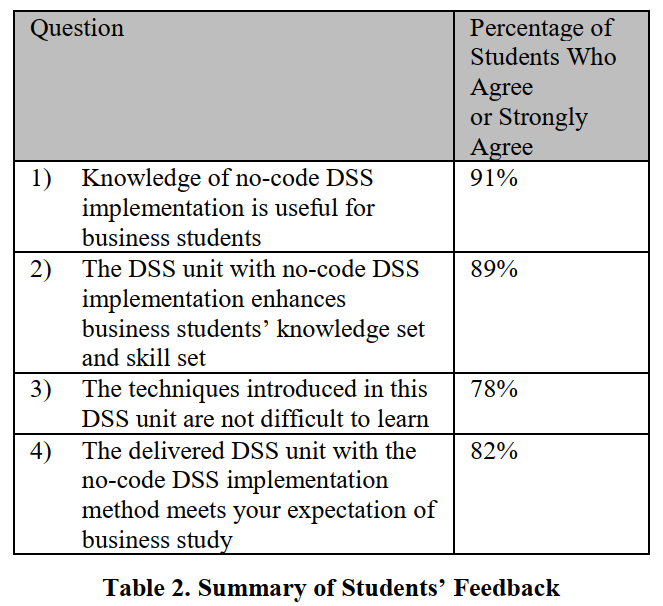
Research Question:

Is it possible to create a 2D top-down shootemup game editor which has no code involved whatsoever? Will it be easy to use with no experience in Game Development and no tutorial? Will the editor allow games created to vary enough to allow for creation of a multitude of games each with their own novel gameplay while still being powerful in that specific area? How does no code development stack up to a traditional learning curve of code based development.

# Literature Review

*“Improving Student Performance by Introducing a No-code Approach: A Course Unit of Decision Support Systems”*

In recent years no code tools have been gaining traction in educational settings as a way to teach both coding and overarching concepts that relate to it. This is primarily useful for students who may not come from a computer science background but who are capable to understanding the material if the initial barrier of not understanding code is taken away. An example of this which I found particularly pertinent to my project is a paper titled “Improving Student Performance by Introducing a No-code Approach: A Course Unit of Decision Support Systems” which was written by Hai Wang and Shouhong Wang (2022). The course which the authors experimented with was a decision support systems (DSS) college course. In the conclusion they noted how the students which completed the no-code unit had overall higher grades and in their opinion a better understanding of the DSS concepts which were being taught than those who had not completed that unit.



Business students feedback studying DSS on how they felt about the no-code unit.

This is particularly relevant to SlayerMaker as a no-code game editor as in simple terms it should make the development process easier for the end user than achieving the same goal with traditional programming. This is amplified by the fact that SlayerMaker is targeted primarily at the target audience which gets the most benefit from no-code tools, people who do not come from a computer science background.

*“No-Code Video Game Development Using Unity and Playmaker”*

The book “No-Code Video Game Development Using Unity and Playmaker” by Mike Kelley covers various aspects of no code development. In particular it focuses on using Unity and Playmaker to achieve this but much of the concepts are not subjective to just this environment and actually are very useful when it comes to understanding no-code game development as a whole. In particular for SlayerMaker Chapter 7 is pertinent as it focuses primarily on user interfaces (UI) again, in Unity. The chapter is a UI 101 class which involves step-by-step instructions on creating various UI elements, some more useful than others. These include panels, text, images, buttons, modal windows and animations. The author also goes into detail about best practices in regard to UI design. These included ensuring UI elements are easy to interact with, using appropriate font sizes and colours and keeping the layout simple and consistent. The chapter lastly covers how to make user interfaces more responsive as well as making them adaptable to different screen sizes and resolutions.

As SlayerMaker allows the user to write no code at all the design principles and best practices that are discussed in Chapter 7 will be useful in helping create a user-friendly UI that makes the entire application easy to use and to navigate. Particularly when it comes to starting with a concept of a game, and seeing that through the entire development process all the way until the end and creation of said game. As the GUI is the number one thing the user will be interacting with, it is pertinent to the user experience (UX) of SlayerMaker that the UI follows these principles and practices, in particular for the target audience of non-technical people with the lack of programming or game development experience.

Replace this text with an appropriate Literature Review.

The literature review places your research in context. You aren’t the first person to investigate or research a particular topic. Present a short literature review with the following goals:

* Give the reader a good overview of the key concepts;
* Describe the most relevant work (in your own words) that other people have done in this area;
* Use proper academic writing with references.
* Show how the existing work influenced your project.

# Evaluation and Discussion

Replace this text with Results and Discussion.

Describe the results using diagrams such as graphs etc. as appropriate, and discuss what the results mean.

Example: Results indicate that once the threshold gets over a certain point it significantly reduces player performance and player experience

**Project Milestones**

|  |  |  |
| --- | --- | --- |
| Feature / Milestone / Component | Date to be completed | Achieved on time? |
| Refined the proposal. Narrowed the scope to two aspects of a no-code editor:  Map Creatuib and User Interface | 21 October | Y |
| Simple version of wall placement | 28 October | Y |
| Click and place walls | 28 October | Y |
| Simple Menu | 14 November | Y |
| State Management | 14 November | Y |
| Research Document Draft | 14 November | Y |
| Other wall placement forms, brushing multiple cells | 25 November | N |
| Path from high to low level game creation: Stages | 25 November | N |
| Finish better menu | 25 November | Y |
| Cell and Grid Rewrite | 25 November | Y |
| Literature Review Draft | 30 November | Y |
| Finish Multiple Brush types | 10 December | Y |
| Finish Stages of game creation from high to low level. Modular, can be added to | 10 December | Y |
| SRS Feature list at current time | 20 January | Y |
| Wall Rewrite | 20 January | N |
| Rubber tool for removing placements | 20 January | Y |
| Simple Player | 27 January | Y |
| Test / Demo Level functionality | 27 January | Y |
| Fixed Toolbar UI | 10 February | N |
| Choice of sprites or walls, curated set | 10 February | Y |
| Enemies, Spawn points, weapon for player | 10 February | Y |
| Wall change again, more efficient | 14 March | Y |
| Add crosshair | 14 March | Y |
| Add shooting towards crosshair | 14 March | Y |
| Enemy Spawners implementation | 14 March | Y |
| Removal / Deletion of spawners with rubber | 24 March | Y |
| Enemy implementation | 24 March | Y |
| Saving to CSV for walls and Enemies | 24 March | Y |
| Loading from CSV to create Games | 24 March | Y |
| Choose which game to play | 24 March | Y |
| Blood Splatter when enemies shot | 24 March | Y |
| Goal/ Sub goal design. High level game types Possible options | 31 March | Y |
| Game types fully implemented | 7 April | Y |
| Ui for game play | 7 April | Y |
| Customisation options | 7 April | Y |
| Items to make games more varied | 14 April | Y |
| Powerups to make the user feel powerful | 14 April | Y |
| Simple network manager to move csv’s to database | 21 April | Y |
| Fill tool | 21 April | Y |
| Finished Documentation | 26 April |  |
| Finished Network Manager | 26 April |  |

**Major Technical Achievements**

I believe I had a good few major technical achievements during the development of SlayerMaker, all of which contributed, to a great end product.

Game Objects:

The game objects of SlayerMaker are instantiated and stored in vector of unique pointers to objects. This means

Network Manager:

**Project Review**

What went right? What went wrong? What (if anything) is still outstanding/missing (i.e., still left to do)? If starting again, how would you approach this project differently? What advice would you have for someone attempting a similar project in the future? Were your technology choices the right or wrong ones? If you chose the wrong technology, provide justifications for why you think this. What were the implications of your technology choices?

# Conclusions

summarise your work and findings.

**Future Work**

Indicate what might be some next steps to try (if a student next year was going to undertake a project in this area what might be an interesting thing for him/her to examine?).

# References

# Appendices

Replace this text with Appendices.

This might include ethics application and other relevant material e.g. copy of any questionnaires used.