University Of Plymouth

School of Engineering,
Computing, and Mathematics

A Co-Operation Level Editor

Osbourne Laud Abraham Clark

10777267

BSc (Hons) Computer Science

Plymouth University
United Kingdom

$??^{th}$ April 2025

Acknowledgements

Abstract

Extra

Word Count:
Code Link:

Contents

Acknowledgements **1**

Abstract **1**

Extra **1**

Contents **2**

List of Figures **4**

1 introduction ¡TODO LOOK AT SOFTWAREDESIGN DOC IN REPO¿ **5**

1.1 Background 5

1.2 Objectives 5

1.3 Deliverables 5

2 Legal, Social, Ethical and Professional Issues **5**

3 Method Of Approach **5**

3.1 technologies 5

3.1.1 unity 5

3.1.2 C# 5

3.1.3 Libraries 6

3.1.4 Co-Operation game 7

3.1.5 github 7

3.2 non-functional requirements 7

3.3 functional requirements 7

3.4 UML 7

3.4.1 classes 7

3.4.2 control flow 7

4 project managment **7**

5 Preperation **7**

5.1 previous attempts 7

6 implementation **7**

6.1 unity 7

6.2 YAML 7

6.2.1 importing 7

6.2.2 exporting 7

6.3 Reverse Engineering of Level Layout 7
6.3.1 object placement 7
6.3.2 object orientation 7
6.4 cacheing for faster loading 7

7 Evaluation 7

8 Conclusion 8
8.1 further Work 8

List of Figures

1 introduction ¡TODO LOOK AT SOFTWAREDESIGN DOC IN REPO¿

1.1 Background

this dissertation idea came about throuhg a problem identified in my second acedemic year while developing a mod for the game Co-Operation created by MINDFEAST. the issue found was the tedious nature of creating levels i general. once levels became big or contain a lot of objects they start to become a mess, and attempts to quickly change a single object can become a massive effort. there are multiple method used for finding and changeing object posistioning:

- counting rows and columns in the game
- crawling throuhg the file until you have found the correct object
- make continuous mental logs of where things have been placed

1.2 Objectives

1.3 Deliverables

2 Legal, Social, Ethical and Professional Issues

3 Method Of Approach

3.1 technologies

3.1.1 unity

to create this project i utilised unity for its easy to use graphics maniplulation methods. From the beginning it was clear a 3D solution was required for the project. If 2D was chosen it would not have given the same feed back or easy usablility to the user. By going 3D i could give the user a closer idea to what the levels will look like in game.

3.1.2 C#

i was required to code within C# as unity was chosen to be developed within. this was benneficial as i am quite well versed in it.

3.1.3 Libraries

importing GLB objects

i have used UnityGLTF to import glb object within to unity at runtime. this allowed for hot reloading objects without the need of re compilation of the entire unity project.
https://github.com/KhronosGroup/UnityGLTF

parsing YAML

to be able to import and export file in YAML format, without haveing to create my own parser from scrtach, i have used YamlDotNet. This is a library built for C# and so worked easily with unity.
https://github.com/aaubry/YamlDotNet

3.1.4 Co-Operation game

3.1.5 github

3.2 non-functional requirements

3.3 functional requirements

3.4 UML

3.4.1 classes

3.4.2 control flow

4 project managment

5 Preperation

5.1 previous attempts

6 implementation

6.1 unity

6.2 YAML

6.2.1 importing

6.2.2 exporting

6.3 Reverse Engineering of Level Layout

6.3.1 object placement

6.3.2 object orientation

6.4 cacheing for faster loading

7 Evaluation

• What went well and what went badly? • Why was this the case? • To what extent was the aspect under consideration responsible (vs. other contributing factors, e.g., your own performance). • Was your experience in line with what might have been expected given the body of knowledge within the literature? • To what extent does the above cause you to reconsider the choices that you made in relation to the given aspect?

8 Conclusion

It is a brief summary of the project and its achievements. Therefore, you should relist your project's objectives and critically (and ruthlessly) evaluate whether you met the objectives

8.1 further Work