

Qt 2D Tile Based Level Editor

Project Research Document

Year IV

Eoin Abbey-Maher

C00217717

Institute of Technology, Carlow

Supervisor: Philip Bourke

# 

# 

# 

Faculty of Science

Open-Book and Remote Assessment Cover Page

**Student Name:** Eoin Abbey-Maher

**Student Number:** C00217717

**Lecturer Name:** Mr Philip Bourke

**Module:** 4th Year Project

**Stage/Year:** 4th Year

**Date:** 28/04/2020

**Declaration**

**This examination/assessment will be submitted using Turnitin as the online submission tool. By submitting my examination/assessment to Turnitin, I am declaring that this examination/assessment is my own work. I understand that I may be required to orally defend any of my answers, to the lecturer, at a given time after the examination/assessment has been completed, as outlined in the student regulations.**

# I. Acknowledgements

I would firstly like to say a thank you to my project supervisor and lecturer Philip Bourke, not only for his continued assistance on this project but also for his assistance and guidance throughout the period of my education. Phil was a valuable contact to have and someone that was there to help answer any questions or problems that I ran into while working on this project or any others. Phil always helped to keep me motivated when it seemed like the project wasn’t working out, and was always willing to give a second opinion and assistance on any issues I faced.

I would like to thank all the lecturers and staff at the Institute of Technology that I had the pleasure of interacting with and learning from during my time as a student there. They helped educate me both as a games developer and as a person, they taught me many lessons and skills that I shall carry with me after I graduate and move onto the next stage of life.

I would like to thank my parents, Sharon Abbey and Brian Maher for their continued support throughout my time in college and throughout this project. A final thank you to my close friends who helped keep me going, without their assistance and input this project would not have been possible.

# 

# II. Abstract

Levels are the environment within which the game takes place and is played, they are used to set a scene, create an atmosphere for the players and guide them to their goal.

The Qt 2D tile-based level editor is a standalone piece of software which is cross-platform and runs on both Linux and Windows, the software allows developers to quickly and graphically construct a tile-based level for their game rather than having the need to hand place every object using code.

The hope is that this project will tackle long development times for creating new levels by allowing developers to quickly see how their levels look and feel, the level editor will also allow artists and members of the design team to quickly put together a mock-up of the level without having any code or development experience.

The level editor could also be packaged as a program within a game that allows players to build their own custom levels and play them or share them with friends and other members of the game community. This would be helpful to prolong a game's lifetime and allow players to show their own creativity as we have seen before in games like *[13]*“*Super Mario Maker*” and *[14]*“*Little Big Planet”.*

# III. Table Of Contents

[**I. Acknowledgements**](#_qavz0826qmrg) **2**

[**II. Abstract**](#_3imvc9i5yotq) **3**

[**III. Table Of Contents**](#_mvmcpz7uv5mm) **3**

[**1. Project Introduction**](#_vqw22rx85hum) **5**

[1.1 Why This Project Was Chosen](#_39wh8fenxrvl) 5

[1.2 Potential Impact](#_tjgz1vy72hz8) 5

[1.3 Outline what is being done](#_d28d9lyl2f3) 5

[**2. Background**](#_a1k6cbc4skc8) **7**

[**3. Literature Review**](#_amv9izsldinj) **8**

[3.1 Literature Review Overview](#_1uedhl8gev0i) 8

[3.2 Cross-Platform Development](#_d1t1h03h9l0o) 8

[3.3 The Qt Development Environment](#_o7qkfplwdiyh) 8

[3.4 A Brief History Of Level Editors in Gaming](#_spqc5nq9xb3d) 9

[**4. Study**](#_qnz4uwdrt5ma) **12**

[What Am I Testing?](#_tq23xct2ugok) 12

[Testing The Amount Of Time To Build A Level](#_4zje6bih1mta) 12

[**5. Project Description**](#_ickyiii58ld) **13**

[5.1 Description of Final Project](#_yw6vqsmn4fr5) 13

[5.2 Personal and Technical Learnings](#_wj15ljey3ag6) 15

[**6. Project Milestones**](#_3wap9mrx6g63) **16**

[Presentations and Deliverables](#_xrwic8eq3fyk) 16

[**7. Results and Discussions**](#_rfimhuzcwm7y) **17**

[Project User Test Results](#_viex9ya8tl4h) 17

[Testing The Amount Of Time To Build A Level Results](#_x5o9mrdj6ka) 17

[**8. Project Review and Conclusions**](#_r5jnk4srox8o) **20**

[8.1 What was Successful & What Went Wrong](#_jj1i04couj3j) 20

[8.2 What Could Be Done Differently](#_as0zpd4mtnsq) 20

[8.2 What Could Be added In A Future Version](#_akryg2g3phaw) 20

[8.3 Findings](#_fx6gmchsptw2) 20

[**References & Bibliography**](#_w7azslnr6mf4) **21**

[Referenced Publication](#_57fjwe3secw) 21

[Citation](#_57fjwe3secw) 21

[Reference](#_57fjwe3secw) 21

# 1. Project Introduction

## 1.1 Why This Project Was Chosen

This project was chosen because I feel that whilst working on a game, attempting to code each level by hand is very time consuming and could be made more efficient. A more efficient level designer would allow the developer more time to work on other elements of their game. Due to this I am aiming to create a stand-alone level editor which will allow the developer to place their tiles visually and then export the layout to an XML file to be loaded into their game project.

Whilst working on another college project this year and working alongside external artists to create assets for our game, I found that it was difficult at times to explain to the artists how the map was going to be laid out or for them to give us ideas for the map layout. I feel this could be another usage for my level editor. It would give artists and map editors the ability to make quick changes to the levels without needing any knowledge of code.

Having used programs such as Tiled I also wanted to challenge myself to develop an alternative tile editor within the time we had for the project. I wanted to investigate if, in the given period, it would be possible to build a level editor that could be used to build a level from concept to completion using the SFML library.

## 1.2 Potential Impact

It is important that before diving in and starting to build a game you plan out the tasks hoped to be achieved and give a foundation to build the game on. I hope that the impact of this software will be that I can help speed up the development time of 2D games and also help give a tool to quickly prototype levels.

I also hope that if a developer decides to include this software as a tool with the release of their game that it can be used to prolong the lifetime and allows the games community to build and share their own levels with other players.

## 

## 1.3 Outline what is being done

I am building a standalone cross-platform 2D level editor that allows a developer to quickly create and layout their maps. This tool will also give artists or map designers the possibility to layout maps and designs without needing any technical knowledge of code.

This level editor could also be used by the developers to let their players design their own custom levels and load them into the game to play themselves or share their new levels with their friends and other members of the community.

At the end of this project the goal is to have a standalone program which can carry out the following tasks effectively and efficiently:

* The user will have the ability to layout and design their own custom map to their own specification.
* The user will have the ability to Insert their own Image and use it as a tile.
* The user will have the ability to place players and NPC Nodes that set positions within the SFML project.
* The user will have the ability to reimport a pre-existing map project into the tile editor and re-edit that map

## 

# 

# 2. Background

For my background research, I looked at pre-existing tile editors and other programs that work with a grid layout. I used these programs to try and narrow down an idea of what functionality and features my own program should have. I used Microsoft Excel to see how I could consider doing multiple and single-cell selection and create a way to select what colours and textures the cells should be assigned. I also researched a pre-existing tile editor known as “*Tiled*”, whilst I knew it wouldn’t be possible for me to recreate this editor in the 7 months we had to do this project, it gave me great inspiration and an idea of the features and functionality I’d like my tile editor to have.

|  |  |
| --- | --- |
|  |  |
| 2.a. Microsoft Excel | 2.b. The Tiled Level Editor |

Using Qt and Linux was a new experience and I had to do a large amount of background research into how the Qt IDE worked. I had to research Into how the Qt widgets work, how makefiles can be used and using efficient development methodologies. I had also never worked with a system that uses a “slots and signals” event loop and needed to learn how to connect these signals and write new ones as needed.

Another piece of background research I needed to do was with setting up a virtual machine and running Linux on that. I had done very little work on virtual machines or Linux before working on this p[roject so I had to put research into learning these applications and operating systems.

# 

# 3. Literature Review

## 3.1 Literature Review Overview

The key reason for this literature review is to familiarise me with numerous concepts that would be used as part of my project development. As part of the literature review, I hoped to learn about cross-platform development and the issues and advantages of cross-platform programs. I hope to learn about the Qt Development and Design Environment which I am using for the main development of my program and look at other cross-platform development tools and what they have to offer to developers. I also hope to learn about the history of level editors both standalone and within video games.

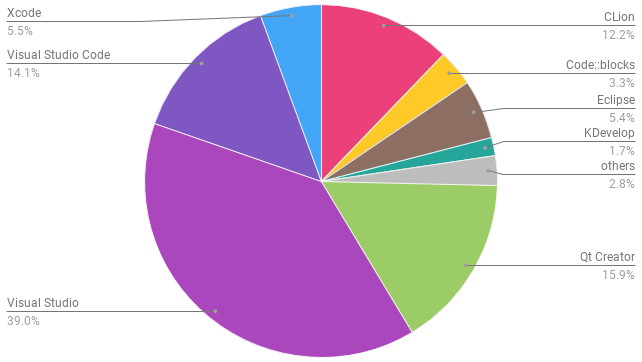
## 3.2 Cross-Platform Development

*[1]“Cross-platform development is the practice of developing software products or services for multiple platforms or software environments.*” Cross-platform applications can work across many platforms such as Linux and Windows, or Android and iOS. Some cross-platform programs can work across mobile and desktop such as programs like “Google Docs” or “Google Sheets”. For this project, I will be working on developing my application for Both Windows and Linux.

Some Advantages of cross-platform development are [2]*“Speed, Reusable Code Components and Reduced Costs”*. Cross-platform development gives a huge boost to the speed in which a piece of software can be developed because it saves on the rewriting and modification of code on different platforms, this advantage also leads us onto the second advantage which is code reusability. Code reusability means that any components that are made can be used again quickly in the future for other platforms and as the bases for other components. The third advantage is Reduced costs, this is because there is less of a need to get different developers based on different platforms and also because the application can be developed in less time the development period costs less for companies.

Although there are many advantages to using cross-platform development, it also has its disadvantages. Some of these are [2]*“Performance challenges, Slow code performance and Limited user experience”*. Cross-platform apps have Performance challenges due to how platforms handle communication between native and non-native components. Slow code performance can be caused by sluggish and inefficient code, This code has to be used as it is not guaranteed that the more efficient way of carrying out a process may not work on all platforms that the program is developed for. The User Experience of a cross-platform application can be limited due to the fact that platform native components cannot be utilised because they are not available across all platforms. Hardware limitations, Screen Layouts and other functionalities can also limit and affect the user experience of a cross-platform application.

## 3.3 The Qt Development Environment

[3]“*Qt Creator is a cross-platform integrated development environment (IDE) built for the maximum developer experience. Qt Creator runs on Windows, Linux, and macOS desktop operating systems, and allows developers to create applications across desktop, mobile, and embedded platforms*. “

[4]*As of 2018, it was said that Qt had a 15.9% market share in “The most used C/C++ IDEs”, with Visual Studio having the largest User base at 39%.* Qt has been used by large companies in the development of their software, Just to list a few Qt has been used by :

*[5]Saab Technologies for their aerial and marine traffic management, Mercedes-Benz’ uses Qt to develop its premium In-car UI system and Autodesk has used Qt to develop the Ui for its Infrastructure design.* When talking about Qt all these companies talked about its large amount of documentation and its ease of use as the reasons behind why they chose to work with the Qt Development Environment.

The Qt Environment uses signals and slots for event tracking rather than an event loop, these signals and slots are used for objects to communicate with each other and to register user interactions on the application window.

The Qt Environment offers the ability to work directly with the code and also using a GUI style editor to layout your application in a quick and easy way. These features speed up the development time and make it easy for developers to see the changes in their work quickly and efficiently. Qt also Includes its own testing features and functionality to Quickly write Unit tests for your product to ensure that your software and methods are functioning how they are meant to.

## 3.4 A Brief History Of Level Editors in Gaming

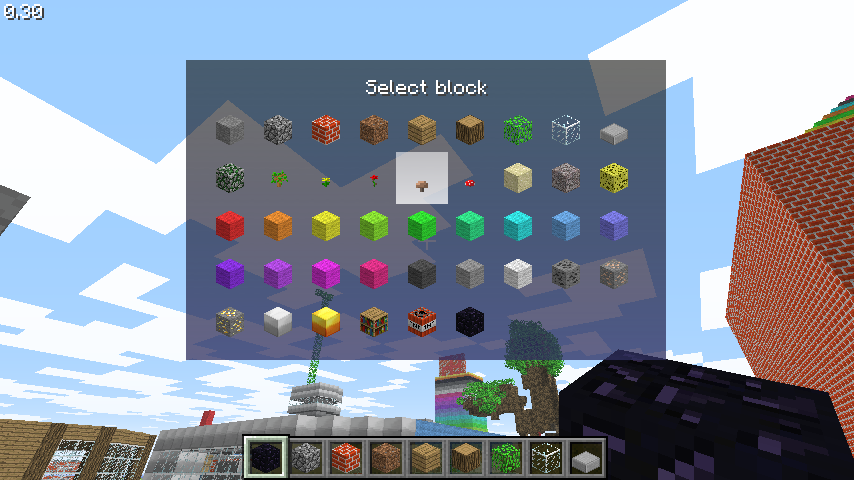
Video games have an ability unlike many other forms of media, [6]“*In most media, such as books, movies, fine arts, and so forth, the creativity, is left almost entirely to the artists or the people who produced the work in question. However, in games, it is possible to give players the building blocks of creativity-the raw materials from which they can construct or create something in the course of playing the game”.* This ability to create and modify game worlds that developers give to their player allows them to add their own creativity and personal touch to the games that they love.

One of the first level editors in video games was introduced in 1981, The game [7]“*K.C. Munchkin*”, which was published for the Odyssey 2 and shared many of the same concepts as Pac-Man, was released with 4 pre-made mazes but also included a mode where players could create their own mazes using programming by starting from a pre-existing maze or starting the whole level from scratch. 

This game was followed in 1983 by a game that would revolutionise level designers within games forever, The game was named “*Lode Runner*” and was so revolutionary for level editors because it allowed players to create their own levels without the need for any programming. [8] *Due to the success of this level editor, many video game magazines launched competitions and gave out prizes to fans for building and submitting their own levels for Lode Runner.*

Lode Runners Developers “*Ariolasoft*” then offered the fans the chance to create and submit levels to be in with a chance to be included in Lode Runner’s sequel, “*Championship Lode Runner*”.

After the success of these level editors and builders, many games followed suit throughout the ’80s and ’90s but the next big revolutionary and memorable level builder according to many didn’t come until 2000, in the form of “*Tony Hawk’s Pro Skater 2*”. This game was revolutionary and had huge replayability because of the creation of new levels. THPS2 also features local multiplayer which allows the players to build their levels and sit and watch as their friends enjoyed their creations. 

Released as a standalone game on Windows in 2006, “*Garry’s Mod*” became another big hitter with level building fans. Allowing fans to play with the physics and items of Valve’s Source engine the game was incredibly popular and still continues to have a loyal fan base to this day. Peaking in 2015 with concurrent players hitting 73.5K users, [9]*the game continues to grasp fans having a peak concurrent player count of 59K players for the month of April 2020*, showing almost 15 years later there is still a demand for games that allow players to build and share their own creations.

It would be impossible to write about the history of level editors and level creation without mentioning “Minecraft”. The game was first released as an alpha in May of 2009. *When it got its full release in November of 2011, the game blew up going from [10] “1000 logins per second to 4000 logins per second”, an article released at this time on “VentureBeat” talks about Minecraft getting 240 million Logins per month. The game was so successful due to it being “an open-ended game that encourages users to be creative”*. Players decided what they wanted to build and how they wanted to play. The creative abilities of Minecraft and its community were truly highlighted by the huge amount of custom levels and Game modes designed and shared by members of the Minecraft community. Minecraft is still one of the leaders in the sandbox and creative genre almost 10 years later, in *[11]2019 it was reported that Minecraft still had a monthly player base of 112 million users* and has over *[12]62K average viewers a month on Twitch.*

Following the success of other Sandbox and level building games, *[13]Nintendo has released its popular Super Mario Maker franchise* with the latest game being released in June of 2019 on the Nintendo Switch. Super Mario Maker allows users to build their own levels and share them With friends and the rest of the community with special ID codes.

Looking at this brief history of level editors and level designers have been used in video games in the past, we can see how they can be a very successful way for developers to build a community around their game and also ensure the longevity of their game even after they stop adding new developments themselves.

# 

# 4. Study

## What Am I Testing?

For my testing, I want to be able to test that I have met the goals I set out in the Project introduction and that I can successfully complete each of these goals to a high standard. A reminder again of these goals are as follows:

* The user will have the ability to layout and design their own custom map to their own specification.
* The user will have the ability to Insert their own Image and use it as a tile.
* The user will have the ability to place players and NPC Nodes that set positions within the SFML project.
* The user will have the ability to reimport a pre-existing map project into the tile editor and re-edit that map

As a secondary test, I wanted to back up the idea that using these level editors leads to quick layout and development of 2D tile-based levels. I have outlined the “time to build a level” test below and I will be addressing the results and conclusions of this test later in this paper.

## Testing The Amount Of Time To Build A Level

To test out my level editor I wanted to test the amount of time it took me to build a varied amount of map sizes and move them from my level editor and into my SFML test game that I have test built.

To ensure that the times are fair I tried to keep each map the same design despite the difference in map sizes.

Each map featured the player position node and 3 NPC nodes.

I also wanted to test the difference in the time it took me to build the map using the tiles that come pre-packaged with the level editor versus loading in a single custom file and using that as the floor in the level. I have also included a screenshot of the 2 “10 x 10 maps” built with the pre-packaged tiles and the custom user tile.

I have written the results of this test, graphed the time taken and given my conclusion, both on this test and the general project in a later section of this paper.

# 

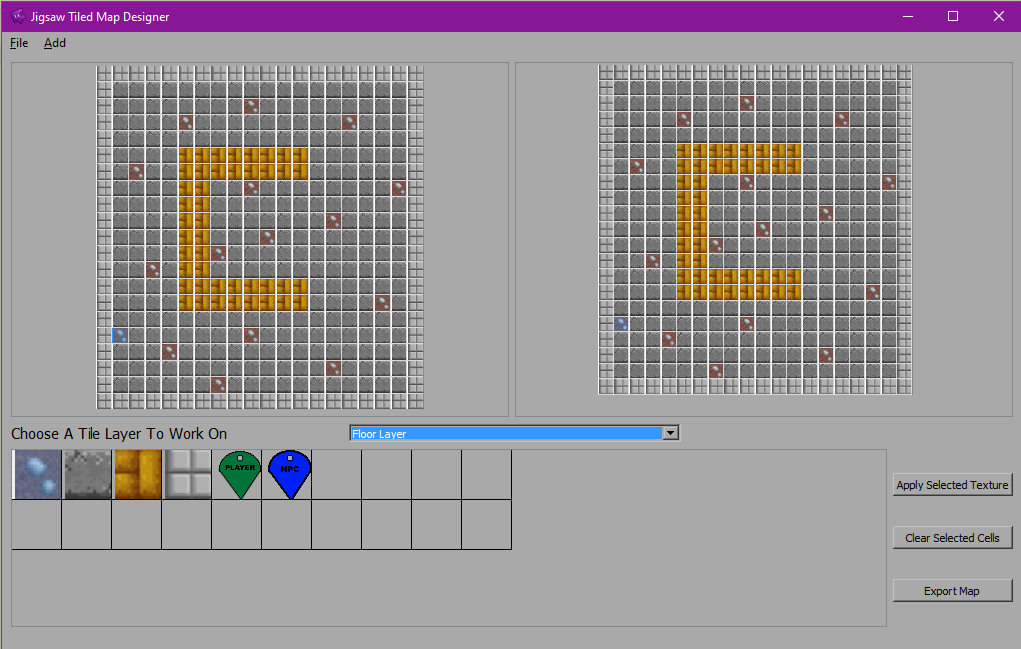
# 5. Project Description

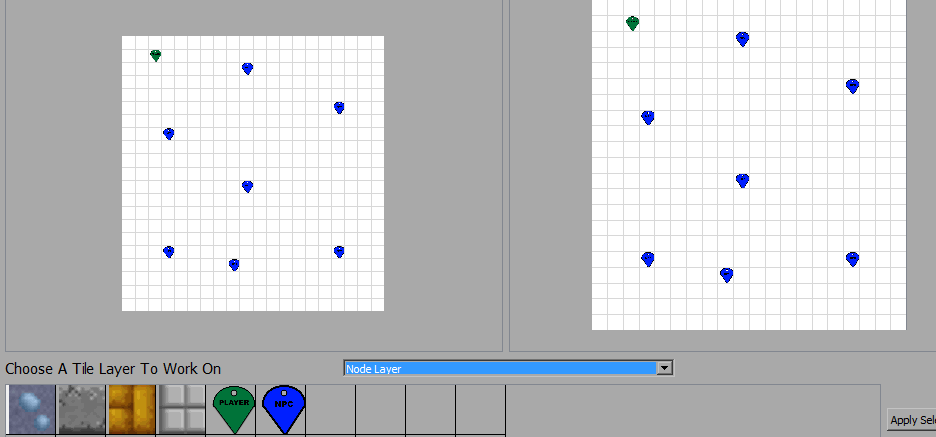
## 5.1 Description of Final Project

The finalized application which I have developed is a cross-platform, 2D tile based level editor. The application has been released on both windows and Linux Ubuntu. I tested the project on Linux Ubuntu 18.04.4 and Windows 10.

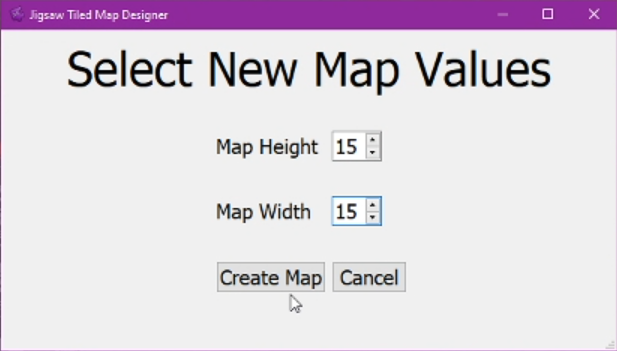
I was successful in implementing the features I hoped to implement when I first started development. Below is a Screenshot and brief description of each feature of the level editor and how it can be used.

The project has multiple features as shown below:

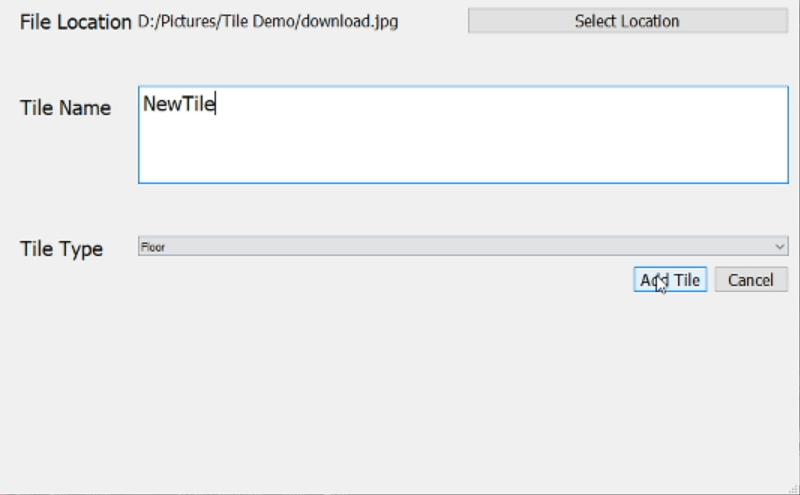
The level editor features a grid-based layout with two views. The first view as can be seen on the left allows for the user to interact with the grid and place their tiles. The left screen also allows for users to zoom in and out along with moving the camera with the “WASD” keys making it easier for users to work on a larger level and to make tile selection easier. The screen that is visible on the right side can also be interacted with. This screen shows a permanent full view of the user’s map, making it easier for the user to see where they are working when zoomed in.

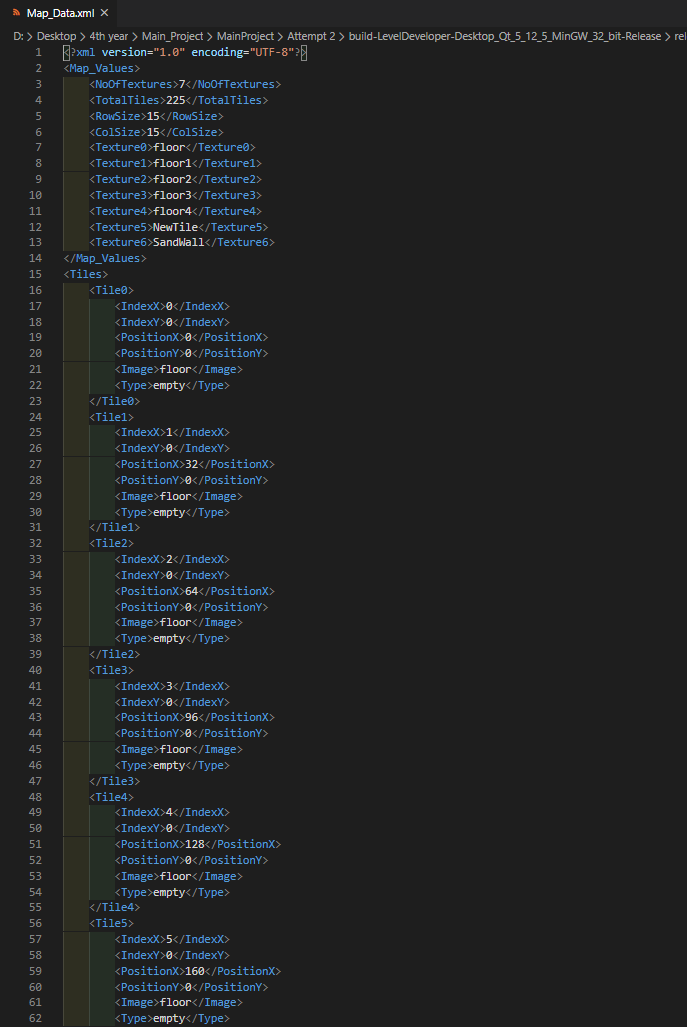


The level editor allows the user to place player and NPC nodes on a top layer of the map. The data related to these nodes are then exported to the XML file with the rest of the tile data. These nodes can then be loadedinto the game project and used to quickly place and assign the positions of the player and enemies in the level.



With the tile editor, users can set there own custom grid size, the minimum size being 1 tile x 1 tile and the maximum size being 99 tiles x 99 tiles. This feature allows for extra customizability of level shapes and sizes.

Users of the application can add their own Images to be used as tiles within their levels. The level editor allows a number of image types, including jpeg, bitmap and transparent png. The user can give their new tile a custom name and also set the tiles type. “Tile Type” is a variable that can be used for collisions when the level has been brought into the external game project and can be accessed through the XML. When the new tile is loaded the image is saved into the map export folder to ensure that if the project is moved from the pc or the original image is deleted the project will still work and have access to the texture.

When the user is happy with the level that they have created, they then have the option to export it to an XML file that they can import into their game project. The XML file holds the name of each texture, how many textures there are, and the size of the grid. The XML also includes the data for each tile and node, including the texture name, tile type, and tile position. This information allows for easy reconstruction when brought into an SFML or other game project.

Users can reimport their levels back into the level editor using the “Map\_Export“ folder. This allows for levels to be quickly updated and redesigned without the user having to start the level from scratch every time. This folder also holds all tile images meaning that the user does not have to set up all the image paths every time they open the map.

## 

## 5.2 Personal and Technical Learnings

I feel I had a number of personal learnings whilst working on this project that I can carry with me into the future. While working on this I learned how to work under pressure and under time constraints and deliverables. I learned how to quickly learn and use software that I have never used before. Along with the previous learnings, I also learned how to keep myself motivated and stay working when I hit issues and roadblocks within the development lifecycle.

Along with personal learnings, I also had many technical learnings whilst working on the project. The major thing I learned was how the Qt development environment works and how to use a signal and slot events loop instead of the usual game use that I am used to using. I also learned how to set up and develop on Linux whilst using Oracle virtual box to run the virtual machine with a version of Linux Ubuntu.

Another technical learning I had was working in a different style of update systems, rather than working a game or software loop that I would be used to using, Qt instead uses “slots and signals” for its updates. I found this to be confusing initially but by the end of the system, I understood how it worked and was comfortable with that update type.

# 

# 6. Project Milestones

## Presentations and Deliverables

We had multiple milestones and meetings to stick to whilst working on this project. Every Monday I had a meeting with my supervisor to show them the work that I had done the week before and tell him the work I had intended on carrying out the week after.

During the project, we had 2 presentations where we demonstrated the work we had done up to that point. For the first presentation, I had put a lot of work into the core feature of my application, tile selection and having it running cross-platform on both Linux and Windows. My key features I had aimed to get working after the first demonstration was to have the level writing to an XML file and custom level sizes.

For my second presentation, I had completed a lot more of the development of the application and was keeping well to the timeline and having the work I needed to do.

In the second presentation, I was able to demonstrate my finalised tile selection feature and I could successfully export my map and build it in my SFML test level. I had also successfully allowed users to add their own images as tiles and set their own map sizes.

Our final demonstration was the end of April 2020, I set myself a target of completing all the features I wanted to have by the middle of April to allow me to go back and fix any bug issues or any problems that were taking up more time than I had originally anticipated. Sticking to this timeline I was unable to add the abilities for users to add their own layers to the map.

I feel that I stuck to the timeline for deliveries as best as I could and met most of the goals as I had hoped. I feel that due to the nature of the type of application there was always going to be features or elements that I would like to have added but would not have been realistic in the time that I had for development.

# 

# 7. Results and Discussions

Below I show the results of the tests I have outlined in the study section of this paper.

A reminder that The First test was an overall test of the application and setting out to complete the goals I had set as a measure of success when I first planned out this application.

## Project User Test Results

|  |  |
| --- | --- |
| **Test Goals** | **Outcome** |
| The user will have the ability to layout and design their own custom map to their own specification. | Success |
| The user will have the ability to Insert their own Image and use it as a tile. | Success |
| The user will have the ability to place players and NPC Nodes that set positions within the SFML project. | Success |
| The user will have the ability to reimport a pre-existing map project into the tile editor and re-edit that map | Success |

After testing the application, you can see that I was successfully able to complete the goals I had set out to achieve when I started development on the level editor. I feel that this tool gives users the ability, to build a custom level and then import that level into their own game project.

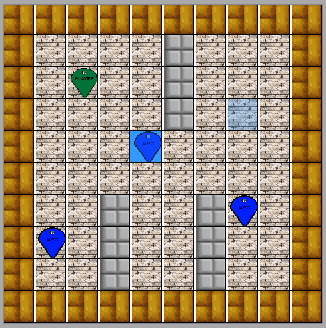
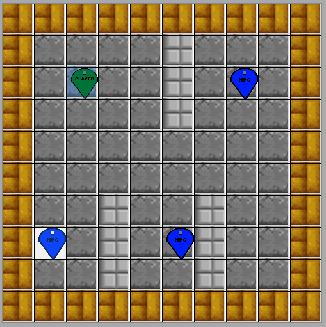
Whilst I feel that I was able to successfully construct a 2D tile based level editor in the given time period, I feel that in future developments, there are still more features that could be added to allow for even more customization to a user level.

## Testing The Amount Of Time To Build A Level Results

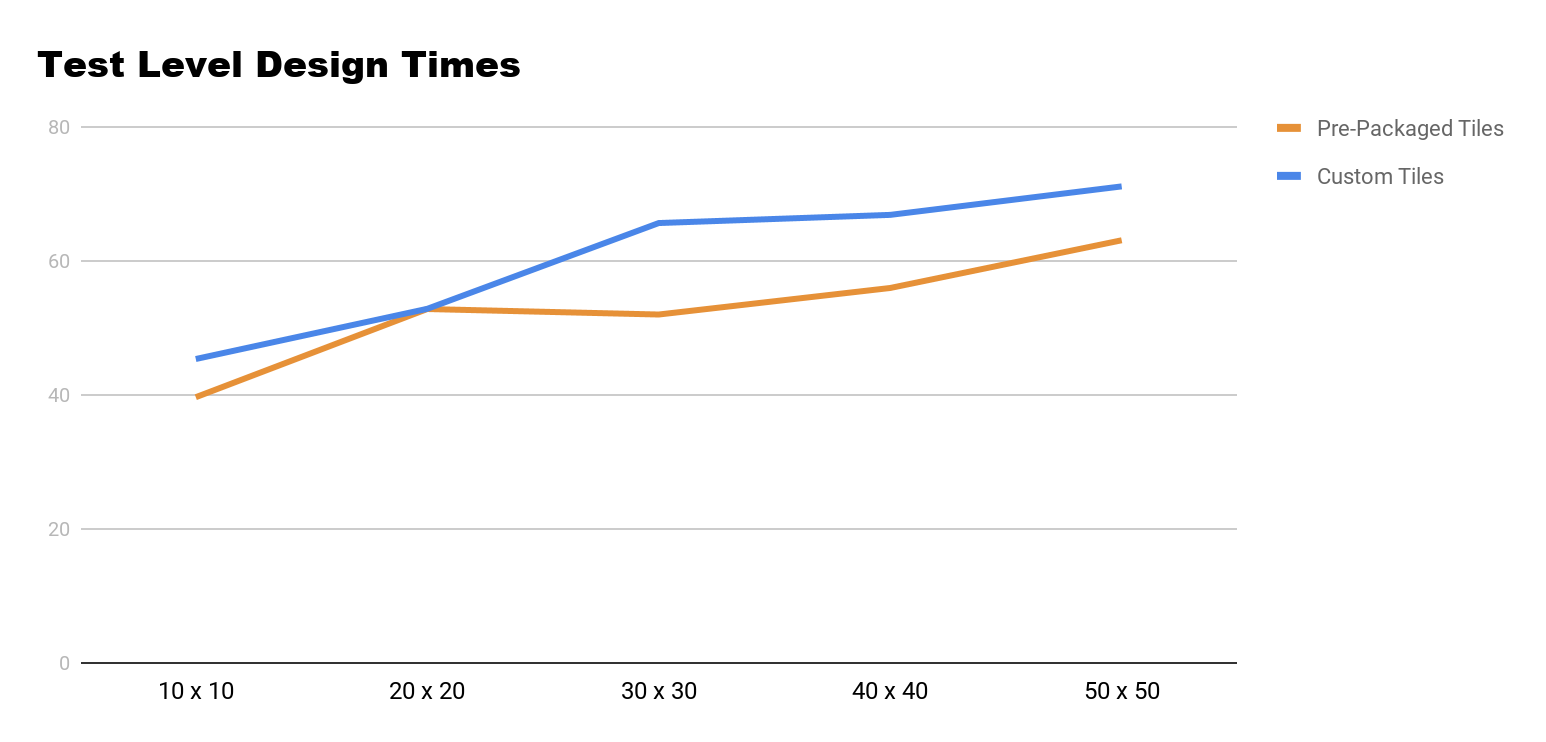
The below results outline the amount of time it took me to build a level in a grid of different sizes and the conclusions that I drew from this test. The full description of this test and the constraints I set to make it fair can be seen fully in the “Study” section of this paper but I shall outline a brief description below.

To ensure that the times are fair I tried to keep each map the same design despite the difference in map sizes. Each map featured the player position node and 3 NPC nodes. The time shows how long it took to launch the level editor, build the map and then export it to my SFML project and open it there.

Pre-Packaged Tiles: Custom User Tile:



|  |  |  |
| --- | --- | --- |
| **Map Size** | **Custom Tile** | **Time Taken** |
| 10 x 10 | YES | 45.36s |
| 10 x 10 | NO | 39.61s |
| 20 x 20 | YES | 52.85s |
| 20 x 20 | NO | 52.82s |
| 30 x 30 | YES | 1m 05s |
| 30 x 30 | NO | 52s |
| 40 x 40 | YES | 1m 06s |
| 40 x 40 | NO | 55.96s |
| 50 x 50 | YES | 1m 11s |
| 50 x 50 | NO | 1m 03s |



The average time to create a level using the pre-packaged tiles was: 52.68s

The average time to create a level using a custom tile was: 1m 04s

As we can see from looking at the graph and table of results The time it takes to build a level is relatively linear, it took approximately an extra 10 seconds to build the level when loading in a single extra tile, due to these tests being carried out by a human rather than the computer there is an extra margin of error that we have to take into account due to the tests been carried out by a human.

During testing I found that as the size of the grid got larger it the program was slightly slower to fill all the grid slots with the selected tile, I believe this may have occurred due to the whole table being filled through a single “For Loop”. A way to tackle this problem and perhaps make it more efficient would be to separate the tables out into a quarter of the grid which would allow for the fill algorithm to only have to check a smaller amount of tiles and therefore take less time to assign the tiles to them.

I feel that these results show that it is much faster and more efficient to build and layout a level using a tile-based level editor rather than trying to layout the level using purely coding and placing each tile, player and enemy by hand.

# 

# 8. Project Review and Conclusions

## 8.1 What was Successful & What Went Wrong

I feel that in general, the majority of the Project went as I had intended. I achieved many of the goals that I set out to achieve at the beginning of this development. At the beginning of the project, I ran into some problems whilst setting up my development environments and trying to learn how all the IDEs and software which I was using worked. Due to my inexperience in using the Qt development environment, I restarted my project a number of times in the early stages as I felt that the foundation I had built would make things more difficult to advance on the further I got into the Project.

## 8.2 What Could Be Done Differently

As outlined above, early in the development cycle I was forced to restart my project due to it not having the best foundation to continue development on, This taught me that if I were to attempt this project again from the start, I would spend more time planning out my foundations and getting to know the Qt environment, rather than just building each element without thinking about how it will affect or parts.

## 8.2 What Could Be added In A Future Version

Due to the application being built on the Qt IDE, it is very easy to port to many platforms. In the future, I would continue to develop the program to work on More Operating systems and port it to work on the Mac Operating System. Unfortunately, I was unable to have it as part of my project due to not having access to a Mac development device to test and develop on.

In the future, I would also add some extra functionality such as the users having the ability to add their own nodes and their own layers to the maps. I would also allow users to be able to freely place Images like stickers rather than them being selected like the tiles on a grid.

After adding the features laid out above I would add some smaller UX features such as button shortcuts to help speed up the users work time even more.

## 8.3 Findings

I found that having completed the level editor it made the level design much faster and much more efficient rather than trying to hand place all of the tiles along with the NPCs and the player.

I also found that it was, in fact, possible to build a level editor, in a 7 month period, that would allow a developer to create their level and then carry that across into SFML to be ready to use in a game. Despite it being possible to build the level editor in this time, I believe that it would take more time and more user feedback to be able to build a level editor application that has all the features to meet every goal that a developer needs.

# References & Bibliography

|  |  |  |
| --- | --- | --- |
| Referenced Publication | Citation | Reference |
| [1]Website | Techopedia | Techopedia. Cross-Platform Development [Online] (<https://www.techopedia.com/definition/30026/cross-platform-development>). (Accessed 5th December 19). |
| [2]Website | Rishabh Software(29/01/19) | Rishabh Software. (29th January 2019) Pros And Cons Of Cross-Platform Mobile Application Development[Online] (<https://www.rishabhsoft.com/blog/pros-cons-cross-platform-mobile-app-development>).(Accessed 5th December 2019). |
| [3]Website |  | Qt, Development Tools [online] (https://www.qt.io/development-tools ). (Accessed 29th April 2020). |
| [4]Website | Coppola D. | Bits Of Bytes, Davide Coppola.(“Market share of the most used C/C++ IDEs in 2018, statistics and estimates“).[online](<http://blog.davidecoppola.com/2018/02/market-share-most-used-c-cpp-ides-in-2018-statistics-estimates/>). (Accessed 2nd May 2020). |
| [5]Website |  | Qt Resources, Customer Success Stories. [online] (<https://resources.qt.io/customer-stories-all>). (accessed 2nd May 2020). |
| [6] Book | David P. & Russell D.(2009) | David P. and Russel D. (2009). David Perry on Game Design: A Brainstorming Toolbox. Boston. Course Technology CENGAGE Learning. |
| [7]Website |  | MobyGames, K.C. Munchkin [online] (<https://www.mobygames.com/game/kc-munchkin>). (Accessed 2nd May 2020). |
| [8]Video | Miller J (5/03/2020) | James Miller, Level Editors In VideoGames(5th March 2020).[youtube](<https://www.youtube.com/watch?v=akgdaexMt7E>). (Accessed 2nd May 2020). |
| [9]Website |  | SteamCharts, Garry's Mod [online] (<https://steamcharts.com/app/4000>). (Accessed 2nd May 2020) |
| *[10]*Website | Crawley D. (21/11/11) | VentureBeat, Dan Crawley(“ Indie game Minecraft receiving over 240 million logins every month”)(November 21st 2011). [online] ( <https://venturebeat.com/2011/11/21/indie-game-minecraft-receiving-over-240-million-logins-every-month/> ). (Accessed 2nd May 2020). |
| *[11]*Website | Fingas J. (15/09/19) | Engadget, Jon Fingas. (“‘Minecraft' now has 112 million Monthly Players”)(September 15th 2019)[online](<https://www.engadget.com/2019-09-15-minecraft-112-million-monthly-players.html?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAJyagQGoRgWksBjr4JAag3VuAYOD4V-GslTuaT2aMEUm1AXtNaDWaUTPJU2MWj66Z49t59n2-UHIgNUm9HRBoO7Bxlv3xeIegKYVyv54ZEYnYMku4mHvYGmnqT9njIkxz5WvVSf2SUeX02EbAja4u_yPgigyJqpefwTFqQayATyw>).(Accessed 2nd May 2020). |
| *[12]*Website |  | Sullygnome Twitch statistics, Minecraft [online] (<https://sullygnome.com/game/Minecraft> ). (Accessed 2nd May 2020). |
| *[13]*Website |  | Super Mario Maker,Make [online](<https://supermariomaker.nintendo.com/make/>).(Accessed 2nd May 2020). |
| *[14]*Website |  | Little Big Planet, Levels [online] (<https://lbp.me/levels?p=1&l=12>). (Accessed 2nd May 2020). |