

Qt 2D Tile Based Level Editor

Software Requirements Specification

Year IV

Eoin Abbey-Maher

C00217717

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 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. Project 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 [1]“*Super Mario Maker*” and [2]“*Little Big Planet”.*

# III. Table Of Contents

[**I. Acknowledgements**](#_thqjj9u48okq) **3**

[**II. Project Abstract**](#_vqrjjof656po) **4**

[**III. Table Of Contents**](#_ixokba5e7db1) **5**

[**1. Project Introduction**](#_mvmcpz7uv5mm) **7**

[1.1 Why This Project Was Chosen](#_39wh8fenxrvl) 7

[1.2 Potential Impact](#_tjgz1vy72hz8) 7

[1.3 Outline what is being done](#_d28d9lyl2f3) 7

[**2. Background**](#_a1k6cbc4skc8) **9**

[**3. Project Description**](#_t0z6xv3xr785) **10**

[3.1 Description of Final Project](#_yw6vqsmn4fr5) 10

[3.2 Personal and Technical Learnings](#_wj15ljey3ag6) 11

[**4. Overview**](#_e6zrq0nenwu2) **13**

[Philosophy](#_y8r4glqussin) 13

[Philosophical point #1](#_n8yf5xson9s1) 13

[Philosophical point #2](#_hin1o1q1wyf1) 13

[Philosophical point #3](#_9im2xbej0qfo) 13

[Common Questions](#_24uhha6tob2a) 13

[What is the project?](#_kb2cridd8es) 13

[Why did I choose this project?](#_eks1o0d3uaru) 13

[What is the main function of the application?](#_etfmy0dpqjw9) 13

[**5. Who is going to be using this application**](#_ki1noonekqr7) **15**

[**6. Context Diagram and Use Cases**](#_w093sbrt10u6) **16**

[**7. Metrics**](#_cwj20tfyn0dy) **17**

[**8. Is there a precedent for this application?**](#_ddxs3dgjbfwk) **18**

[**9. Design Manual**](#_vry1glqm8veb) **19**

[**10. Project Milestones**](#_8o8w45u35j13) **20**

[Presentations and Deliverables](#_xrwic8eq3fyk) 20

[**11. Project Review and Conclusions**](#_k21yrg9h9pi5) **21**

[11.1 What was Successful & What Went Wrong](#_jj1i04couj3j) 21

[11.2 What Could Be Done Differently](#_as0zpd4mtnsq) 21

[11.2 What Could Be added In A Future Version](#_akryg2g3phaw) 21

[11.3 Findings](#_fx6gmchsptw2) 21

[**12. References & Bibliography**](#_1b0bs8ck1zn3) **22**

[Referenced Publication](#_57fjwe3secw) 22

[Reference](#_57fjwe3secw) 22

# 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 add their own layers to the 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.

# 

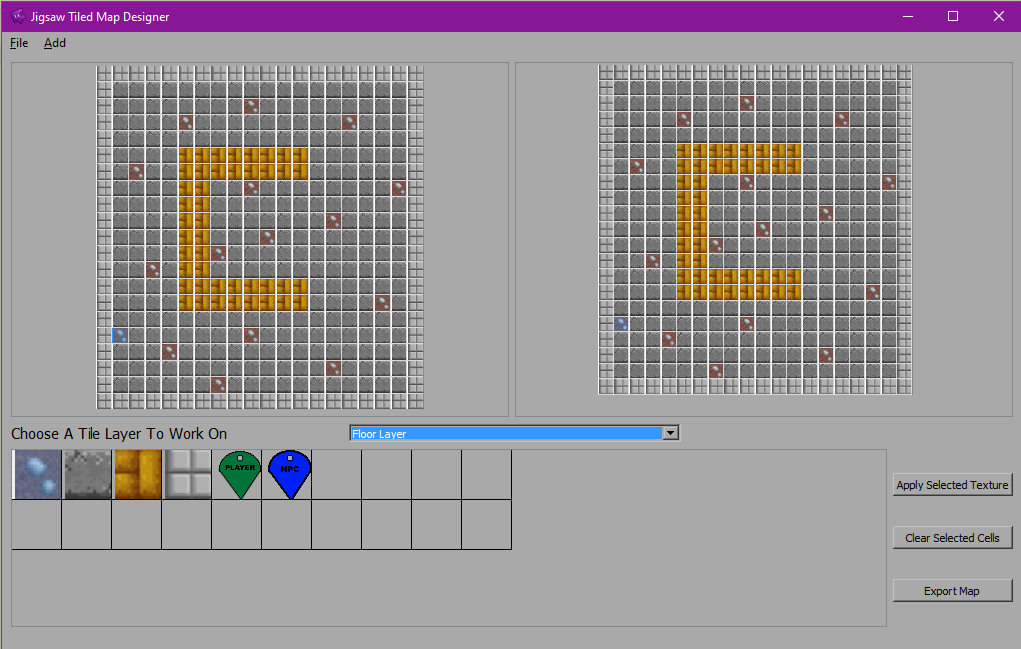
# 3. Project Description

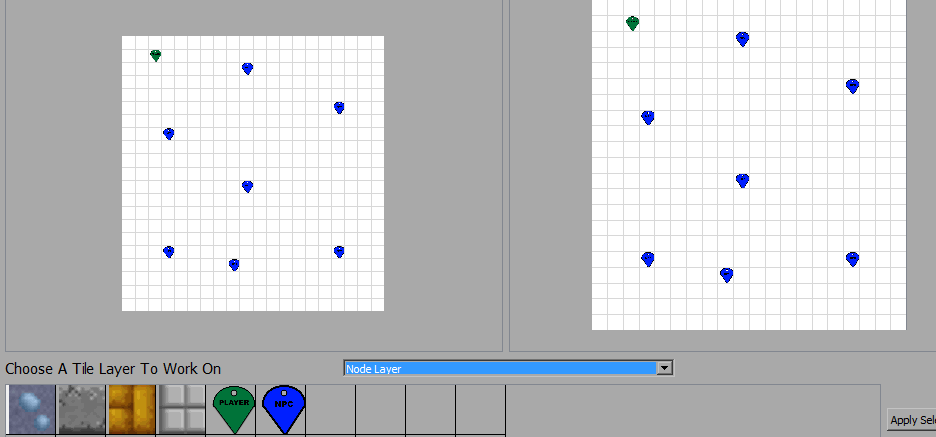
## 3.1 Description of Final Project

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.

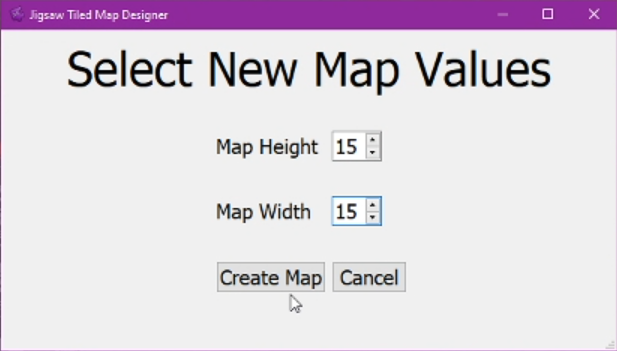
I was successful in implementing the features I hoped to implement when I first started development. Below is a Screenshot and brief description of each feature of the level editor and how it can be used.

The project has multiple features as shown below:

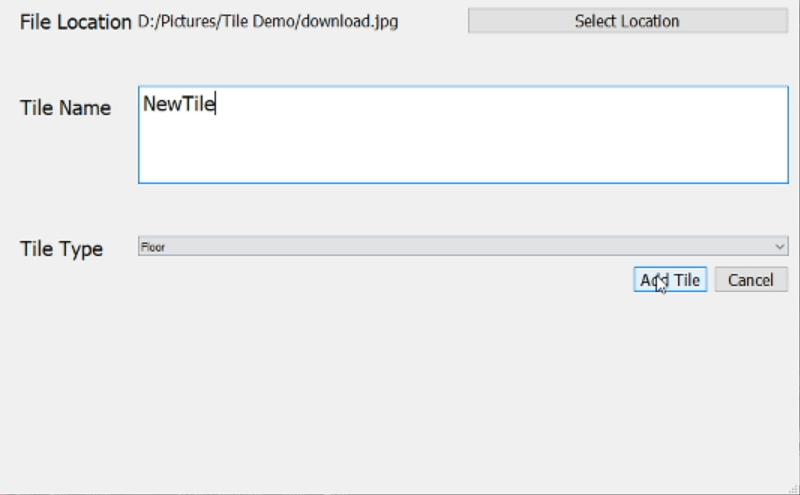
The level editor features a grid-based layout with two views. The first view as can be seen on the left allows for the user to interact with the grid and place their tiles. The left screen also allows for users to zoom in and out along with moving the camera with the “WASD” keys making it easier for 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.

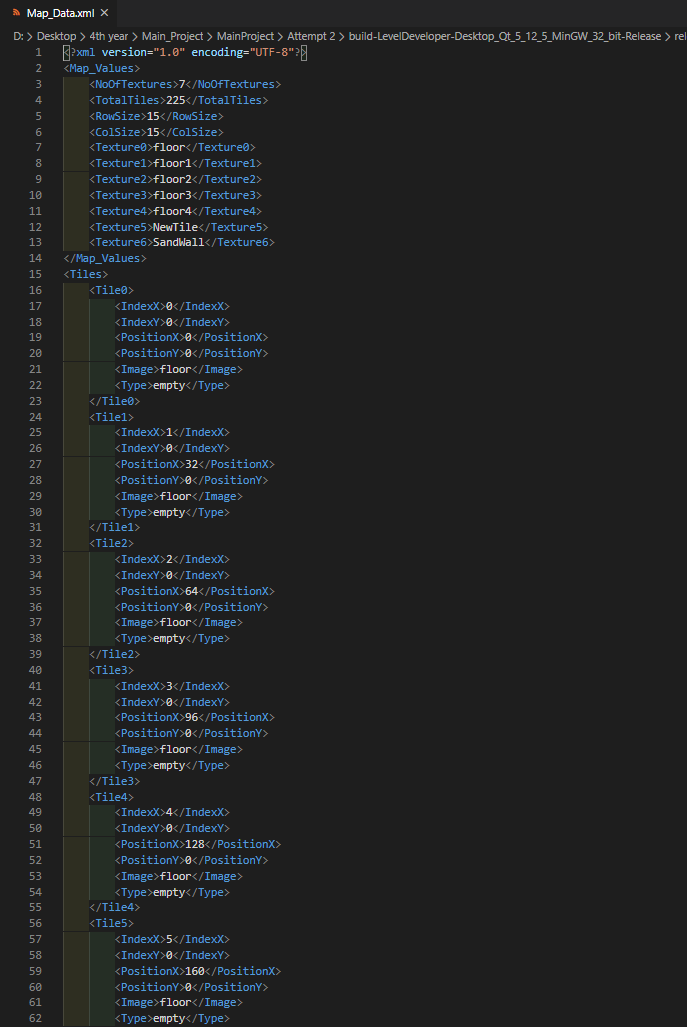


The level editor allows the user to place player and NPC nodes on a top layer of the map. The data related to these nodes are then exported to the XML file with the rest of the tile data. These nodes can then be loadedinto the game project and used to quickly place and assign the positions of the player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

## 

## 3.2 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.

# 

# 4. Overview

## Philosophy

### Philosophical point #1

The Application is trying to Help speed up the development time of 2D tile based levels for a usage in video games. This encompasses map sizes, tile types and player and NPC positions. Despite there already being tile based level editors available to the public, I am not trying to challenge these other editors but offer an alternative option for users to try and use if it fits to their personal needs.

### Philosophical point #2

This Tile editor is built for, and has been tested on, Windows 10 and Linux Ubuntu 18.04.4. I was unable to develop for the Mac OS due to a lack of access to Mac hardware and software for development purposes.

### Philosophical point #3

My software goals are to build a piece of software that can speed up the development time of levels and can give artists access to a software that allows them to plan out their levels without any knowledge of code.

## Common Questions

### What is the project?

The project is a 2D tile based level editor, it can be used for designing 2D game levels from a side-on or top-down perspective. It could be used to quickly design games in the style of “Super Mario” and other platformer side scrollers, or to design games like the Binding of Issaac which features a top down design showing a full room or dungeon.

### Why did I choose this project?

I chose this project because I wanted to make a program to quickly speed up the development time of levels for 2D games. I also chose to work on this project because I have always had an interest in the graphical and UI side of video games. Part of what affects the feeling of a video game is how the game's level is laid out.

### What is the main function of the application?

The main functionality of the application is the ability to set the size of your level and then layout the tiles that you want where you want, you can also then set them tile types and load in your own images to use as tiles on your level. When you've finished building your level you can export it to an XML file to be imported into your game.

# 

# 5. Who Is Going To Be Using This Application

This application has multiple possible user groups.

The first usage group I will cover is if the application is being used for the initial development stage of the game. In this case the game would be used by level designers or developers to lay out their levels and prototype what works and what does not work. The application can also be used by artists on the team to see how new tile designs work together and get an idea of how the level will be designed.

The second possible user group is if a developer decides to package this level builder with their game and releases it to the public. In this case the main users would be the players and the community of the game. In this case players could share their levels and create their own levels for the game

# 

# 6. Context Diagram and Use Cases

## Context Diagram

## 

## Use Case

# 

# 

# 7. Metrics

The measurement for success of this project is whether or not a user can successfully design a 2D tile-based level, export it to an xml file and then import it into a test project. To make this measure of success more specific I have laid out a number of tasks that my project should be able to achieve below :

* 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

Also due to this project wanting to speed up the development time of 2D levels, I want users, on their 10th time to run the program to be able to build their level in less than five minutes. I also want this goal to apply to all map sizes.

# 

# 8. Is there a precedent for this application?

The Inspiration behind this project was the “Tiled” Level Editor. Whilst working with the Tiled editor I enjoyed it and wanted to challenge myself to try to build a software that worked as well and sped up the development time of Levels. I knew when I started this project that It would be difficult for me to build something as complete as the Tiled editor in the amount of time that we had for the project. Due to this I tried to focus on the key elements needed to build a level and make the development time faster. Rather than challenge Tiled for the top spot, I wanted to offer an alternative that did the job just as well.

A second inspiration for this project was working on projects in college and in my personal time over the last 4 years, it always bothered me how long it took to build the construct game levels. Whilst I was in second year I worked on a much simpler version of a level editor by using a notepad file and drawing out the grids with different numbers representing different textures. I saw how this simple system was already helping with the speed of development, but I felt it was still taking me too long to draw the grid out with numbers, this was the reason I wanted to develop a visual version of my tile editor that would also allow me to set tile types, place my player and NPC positions, and be able to quickly swap out textures and edit my map sizes.

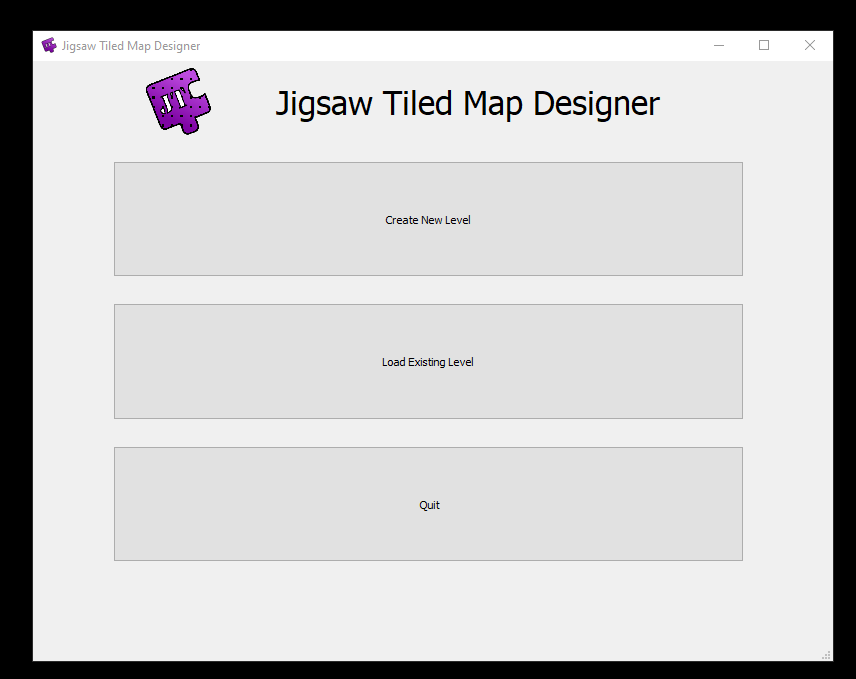
# 

# 9. Design Manual

## Using The Jigsaw Tile Editor

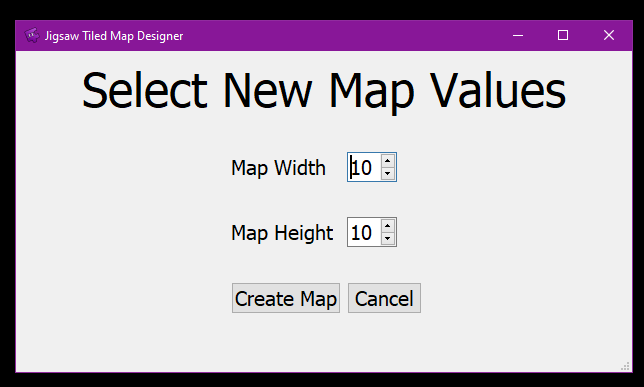
## Getting Started

When you first Launch the program you are greeted by a starting screen that has 3 options: “Create New Level”, “Load Existing Level” and “Quit”. To create Your Knew level Click “Create New Level”.



You will then be taken to set your map Height and Width

Set your Map Dimensions and click the “Create Map” button



You can now see the main map editing screen, read onto the next part of the manual to learn how to select and assign textures.

## 

## Assigning Textures

Ensure that the layer selector is set to “Floor Layer”

## 

Select one of the tiles from a row of textures that you would like to use, If you hover over the tile you can see its “Type”.



Highlight the tiles you would like to set the texture to.

Click the “Apply Selected Texture “ button to apply textures to the highlighted cells



## Adding A New Tile

Click the “Add New Tile” Button



Click The “Select Location” button to search for image location



Navigate to the File location and Choose your image



Set the tile's new name in the “Tile Name” text box and choose the TileType from the drop down menu.

Click “Add Tile” and your new tile should be added to the selection table.

## Adding Player and NPC Nodes

Change the layer selector to be on the Node Layer

## 

Choose the Player or Npc Node in the tile selection table.

Select the Cell you want the node to go and click “Apply Selected Texture” or double left mouse click.

## Exporting Your Map

Click the “Export Map“ Button.



Your “MapExport” Folder can now be found in the root directory of your application

**\*\* N.B. Do Not Change The Name Of This Folder Or Its Contents** \*\*



## 

## Opening Existing Levels

On the main menu click the “Load Existing Level” Button

## 

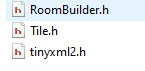
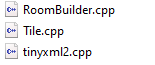
Navigate to your MapExport Folder and select the entire folder

This should now open the Level editor with the tiles and Nodes of the level filled in.

## SFML Setup

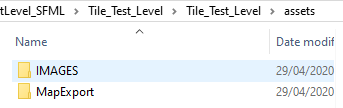
## Setting up the Files

Ensure you’ve included the Room Builder, Tile and Tinyxml2 files in your project and that they can be accessed



`

In your “assets” Folder place your “MapExport” folder



# 

## Setting up Your Game

Include “Roombuilder.h”

Create a “RoomBuilder” object in your Game.h.

In the constructor of your game class construct your room builder and pass back your sf::RenderWindow

Load your file using RoomBuilder::loadfile(FileAddress)

If you have a Player and NPCs use RoomBuilder::loadFile(fileAddress, Vector<Enemy>, Player)

If not use RoomBuilder::loadFile(fileAddress)

In your render function add RoomBuilder::render()

## Setting up Collisions

Set Up a Collision check function

Pass the RoomBuilder Object back to this function

The following piece of code can be used for collisions to check the tile type:

int t = 0;

for (int r = 0; r < t\_roombuilder.m\_mapHeight; r++)

{

for (int c = 0; c < t\_roombuilder.m\_mapWidth; c++)

{

if(m\_bodyShape.getGlobalBounds().intersects(t\_roombuilder.m\_tiles[t].m\_bodySquare.getGlobalBounds()) && t\_roombuilder.m\_tiles[t].m\_type == TileType::WALL)

{

m\_position = m\_lastPosition;

m\_bodyShape.setPosition(m\_position);

}

t++;

}

m\_bodyshape refers to the players sf::RectangleShape.

Room Builder has a vector of tiles for each table map, each of these tiles has a “tile type” which declares if the tile is a Wall or Floor Tile.

\*The included RoomBuilder and Tile files are only a foundation for reading the XML File and Setting up the tiles in the SFML Example level. There may be features or requirements overlooked that need to be added to make these files work better with your project needs, or to connect better with additional AI pathfinding methods and other external libraries.

# 10. 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.

# 

# 

# 11. Project Review and Conclusions

## 11.1 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.

## 11.2 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.

## 11.2 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.

## 11.3 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.

# 12. References & Bibliography

|  |  |
| --- | --- |
| Referenced Publication | Reference |
| *[1]*Website | Super Mario Maker,Make [online](<https://supermariomaker.nintendo.com/make/>).(Accessed 2nd May 2020). |
| *[2]*Website | Little Big Planet, Levels [online] (<https://lbp.me/levels?p=1&l=12>). (Accessed 2nd May 2020). |

### 