**SCHOOL OF COMPUTING**

**UNIVERSITY OF TEESSIDE**

**MIDDLESBROUGH**

**TS1 3BA**

**Multimodal Interfaces for Games (COM2061)**

**ICA Component 1  
Graphical User Interface for a Games Tool**

**<Matthew King-Mason>**

**<N3067685>**

**10th December 2014**

Table of Contents

[1 Introduction 3](#_Toc378578913)

[2 Software Design 4](#_Toc378578914)

[3 Results 5](#_Toc378578915)

[4 Conclusion 6](#_Toc378578916)

[5 REFERENCES/BIBLIOGRAPHY 7](#_Toc378578917)

# Introduction

I designed my editor to work with my game “Mary-Jordan Magee” I will be making in my Games Engine Construction Module, the game will be in the style of “Super Mario” and will be a side scroller. The game will feature a knight character which be the main playable character, several power-ups, and the objective will be to reach the end of the level where the player will rescue the damsel. My editor will be able to; alter the player colour and some of it’s’ mechanics, it will also be able to create a map for a level and pick the back ground for that level. Finally my editor will be able alter the number and frequency of the enemies.

# Software Design

The first feature I designed was a tabular view to hold each section of the editor. Each tab is appropriately named so the user can easily find the section they are looking for.

The next feature I designed was the character colour changer I originally designed the feature to be able to change the colour of the character programmatically so the user could choose any colour. However after looking further into the design and implementation of this I chose to scale this feature down to using a list view of colours and having pre-set colours. The colour of the character is displayed as sprite image next to the list and is loaded by pressing a button underneath the list. I chose to include the button as it prevents accidental changes in the character colour before saving. This feature is saved in an xml document using the path of the image as the variable. The path in the xml document is: “Mary-Jordan Magee/Player/Colour”, I chose this path as it is simple and self-explanatory. (See figures 1 & 2)

I then designed was the character variables, these are; the number of lives, the run speed and the jump height. The number of lives is set by the user inputting a number into a text box. The text box length is clamped at three characters so the value cannot go above nine hundred and ninety nine. The text box does not allow the user to input any characters other than numbers, if the user tries nothing happens. (See figures 3 & 4)

Both the run speed and jump height are combo boxes, I chose combo boxes over a text box in this case as I wanted the user to pick a pre-set value that wasn’t too big and had sensible values. (See figure 5)

The path for saving the character variables data is similar to that of the character colour, the final “Colour” is replaced by “Lives”, “Run”, and “Jump”.

I then decided to implement a menu system so the user could easily find where to save and load their character data and map data. I decided to save and load from different xml documents for the character data and the map data as the xml documents will be used in different areas of the game so it is easier to split them up. (See figure 6)

The next feature I made was the background, this feature was slightly different to the rest of my editor as I chose not to have this save in the Xml. I instead decided that the user will load an image into the editor, then when they click save it will resave the image into the correct place with the correct name and in the file format “.png”. This provided the problem that I wasn’t sure how to save the image let alone save it as a different format. The solution however proved to be easier than first expected and I was able to keep the functionality. (See figures 7 & 8)

My next design feature was the enemies, this was a relatively simple design consisting of three combo boxes and three sliders whose values are displayed in text boxes. The combo boxes contain the relative number of enemies the game will spawn and the sliders have values between one and one hundred for the frequency in which the enemies will spawn. The path for the saving is split into three elements under the root node; “WeakEnemy”, “MedEnemy”, and “ToughEnemy”. Then the values for the enemy count and frequency are stored under them; “Count” and “Frequency”. (See figures 10 & 11)

My final piece of functionality was the map editor, I had originally designed the map editor so that the each tile would only be allowed to be placed if the adjacent tile was compatible. However I decided against this system in the end as I had chosen a large sprite sheet with a lot of different tiles many of which are power ups, the addition of the power-ups were the main deciding factor in redesigning the map editor.

My map editor is a combination of a list view holding the tiles and a canvas which has a child grid which in turn has a child data grid. The canvas width is decided by the user depending on how long they want their level to be, the maximum length for a level is two hundred as if it is much bigger the application takes too long to process. The purpose of the grid is so the user can only place tiles in a singular section instead of on top of each other. The data grid is so that the user can see the grid making it aesthetically pleasing. The user is able to place a singular tile on the grid or thy may “paint” tiles on to the grid if they wish to place more than one of the same tile. (See figures 11, 12 & 13)

The saving and loading of the map proved to be a difficult task as the canvas wasn’t a set width so often the editor tried to save and load outside the bounds of the array which held the x and y values for the index of the tile. The map XML has the path “Mary-JordanMagee/Map/Tiles” and saving uses the set attribute method of “XDocument” so each element has an X, Y, and TileID value. By default the TileID is set to zero as this is where the background will be in view in the game. The editor automatically adds a blue tile when loading to visualise this. (See figure 14)

To help the user I designed a new window that appears at when the program is started, and it can be started by the user. The window has several text blocks containing instructions on how to use the editor. (See Figure 15)

# Results

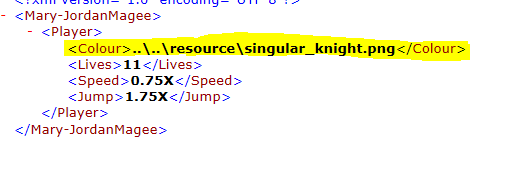
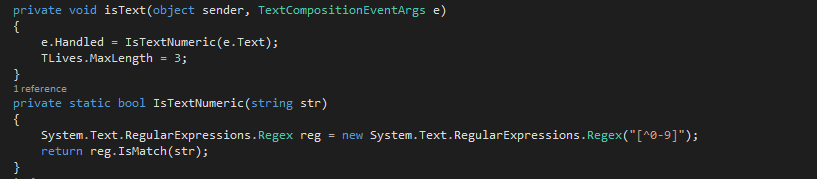
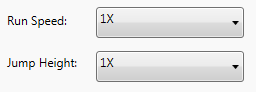
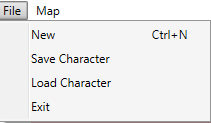
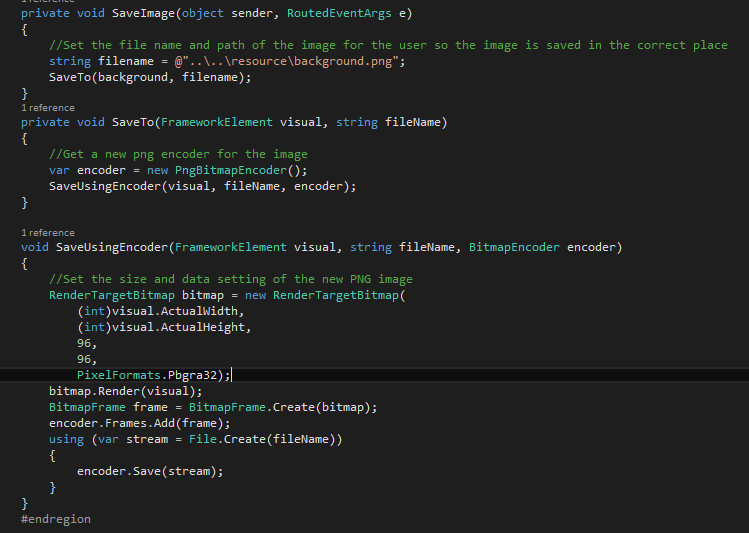
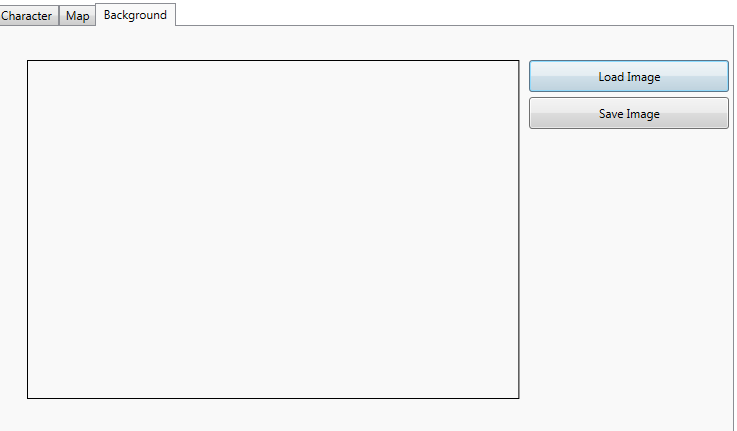


Figure 8

Figure 7

Figure 6

Figure 5

Figure 4

Figure 3

Figure 2

Figure 1

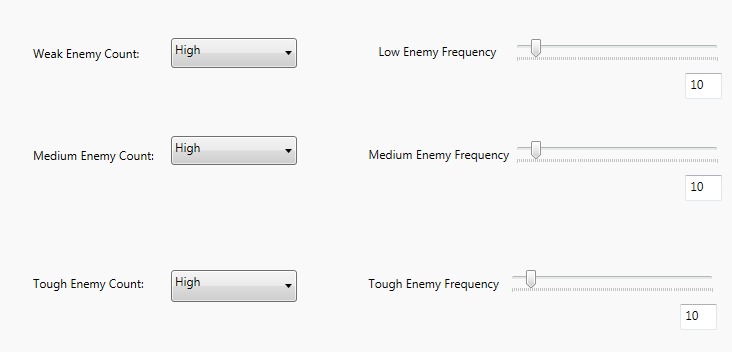


Figure 9

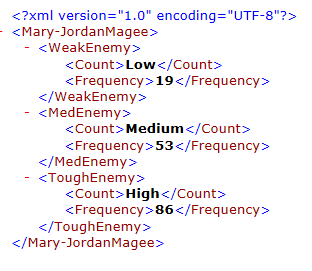


Figure 10

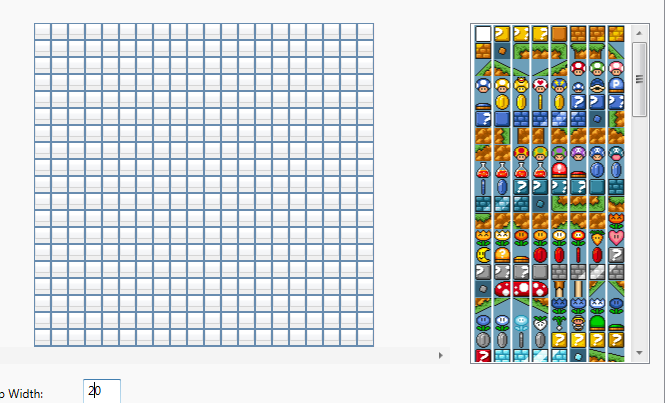


Figure 11

Figure 12



Figure 13

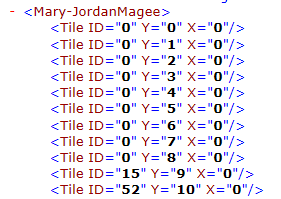


Figure 14

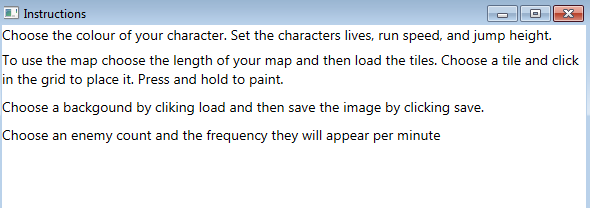
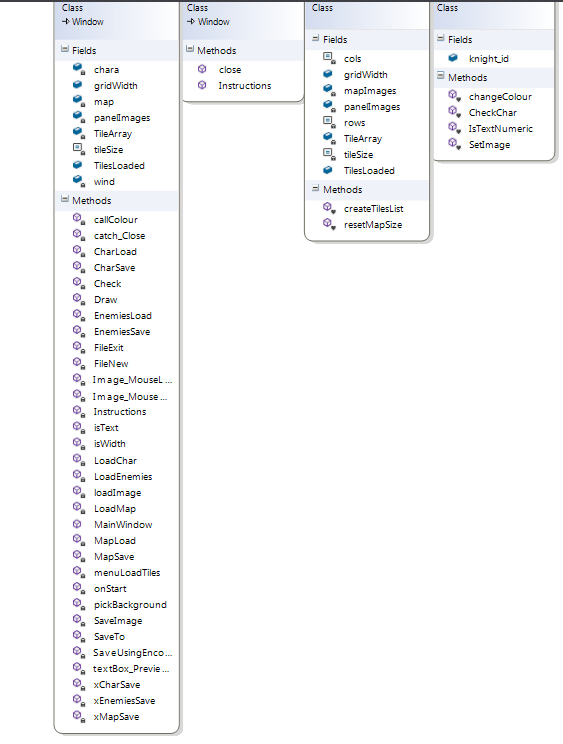


Figure 15



# Conclusion

Overall I managed to achieve most of what I wanted to in my editor, the only things I failed to implement were the advanced colour changer, and the tile checking system. The editor allows the user to choose their colour and set some character mechanics. They are able to set the number and frequency of different levels of enemies, and finally the user is able to create a map and choose the background for the level.