Maze Game Testing Document

By Rory Gallie

# 3. Goals and vision

The goal of this project is to have a working and tested maze game that is fully functional including some design given from the requirements brief.

The goal of this document is to outline the testing process of the “maze game” that is required for finding bugs within the application.

# 4. Test Plan

4.1 Objectives

The main objective of this test plan is to discover any issues or bugs within the newly created maze game.

This test plan supports the following objectives:

* To define the scope of what will be tested.
* Specify the approach taken to testing.
* What deliverables can be expected from test activities.

4.2 Scope

The scope of this test plan is the newly created maze game from unit testing perspective and manual testing within the User Interface.

Testing the Maze will be by the following means:

* System testing functionality.
* Testing the User Interface manually.
* Beta testing

I would regard testing the user Interface tests as a type of use case for the system as well.

4.3 Testing strategy

This project follows a reactive testing strategy, meaning that test activities are started after design and coding are completed.

All unit testing will use the following library for testing:

* NUnit – version 3.12.0
* NUnit3TestAdapter – version 3.15.1

4.3.1 Unit testing

Unit testing will test the following within code:

* Test the Rooms total amount.
* Test the rooms have correct map dimensions from loaded file.
* Test the game regions are correct to the picture box.
* Test the player movement is working correctly.
* Test the players actions to attack or pickup coins works correctly
* Test the coin value is correct
* Test the files loaded from the levels (rooms) directory is correct.

4.3.2 User interface testing

The user interface testing will be to test the following during one run of the main application:

* Can the user move to a new room if a threat does not exist in current room.
* Can the user collect coins and kill threats
* Can the game be restarted at any point
* Can the user complete the game

*4.3.2 Beta Testing*

Beta tests will be done by providing a few colleagues with access to the maze. This will be to get their opinion and any bugs they may have found during the experience.

# 5. Tests cases with results

5.1 Unit tests

|  |  |  |  |
| --- | --- | --- | --- |
| Test name | Test process | Expected results | Actual results |
| Test Rooms Total | Run test in test explorer in visual studio | Test should check the rooms total amount of rooms | Test passed with correct values |
| Test Room Dimensions | Run test in test explorer in visual studio | Test should check the rooms map dimensions are the same as loaded map dimensions | Test passed with correct values |
| Test Game Regions | Run test in test explorer in visual studio | Test should check the game regions are correct to the Room map regions | Test passed with correct values |
| Test Player Move | Run test in test explorer in visual studio | Test should check that on a player move, location has changed | Test passed with correct values |
| Test Player Action | Run test in test explorer in visual studio | Test should check that on a player action, action is performed | Test passed with correct values |
| Test Coin Value | Run test in test explorer in visual studio | Test should check coin value exists from selected random seed. | Test passed with correct values |
| Test Load | Run test in test explorer in visual studio | Test should check correct amount of files are loaded from folder. | Test passed with correct values |

5.2 Manual Tests

|  |  |  |  |
| --- | --- | --- | --- |
| Test name | Test Process | Expected results | Actual results |
| Can move new room | Start MazeGame.exe  Hit start on menu form  Use arrow keys to move to a passage | Player initially gets a popup window if threat exists if not can move to new room | Player does get popup and if threat is killed player can move to next room |
| Can kill threats an pickup coins | Start MazeGame.exe  Hit start on menu form  Use arrow keys to move to a threat  Press spacebar to kill threat | Threat popup appears saying the threat has been destroyed and how much life value has been lost. | Threat is destroyed from spacebar press and popup states that health has been lost, that health amount is also removed from health box. |
| Can be restarted | Start MazeGame.exe  Hit start on menu form  Click the button “start new game” | Button is clicked and new game is initiated | Button is clicked and game is reset back to base with all values being reset. |
| Can the user complete the game | Start MazeGame.exe  Hit start on menu form  Play the maze to room 7 | In room 7 if a threat exists player cannot use exit block however if threat is destroyed player can exit and complete the game | Player is able to complete the game with a popup displaying their current health and wealth. There is also an ok button to exit game. |

# 6.0 Beta Testing

Beta testing will be performed by 2 different colleagues at my current workplace.

Firstly, Corentin beta tested the game and had the following to say:

*“The game start-up screen showed instructions on how to move and interact with the various elements of the game (coins and threats), which were helpful to learn how to use the game and progress. The controls were reactive, messages were displayed to explain the outcome of my actions. Overall the game was easy to understand and straightforward to play. I noticed a bug that occurred when pressing any key which was not one of the keys used by the game, in which case the player would move in the direction of its last movement. Otherwise there was no issue.”*

With the bug found by Corentin, this has now been solved and relevant information surrounding this can be found in section 7 of this document.

The statement from Afaq is as follows:

I enjoyed doing beta testing for your game. However, I found one bug when resizing the main window. The rest of the GUI remained the same size.

The bug from Afaq has now been solved and is documented in section 7 of this document.

# 7.0 Bugs found and changes done.

7.1.0 Exit location bug

A bug was found with the exit block code during manual tests, if a user was to destroy all threats in the room with an exit block any passage in the same location as an exit block on other maps would then become an exit block.

This was due to the code in GenerateGame.cs having

Rooms[7].NoThreats == true

in the if statement, this was then changed to

Rooms[CurrentRoom].Map == Rooms[7].Map && Rooms[7].NoThreats == true

This was done to check if the user is also in the same room.

7.1.1 Coin Value and render Bug

A bug was found during the manual testing section that coin values would be different to the displayed colour on screen and upon entering / exiting a room these would change every time allowing a player to always get a gold coin.

To solve this issue all information regarding coins required some design changes. The following changes occurred to solve this bug:

* Changes to the LinkRooms function
* Removal of the UpdateCoinData function
* GenerateGame function changes
* Changes to rendermap function

LinkRooms changes consisted of making the system more dynamic to the current room. The bug found was due to this function not updating the room to be one of the stored data in the array of room objects. But updating the current rooms map to the new room map it should move too.

To solve this, I added a new line to make the current room be the new room once the if statement checks parameters are correct:

CurrentRoom = UpdatedRoom.North;

This also meant that the previous bug in section 7.1.0 could also be changed to state the current room over hard coding room 7 to be the room for selection.

I have also made a change so that within each statement, it will return the updated room. This is to stop the code from running other statements once a room has updated and just to return the new room.

UpdateCoinData information was removed with all its information being stored into GenerateGame this has been done so that I can set a number of coins and have them be based on the current room.

As shown with the following line:

public Coin[] Coins = new Coin[8];

Within the Generate Game a for statement was added to loop over each room stored within the room array and then for each coin set the value and location.

I then updated the render function to render coins based on their value in the coin array which is set to be 0-7 which is relevant to each room. This meant I could pass in the current room value to return the correct coin in the current room.

7.1.2 Beta Testing Player movement bug

The bug that Corentin found was a simple fix.

The switch statement for player movement did not set movement to none during the default switch

The following lines of code where added to the default case:

GameMap.PlayerMovement = GenerateGame.Movement.None;

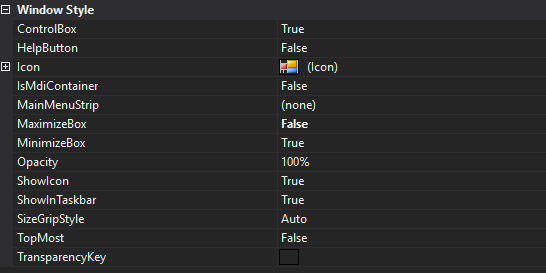
GameMap.PlayerAttack = GenerateGame.PlayerAction.NoAction;

7.1.3 Resize bug

The game form window had the ability to be resized.

This should not have been able by the user.

To solve this bug the following setting was changed using the windows form design editor in visual studio:



The following section of “MaximiseBox” has been changed to false from true, solving the issue.