|  |  |
| --- | --- |
| Program #3 | Planning Document  James Scott  Colin Riley  Stephen Belden  Shaya Wolf  Neil Carrico  04/08/2016 |

**Project 3**

The first thing we did for this part of the program was read in that raw maze information file. Because the values in the file were ints/floats, we were able to use the convertToInt() and convertToFloat() methods included in Java. We read in this file successfully, but our code was ugly. Despite constant efforts to clean and maintain the code there are still blemishes, such as this function, that we will fix when we have time.

Next we drew lines on the tiles. These lines matched with the data inside the file and were drawn correctly. We checked with another group as well to make sure that our layout was not modified. Assuming they didn’t change their code, it should still match.

You should now be able to move around the tiles wherever you would like. This was included in Project 2, but now the tiles have their picture on them and the game has some resemblance to a maze.

Lastly, we added functionality to the reset button. This was done in such a way that the tiles all return to their original positions on the sides of the game board. It is not an undo button.

This program is finally starting to look like we imagined back in February! One issue that we did run into that caused some problems was the same exact code working for some computers in the group, but not others. The gameWindow was having some issues that we will continue working on through the end of the semester.

**Future Plans**

There is still quite a bit to do on this project. We still need to put functionality behind the “New Game” button, win conditions and legal/illegal moves will have to be enforced. We will also need to implement a lot of testing on our final program to find the other bugs that you simply can’t plan for a week out.

Meeting times continue to fluctuate based on class schedules and class presentations. Most work is being done throughout the week separately, and then being combined throughout the week, utilizing Slack. We will continue to meet Friday’s to finish projects before they are due.

**Anonymous Class Explanation**

We were instantiating new action listeners in the actionPerformed() method, which required the use of anonymous classes. This was a problem, so we removed these classes by deleting them and instead called the getActionCommand() method. This was definitely the more proper and clean than what we had. So we will be keeping these changes and avoiding anonymous classes in the future.

**UML**

GameWindow

+ << constructor>>GameWindow

+ actionPerformed (actionevent)

+ setup()

+ addButtons(GridBagConstraints)

+ setClicked(Tile)

+ readInt(FileInputStream)

+ convertToInt(byte[])

+ convertToFloat(byte[])

+ newButton: JButton

+ resetButton: JButton

+ quitButton: JButton

+ lastClicked: Tile

<<Interface>>  
Action Listener

Tile

+ << constructor >> Tile(int, Point[])

+ << constructor >> Tile(int)

+ getID()

+ setID(int)

+ getPoints()

+ isEmpty(): Boolean

+ makeEmpty():void

+ makeLive():void

+ switchState():void

+ reset():void

+ mousePressed():void

+ debugPrint():void

+ mouseClicked(MouseEvent)

+ mouseEntered(MouseEvent)

+ mouseExited(MouseEvent)

+ mousePressed(MouseEvent)

+ mouseReleased(MouseEvent)

- ID: int

- points : Point[]

- isEmpty: Boolean

- border : Border

- NoBorder : Border

**Uses**

JFrame

**Uses**

Line

+ << constructor >> Line(Point, Point)

+ getBegin():Point

+ getEnd():Point

+ debugPrint():void

- begin : Point

- end : Point

<<Interface>>  
Mouse Listener

JLabel