Group Afa

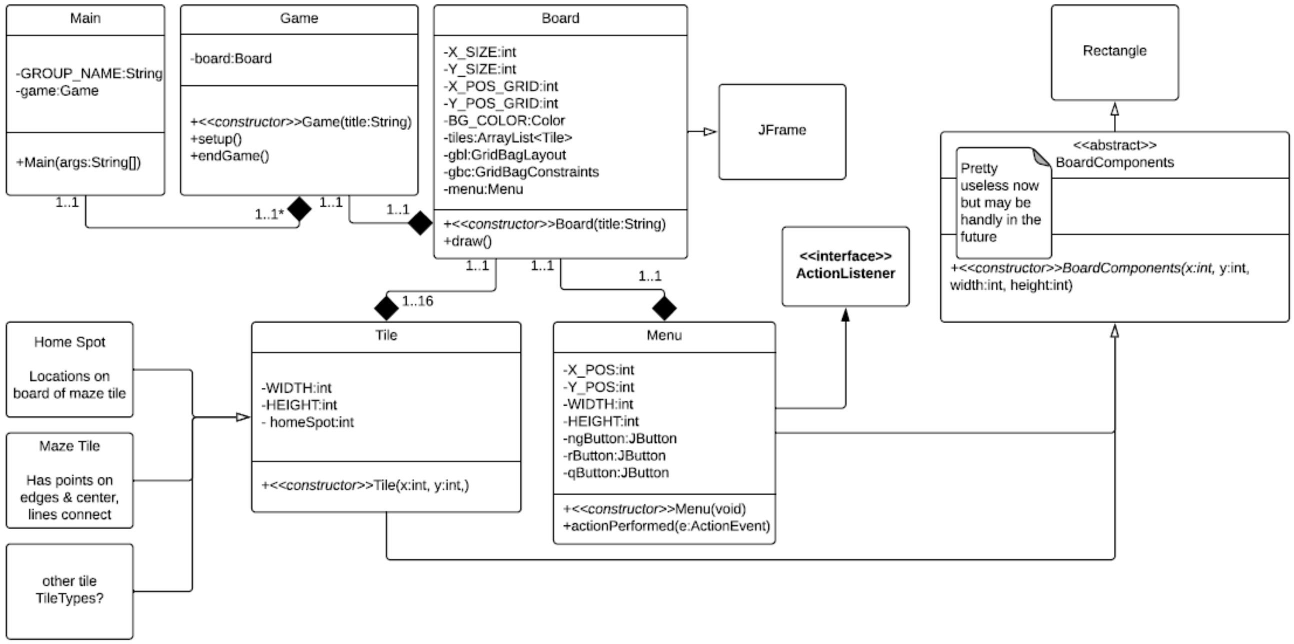
COSC 3011-01

5/13/2019

Jordan Ruckle
Jared Spaulding
William Cary
Shin Yamagami
Miles Golding

A Brief History of our Project

Program #01 began for our group with a mild amount of confusion. Our first meeting resulted in the beginning of our planning for this project. Our first group meeting was set at coal creek and consisted of a detailed discussion of concrete deliverables for the first assignment as well as a small amount of get-to-know-you questions. Group members erroneously interpreted the guidelines presented by Dr. Buckner to only desire detailed planning for the project. The result of this detailed planning resulted in the following UML diagram created with LucidChart.



This original design was only modified slightly over the course of the project and was only modified slightly over the course of the project to conform to recommendations suggested by the client, Dr. Buckner. Though our group seemed to have a good grasp of the planning, we fell short in a few areas on the first assignment. A loose interpretation of the guidelines led to several creative choices being made by the team that should have been left as described by the original request. Buttons were mislabeled and the layout of the board did not conform to the document provided by the client. After an urgent reconvening to address these criticisms, our group adjusted our initial product and begged forgiveness from our client. A planning document was put in place to document our current goal in a concrete fashion and to be a place to consolidate useful deliverables from each new request. Our planning followed a simple layout of instructions simplified from the client's descriptions.

Milestone 1:

Add a menu to window

Add buttons to menu

Make adjustments to the menu so alignment is according to diagram

Make quit button work

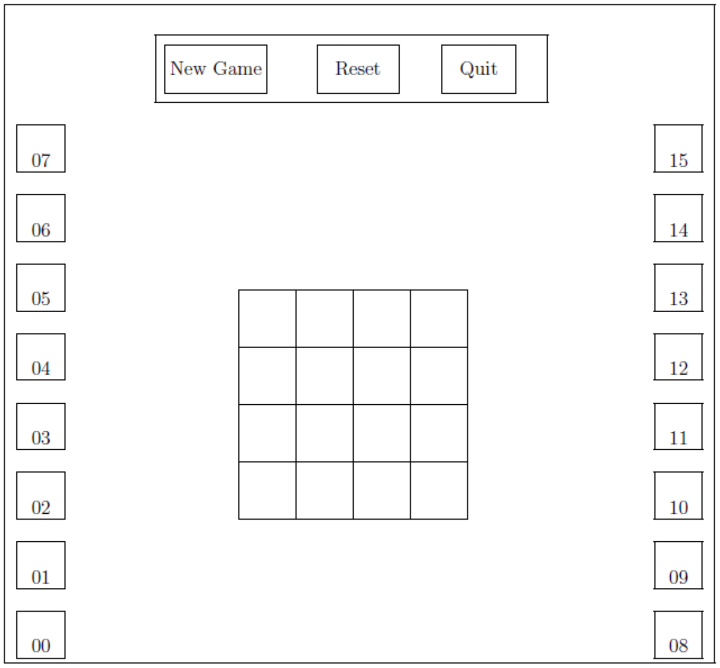
Add 8 Tiles on left and 8 Tiles of the right

Add a grid in the center

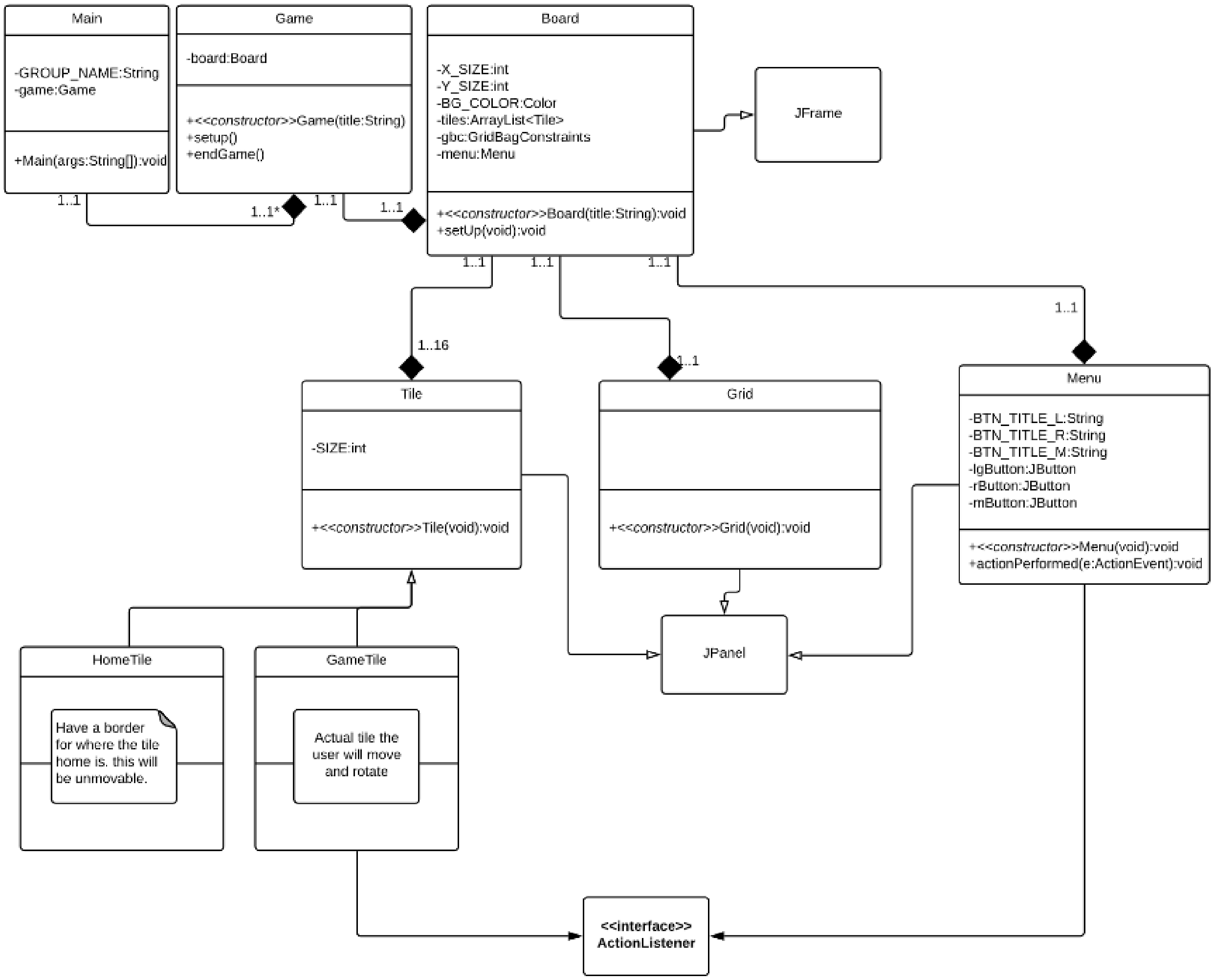
Set title to group name

Change background color

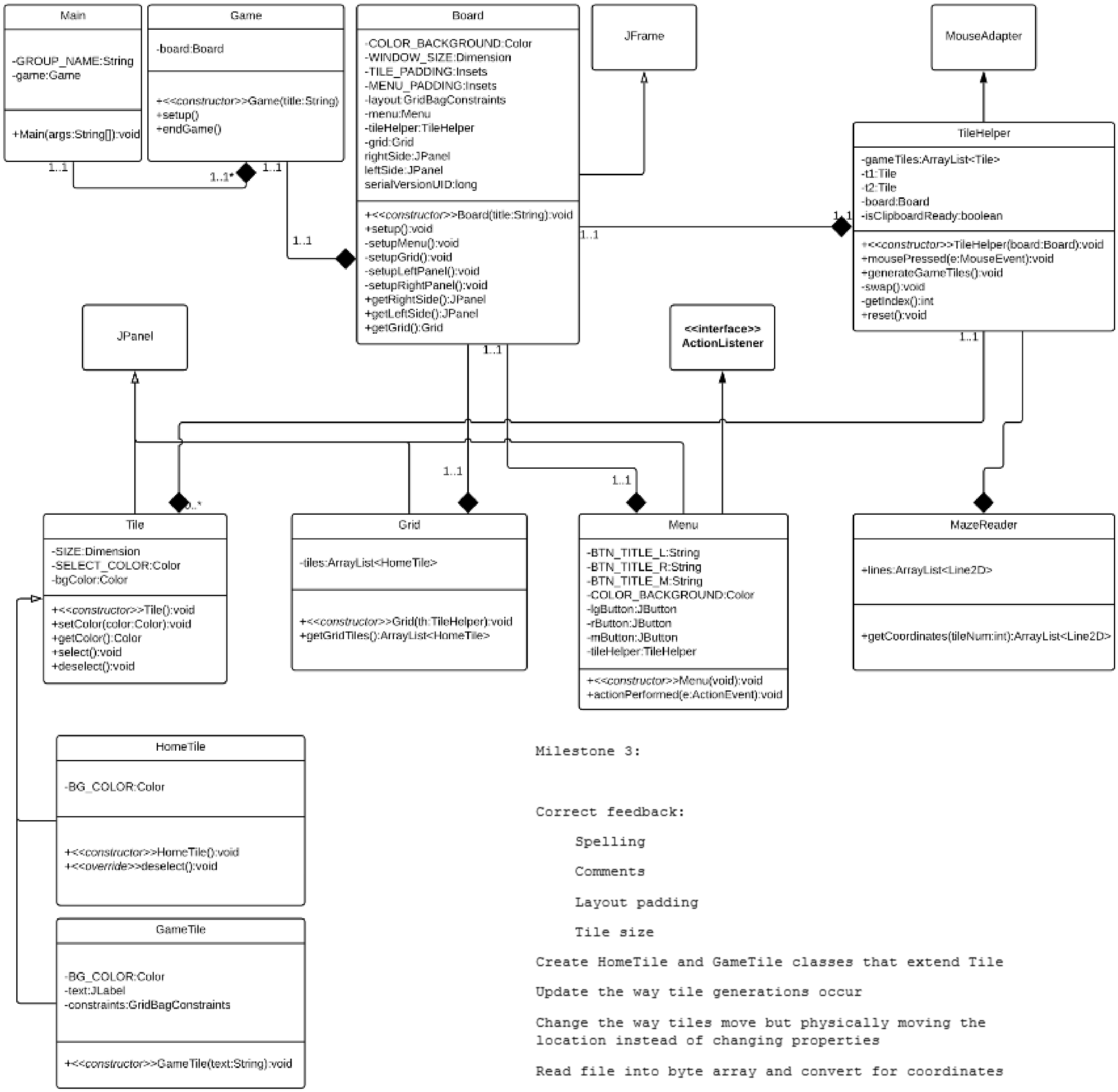
The diagram reference by the Milestone 1: document has been included bellow. This diagram was the cornerstone that we built the initial program to imitate as described by the client's request.



The second program milestone requested that we begin to move tiles and give the user the power to place the tiles on the center game board. Our group was given freedom on the mechanism of which to move tiles and we decided on a simple 'click and place' method. Aside from some minor spelling and layout suggestions the client was well pleased by our progress. The changes to the UML diagram for milestone two are included bellow. Classes implemented an action listener to respond to the user's mouse clicks and various classes were further specified now that our group had a greater vision of where this project would take us.

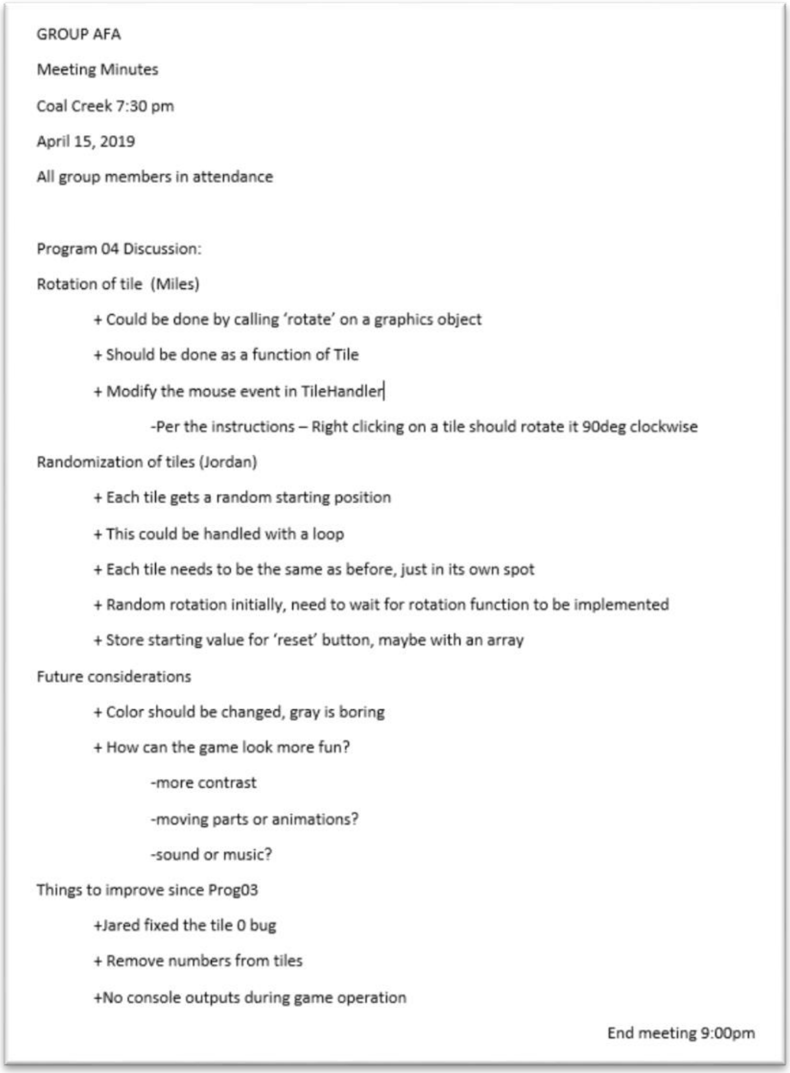


For the third milestone, our group was tasked with generating maze lines on our tiles based on a file provided by the client. This led to the one of the largest additions to the UML diagram. A TileHelper class was added to assist in the tile movement and generation of maze lines. The Board class was also heavily expanded to accommodate for the new initialization system for the board. Our planning document has been included with the UML to show the progress and notable changes. A bug also appeared due to the new tile movement that was documented in our planning. This was quickly solved by a member at the beginning of milestone four.



```
Milestone 3:


Correct feedback:

    Spelling

    Comments

    Layout padding

    Tile size

Create HomeTile and GameTile classes that extend Tile

Update the way tile generations occur

Change the way tiles move but physically moving the
location instead of changing properties

Read file into byte array and convert for coordinates

Draw lines to tiles

Future Work:

    Clean the file reading code!

    Fix the weird bug when moving tile 0 to grid position
0

     Consider where to put the maze check
```

Milestone 4 required a randomization of tile placement and initial rotation for the tiles. The user would also need to have the ability to rotate the tiles for themselves. An exerpt of meeting minutes has been included bellow alongside the planning document to illustrate the conversations had for each in-person meeting. A greater emphasis was put on future work and considerations in order to better meet requests or thoughts that the client may have. Meetings such as the one detailed bellow occurred twice per milestone and usualy included all group members. Minutes were distributed to each member and also kept along with the planning and other documentation instances. The UML for milestone four was mostly unchanged asside from varrious tile controls such as the rotate function.



GROUP AFA

Meeting Minutes

Coal Creek 7:30 pm

April 15, 2019

All group members in attendance

Program 04 Discussion:

Rotation of tile (Miles)

- + Could be done by calling 'rotate' on a graphics object
- + Should be done as a function of Tile
- + Modify the mouse event in TileHandler
    - -Per the instructions – Right clicking on a tile should rotate it 90deg clockwise

Randomization of tiles (Jordan)

- + Each tile gets a random starting position
- + This could be handled with a loop
- + Each tile needs to be the same as before, just in its own spot
- + Random rotation initially, need to wait for rotation function to be implemented
- + Store starting value for 'reset' button, maybe with an array

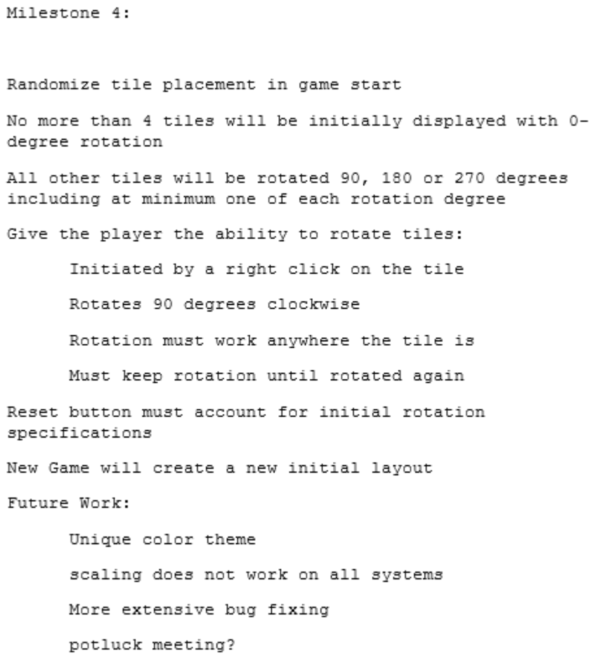
Future considerations

- + Color should be changed, gray is boring
- + How can the game look more fun?
    - -more contrast
    - -moving parts or animations?
    - -sound or music?

Things to improve since Prog03

- +Jared fixed the tile 0 bug
- + Remove numbers from tiles
- +No console outputs during game operation

End meeting 9:00pm



```
Milestone 4:


Randomize tile placement in game start

No more than 4 tiles will be initially displayed with 0-
degree rotation

All other tiles will be rotated 90, 180 or 270 degrees
including at minimum one of each rotation degree

Give the player the ability to rotate tiles:

      Initiated by a right click on the tile

      Rotates 90 degrees clockwise

      Rotation must work anywhere the tile is

      Must keep rotation until rotated again

Reset button must account for initial rotation
specifications

New Game will create a new initial layout

Future Work:

      Unique color theme

      scaling does not work on all systems

      More extensive bug fixing

      potluck meeting?
```

Milestone five led our group to address several bugs and issues that our client had with our last deliverable. These were a minor changes but they still took a priority in the consideration of our current goals. Bellow, the planning document for milestone five shows the steps we took to reach the next goal in our project.

Milestone 5:

Begin by fixing previous errors:

1.) Tile color should flash on invalid move, not background color
2.) Spacing between tiles sets on the left and right sides needs to be increased
3.) Should be able to deselect tiles by clicking on them and rotate tiles without selecting them

The New Game button should be removed. This will be replaced with a File button. Clicking File will provide the user with a dropdown menu. The dropdown menu will contain two options: Load and Save. Clicking either button will bring up a file explorer so that you can choose a file.

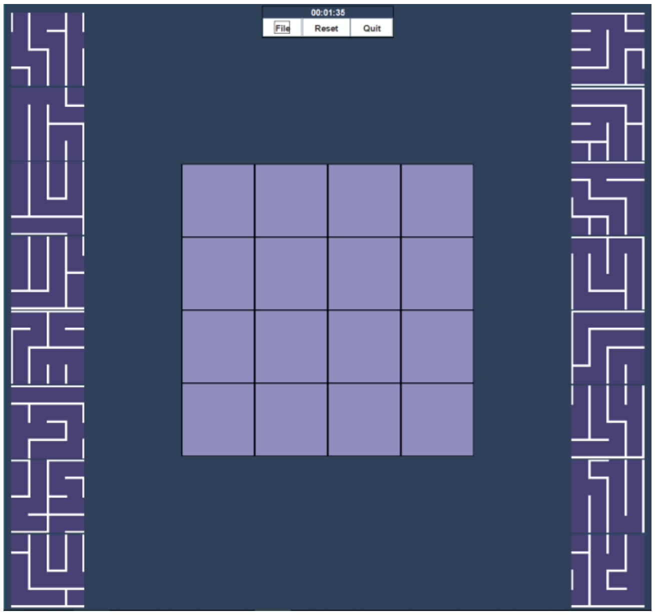
When Load is clicked, choosing a file will load the file. The program will or won't randomize the tiles depending upon the initial four bytes. The user should be asked if they want to save the current game they have open.

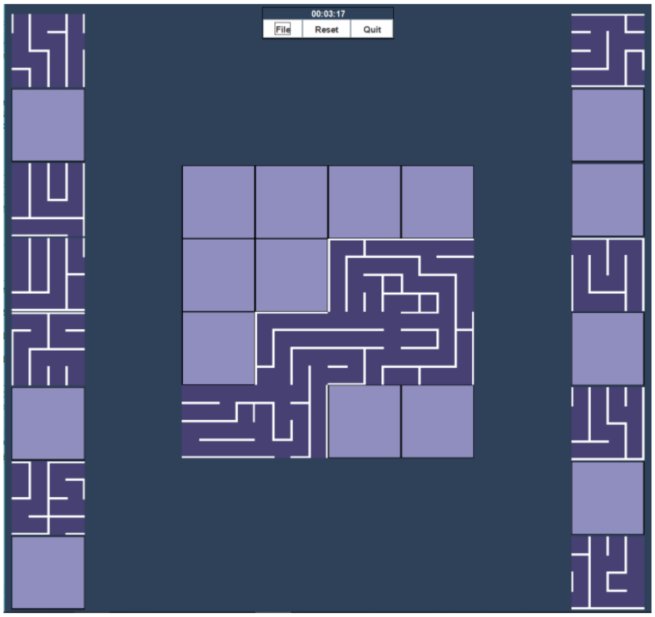
When Save is clicked, choosing a file will save the current game to that file. The file can be any name, extension, and location. The file will contain the appropriate four bytes (to signify it was played) and all of the current tile data as-is.

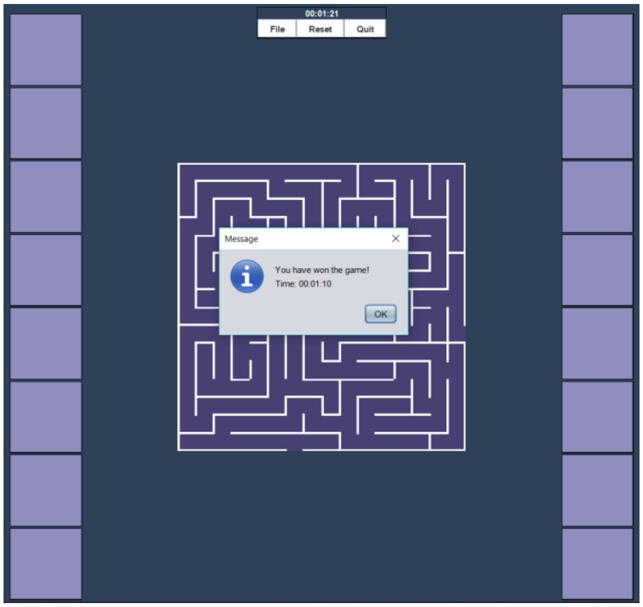
Future work:

- Redo the ReadMaze function
- Cleanup how tiles are created and managed
- Make the entirety of the code more elegant
- Fix some minor visual issues that we're not entirely happy with

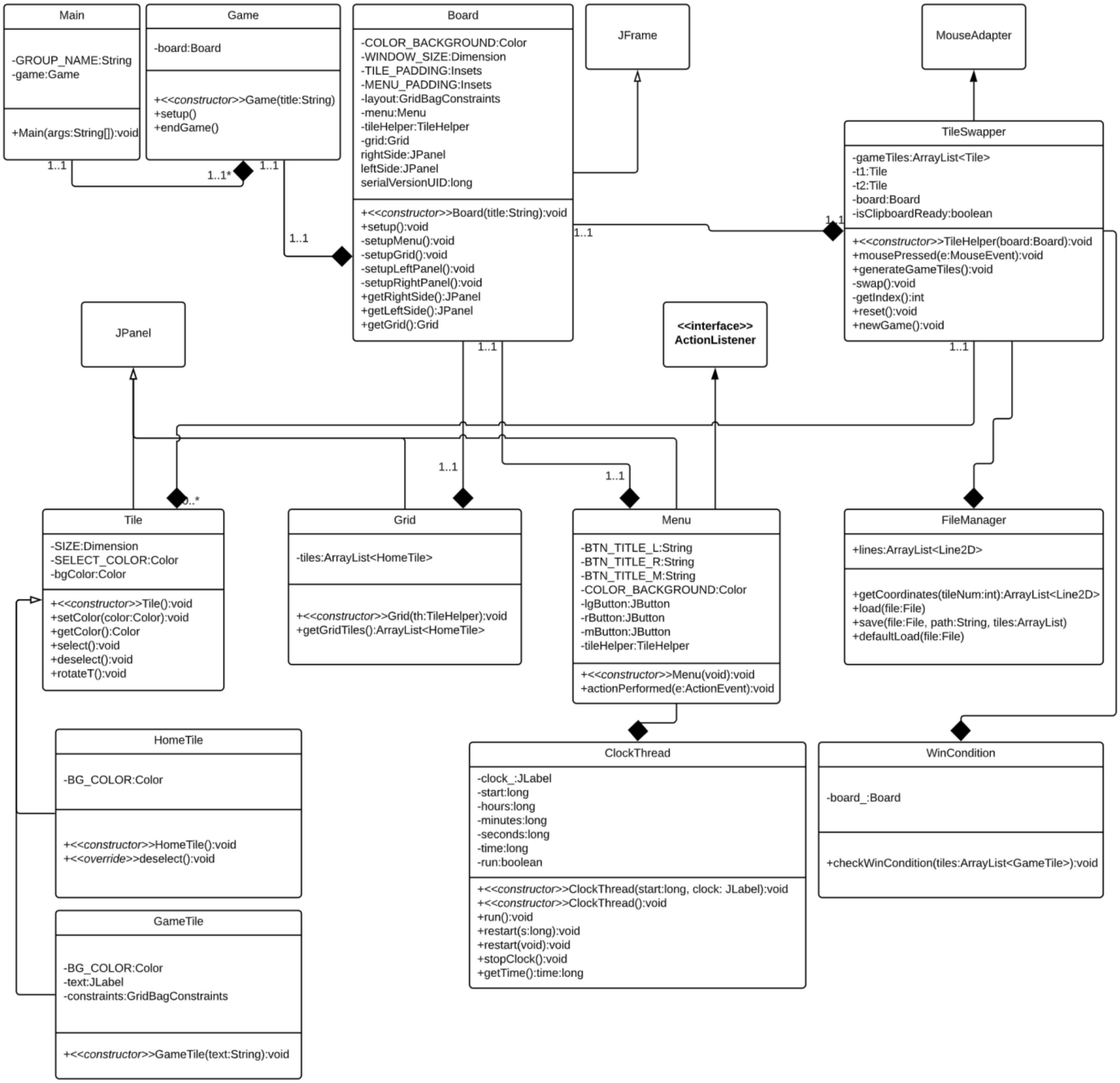
Milestone 6 was the culmination of all of the work we did. The final requests from the client involved a game clock to keep track of the total time spent on solving the maze game as well as a win condition to check when the board was complete. Many options were considered including a solution file and maze solving algorithms but in the end we consulted the client. Though this is likely the end to this project, our group still thought of some future considerations such as the ability for the user to choose a color theme and leaderboards for fastest maze completions. Some images of the final project are included bellow as well as the to-date UML diagram for the current iteration.







You have won the game!
Time: 00:01:10


Main
-GROUP_NAME:String
-game:Game
+Main(args:String[]):void
Game
-board:Board
+<<constructor>>Game(title:String)
+setup()
+endGame()
Board
-COLOR_BACKGROUND:Color
-WINDOW_SIZE:Dimension
-TILE_PADDING:Insets
-MENU_PADDING:Insets
-layout:GridBagConstraints
-menu:Menu
-tileHelper:TileHelper
-grid:Grid
rightSide:JPanel
leftSide:JPanel
serialVersionUID:long
+<<constructor>>Board(title:String):void
+setup():void
-setupMenu():void
-setupGrid():void
-setupLeftPanel():void
-setupRightPanel():void
+getRightSide():JPanel
+getLeftSide():JPanel
+getGrid():Grid
JFrame
MouseAdapter
TileSwapper
-gameTiles:ArrayList<Tile>
-t1:Tile
-t2:Tile
-board:Board
-isClipboardReady:boolean
+<<constructor>>TileHelper(board:Board):void
+mousePressed(e:MouseEvent):void
+generateGameTiles():void
-swap():void
-getIndex():int
+reset():void
+newGame():void
1..1
1..1*
1..1
1..1
1..1
1..1
1..1
1..1
JPanel
<<interface>>
ActionListener
1..1
1..1
0..*
Tile
-SIZE:Dimension
-SELECT_COLOR:Color
-bgColor:Color
+<<constructor>>Tile():void
+setColor(color:Color):void
+getColor():Color
+select():void
+deselect():void
+rotateT():void
Grid
-tiles:ArrayList<HomeTile>
+<<constructor>>Grid(th:TileHelper):void
+getGridTiles():ArrayList<HomeTile>
Menu
-BTN_TITLE_L:String
-BTN_TITLE_R:String
-BTN_TITLE_M:String
-COLOR_BACKGROUND:Color
-lgButton:JButton
-rButton:JButton
-mButton:JButton
-tileHelper:TileHelper
+<<constructor>>Menu(void):void
+actionPerformed(e:ActionEvent):void
FileManager
+lines:ArrayList<Line2D>
+getCoordinates(tileNum:int):ArrayList<Line2D>
+load(file:File)
+save(file:File, path:String, tiles:ArrayList)
+defaultLoad(file:File)
HomeTile
-BG_COLOR:Color
+<<constructor>>HomeTile():void
+<<override>>deselect():void
GameTile
-BG_COLOR:Color
-text:JLabel
-constraints:GridBagConstraints
+<<constructor>>GameTile(text:String):void
ClockThread
-clock_:JLabel
-start:long
-hours:long
-minutes:long
-seconds:long
-time:long
-run:boolean
+<<constructor>>ClockThread(start:long, clock: JLabel):void
+<<constructor>>ClockThread():void
+run():void
+restart(s:long):void
+restart(void):void
+stopClock():void
+getTime():time:long
WinCondition
-board_:Board
+checkWinCondition(tiles:ArrayList<GameTile>):void