



Thoraya Al-Sabti (u6136358)
Samuel Brookes (u5380100)
Baohong Tan (u6126217)



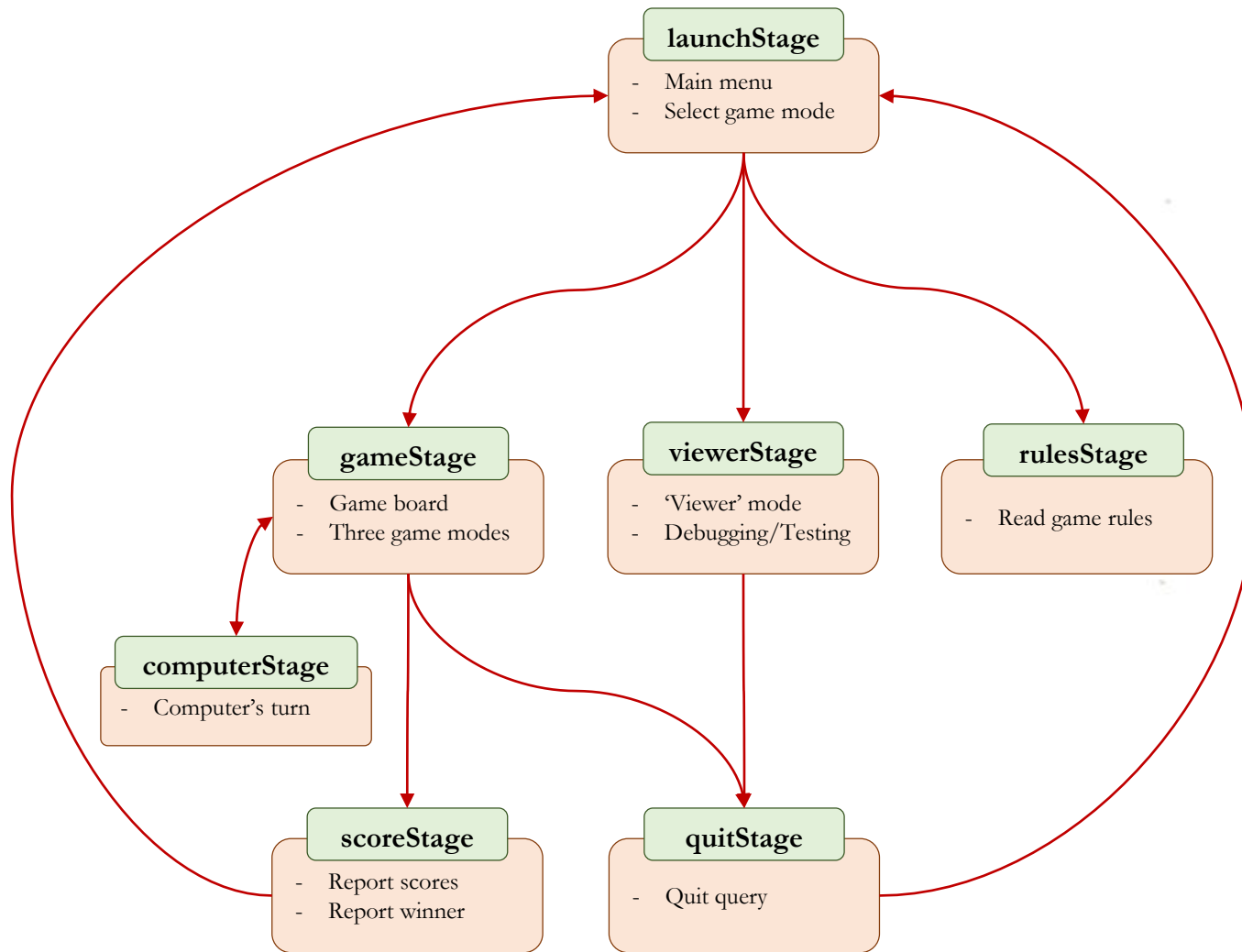
Game Design

UI Design

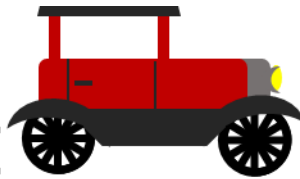
Code Design

Method Example

Demonstration



Game Design



Flow of game

Relationship between Stages

Function of Stages

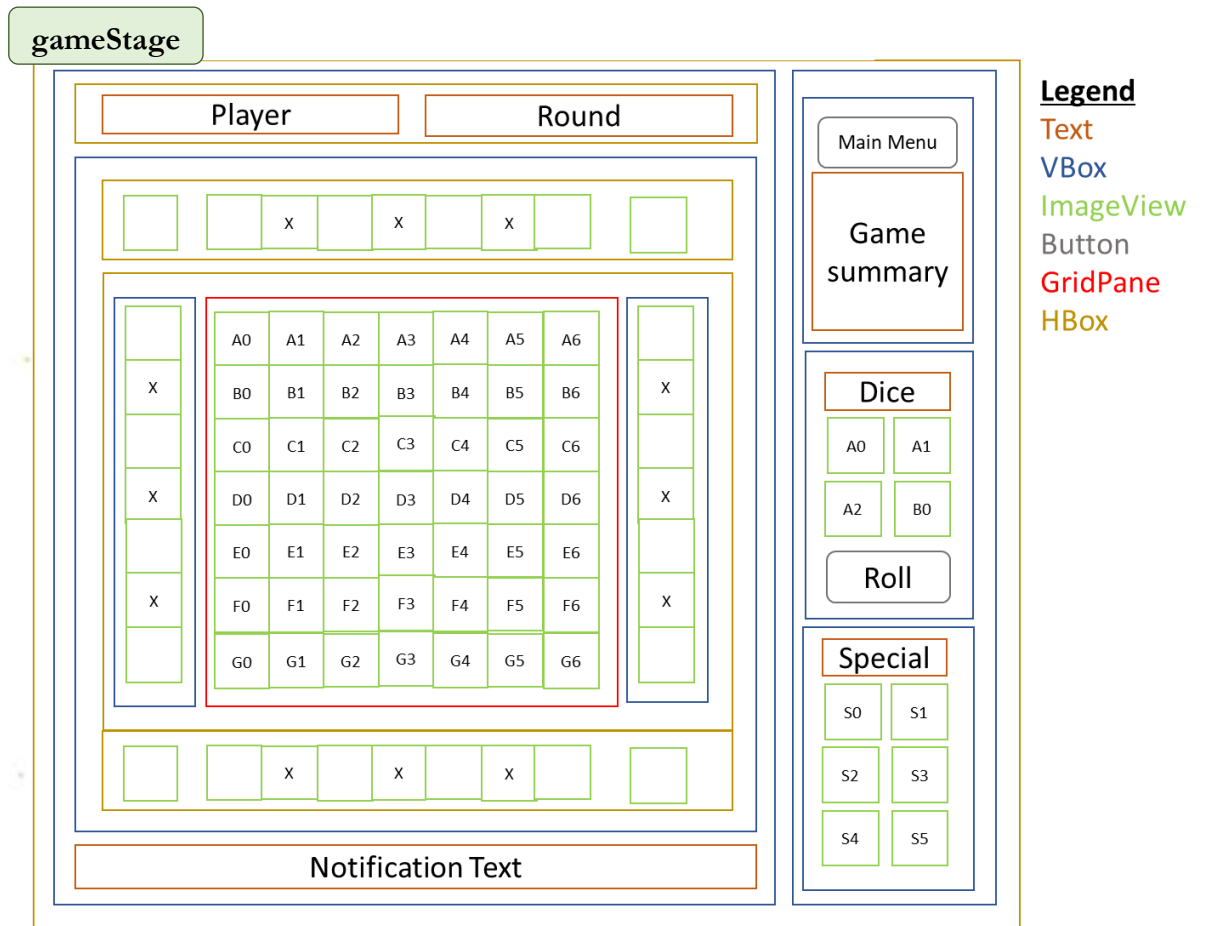
UI Design



Break into components

Find suitable JavaFX classes

Plan hierarchy and layout



UI Design



Final designs

launchStage



- ☒ Single Player
- ☐ Multi-player
- ☐ Computer Opponent
- ☐ Use Viewer

Play

Show Rules

rulesStage



THE AIM of Railroad Ink is to create networks of railroads and highways that connect as many exits as possible.

EXITS can be seen on the green outer edge of the game board.

A TILE can only be placed on the board if there are no illegal connections to other tiles on the board or to an exit.

AN ILLEGAL CONNECTION occurs when two edges (or an edge and an exit) meet at different route types, for example, when an edge with a railroad meets an edge with a highway (or vice versa).

YOU GET EXTRA POINTS for placing tiles on the red centre squares and for your longest highway and railroad.

YOU LOSE POINTS by having unconnected edges on the board at the end of the game.

ONLY ONE SPECIAL TILE can be placed each round, and no more than 3 Special tiles can be placed in a game.

TO ROTATE A TILE, click on it with the right mouse button.

TO PLACE A TILE on the board, click on it with the left mouse button to select it, and then click on the position on the board where you would like it to go.

BEWARE! Once you place a tile, you are unable to remove it, so think carefully before making your move.

Close

quitStage

Are you sure you want to quit?

Yes

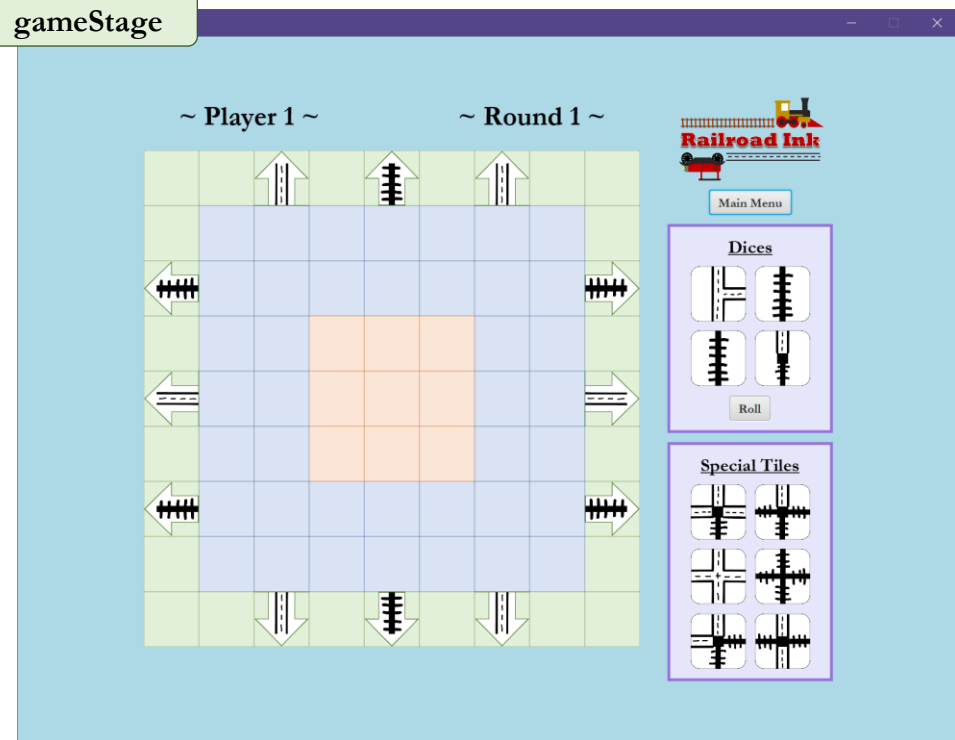
No

UI Design

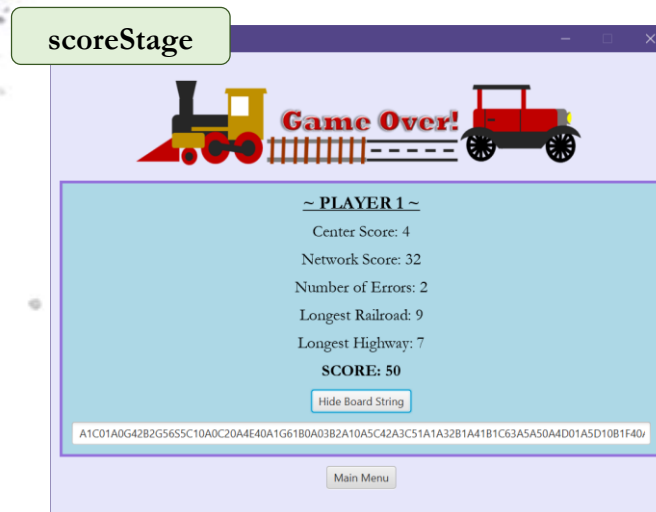


Final designs

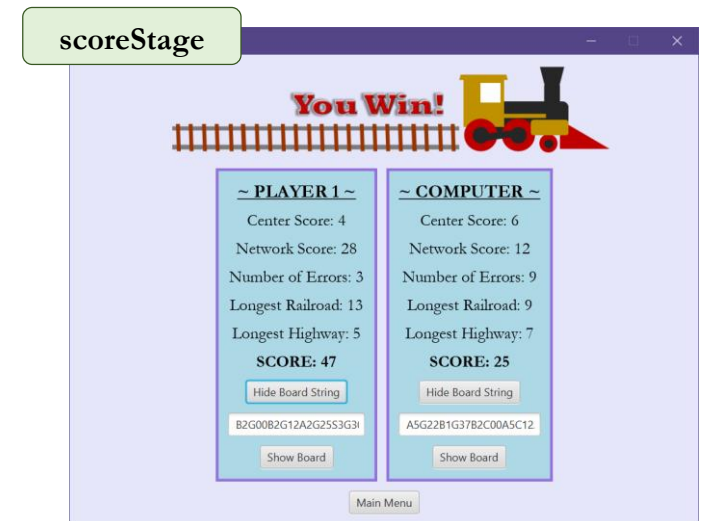
gameStage



scoreStage

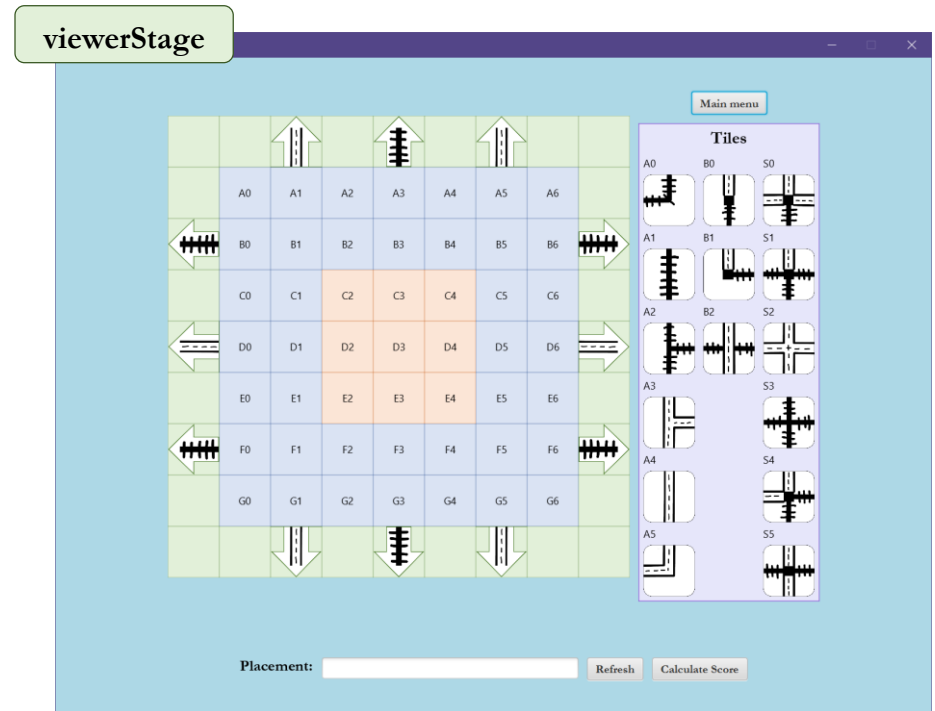
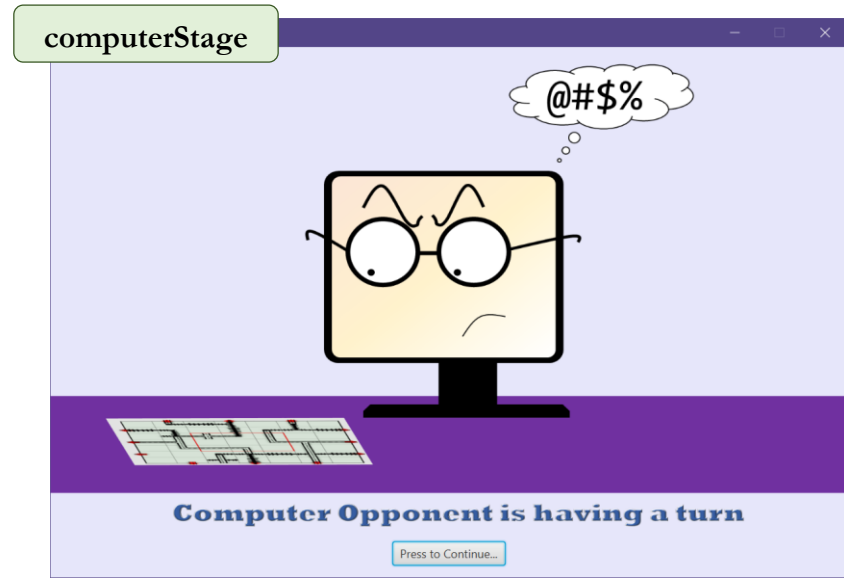


scoreStage





Final designs



UI Classes

Viewer

- Main UI class
- Builds and launches all Stages

ImageHandler

- Image retrieval and manipulation

SpecialTiles

- State and behavior of the special tiles sub-UI

Dices

- State and behavior of dices sub-UI

Primary Game Classes

Board

- Checks placement validity
- Stores all placements

ComputerOpponent

- State and behavior of computer opponent.

ScoreCalculator

- All score calculation methods

Tile

- State and behavior of tiles

Utility Classes

Iterator

- Enable iteration around the edges of a tile

Placement

- Enable easy access to components of a placement string

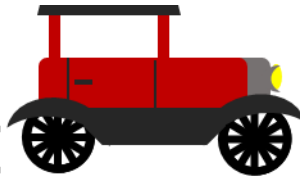
RailroadInk

- Utility methods

PlayerData

- Store game data for each player

Code Design

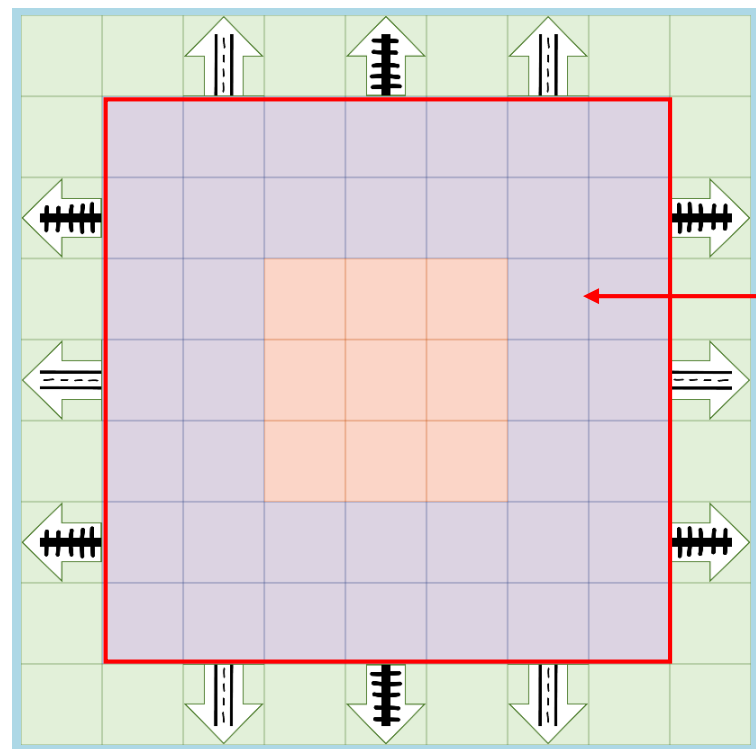


Classes

Code Example



Tile Placement



Viewer

.setUpDropTarget()



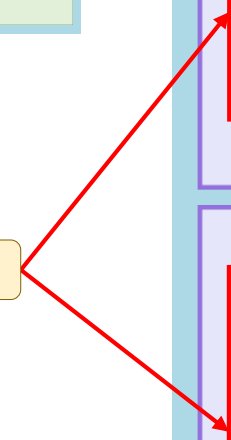
Viewer

.setUpSelectAndRotate()

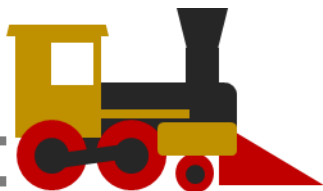
Dices

Roll

Special Tiles

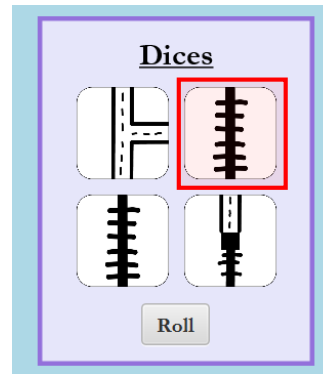


Code Example



Tile Placement

Player clicks on
tile with
left mouse
button

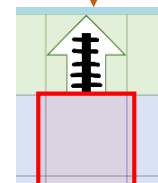


Viewer

New Placement object created

new Placement

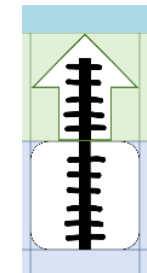
Player clicks on
board
with left mouse
button



Viewer

Placement object updated with
coordinates

Placement.updateCoordinates()



Illegal placement!

Board

If placement is
legal .addTile()
returns **true**

Board

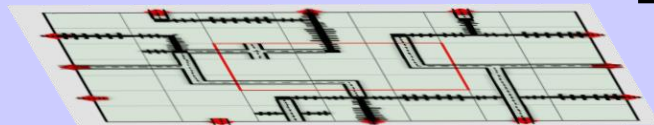
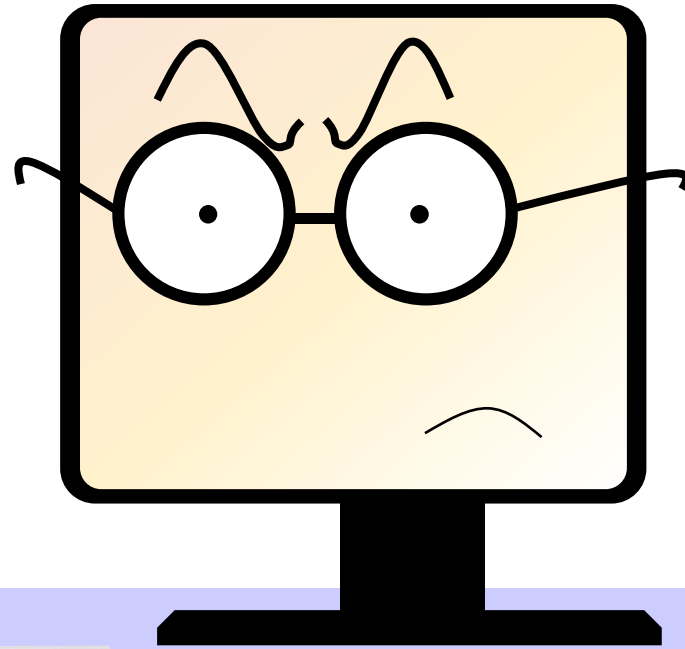
If placement is
illegal .addTile()
returns **false**

PlayerData

Board.addTile()

Placement is passed to that
player's Board object

Please shut up and just
show me...



Demonstration

