Focus Game

Benjamin Samuel, Joanne Louie,
Mithun Comar

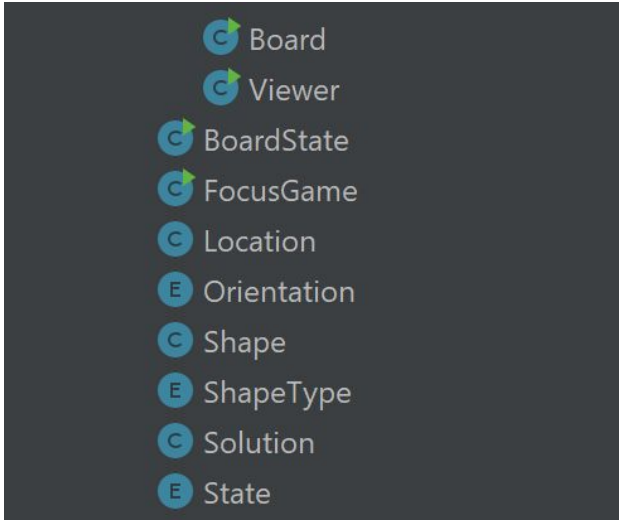
The Challenge

Our challenge was to implement a fully functioning game of IQ Focus.

We unsurprisingly faced difficult problems along the way.

Designing the Classes

- BoardState
  - The building block of our tasks
  - Functions inside update the board with “States” (colours)
  - The functions also check for overlap, and if pieces are on the board
- ShapeType
  - Uses a similar function to “stateFromOffset” in Assignment 1, except it’s implemented to work with this game
- The others
  - Includes our objects (the shapes), and basic methods to figure out small things like orientation



Data Structures

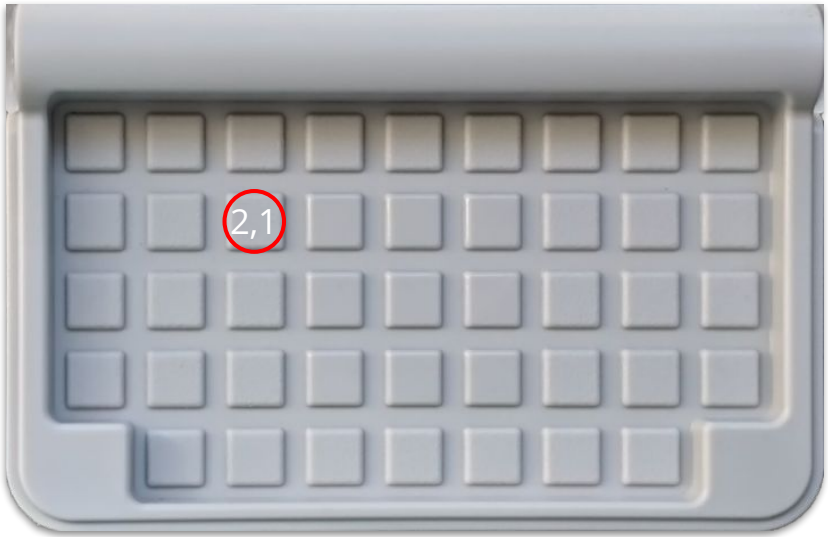
- 2d array to represent the board
  - A 5x9 array, where all indexes are “null” initially

Issues Encountered

- Task 6
  - Originally, the function used to generate viable placements was too slow to pass the given tests
  - The code required optimisation to avoid timing out
- JavaFX
  - JavaFX was significantly more challenging to use and more time consuming compared to earlier tasks
  - Task 7 and 8 were completed by breaking apart the required functions into multiple inner classes and methods
- Task 9
  - As with task 6, the function written for task 9 would timeout when the tests were run
  - Ensuring that the code produced the correct solution set also became an issue

Beating the Problem

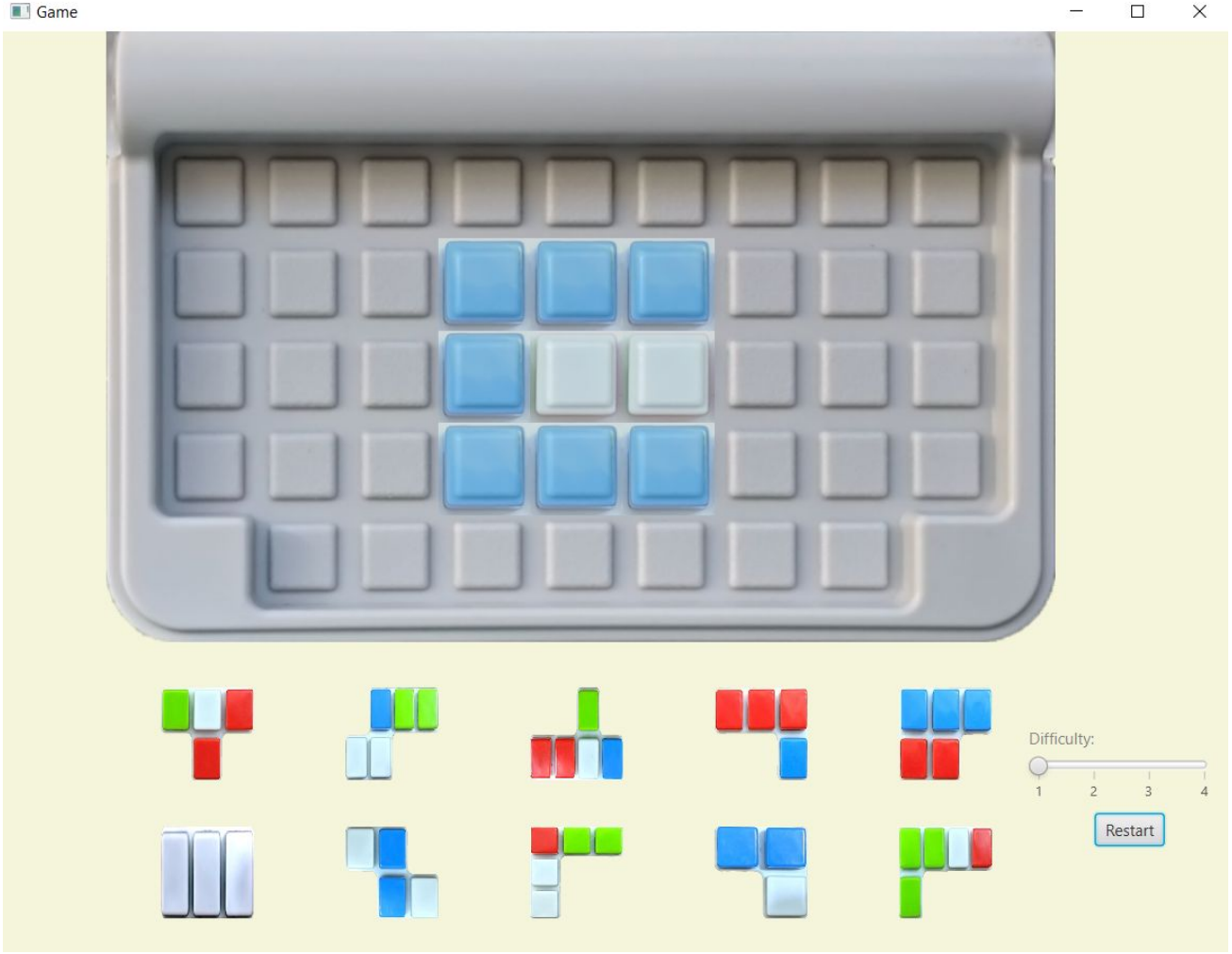
- *Checking a placement covers the given cell:*
  - Step 1: update the board state with the possible piece placement
  - Step 2: simply check if the index of the board which corresponds to the cell isn't null (isn't not covered):
    e.g board[1][2] != null;
- *Checking consistency*
  - Step 1: update the board state
  - Step 2: check if it's covering any of the challenge cells. If so, then check consistency



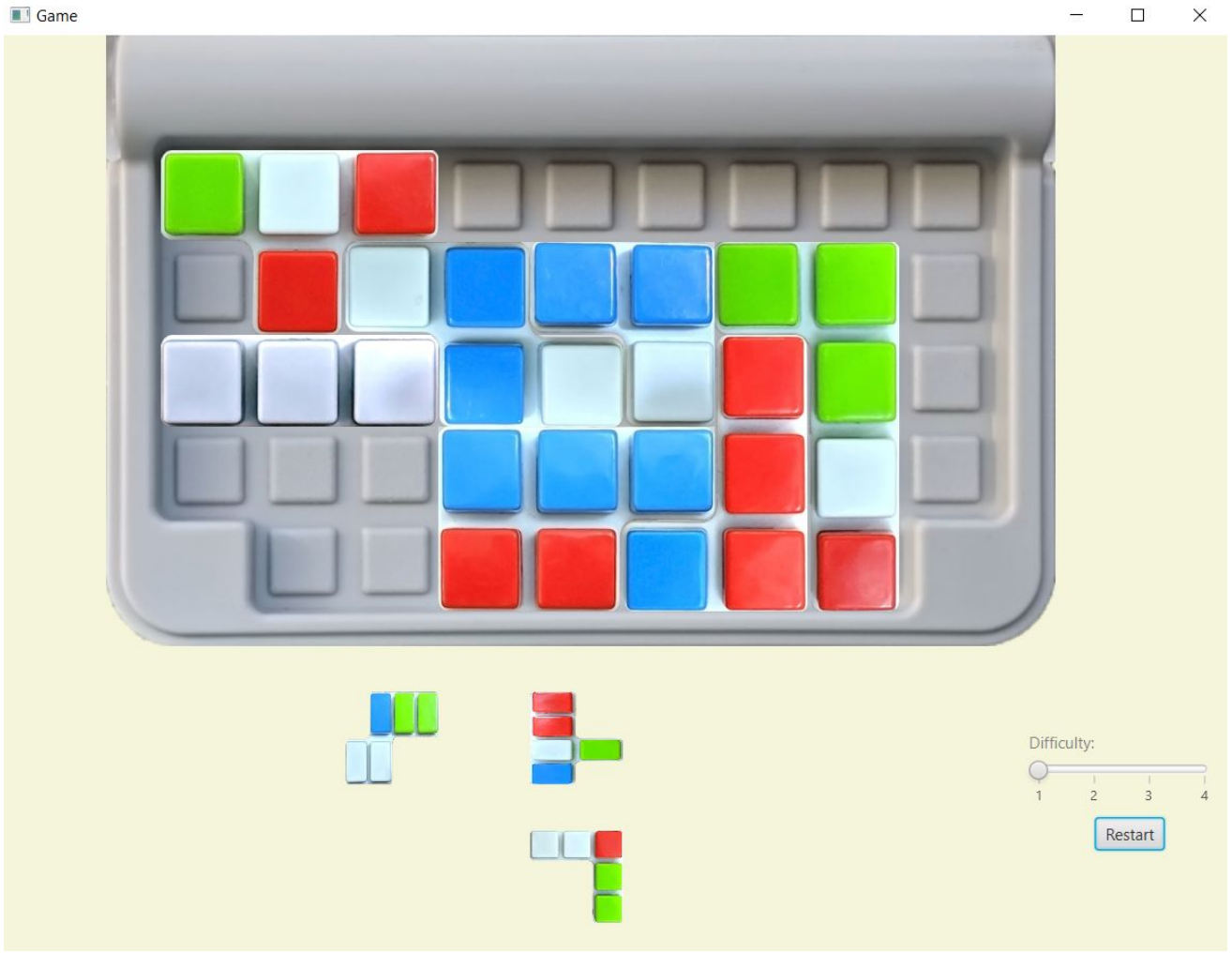
Difficulty in JavaFX

Writing code for JavaFX came with many smaller challenges

- *Rotating a placement piece in the game window would cause the piece to shift away from its original position*
  - 4 copies of each shape were made, one for each orientation and added to the assets folder. When a shape was rotated, instead of rotating the image itself, a new image was generated corresponding to the desired orientation.
- *Adjusting the scale of the game window and the shapes*
  - All measurements and coordinates were derived from the fixed width and height of the board and the squares inside the board. Relative distances between for the starting pieces were found the same way.
- *Creating the working game*
  - The code was broken down into multiple inner classes and methods and the task was completed through solving smaller problems with the code



The Game on Startup



Pieces Placed Randomly on the Board