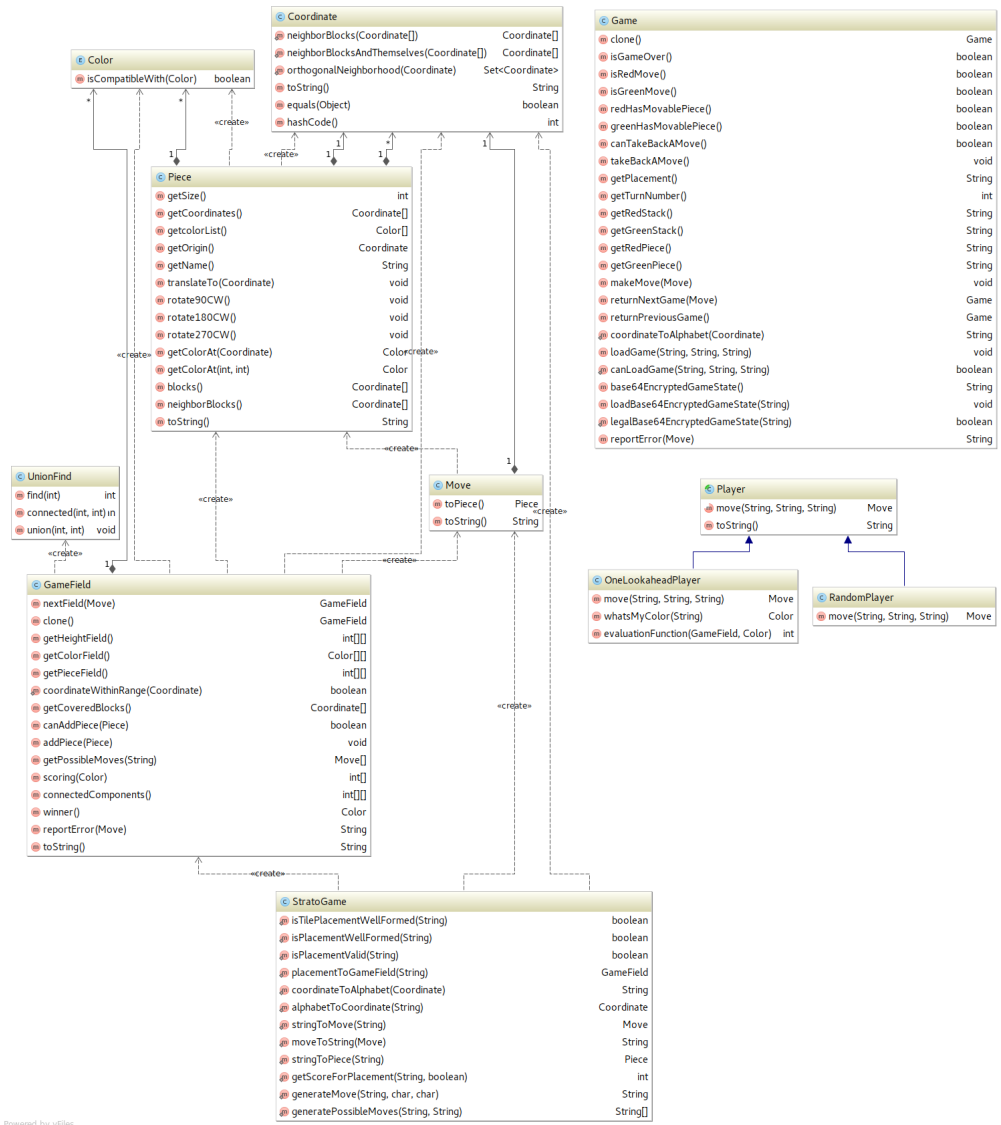
A Java Implementation of Stratopolis

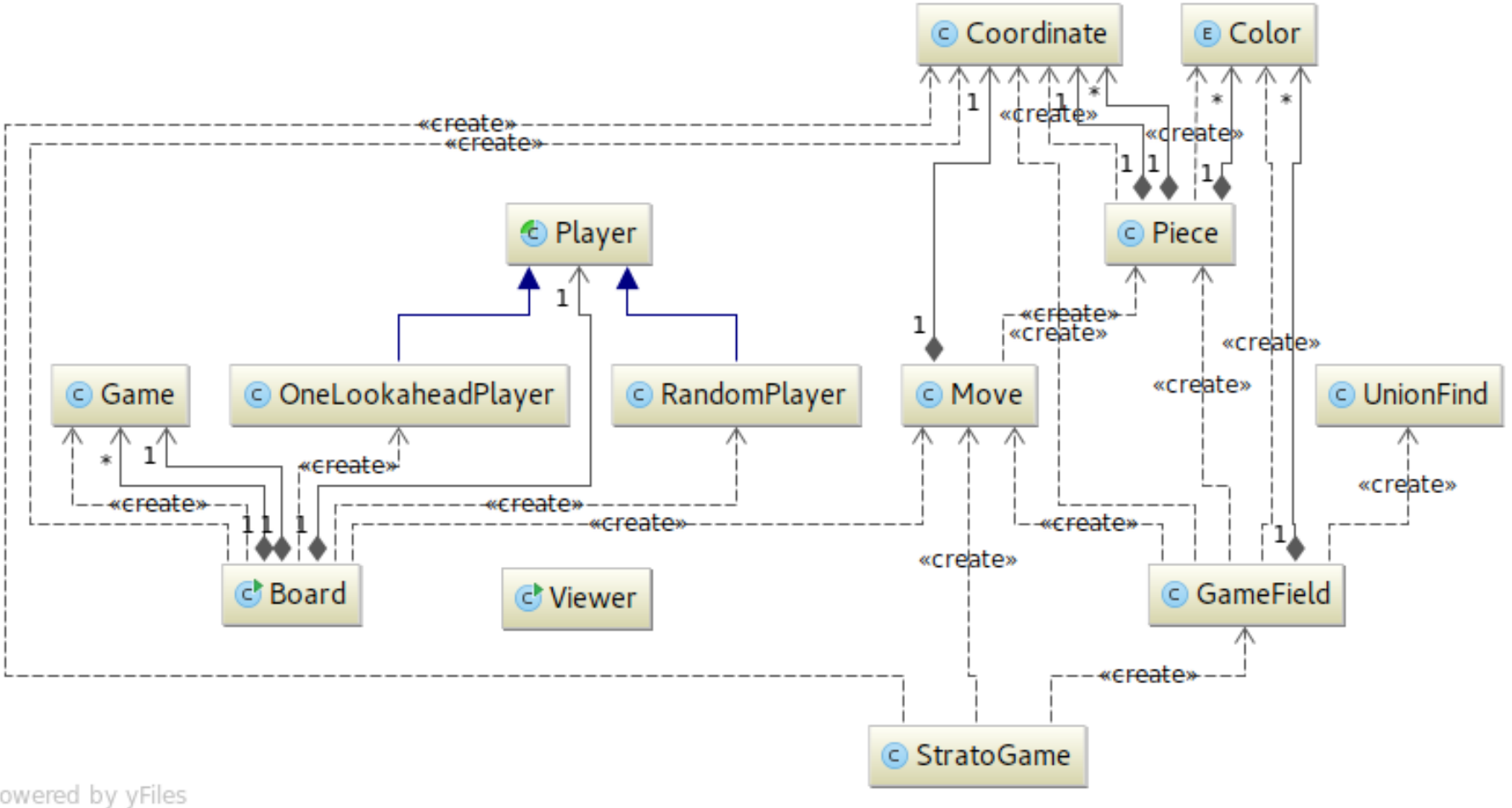
Yuxi Liu, Xinyi Qian, Woojin Ra

10 minutes, 19 October, 2016

UML (1 minute)


Coordinate
neighborBlocks(Coordinate[]) Coordinate[]
neighborBlocksAndThemselves(Coordinate[]) Coordinate[]
orthogonalNeighborhood(Coordinate) Set<Coordinate>
toString() String
equals(Object) boolean
hashCode() int
Color
isCompatibleWith(Color) boolean
Game
clone() Game
isGameOver() boolean
isRedMove() boolean
isGreenMove() boolean
redHasMovablePiece() boolean
greenHasMovablePiece() boolean
canTakeBackAMove() boolean
takeBackAMove() void
getPlacement() String
getTurnNumber() int
getRedStack() String
getGreenStack() String
getRedPiece() String
getGreenPiece() String
makeMove(Move) void
returnNextGame(Move) Game
returnPreviousGame() Game
coordinateToAlphabet(Coordinate) String
loadGame(String, String, String) void
canLoadGame(String, String, String) boolean
base64EncryptedGameState() String
loadBase64EncryptedGameState(String) void
legalBase64EncryptedGameState(String) boolean
reportError(Move) String
Piece
getSize() int
getCoordinates() Coordinate[]
getcolorList() Color[]
getOrigin() Coordinate
getName() String
translateTo(Coordinate) void
rotate90CW() void
rotate180CW() void
rotate270CW() void
getColorAt(Coordinate) Color
getColorAt(int, int) Color
blocks() Coordinate[]
neighborBlocks() Coordinate[]
toString() String
UnionFind
find(int) int
connected(int, int) bo
union(int, int) void
Move
toPiece() Piece
toString() String
Player
move(String, String, String) Move
toString() String
OneLookaheadPlayer
move(String, String, String) Move
whatsMyColor(String) Color
evaluationFunction(GameField, Color) int
RandomPlayer
move(String, String, String) Move
GameField
nextField(Move) GameField
clone() GameField
getHeightField() int[][]
getColorField() Color[][]
getPieceField() int[][]
coordinateWithinRange(Coordinate) boolean
getCoveredBlocks() Coordinate[]
canAddPiece(Piece) boolean
addPiece(Piece) void
getPossibleMoves(String) Move[]
scoring(Color) int[]
connectedComponents() int[][]
winner() Color
reportError(Move) String
toString() String
StratoGame
isTilePlacementWellFormed(String) boolean
isPlacementWellFormed(String) boolean
isPlacementValid(String) boolean
placementToGameField(String) GameField
coordinateToAlphabet(Coordinate) String
alphabetToCoordinate(String) Coordinate
stringToMove(String) Move
moveToString(Move) String
stringToPiece(String) Piece
getScoreForPlacement(String, boolean) int
generateMove(String, char, char) String
generatePossibleMoves(String, String) String[]
«create»
Powered by yFiles

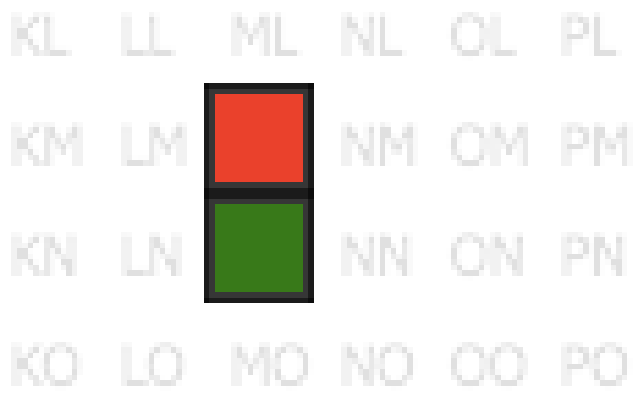
UML (1 minute)


Coordinate
Color
Player
Piece
Game
OneLookaheadPlayer
RandomPlayer
Move
UnionFind
Board
Viewer
GameField
StratoGame
«create»
1
*
Powered by yFiles

GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

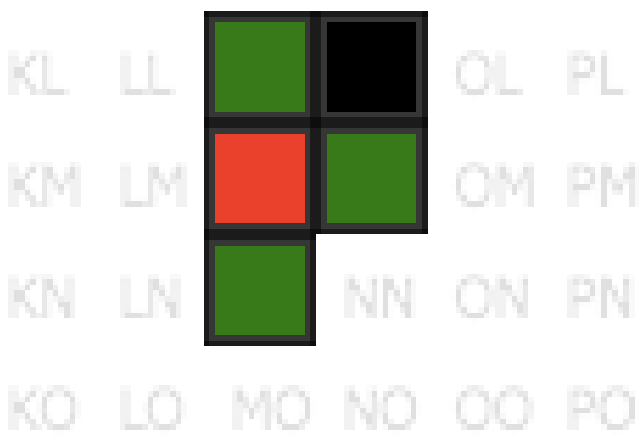
$$\begin{bmatrix} B & B & B & B & B & B \\ B & B & R & B & B & B \\ B & B & G & B & B & B \\ B & B & B & B & B & B \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

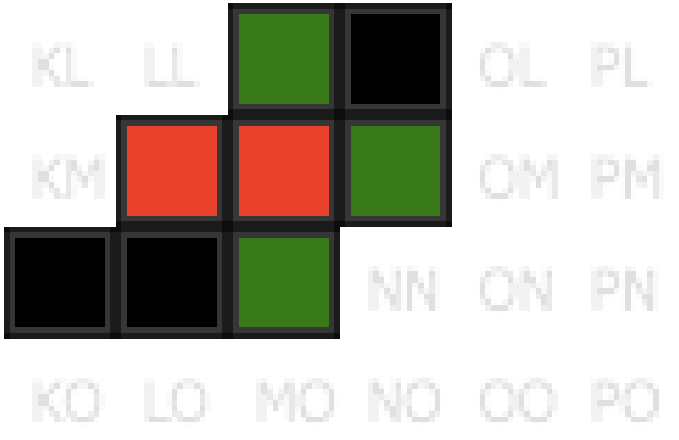
$$\begin{bmatrix} B & B & G & B & B & B \\ B & B & R & G & B & B \\ B & B & G & B & B & B \\ B & B & B & B & B & B \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 & 1 & -1 & -1 \\ -1 & -1 & 0 & 1 & -1 & -1 \\ -1 & -1 & 0 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

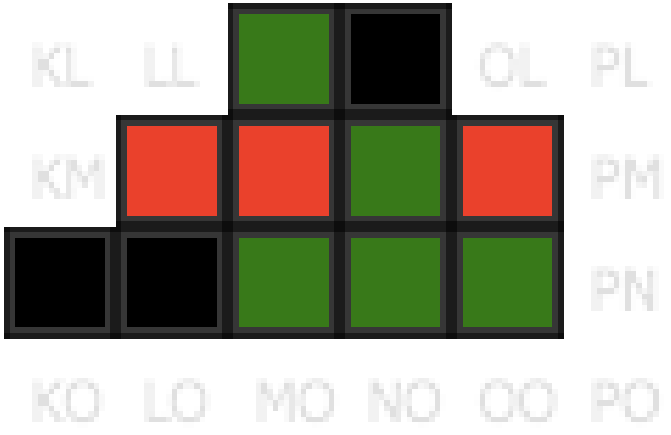
$$\begin{bmatrix} B & B & G & B & B & B \\ B & R & R & G & B & B \\ B & B & G & B & B & B \\ B & B & B & B & B & B \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 & 1 & -1 & -1 \\ -1 & 2 & 0 & 1 & -1 & -1 \\ 2 & 2 & 0 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

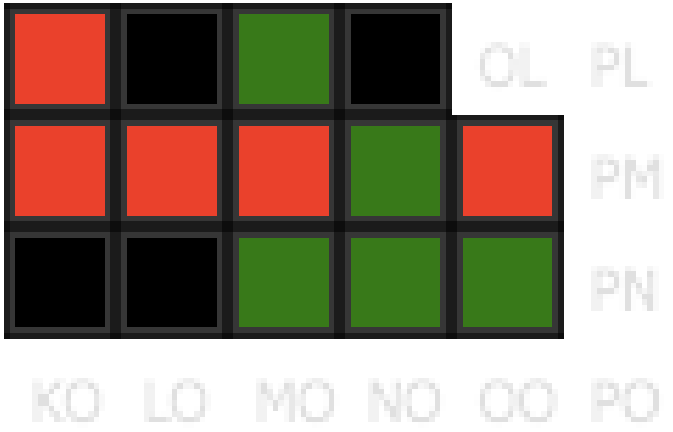
$$\begin{bmatrix} B & B & G & B & B & B \\ B & R & R & G & R & B \\ B & B & G & G & G & B \\ B & B & B & B & B & B \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 & 1 & -1 & -1 \\ -1 & 2 & 0 & 1 & 3 & -1 \\ 2 & 2 & 0 & 3 & 3 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

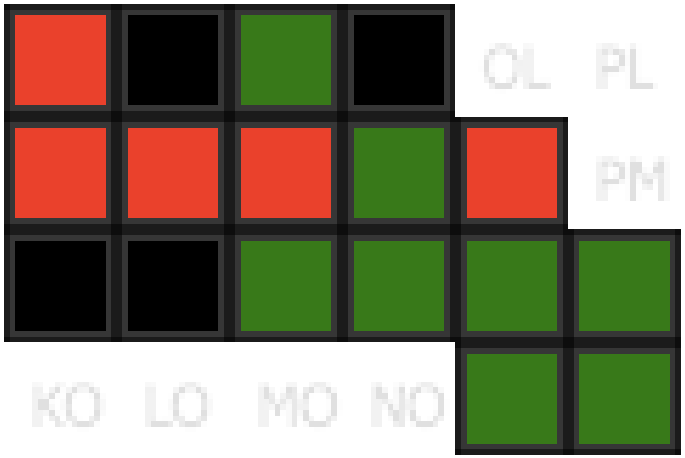
$$\begin{bmatrix} R & B & G & B & B & B \\ R & R & R & G & R & B \\ B & B & G & G & G & B \\ B & B & B & B & B & B \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 4 & 4 & 1 & 1 & -1 & -1 \\ 4 & 2 & 0 & 1 & 3 & -1 \\ 2 & 2 & 0 & 3 & 3 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

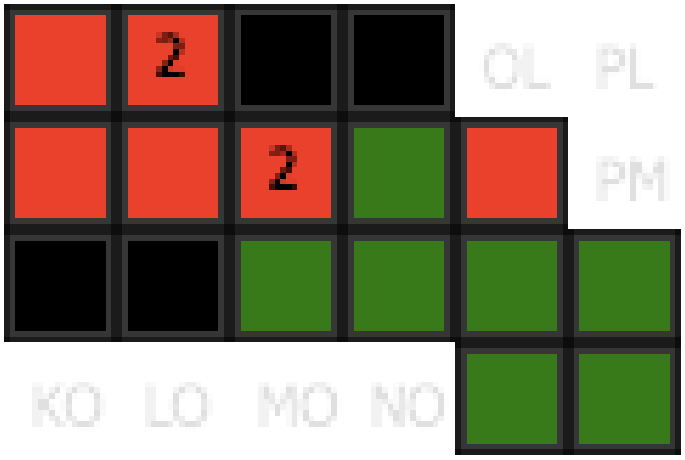
$$\begin{bmatrix} R & B & G & B & B & B \\ R & R & R & G & R & B \\ B & B & G & G & G & G \\ B & B & B & B & G & G \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 4 & 1 & 1 & -1 & -1 \\ 4 & 2 & 0 & 1 & 3 & -1 \\ 2 & 2 & 0 & 3 & 3 & 5 \\ -1 & -1 & -1 & -1 & 5 & 5 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

$$\begin{bmatrix} R & R & B & B & B & B \\ R & R & R & G & R & B \\ B & B & G & G & G & G \\ B & B & B & B & G & G \end{bmatrix} \begin{bmatrix} 1 & 2 & 2 & 1 & 0 & 0 \\ 1 & 1 & 2 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 6 & 6 & 1 & -1 & -1 \\ 4 & 2 & 6 & 1 & 3 & -1 \\ 2 & 2 & 0 & 3 & 3 & 5 \\ -1 & -1 & -1 & -1 & 5 & 5 \end{bmatrix}$$



GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

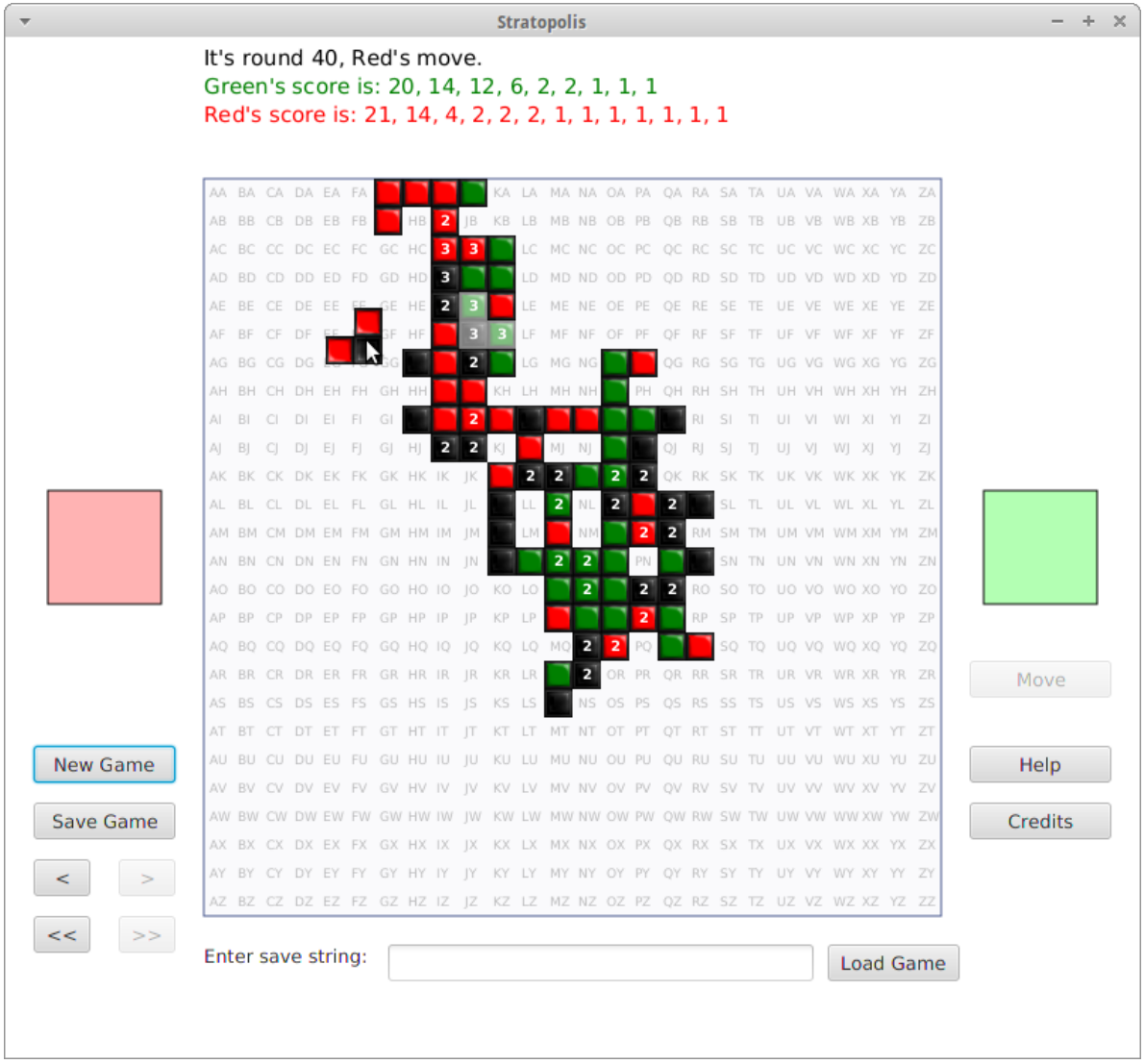
$$\begin{bmatrix} R & R & B & B & B & B \\ R & R & R & G & R & G \\ B & B & G & G & G & G \\ B & B & B & B & G & G \end{bmatrix} \begin{bmatrix} 1 & 2 & 2 & 1 & 1 & 1 \\ 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 6 & 6 & 1 & 7 & 7 \\ 4 & 2 & 6 & 1 & 3 & 7 \\ 2 & 2 & 0 & 3 & 3 & 5 \\ -1 & -1 & -1 & -1 & 5 & 5 \end{bmatrix}$$

GameField class (2 minute)

The GameField class represents the game board. It can decide whether a piece can be played on the board, and it can score the board.
A Gamefield has three matrices, representing color, height, and index.

$$\begin{bmatrix} R & R & B & B & B & B \\ R & R & R & G & R & G \\ R & R & G & G & G & G \\ B & B & B & B & G & G \end{bmatrix} \begin{bmatrix} 1 & 2 & 2 & 1 & 1 & 1 \\ 2 & 1 & 2 & 1 & 1 & 1 \\ 2 & 2 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 6 & 6 & 1 & 7 & 7 \\ 8 & 2 & 6 & 1 & 3 & 7 \\ 8 & 8 & 0 & 3 & 3 & 5 \\ -1 & -1 & -1 & -1 & 5 & 5 \end{bmatrix}$$

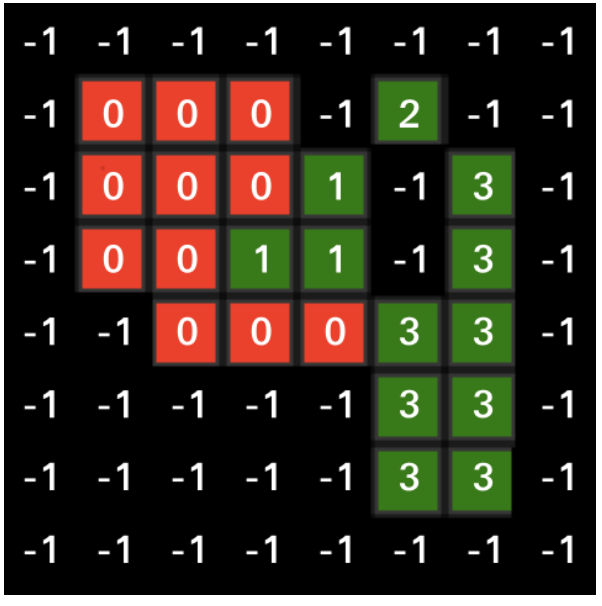
GUI screenshot (10 seconds)



Connected Component Labelling (1 minutes)

To score the board, we must solve the algorithmically interesting problem: how to find out the connected color regions of a board?

(a) Input: the color matrix



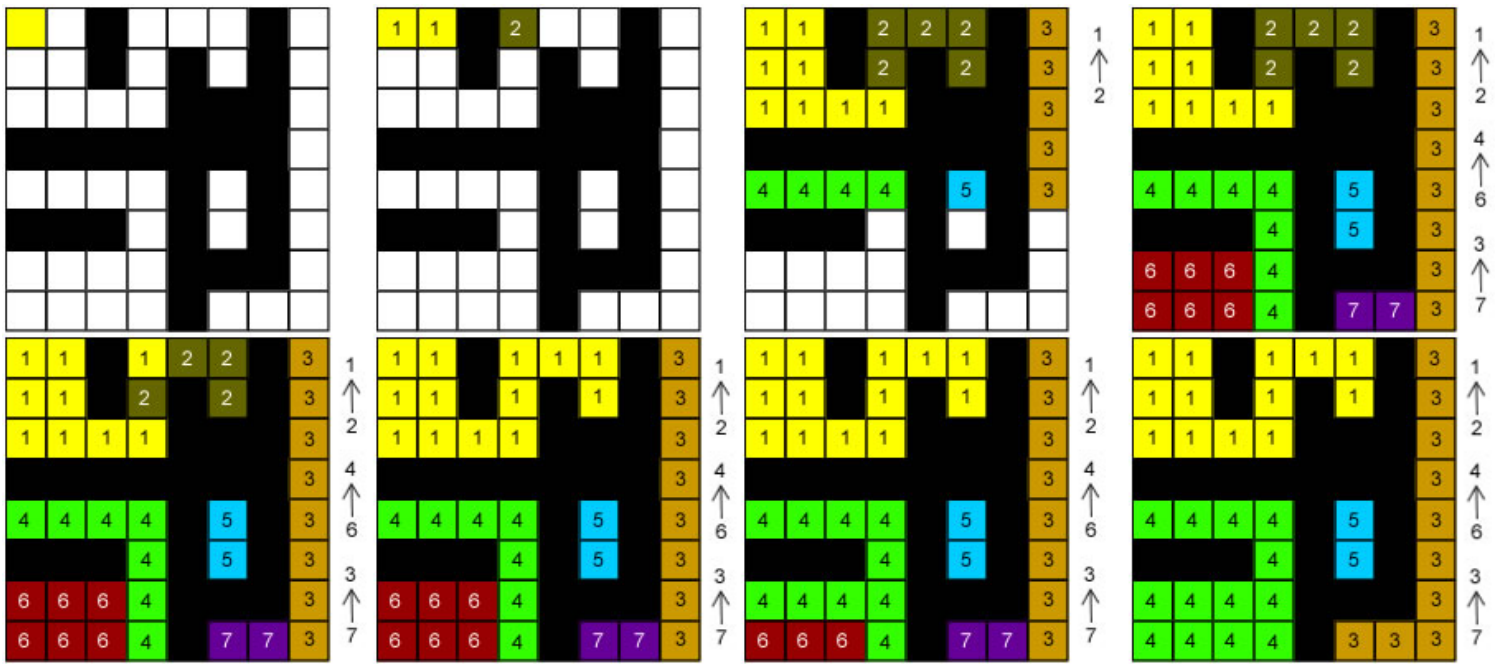
(b) Output: the component label matrix.

Connected Component Labelling. The background gets label -1.

2 Pass Union-find CCL Algorithm (20 seconds)

We used a two pass algorithm that uses an implementation of the set ADT, called "union-find".

Solves the problem in linear space and nearly in linear time.

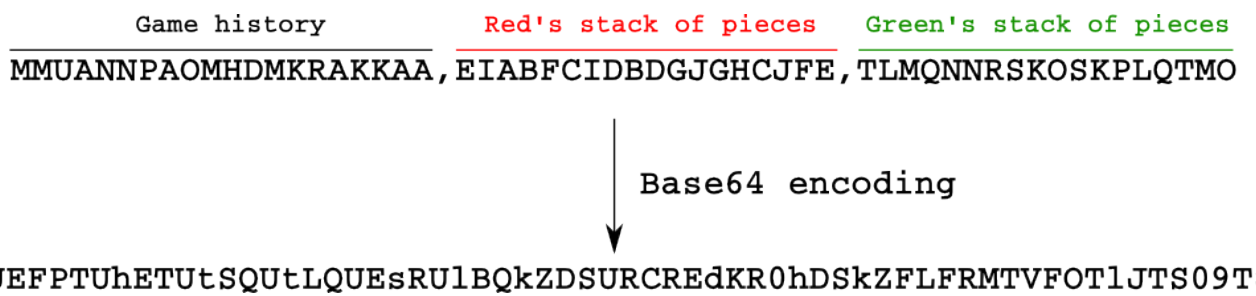


Pictures copied from http://aishack.in/tutorials/labelling-connected-components-example/

Search "union find connected components labelling" for details.

Base64 encoding (20 seconds)

We implemented game saving and loading by converting the game state into a single savestring, which can later be loaded to recreate the game state.



The savestring is encoded in Base64 to discourage cheating.

Demonstration (4 minutes)

Green is human, red is OneLookaheadPlayer.
Demonstrate error warning, move highlighting, forward and backward stepping, savestring, help screen, and credit screen.

Q&A (1 minute)

Any questions?