recreated in Java

By Timothy Bui

Objective

- Attempt to faithfully recreate 2048 in Java from scratch
- Have a functioning puzzle game just like the real game
- Minimize how many times I have to go to Stack Overflow for small details (lol)

Creating the board

1. Set layout
2. Add board panel to center panel
3. Initialize new JPanels in the 2d JPanel array(s)
4. Spawn 2 starting tiles using the refresh() method
5. Set score to 0
6. setFocusable and requestFocusInWindow required for the KeyListener to work

```
,
//create the game board and spawn two numbers in random locations
public void createBoard() {
    gameBoard.setLayout(new GridLayout(4, 4, 5, 5));
    gameBoardBorder.add(gameBoard, BorderLayout.CENTER);
    for (int y = 0; y < 4;  y++) {
        for(int x = 0; x < 4; x++) {
            backgroundPanels[y][x] = new JPanel();
            backgroundPanels[y][x].setLayout(new FlowLayout());
            backgroundPanels[y][x].setBackground(gray.brighter());
            blocks[y][x] = new JLabel(" ");
            blocks[y][x].setFont(new Font("Arial", Font.BOLD, 30));
            numbers[y][x] = 0;
            backgroundPanels[y][x].add(blocks[y][x]);
            gameBoard.add(backgroundPanels[y][x]);
        }
    }
    refresh();
    refresh();
    current = 0;
    currentScore.setText("" + current);
    this.setFocusable(true); //from StackOverflow
    this.requestFocusInWindow(); //from StackOverflow
}
```

The Refresh Method

- Spawn a random tile
- Update the tile colors
- Keep track of progress/ win condition

```
//spawn random tiles as well as update the tile colors
//also check win conditions
public void refresh() {
    int counter = 0;
    for (int y = 0; y < 4; y++){
        for (int x = 0; x < 4; x++) {
            if (numbers[y][x] > highestTile) highestTile = numbers[y][x];
            if(numbers[y][x] == 0) {
                counter++;
                backgroundPanels[y][x].setBackground(gray);
            } //Set colors for tiles
            else if (numbers[y][x] == 2) {
                backgroundPanels[y][x].setBackground(white);
            }
            else if (numbers[y][x] == 4) {
                backgroundPanels[y][x].setBackground(yellow.darker());
            }
            else if (numbers[y][x] == 8) {
                backgroundPanels[y][x].setBackground(orange);
            }
            else if (numbers[y][x] == 16) {
                backgroundPanels[y][x].setBackground(orange.brighter());
            }
            else if (numbers[y][x] == 32) {
                backgroundPanels[y][x].setBackground(orange.darker());
            }
            else if (numbers[y][x] == 64) {
                backgroundPanels[y][x].setBackground(red);
            }
            else if (numbers[y][x] >= 128) {
                backgroundPanels[y][x].setBackground(yellow.brighter());
            }
            else if(numbers[y][x] >= 4096) {
                backgroundPanels[y][x].setBackground(Color.BLACK);
            }
        }
    } //randomize spawn location of 2 tiles
    int y = (int) (Math.random() * 4);
    int x = (int) (Math.random() * 4);
    while (numbers[y][x] != 0) {
        if (counter == 0) break;
        y = (int) (Math.random() * 4);
        x = (int) (Math.random() * 4);
    } //add new "2" tile
    if (counter != 0) {
        numbers[y][x] = 2;
        blocks[y][x].setText("" + numbers[y][x]);
        backgroundPanels[y][x].setBackground(white);
    } //if 2048 is reached
    if(highestTile >= 2048) add(congratsPanel, BorderLayout.SOUTH);
}
```

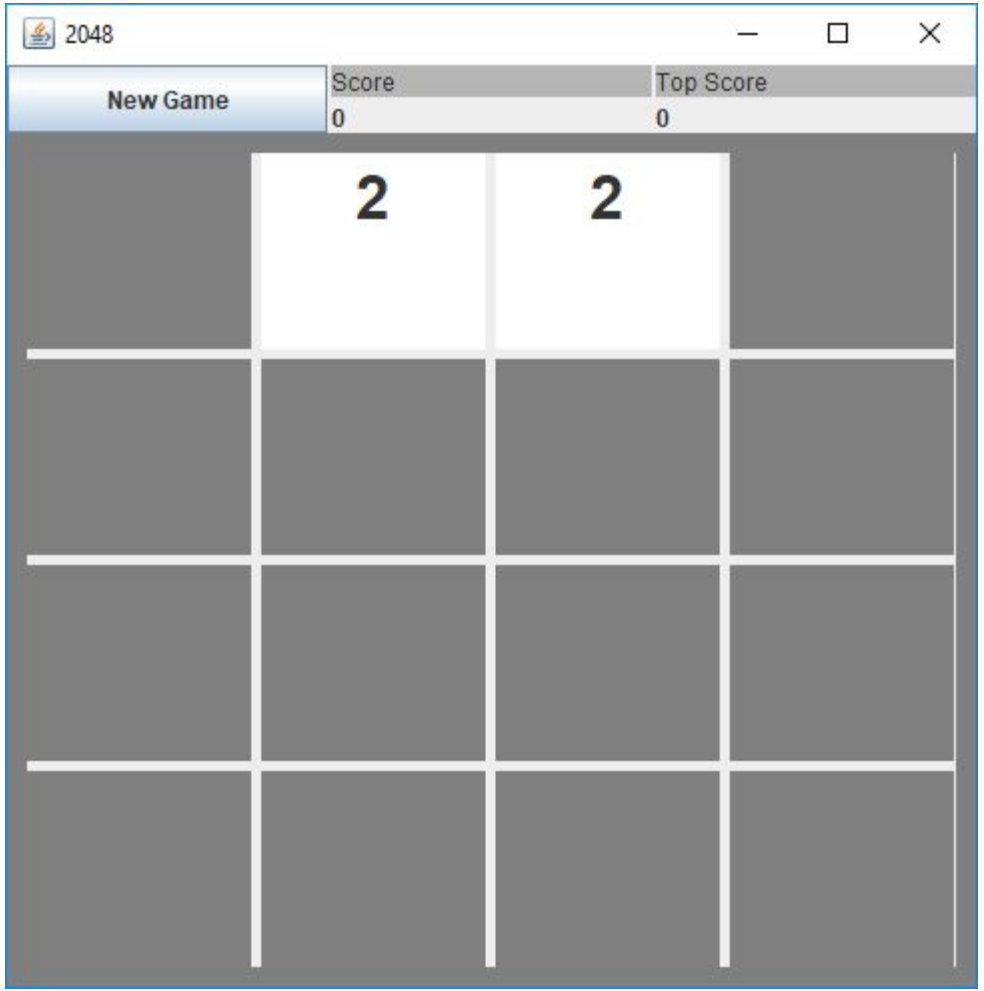
Example of Tile Movement

- Numbers array holds integer values
- “z” integer “moves left” until the next occupied tile
- Tiles are then shifted over
- Matching tiles to the right are merged and text is updated
- Score is tracked

```
else if (b == KeyEvent.VK_RIGHT) { //move right
    for(int y = 0; y < 4; y++) {
        for (int x = 3; x > 0; x--) {
            //find the nearest occupied tile and move it to this empty tile
            if(numbers[y][x] == 0) {
                int z = x;
                while (z > 0) {
                    if (numbers[y][z] == 0) z--;
                    else break;
                }
                numbers[y][x] = numbers[y][z];
                numbers[y][z] = 0;
                if(numbers[y][x] != 0) blocks[y][x].setText("" + numbers[y][x]);
                blocks[y][z].setText("");
            } //merge two equal tiles
            if (numbers[y][x] == numbers[y][x - 1] && numbers[y][x] != 0) {
                numbers[y][x] *= 2;
                plus += numbers[y][x];
                numbers[y][x - 1] = 0;
                blocks[y][x].setText("" + numbers[y][x]);
                blocks[y][x - 1].setText("");
            }

        }
    }
    refresh();
}
```

New Game

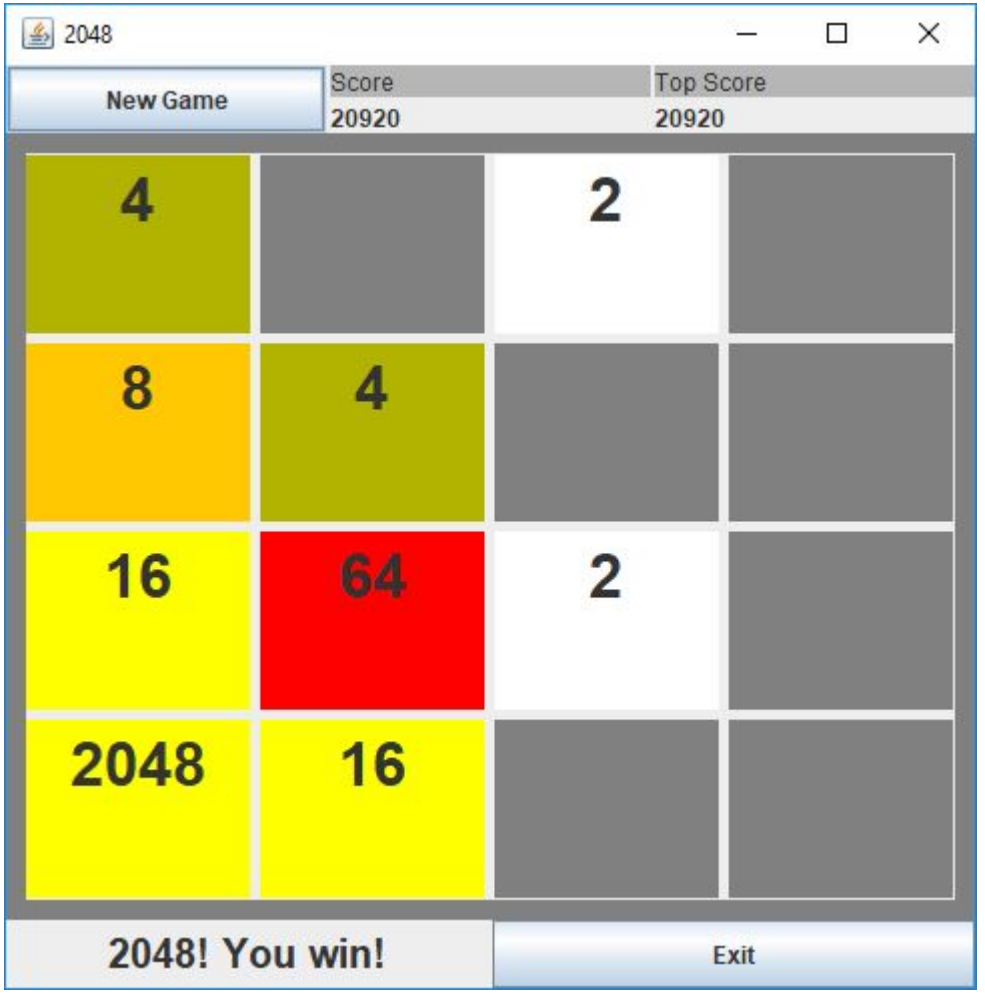


Full board


2048
New Game
Score
828
Top Score
828
2
32
8
2
4
64
32
4
2
8
16
8
8
32
4
2

Winning the game

- Can still continue to play if player chooses to



Clicking new game button