

```

1  /*
2   * Timothy Bui
3   * CPSC-223J-01 TuTh 8:00 - 9:50 AM
4   * Final Project: 2048 recreated in Java
5   */
6  import javax.swing.*;
7  import java.awt.*;
8  import java.awt.event.*;
9  import javax.swing.JOptionPane;
10 public class twenty48 extends JFrame implements ActionListener, KeyListener {
11     JPanel topPanel = new JPanel();
12     JButton newGame = new JButton("New Game"); //New Game Button
13     //Current Score
14     JPanel currentScorePanel = new JPanel();
15     JTextArea currentScoreLabel = new JTextArea("Score");
16     int current = 0;
17     JLabel currentScore = new JLabel("" + current);
18     //Top Score
19     JPanel topScorePanel = new JPanel();
20     JTextArea topScoreLabel = new JTextArea("Top Score");
21     int top = current;
22     JLabel topScore = new JLabel("" + top);
23
24     //Game Board
25     Color gray = Color.GRAY; Color yellow = Color.YELLOW;
26     Color white = Color.WHITE; Color pink = Color.PINK;
27     Color orange = Color.ORANGE; Color red = Color.RED;
28     JPanel gameBoardBorder = new JPanel();
29     JPanel gameBoard = new JPanel();
30     JPanel[][] backgroundPanels = new JPanel[4][4];
31     JPanel topBorder = new JPanel(), rightBorder = new JPanel(),
32         bottomBorder = new JPanel(), leftBorder = new JPanel();
33
34     //Game Tiles
35     JLabel[][] blocks = new JLabel[4][4];
36     int[][] numbers = new int[4][4];
37
38     //win condition

```

```

39     int highestTile = 0;
40     JPanel congratsPanel = new JPanel();
41     JPanel textPanel = new JPanel();
42     JLabel congrats = new JLabel("2048! You win!");
43     JButton exit = new JButton("Exit");
44
45     public twenty48() {
46         super("2048");
47         setSize(500,500);
48         setDefaultCloseOperation(EXIT_ON_CLOSE);
49         setLayout(new BorderLayout());
50         this.addKeyListener(this);
51         this.setFocusable(true); //from StackOverflow
52         this.requestFocusInWindow(); //from StackOverflow
53
54         //Top Panel
55         topPanel.setLayout(new GridLayout(1,3,2,2));
56         newGame.addActionListener(this);
57         topPanel.add(newGame);
58         currentScoreLabel.setBackground(gray.brighter());
59         currentScorePanel.setLayout(new GridLayout(2,1,2,2));
60         currentScorePanel.add(currentScoreLabel);
61         currentScorePanel.add(currentScore);
62         topScoreLabel.setBackground(gray.brighter());
63         topPanel.add(currentScorePanel);
64         topScorePanel.setLayout(new GridLayout(2,1,2,2));
65         topScorePanel.add(topScoreLabel);
66         topScorePanel.add(topScore);
67         topPanel.add(topScorePanel);
68         add(topPanel, BorderLayout.NORTH);
69
70         //Game Board Panel
71         gameBoardBorder.setLayout(new BorderLayout());
72         add(gameBoardBorder, BorderLayout.CENTER);
73
74         //borders surrounding game board
75         topBorder.setBackground(gray);
76         rightBorder.setBackground(gray);

```

```

77     bottomBorder.setBackground(gray);
78     leftBorder.setBackground(gray);
79     gameBoardBorder.add(topBorder, BorderLayout.NORTH);
80     gameBoardBorder.add(rightBorder, BorderLayout.EAST);
81     gameBoardBorder.add(bottomBorder, BorderLayout.SOUTH);
82     gameBoardBorder.add(leftBorder, BorderLayout.WEST);
83     createBoard();
84
85     //Congratulations
86     congratsPanel.setLayout(new GridLayout(1,2,2,2));
87     congrats.setFont(new Font("Arial", Font.BOLD, 20));
88     textPanel.add(congrats);
89     congratsPanel.add(textPanel);
90     exit.addActionListener(this);
91     congratsPanel.add(exit);
92     if(highestTile >= 2048) add(congratsPanel, BorderLayout.SOUTH);
93 }
94 //create the game board and spawn two numbers in random locations
95 public void createBoard() {
96     gameBoard.setLayout(new GridLayout(4, 4, 5, 5));
97     gameBoardBorder.add(gameBoard, BorderLayout.CENTER);
98     for (int y = 0; y < 4; y++) {
99         for(int x = 0; x < 4; x++) {
100             backgroundPanels[y][x] = new JPanel();
101             backgroundPanels[y][x].setLayout(new FlowLayout());
102             backgroundPanels[y][x].setBackground(gray.brighter());
103             blocks[y][x] = new JLabel(" ");
104             blocks[y][x].setFont(new Font("Arial", Font.BOLD, 30));
105             numbers[y][x] = 0;
106             backgroundPanels[y][x].add(blocks[y][x]);
107             gameBoard.add(backgroundPanels[y][x]);
108         }
109     }
110     refresh();
111     refresh();
112     current = 0;
113     currentScore.setText("" + current);
114     this.setFocusable(true); //from StackOverflow

```

```

115         this.requestFocusInWindow(); //from StackOverflow
116     }
117     //spawn random tiles as well as update the tile colors
118     //also check win conditions
119     public void refresh() {
120         int counter = 0;
121         for (int y = 0; y < 4; y++){
122             for (int x = 0; x < 4; x++) {
123                 if (numbers[y][x] > highestTile) highestTile = numbers[y][x];
124                 if(numbers[y][x] == 0) {
125                     counter++;
126                     backgroundPanels[y][x].setBackground(gray);
127                 } //Set colors for tiles
128                 else if (numbers[y][x] == 2) {
129                     backgroundPanels[y][x].setBackground(white);
130                 }
131                 else if (numbers[y][x] == 4) {
132                     backgroundPanels[y][x].setBackground(yellow.darker());
133                 }
134                 else if (numbers[y][x] == 8) {
135                     backgroundPanels[y][x].setBackground(orange);
136                 }
137                 else if (numbers[y][x] == 16) {
138                     backgroundPanels[y][x].setBackground(orange.brighter());
139                 }
140                 else if (numbers[y][x] == 32) {
141                     backgroundPanels[y][x].setBackground(orange.darker());
142                 }
143                 else if (numbers[y][x] == 64) {
144                     backgroundPanels[y][x].setBackground(red);
145                 }
146                 else if (numbers[y][x] >= 128) {
147                     backgroundPanels[y][x].setBackground(yellow.brighter());
148                 }
149                 else if (numbers[y][x] >= 4096) {
150                     backgroundPanels[y][x].setBackground(Color.BLACK);
151                 }
152             }

```



```

153     } //randomize spawn location of 2 tiles
154     int y = (int) (Math.random() * 4);
155     int x = (int) (Math.random() * 4);
156     while (numbers[y][x] != 0) {
157         if (counter == 0) break;
158         y = (int) (Math.random() * 4);
159         x = (int) (Math.random() * 4);
160     } //add new "2" tile
161     if (counter != 0) {
162         numbers[y][x] = 2;
163         blocks[y][x].setText("" + numbers[y][x]);
164         backgroundPanels[y][x].setBackground(white);
165     } //if 2048 is reached
166     if(highestTile >= 2048) add(congratsPanel, BorderLayout.SOUTH);
167 }
168 @Override //new game and exit buttons
169 public void actionPerformed(ActionEvent a) {
170     Object b = a.getSource();
171     if (b == newGame) {
172         gameBoard.removeAll();
173         gameBoard.revalidate(); //from Stack Overflow
174         createBoard();
175     }
176     else if (b == exit) super.dispose();
177 }
178 @Override
179 public void keyTyped(KeyEvent a) {}
180 @Override
181 public void keyPressed(KeyEvent a) {
182     int plus = 0;
183     int b = a.getKeyCode();
184     int temp = 0;
185     if (b == KeyEvent.VK_UP) { //move up
186         for(int y = 1; y < 4; y++) {
187             for (int x = 0; x < 4; x++) {
188                 temp = y;
189                 if(numbers[y][x] != 0) {
190                     while (temp > 0 && numbers[temp - 1][x] == 0) temp--;

```

```

191         numbers[temp][x] = numbers[y][x];
192         if (numbers[temp][x] != 0) blocks[temp][x].setText("" + numbers[temp][x]);
193         if (temp != y) {
194             numbers[y][x] = 0;
195             blocks[y][x].setText("");
196         } //merge two equal tiles
197         if (temp > 0 && numbers[temp - 1][x] == numbers[temp][x]) {
198             numbers[temp - 1][x] += numbers[temp][x];
199             plus += numbers[temp - 1][x];
200             if (numbers[temp - 1][x] != 0) blocks[temp - 1][x].setText("" + numbers[temp - 1][x]);
201             numbers[temp][x] = 0;
202             blocks[temp][x].setText("");
203         }
204     }
205 }
206 }
207 refresh();
208 }
209 else if (b == KeyEvent.VK_RIGHT) { //move right
210     for(int y = 0; y < 4; y++) {
211         for (int x = 3; x > 0; x--) {
212             //find the nearest occupied tile and move it to this empty tile
213             if(numbers[y][x] == 0) {
214                 int z = x;
215                 while (z > 0) {
216                     if (numbers[y][z] == 0) z--;
217                     else break;
218                 }
219                 numbers[y][x] = numbers[y][z];
220                 numbers[y][z] = 0;
221                 if(numbers[y][x] != 0) blocks[y][x].setText("" + numbers[y][x]);
222                 blocks[y][z].setText("");
223             } //merge two equal tiles
224             if (numbers[y][x] == numbers[y][x - 1] && numbers[y][x] != 0) {
225                 numbers[y][x] *= 2;
226                 plus += numbers[y][x];
227                 numbers[y][x - 1] = 0;
228                 blocks[y][x].setText("" + numbers[y][x]);

```

```

229         blocks[y][x - 1].setText("");
230     }
231 }
232 }
233 }
234     refresh();
235 }
236 else if (b == KeyEvent.VK_DOWN) { //move down
237     for(int y = 2; y >= 0; y--) {
238         for (int x = 0; x < 4; x++) {
239             temp = y;
240             if(numbers[y][x] != 0) {
241                 while (temp < 3 && numbers[temp + 1][x] == 0) temp++;
242                 numbers[temp][x] = numbers[y][x];
243                 if (numbers[temp][x] != 0) blocks[temp][x].setText("" + numbers[temp][x]);
244                 if (temp != y) {
245                     numbers[y][x] = 0;
246                     blocks[y][x].setText("");
247                 } //merge two equal tiles
248                 if (temp < 3 && numbers[temp + 1][x] == numbers[temp][x]) {
249                     numbers[temp + 1][x] += numbers[temp][x];
250                     plus += numbers[temp + 1][x];
251                     if (numbers[temp + 1][x] != 0) blocks[temp + 1][x].setText("" + numbers[temp + 1][x]);
252                     numbers[temp][x] = 0;
253                     blocks[temp][x].setText("");
254                 }
255             }
256         }
257     }
258     refresh();
259 }
260 else if (b == KeyEvent.VK_LEFT) { //move left
261     for(int y = 0; y < 4; y++) {
262         for (int x = 0; x < 4; x++) {
263             if(numbers[y][x] == 0) {
264                 int z = x;
265                 while (z < 3) {
266                     if (numbers[y][z] == 0) z++;

```

```

267         else break;
268     }
269     numbers[y][x] = numbers[y][z];
270     numbers[y][z] = 0;
271     if(numbers[y][x] != 0) blocks[y][x].setText("" + numbers[y][x]);
272     blocks[y][z].setText("");
273 } //merge two equal tiles
274 if (x < 3 && numbers[y][x] == numbers[y][x + 1] && numbers[y][x] != 0) {
275     numbers[y][x] *= 2;
276     plus += numbers[y][x];
277     numbers[y][x + 1] = 0;
278     blocks[y][x].setText("" + numbers[y][x]);
279     blocks[y][x + 1].setText("");
280 }
281 }
282 }
283 }
284     refresh();
285 } //press escape to close program
286 else if (b == KeyEvent.VK_ESCAPE) super.dispose();
287 current += plus; //total score gained after each key press
288 currentScore.setText("" + current); //update current score
289 if (current > top) { //if new top score is reached
290     top = current;
291     topScore.setText("" + top);
292 }
293 }
294 @Override
295 public void keyReleased(KeyEvent a) {}
296 public static void main(String[] args) {
297     twenty48 test = new twenty48();
298     test.setVisible(true);
299 }
300 }

```








