GitHub Stuff\project-4---tic-tac-toe-WildestPantaloons\TicTacToeGUI.java

```
1  import java.awt.BorderLayout;
2  import java.awt.Dimension;
3  import java.awt.Font;
4  import java.awt.GridLayout;
5  import java.awt.Point;
6  import java.awt.event.ActionEvent;
7  import java.awt.event.ActionListener;
8  import java.util.Random;
9  import javax.swing.BoxLayout;
10 import javax.swing.JButton;
11 import javax.swing.JFrame;
12 import javax.swing.JLabel;
13 import javax.swing.JPanel;
14 import javax.swing.JTextArea;
15
16 /**
17  * GUI for playing a TicTacToeGame.
18  * @author mvail
19  */
20 public class TicTacToeGUI extends JPanel {
21     private static final long serialVersionUID = 1L;
22     private final String PLAYER = "X";
23     private final String COMPUTER = "O";
24     private final String OPEN = "";
25     private final int DIM = 3;
26     private GridButton[][] gameGrid;
27     private JTextArea movesTextArea;
28     private JButton newGameButton;
29     private TicTacToeGame game;
30
31     /** Initialize the GUI. */
32     public TicTacToeGUI() {
33         game = new TicTacToeGame();
34         Font bigFont = new Font("Serif", Font.PLAIN, 48);
35         Font smallFont = new Font("Serif", Font.PLAIN, 36);
36         gameGrid = new GridButton[DIM][DIM];
37         GridButtonListener buttonListener = new GridButtonListener();
38         for (int row = 0; row < DIM; row++) {
39             for (int col = 0; col < DIM; col++) {
40                 gameGrid[row][col] = new GridButton(OPEN, row, col);
41                 gameGrid[row][col].addActionListener(buttonListener);
42                 gameGrid[row][col].setPreferredSize(new Dimension(128,128));
43                 gameGrid[row][col].setFont(bigFont);
44             }
45         }
46
47         this.setLayout(new BorderLayout());
48
49         JPanel controlsPanel = new JPanel(); //default FlowLayout
50         newGameButton = new JButton("New Game");
51         newGameButton.setFont(bigFont);
52         newGameButton.addActionListener(new NewGameButtonListener());
53         controlsPanel.add(newGameButton);
```

```
54          this.add(controlsPanel, BorderLayout.SOUTH);
55
56          JPanel movesPanel = new JPanel(); //vertical BoxLayout
57          movesPanel.setLayout(new BoxLayout(movesPanel, BoxLayout.Y_AXIS));
58          JLabel movesLabel = new JLabel("Moves");
59          movesLabel.setFont(smallFont);
60          movesPanel.add(movesLabel);
61          movesTextArea = new JTextArea();
62          movesTextArea.setPreferredSize(new Dimension(256,512));
63          movesTextArea.setFont(smallFont);
64          movesTextArea.setEnabled(false);
65          movesPanel.add(movesTextArea);
66          this.add(movesPanel, BorderLayout.EAST);
67
68          JPanel gamePanel = new JPanel();
69          gamePanel.setLayout(new GridLayout(DIM,DIM));
70          for (int row = 0; row < DIM; row++) {
71              for (int col = 0; col < DIM; col++) {
72                  gamePanel.add(gameGrid[row][col]);
73              }
74          }
75          this.add(gamePanel, BorderLayout.CENTER);
76      }
77
78      /** Reset the game and corresponding GUI controls. */
79      private void resetGame() {
80          //reset the game
81          game.newGame();
82          //clear the GUI board
83          for(int row = 0; row < DIM; row++) {
84              for (int col = 0; col < DIM; col++) {
85                  gameGrid[row][col].setText(OPEN);
86                  gameGrid[row][col].setEnabled(true);
87              }
88          }
89          //clear visual list
90          movesTextArea.setText("");
91          //refresh GUI
92          revalidate();
93      }
94
95      /** Disable GUI game board and display game results */
96      private void endGame() {
97          //disable game board buttons
98          for (int row = 0; row < DIM; row++) {
99              for (int col = 0; col < DIM; col++) {
100                 gameGrid[row][col].setEnabled(false);
101             }
102         }
103         //display the moves history and winner
104         movesTextArea.setText("");
105         Point[] moves = game.getMoves();
106         for (int i = 0; i < moves.length; i++) {
107             if (i %2 == 0) {
108                 movesTextArea.append("X: ");
109             } else {
```

```
110                 movesTextArea.append("O: ");
111             }
112             movesTextArea.append("row " + moves[i].x + ", col " + moves[i].y + "\n");
113         }
114         if (game.getGameState() == TicTacToe.GameState.TIE) {
115             movesTextArea.append("NO WINNER\n");
116         } else if (game.getGameState() == TicTacToe.GameState.X_WON) {
117             movesTextArea.append("Winner: X\n");
118         } else if (game.getGameState() == TicTacToe.GameState.O_WON) {
119             movesTextArea.append("Winner: O\n");
120         } else {
121             movesTextArea.append("ERROR\n");
122         }
123     }
124 
125     /** Start the GUI
126      * @param args unused
127      */
128     public static void main(String[] args) {
129         JFrame frame = new JFrame("Tic Tac Toe");
130         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
131         frame.getContentPane().add(new TicTacToeGUI());
132         frame.pack();
133         frame.setVisible(true);
134     }
135 
136     /**
137      * Private inner class to respond to newGameButton clicks.
138      */
139     private class NewGameButtonListener implements ActionListener {
140         @Override
141         public void actionPerformed(ActionEvent e) {
142             resetGame();
143         }
144     }
145 
146     /**
147      * Private inner class to respond to grid button clicks.
148      * Update 'gameGrid' if the button is not already claimed.
149      * Make a computer move after player moves.
150      * Check for game over conditions.
151      */
152     private class GridButtonListener implements ActionListener {
153         @Override
154         public void actionPerformed(ActionEvent arg0) {
155             GridButton button = (GridButton)(arg0.getSource());
156             if (!game.gameOver()) {
157                 //call choose(X, row, col) corresponding to clicked button
158                 //  if the position is successfully claimed
159                 if (game.choose(TicTacToe.BoardChoice.X, button.getRow(), button.getCol())) {
160                     button.setText(PLAYER);
161                     if (game.gameOver()) { //did the player just win?
162                         endGame();
163                     } else { //make a random move for the computer
164                         Random rand = new Random();
165                         boolean done = false;
```

```
166                         while (!done) {
167                             int cRow = rand.nextInt(DIM);
168                             int cCol = rand.nextInt(DIM);
169                             if (game.choose(TicTacToe.BoardChoice.O, cRow, cCol)) {
170                                 gameGrid[cRow][cCol].setText(COMPUTER);
171                                 done = true;
172                             }
173                         }
174                         if (game.gameOver()) { //did the computer just win?
175                             endGame();
176                         }
177                     }
178                 }//if player clicked an unclaimed position
179             }//if game not over
180         }//actionPerformed()
181     }//GridButtonListener
182 
183     /**
184      * A button that knows its row,col coordinates
185      */
186     private class GridButton extends JButton {
187         private static final long serialVersionUID = 1L;
188         private int row;
189         private int col;
190 
191         /**
192          * Create this button with given text and coordinates.
193          * @param row row coordinate for this button
194          * @param column column coordinate for this button
195          */
196         public GridButton(String text, int row, int column) {
197             super(text);
198             this.row = row;
199             this.col = column;
200         }
201 
202         /** @return the row */
203         public int getRow() {
204             return row;
205         }
206 
207         /** @return the col */
208         public int getCol() {
209             return col;
210         }
211     } //GridButton
212 } //TicTacToeGUI
213 
```