package product;

import java.awt.\*;

import java.awt.event.\*;

import java.util.\*;

import javax.swing.\*;

import product.Game.GameState;

public class GUI implements ActionListener {

    JFrame frame = new JFrame();

    JPanel title\_panel = new JPanel();

    JPanel button\_panel = new JPanel();

    JPanel toppanel = new JPanel();

    JLabel toptext = new JLabel();

    JPanel leftpanel = new JPanel();

    JLabel lefttext = new JLabel();

    JPanel rightpanel = new JPanel();

    JLabel righttext = new JLabel();

    JLabel gameStatusBar = new JLabel();

    JLabel ptext = new JLabel();

    JButton[][] buttons = new JButton[8][8];

    Random random = new Random();

    JRadioButton bS = new JRadioButton("S");

    JRadioButton bO = new JRadioButton("O");

    JRadioButton rS = new JRadioButton("S");

    JRadioButton rO = new JRadioButton("O");

    JRadioButton sgame = new JRadioButton("Simple game");

    JRadioButton ggame = new JRadioButton("General game");

    JRadioButton bhuman = new JRadioButton("Human");

    JRadioButton bcomputer = new JRadioButton("Computer");

    JRadioButton rhuman = new JRadioButton("Human");

    JRadioButton rcomputer = new JRadioButton("Computer");

    private Game game;

    public GUI() {

        this(new Game());

    }

    public GUI(Game game) {

        this.game = game;

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        frame.setSize(600, 600);

        frame.getContentPane().setBackground(Color.cyan);

        frame.setLayout(new BorderLayout());

        frame.setVisible(true);

        // Game selection buttons

        ButtonGroup selectGame = new ButtonGroup();

        selectGame.add(sgame);

        selectGame.add(ggame);

        // Blue Player S or O

        ButtonGroup blue = new ButtonGroup();

        blue.add(bS);

        blue.add(bO);

        ButtonGroup blueside = new ButtonGroup();

        blueside.add(bhuman);

        blueside.add(bcomputer);

        // Red Player S or O

        ButtonGroup red = new ButtonGroup();

        red.add(rS);

        red.add(rO);

        ButtonGroup redside = new ButtonGroup();

        redside.add(rhuman);

        redside.add(rcomputer);

        // Game Mode selection

        toppanel.setLayout(new BorderLayout());

        toppanel.setBounds(100, 500, 100, 800);

        JPanel top = new JPanel(new GridLayout(0, 3));

        toptext.setText("SOS");

        top.add(toptext);

        toptext.setHorizontalAlignment(JLabel.CENTER);

        top.add(sgame);

        top.add(ggame);

        toppanel.add(top);

        // Blue Player side

        leftpanel.setLayout(new BorderLayout());

        leftpanel.setBounds(0, 0, 50, 400);

        JPanel leftside = new JPanel(new GridLayout(6, 0));

        lefttext.setText("Blue player");

        lefttext.setHorizontalAlignment(JLabel.CENTER);

        leftside.add(lefttext);

        ((AbstractButton) leftside.add(bhuman)).setHorizontalAlignment(JLabel.CENTER);

        ((AbstractButton) leftside.add(bS)).setHorizontalAlignment(JLabel.CENTER);

        ((AbstractButton) leftside.add(bO)).setHorizontalAlignment(JLabel.CENTER);

        ((AbstractButton) leftside.add(bcomputer)).setHorizontalAlignment(JLabel.CENTER);

        leftpanel.add(leftside);

        // Red Player side

        rightpanel.setLayout(new BorderLayout());

        rightpanel.setBounds(100, 500, 100, 800);

        JPanel rightside = new JPanel(new GridLayout(6, 0));

        righttext.setText("Red player");

        righttext.setHorizontalAlignment(JLabel.CENTER);

        rightside.add(righttext);

        ((AbstractButton) rightside.add(rhuman)).setHorizontalAlignment(JLabel.CENTER);

        ((AbstractButton) rightside.add(rS)).setHorizontalAlignment(JLabel.CENTER);

        ((AbstractButton) rightside.add(rO)).setHorizontalAlignment(JLabel.CENTER);

        ((AbstractButton) rightside.add(rcomputer)).setHorizontalAlignment(JLabel.CENTER);

        rightpanel.add(rightside);

        // Turn Panel at bottom

        gameStatusBar.setBackground(Color.DARK\_GRAY);

        gameStatusBar.setForeground(Color.white);

        gameStatusBar.setFont(new Font("Ink Free", Font.BOLD, 30));

        gameStatusBar.setHorizontalAlignment(JLabel.CENTER);

        gameStatusBar.setText("Curren turn: ");

        gameStatusBar.setOpaque(true);

        title\_panel.setLayout(new BorderLayout());

        title\_panel.setBounds(0, 0, 800, 100);

        title\_panel.add(gameStatusBar);

        // Adjust the Status Bar

        {

            if (game.getGameState() == GameState.PLAYING) {

                gameStatusBar.setForeground(Color.BLACK);

                if (game.getTurn() == "Blue") {

                    gameStatusBar.setText("Blue's Turn");

                } else {

                    gameStatusBar.setText("Red's Turn");

                }

            } else if (game.getGameState() == GameState.DRAW) {

                gameStatusBar.setForeground(Color.RED);

                gameStatusBar.setText("It's a Draw! Click to play again.");

            } else if (game.getGameState() == GameState.BLUE\_WON) {

                gameStatusBar.setForeground(Color.RED);

                gameStatusBar.setText("Blue Won! Click to play again.");

            } else if (game.getGameState() == GameState.RED\_WON) {

                gameStatusBar.setForeground(Color.RED);

                gameStatusBar.setText("Red Won! Click to play again.");

            }

        }

        // Grid-panel

        button\_panel.setLayout(new GridLayout(game.getTotalRows(), game.getTotalColumns()));

        button\_panel.setBackground(Color.gray);

        frame.add(title\_panel, BorderLayout.SOUTH);

        frame.add(toppanel, BorderLayout.NORTH);

        frame.add(leftpanel, BorderLayout.WEST);

        frame.add(rightpanel, BorderLayout.EAST);

        frame.add(button\_panel);

        // numbers of cells

        for (int i = 0; i < game.getTotalRows(); i++) {

            for (int j = 0; j < game.getTotalColumns(); j++) {

                buttons[i][j] = new JButton();

                button\_panel.add(buttons[i][j]);

                buttons[i][j].setFont(new Font("MV Boli", Font.BOLD, 24));

                buttons[i][j].setFocusable(false);

                buttons[i][j].addActionListener(this);

                buttons[i][j].setBackground(Color.white);

            }

        }

    }

    public JRadioButton getLetterSelected(JRadioButton sBtn, JRadioButton oBtn) {

        if (sBtn.isSelected()) {

            return sBtn;

        }

        return oBtn;

    }

    @Override

    public void actionPerformed(ActionEvent e) {

        if (game.getGameState() == GameState.PLAYING) {

            for (int i = 0; i < game.getTotalRows(); i++) {

                for (int j = 0; j < game.getTotalColumns(); j++) {

                    if (e.getSource() == buttons[i][j]) {

                        if (game.getTurn() == "Blue") {

                            if (buttons[i][j].getText().equals("")) {

                                gameStatusBar.setText("RED");

                                buttons[i][j].setForeground(Color.BLUE);

                                // buttons[i][j].setText("S");

                                buttons[i][j].setText(getLetterSelected(bS, bO).getText());

                                game.turn = (game.turn == "Blue") ? "Red" : "Blue";

                            }

                        } else {

                            if (buttons[i][j].getText() == "") {

                                gameStatusBar.setText("BLUE");

                                buttons[i][j].setForeground(Color.RED);

                                // buttons[i][j].setText("O");

                                buttons[i][j].setText(getLetterSelected(rS, rO).getText());

                                game.turn = (game.turn == "Red") ? "Blue" : "Red";

                            }

                        }

                        if (game.hasWon(buttons)) {

                            if (sgame.isSelected()) {

                                game.turn = (game.turn == "Red") ? "Blue" : "Red";

                                game.currentGameState = (game.getTurn() == "Blue") ? GameState.BLUE\_WON

                                        : GameState.RED\_WON;

                                for (int r = 0; r < game.getTotalRows(); r++) {

                                    for (int c = 0; c < game.getTotalColumns(); c++) {

                                        buttons[r][c].setEnabled(false);

                                    }

                                }

                                gameStatusBar.setText(game.getGameState() + " Won! Click to play again.");

                                game.resetGame();

                            } else if (ggame.isSelected()) {

                                int blueCount = 0;

                                int redCount = 0;

                                int count;

                                {

                                    gameStatusBar.setText(game.getTurn() + " Won!");

                                    count = blueCount = (game.getTurn() == "Blue") ? ++blueCount : ++redCount;

                                    gameStatusBar.setText(game.getTurn() + "Count : " + count);

                                    // game.resetGame();

                                }

                            }

                        }

                        else if (game.isDraw()) {

                            game.currentGameState = GameState.DRAW;

                            gameStatusBar.setText(game.getGameState() + "! Click to play again.");

                            game.resetGame();

                        }

                    }

                }

            }

        }

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(new Runnable() {

            public void run() {

                new GUI(new AutoGame("Blue"));

            }

        });

    }

 Random random = new Random();

                                int so = random.nextInt(2);

                                int position = random.nextInt(game.getTotalRows() \* game.getTotalColumns());

                                buttons[position / game.getTotalRows()][position / game.getTotalColumns()]

                                        .setText(getLetterSelected(bS, bO).getText());

    if (bcomputer.isSelected()) {

        // if (game.getGameState() == GameState.PLAYING) {

        //     for (int i = 0; i < game.getTotalRows(); i++) {

        //     for (int j = 0; j < game.getTotalColumns(); j++) {

        // if (e.getSource() == buttons[i][j]) {

        //     if (game.getTurn() == "Blue") {

        Random random = new Random();

        int position = random.nextInt(game.getTotalRows() \* game.getTotalColumns());

        makeMove(position / game.getTotalRows(), position % game.getTotalColumns());

    public void makeMove(int row, int column) {

        makeMove(row, column);

        if (game.getTurn() == "Blue" && game.getGameState() == GameState.PLAYING) {

        makeAutoMove();

        }

    }

    private void makeAutoMove() {

        if (!makeWinningMove()) {

        if (!blockOpponentWinningMove())

            makeRandomMove();

    }

    }

    private boolean makeWinningMove() {

        return false;

    }

    private boolean blockOpponentWinningMove() {

        return false;

    }

    private void makeRandomMove() {

    int numberOfEmptyCells = game.getNumberOfEmptyCells();

    Random random = new Random();

    int targetMove = random.nextInt(numberOfEmptyCells);

    int index = 0;

    for (int row = 0; row < game.getTotalRows(); ++row) {

    for (int col = 0; col < game.getTotalColumns(); ++col) {

    if (buttons[row][col].getText() == "") {

    if (targetMove == index) {

            makeMove(row, col);

            gameStatusBar.setText("RED");

    buttons[row][col].setForeground(Color.BLUE);

    buttons[row][col].setText(getLetterSelected(bS, bO).getText());

    game.turn = (game.turn == "Blue") ? "Red" : "Blue";

    return;

    } else

    index++;

    }

    }

    }

    }

    if (buttons[i][j].getText().equals("")) {

    gameStatusBar.setText("RED");

    buttons[i][j].setForeground(Color.BLUE);

    // buttons[i][j].setText("S");

    buttons[i][j].setText(getLetterSelected(bS, bO).getText());

    game.turn = (game.turn == "Blue") ? "Red" : "Blue";

    }

    } else {

    if (buttons[i][j].getText() == "") {

    gameStatusBar.setText("BLUE");

    buttons[i][j].setForeground(Color.RED);

    // buttons[i][j].setText("O");

    buttons[i][j].setText(getLetterSelected(rS, rO).getText());

    game.turn = (game.turn == "Red") ? "Blue" : "Red";

    }

    }

    }

    }

    }

    }}

}