Object Oriented Language (Java) Practice

Program Documentation

Student ID: 202237022

Name:MD MAHADI HASAN(马博文)

1. Program functions：

#### Introduction

This Java Swing application exemplifies effective UI design and layout management techniques while integrating a Tic-Tac-Toe game. It aims to provide a comprehensive showcase of Java Swing capabilities, emphasizing clarity, responsiveness, and user-friendly interaction.

#### Components and Features

**Status and Score Labels**

* 1. **Status Label**: Positioned at the top, it updates dynamically to reflect the current player's turn (either "Player X's turn" or "Player O's turn"). The label uses a dark cyan background with white text for high visibility and maintains consistency throughout the application.
  2. **Score Label**: Located at the bottom, it displays the scores of Player X and Player O in a centered format ("Player X: 0 | Player O: 0"). This label features a light gray background to ensure readability and serves as a persistent reference to ongoing game statistics.

**Tic-Tac-Toe Game**

* 1. **Game Board**: Implemented with a 3x3 grid of JButton components, representing the playable area for the Tic-Tac-Toe game. Each button is styled with a plain Arial font and a sizeable dimension (40 points), ensuring clear visibility and touch responsiveness.
  2. **Player Interaction**: Players alternate turns by clicking on empty buttons. The application distinguishes between Player X (blue "X" text) and Player O (red "O" text) moves, updating the status label accordingly.
  3. **Win Condition**: The application checks for win conditions after each move, including horizontal, vertical, and diagonal alignments of "X" or "O". Upon detecting a win, the respective player's score increments, and the game board disables further moves until reset.
  4. **Reset Functionality**: Allows players to restart the game at any point, resetting the game board, scores, and initial status ("Player X's turn").

**Layout Panels**

* 1. **Grid Layout Panel**: Displays a 12x12 grid of JLabel components, each with alternating background colors (black and white). It responds to mouse hover events by highlighting the hovered label in gray, enhancing user interactivity and visual feedback.
  2. **Null Layout Panel**: Uses absolute positioning (setLayout(null)) to place JLabel, JTextField, JButton, JCheckBox, and JComboBox components at precise coordinates. This layout method demonstrates control over component placement for specific UI designs.
  3. **Flow Layout Panel**: Arranges four JButton components ("Button 1" to "Button 4") horizontally within the panel. Each button triggers a message dialog upon click, showcasing event handling in a dynamically resizable layout.
  4. **Card Layout Panel**: Incorporates a JPanel with CardLayout, facilitating seamless switching between two distinct cards ("Card 1" and "Card 2"). A JButton labeled "Switch Card" allows users to toggle between the two cards, illustrating CardLayout's utility for managing multiple views within a single container.

**Tabbed Interface**

* 1. **JTabbedPane**: Centralizes navigation across different panels, providing a tabbed interface that includes:
     1. **Tic-Tac-Toe Tab**: Direct access to the interactive Tic-Tac-Toe game board.
     2. **Grid Layout Tab**: Showcases the GridLayout panel with a matrix of colored JLabels.
     3. **Null Layout Tab**: Demonstrates absolute positioning of UI components for customized layouts.
     4. **Flow Layout Tab**: Presents dynamically arranged buttons in a horizontally aligned FlowLayout.
     5. **Card Layout Tab**: Displays the CardLayout panel with switchable cards, emphasizing versatility in managing multiple UI views.

#### Design Choices and Enhancements

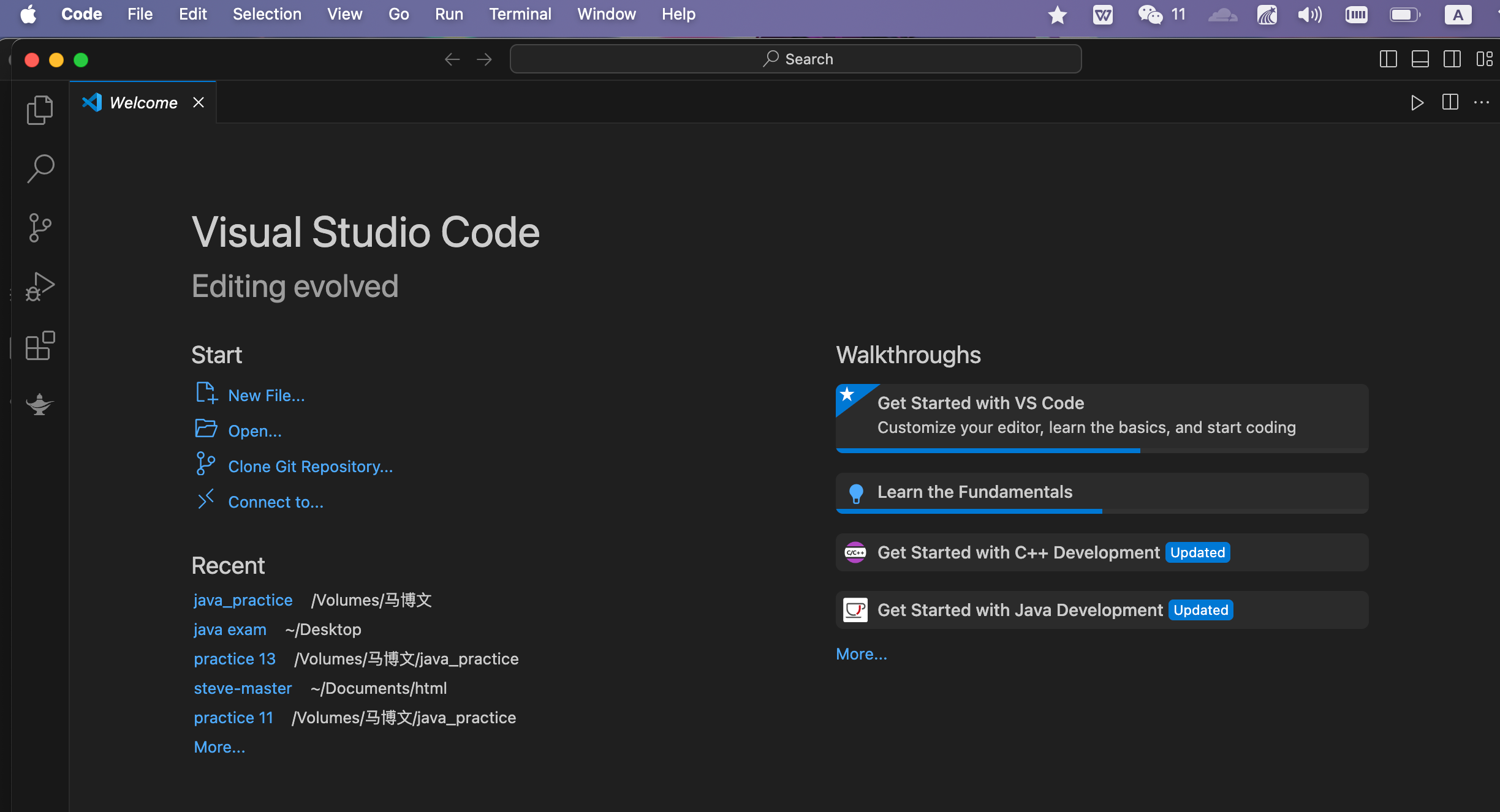
* **Visual Consistency**: Maintains a consistent color scheme (dark cyan, white, light gray) and font (Arial) throughout the application for professional aesthetics and readability.
* **User Engagement**: Incorporates interactive elements such as mouse hover effects (Grid Layout Panel) and message dialogs (Flow Layout Panel buttons) to enhance user engagement and feedback.
* **Code Structure**: Organizes code into separate classes (e.g., PanelGridLayout, PanelNullLayout) and methods with meaningful names and comments, ensuring clarity, maintainability, and ease of extension.
* **Resizable Interface**: Supports dynamic resizing of the main JFrame window, adapting the layout to varying screen dimensions while preserving component alignment and usability.

#### Conclusion

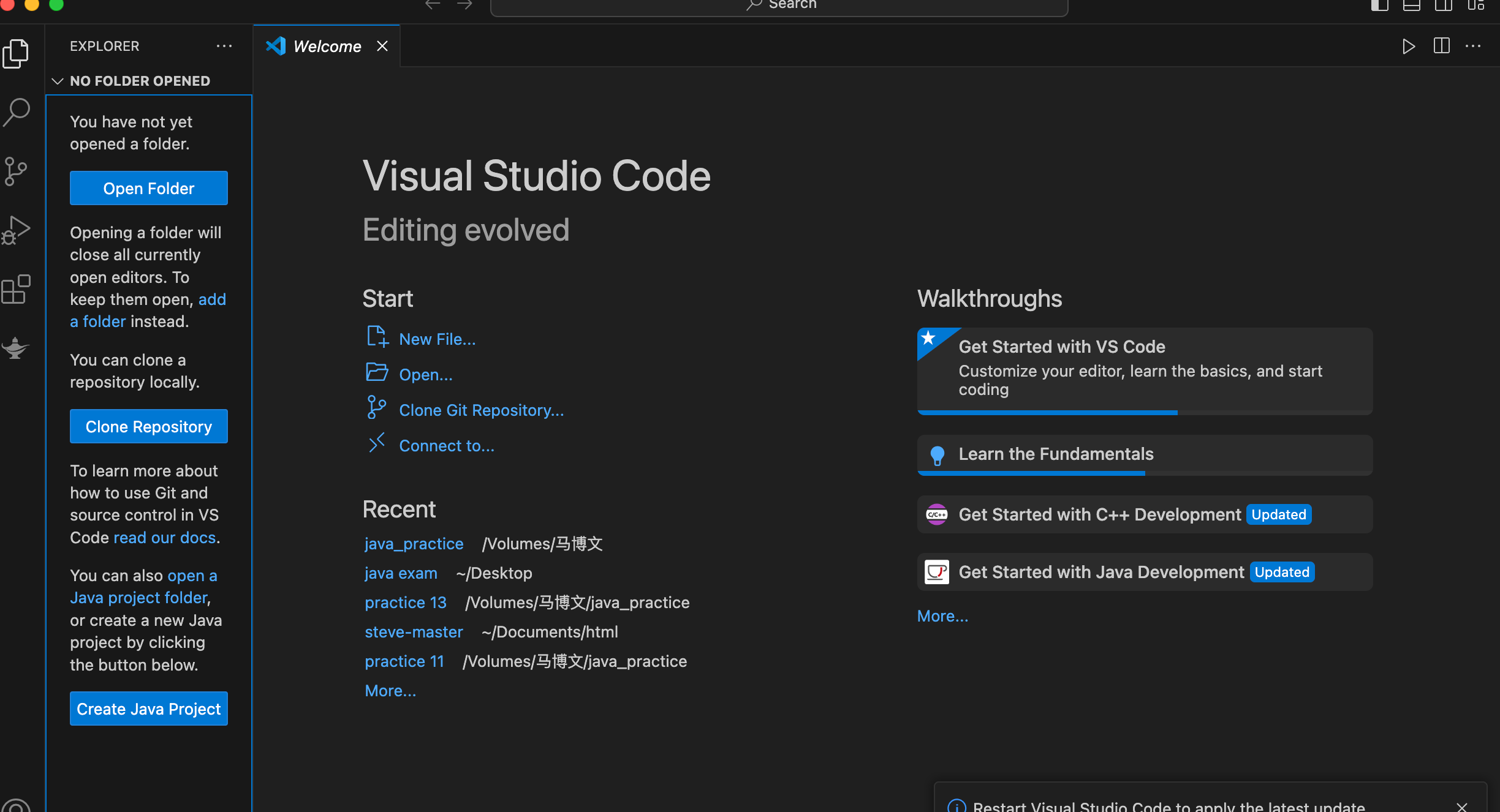
This Java Swing application serves as both an educational tool and a practical example of effective UI design principles and layout management techniques. It demonstrates how to leverage different Swing components and layout managers to create intuitive user interfaces with responsive behaviors. By focusing on clarity, consistency, and interactive features, the application showcases best practices for building robust Java GUI applications.

2. Implementation steps:

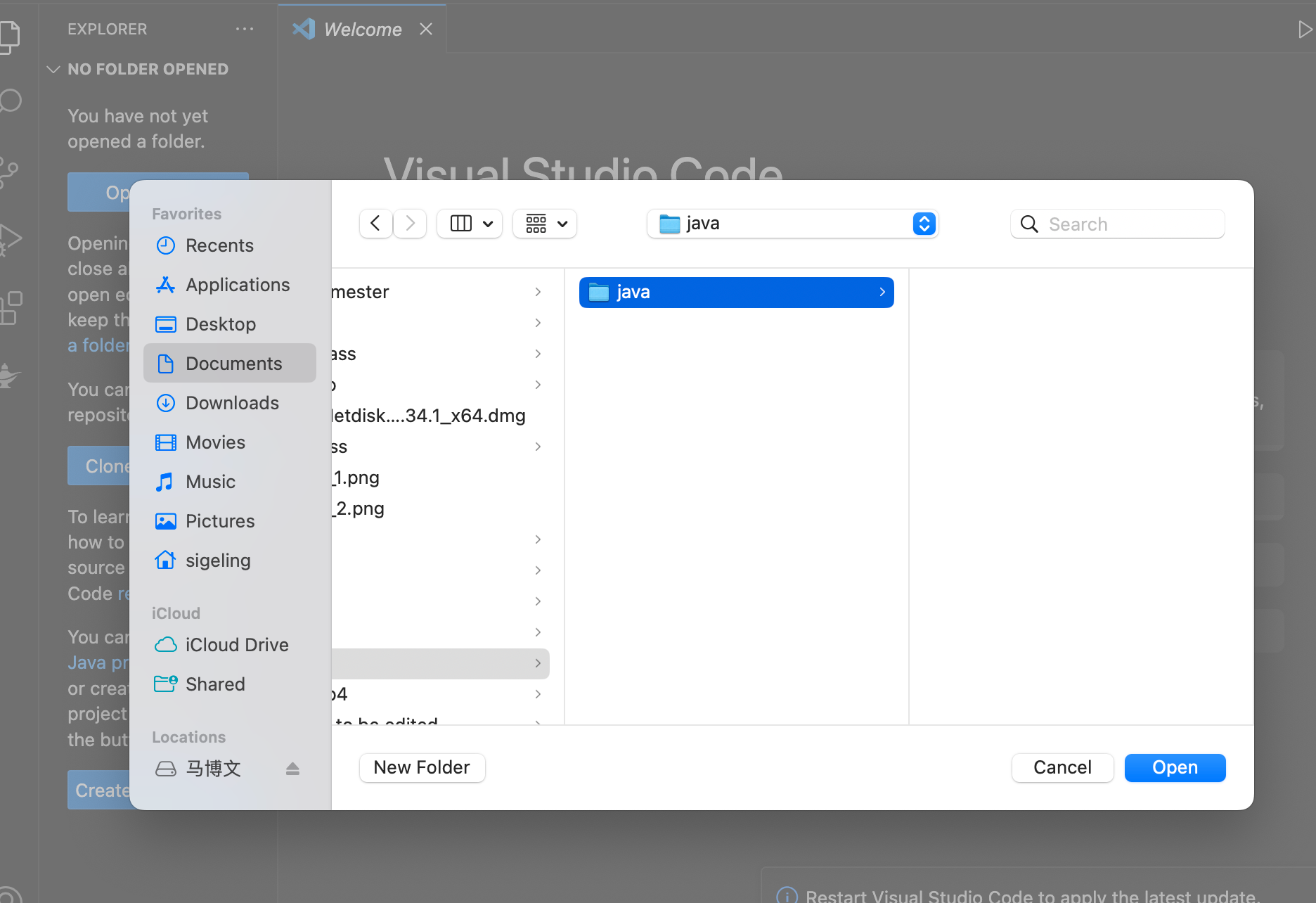
Open vs code IDE:



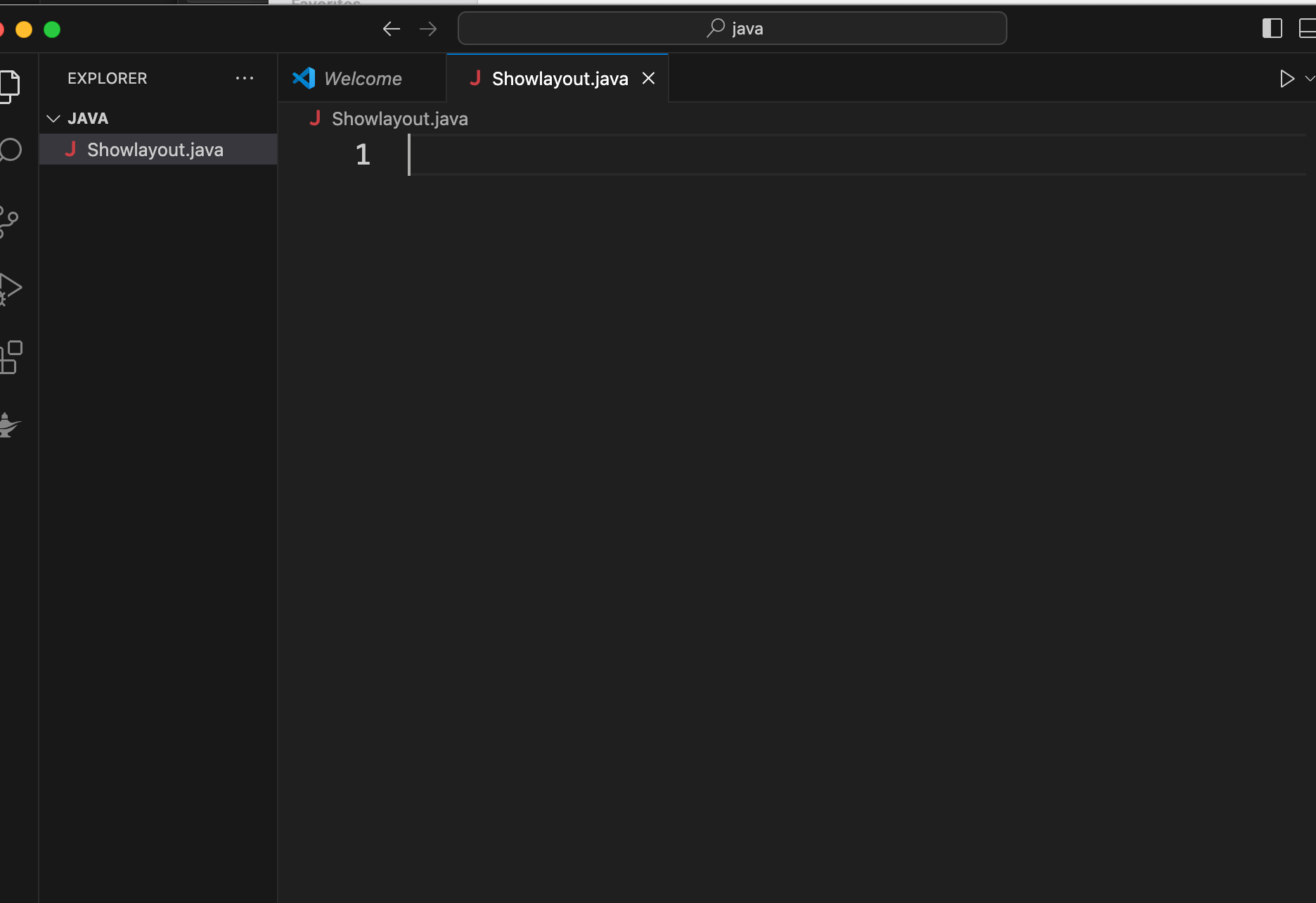
Select OPen Folder:



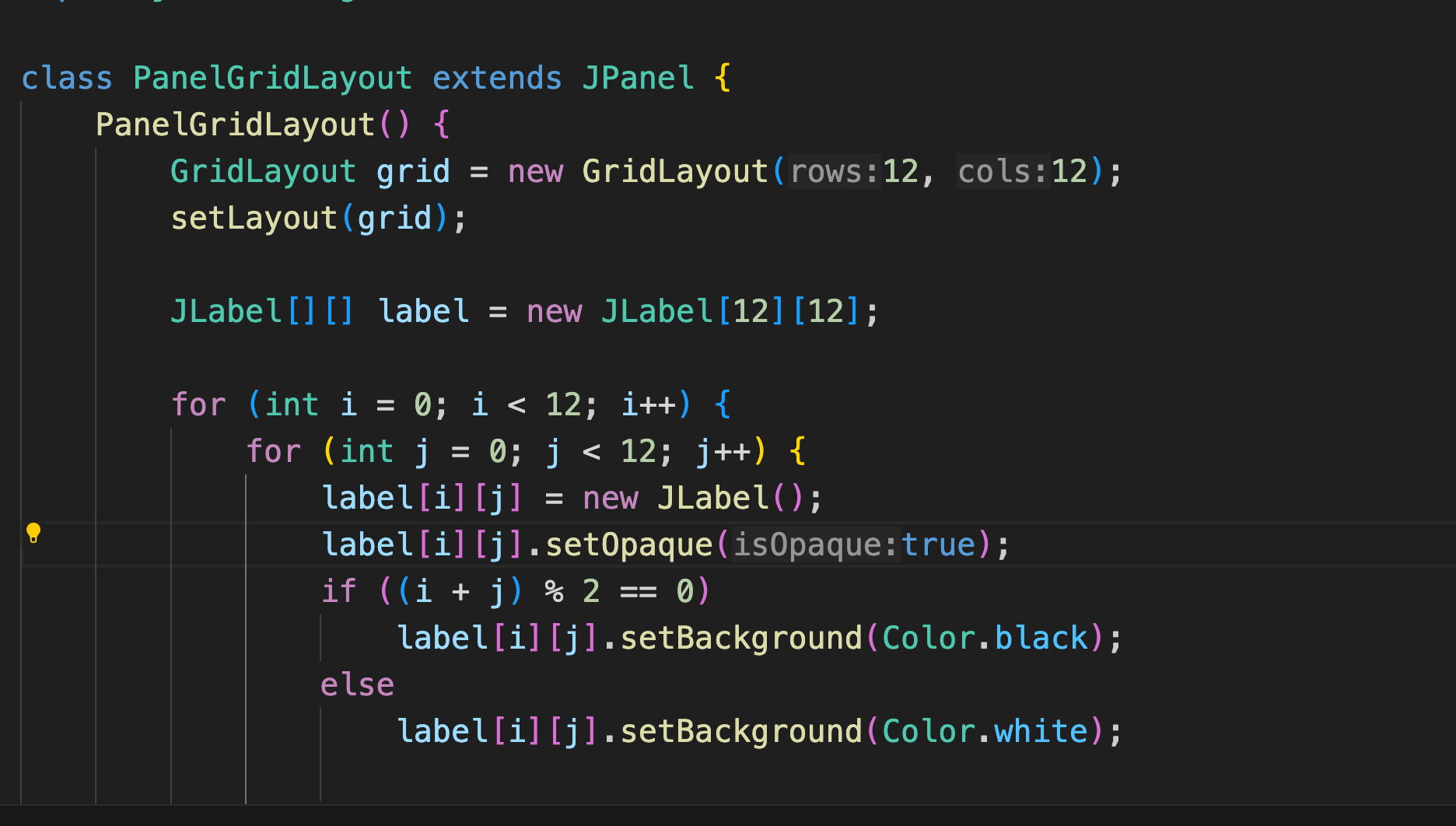
Create new file:



Created java class:

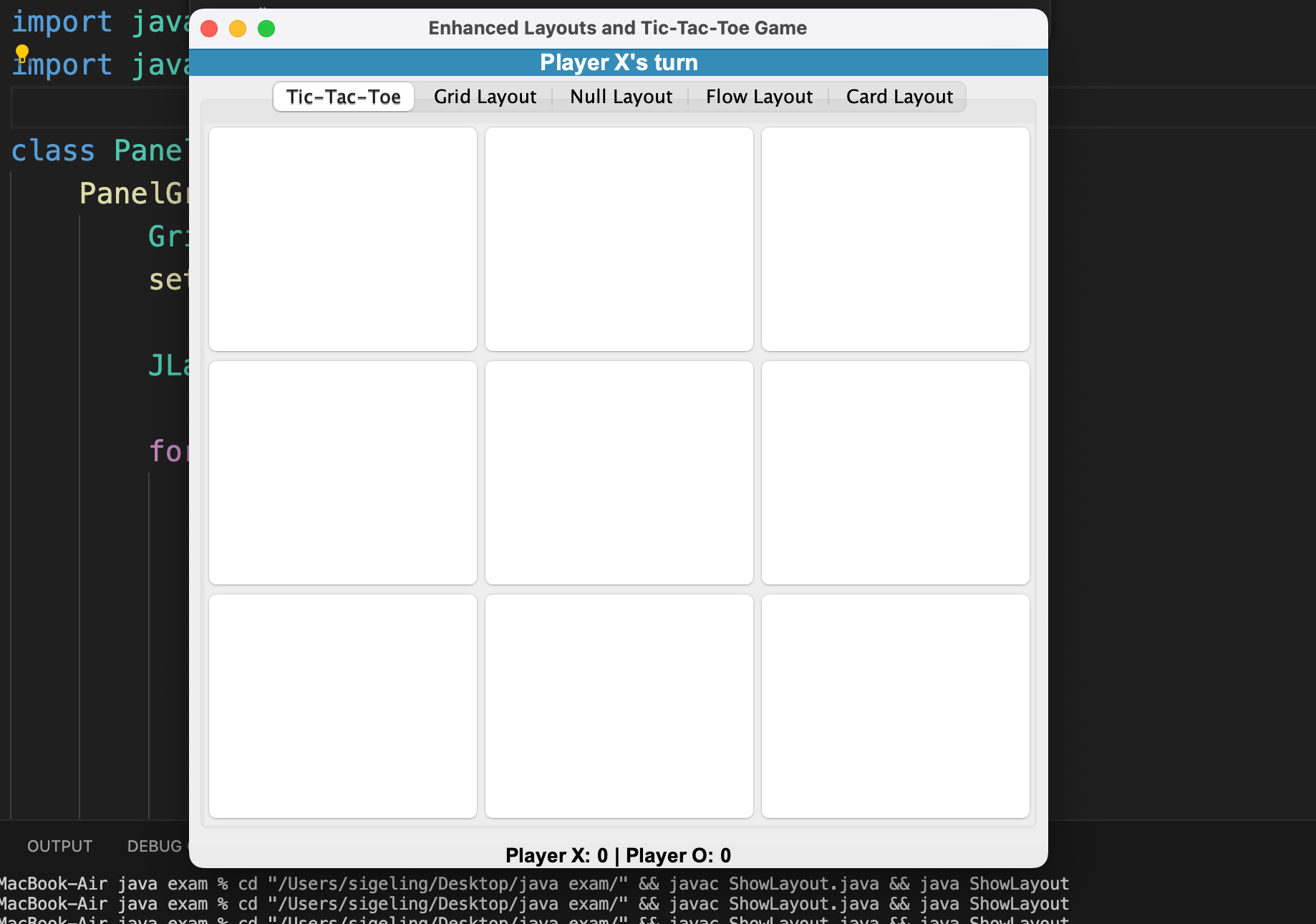


Write code:

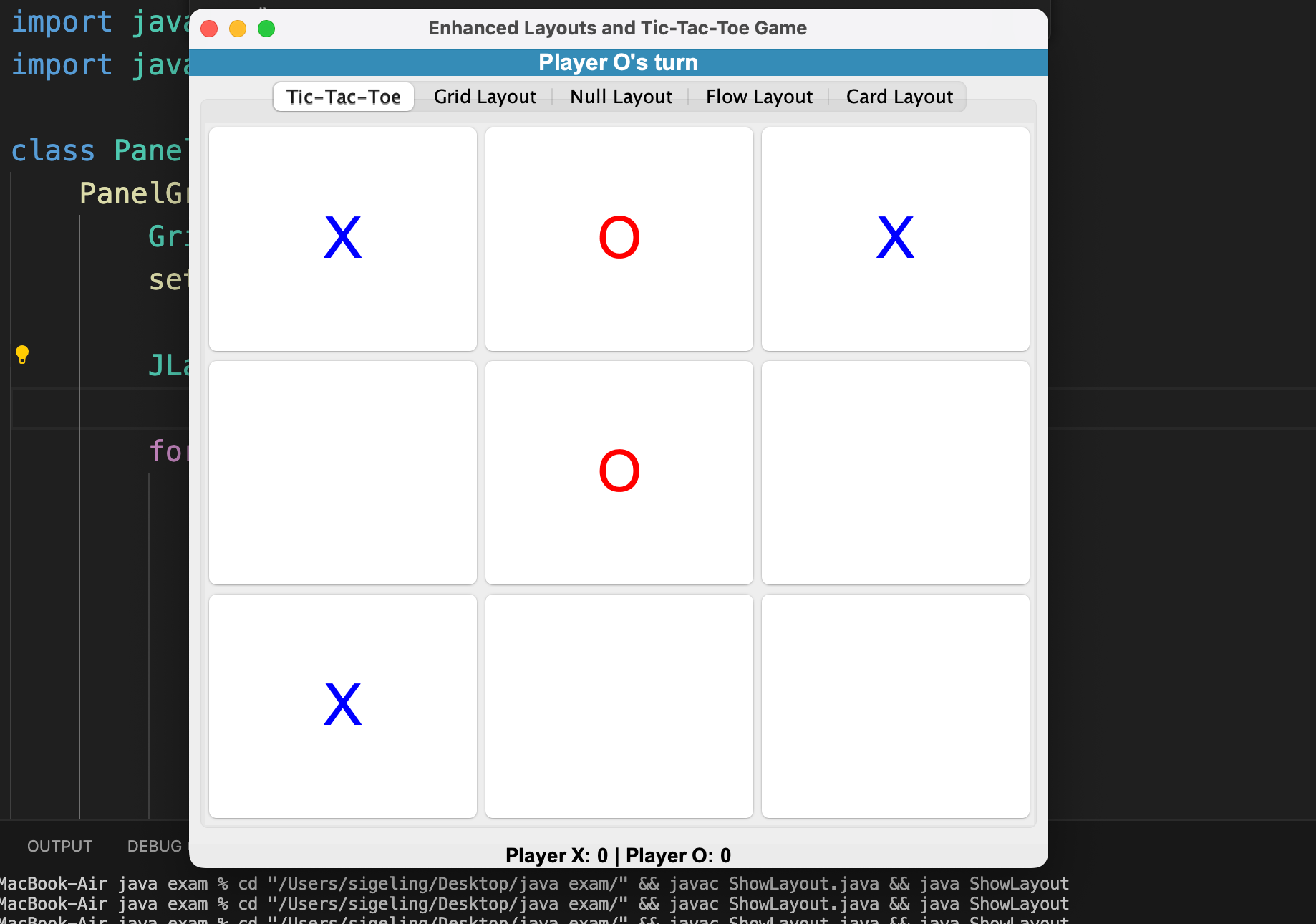


OUTPUT:

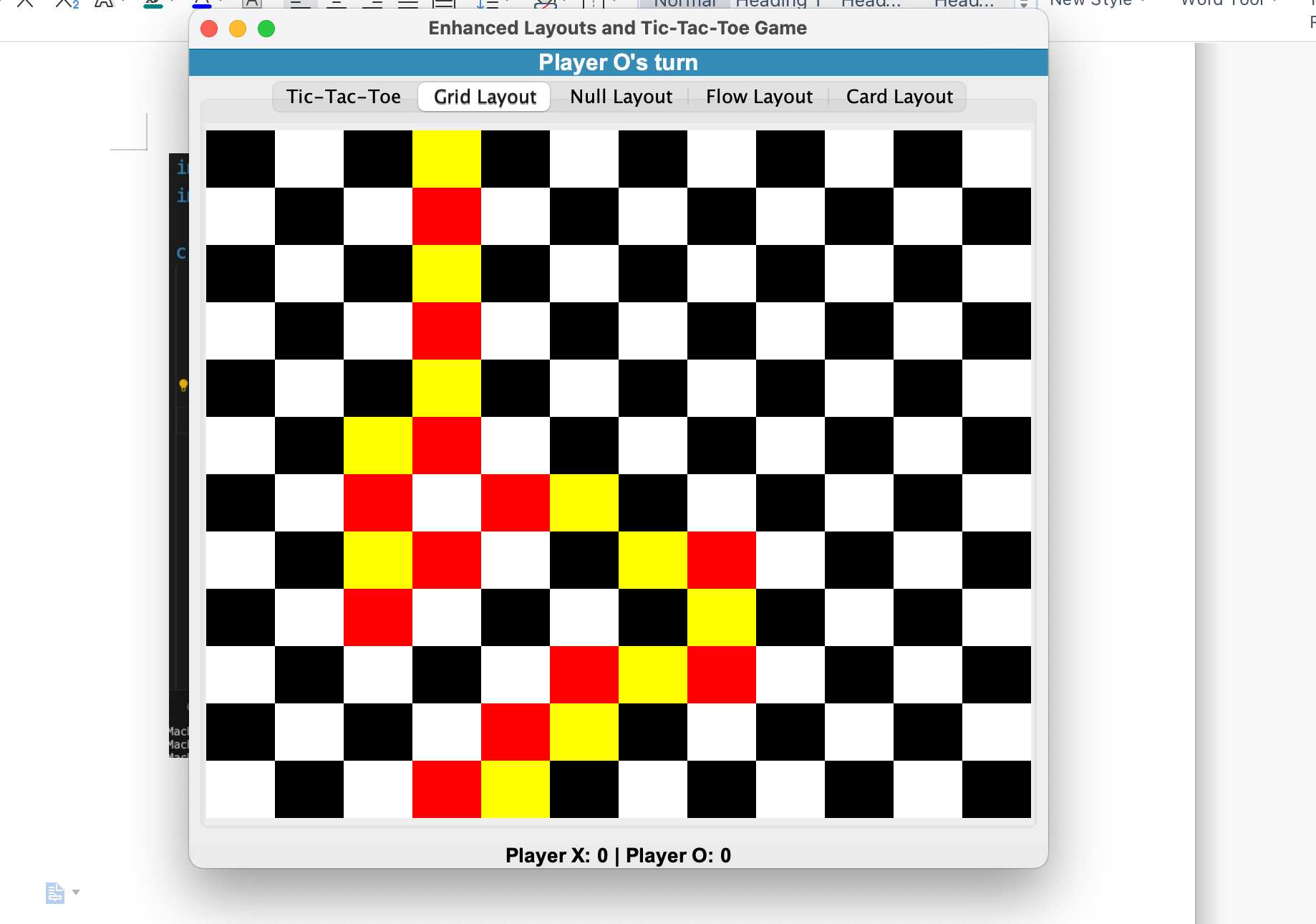
1.First page is Tic\_Tac-Toe game



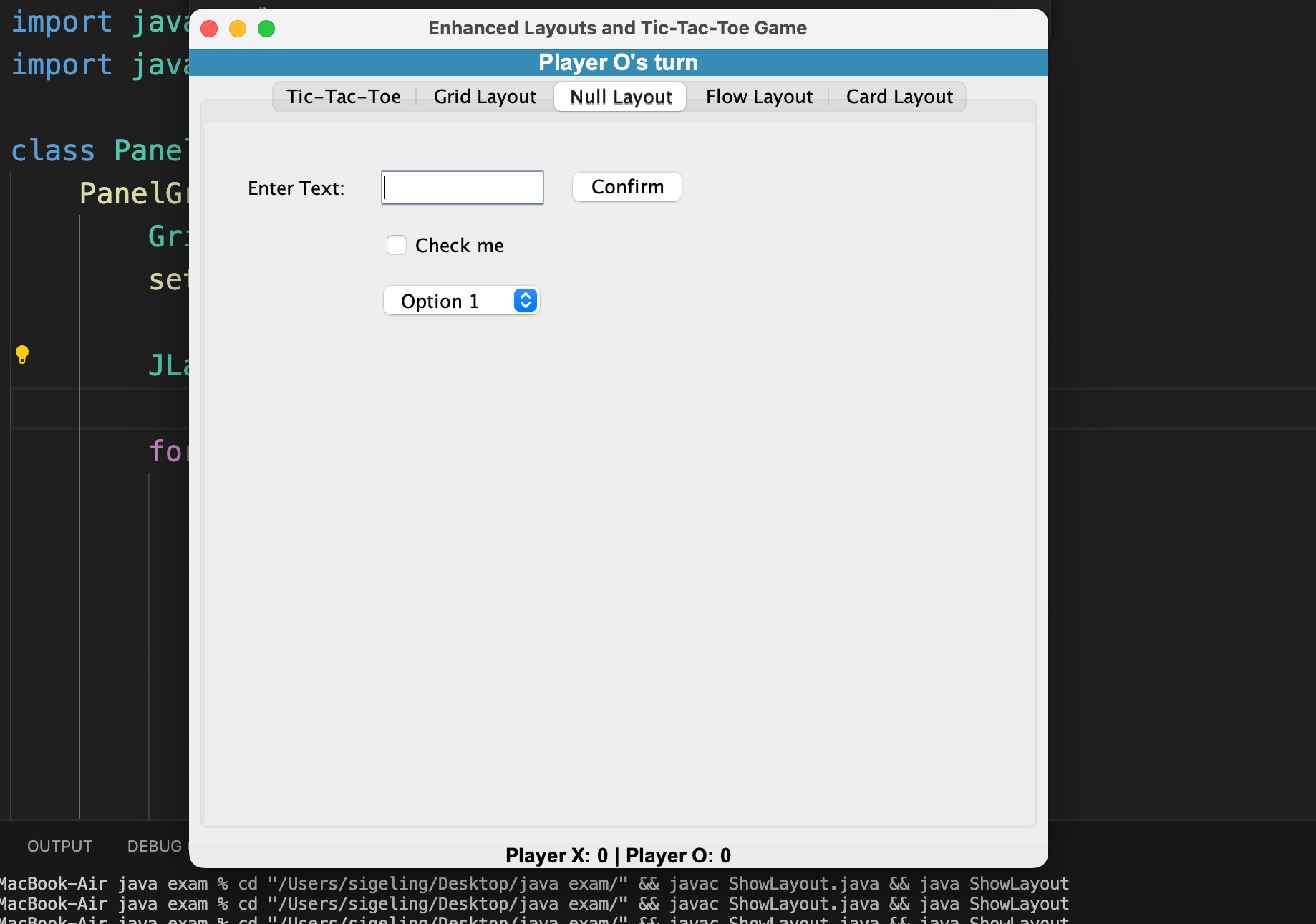
2 player can play this game and it will show you whice player win.



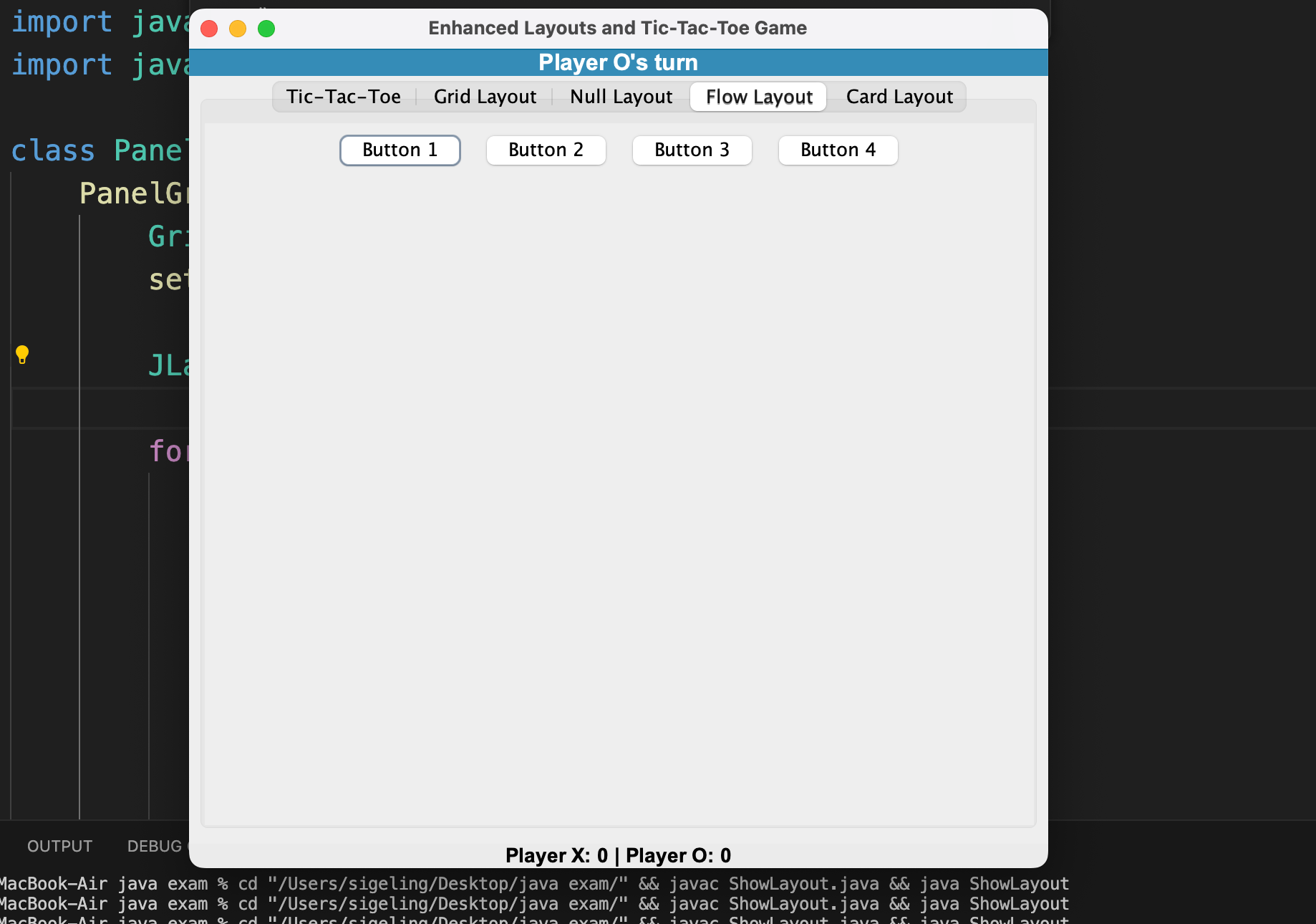
2: 2nd page is gird Layout if you are hover it will change the color



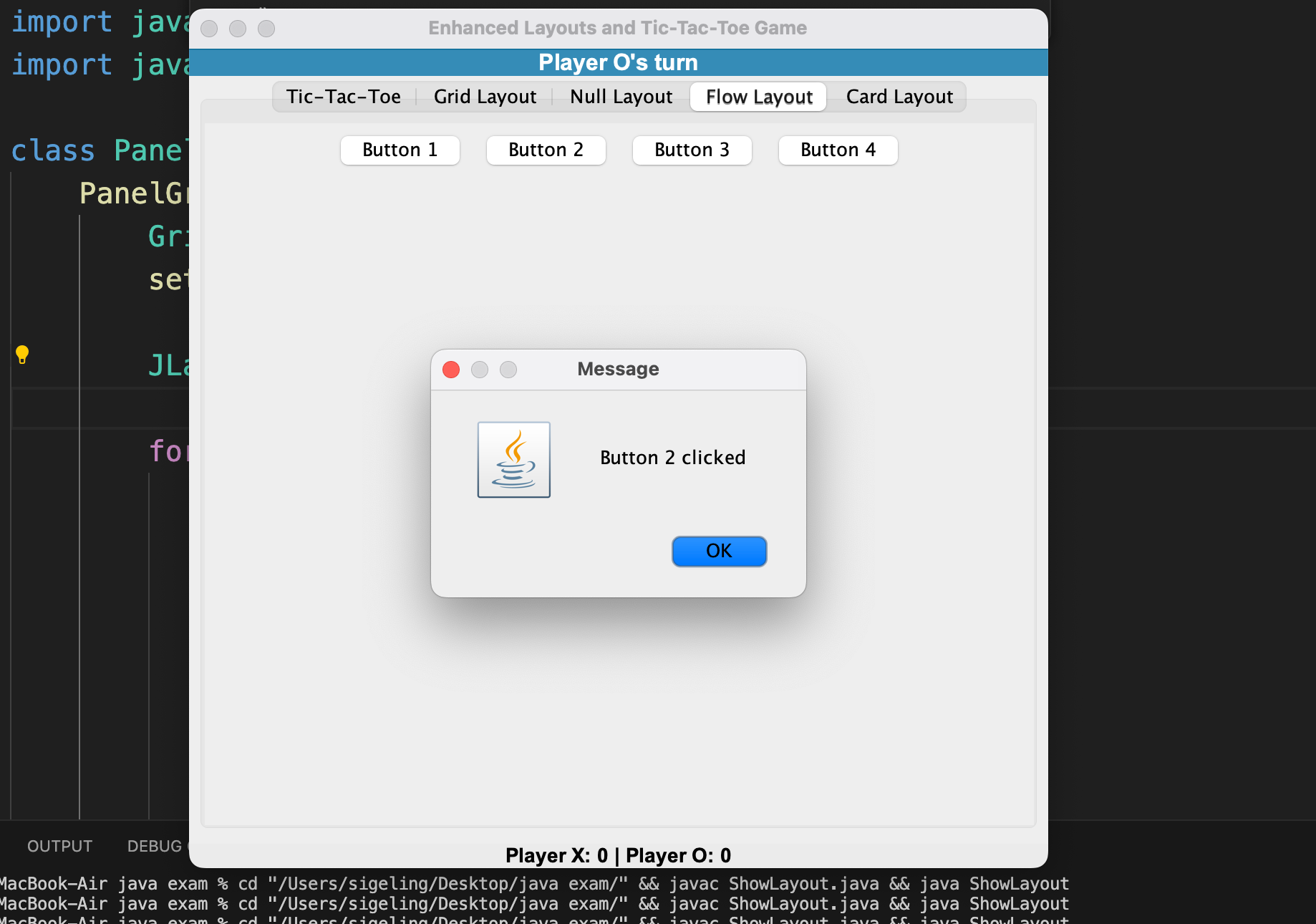
3: 3 page is text box confirm box , check me box, and drop down button,



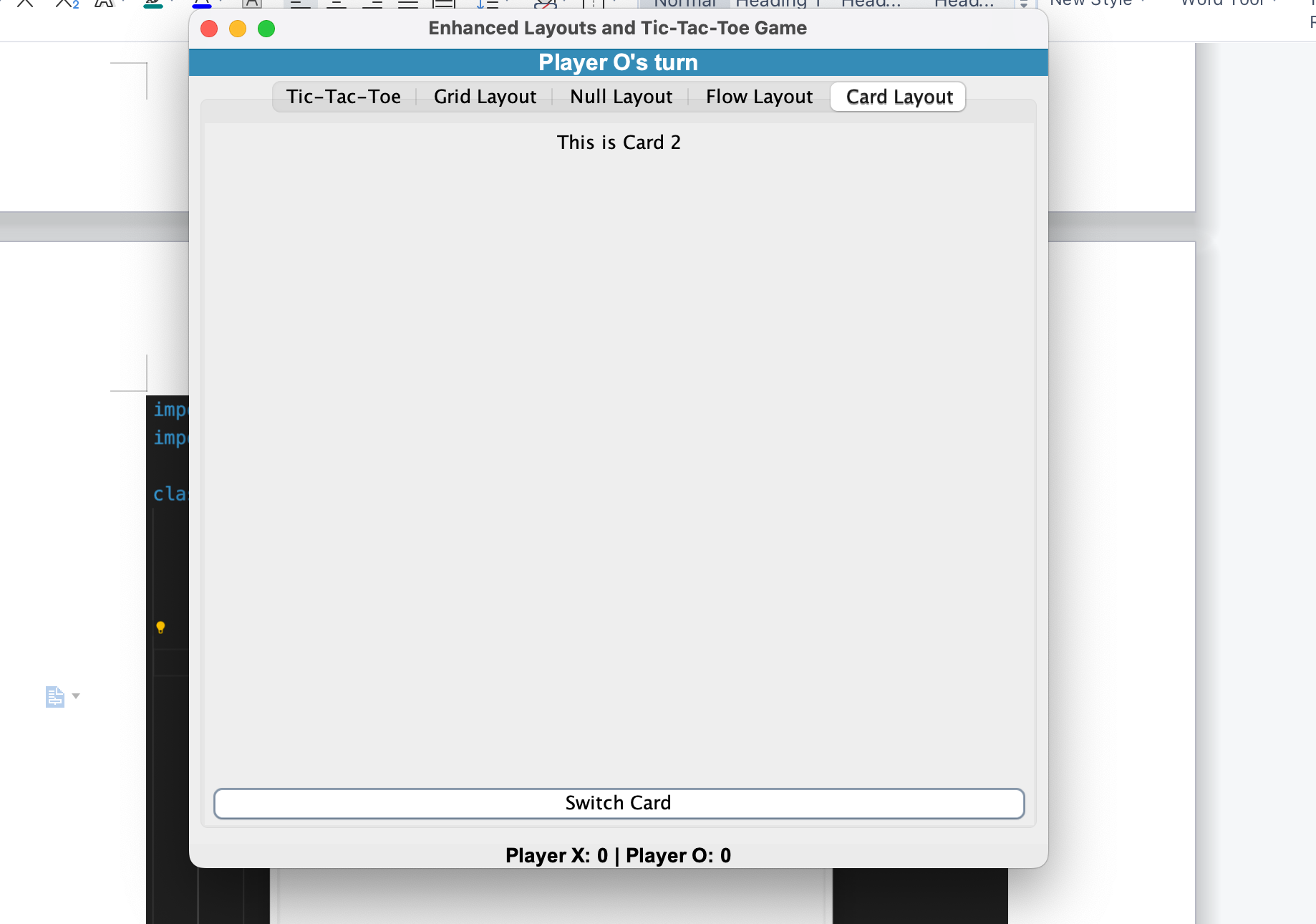
4: 4th page is 4 Button



5: if you press any button it wil open new window with text;



6:5th page is card layout and swith card:



this is my full code:

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

class PanelGridLayout extends JPanel {

PanelGridLayout() {

GridLayout grid = new GridLayout(12, 12);

setLayout(grid);

JLabel[][] label = new JLabel[12][12];

for (int i = 0; i < 12; i++) {

for (int j = 0; j < 12; j++) {

label[i][j] = new JLabel();

label[i][j].setOpaque(true);

if ((i + j) % 2 == 0)

label[i][j].setBackground(Color.black);

else

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

final int row = i;

final int col = j;

label[i][j].addMouseListener(new MouseAdapter() {

@Override

public void mouseEntered(MouseEvent e) {

label[row][col].setBackground(Color.gray);

}

@Override

public void mouseExited(MouseEvent e) {

if ((row + col) % 2 == 0)

label[row][col].setBackground(Color.red);

else

label[row][col].setBackground(Color.yellow);

}

});

add(label[i][j]);

}

}

}

}

class PanelNullLayout extends JPanel {

JButton button;

JTextField text;

JLabel label;

JCheckBox checkBox;

JComboBox<String> comboBox;

PanelNullLayout() {

setLayout(null);

label = new JLabel("Enter Text:");

text = new JTextField();

button = new JButton("Confirm");

checkBox = new JCheckBox("Check me");

comboBox = new JComboBox<>(new String[]{"Option 1", "Option 2", "Option 3"});

add(label);

add(text);

add(button);

add(checkBox);

add(comboBox);

label.setBounds(30, 30, 80, 30);

text.setBounds(120, 30, 120, 30);

button.setBounds(250, 30, 90, 30);

checkBox.setBounds(120, 70, 120, 30);

comboBox.setBounds(120, 110, 120, 30);

}

}

class PanelFlowLayout extends JPanel {

JButton button1, button2, button3, button4;

PanelFlowLayout() {

setLayout(new FlowLayout());

button1 = new JButton("Button 1");

button2 = new JButton("Button 2");

button3 = new JButton("Button 3");

button4 = new JButton("Button 4");

button1.addActionListener(e -> JOptionPane.showMessageDialog(this, "Button 1 clicked"));

button2.addActionListener(e -> JOptionPane.showMessageDialog(this, "Button 2 clicked"));

button3.addActionListener(e -> JOptionPane.showMessageDialog(this, "Button 3 clicked"));

button4.addActionListener(e -> JOptionPane.showMessageDialog(this, "Button 4 clicked"));

add(button1);

add(button2);

add(button3);

add(button4);

}

}

class PanelCardLayout extends JPanel {

CardLayout cardLayout;

JPanel cardPanel;

PanelCardLayout() {

cardLayout = new CardLayout();

cardPanel = new JPanel(cardLayout);

JPanel card1 = new JPanel();

card1.add(new JLabel("This is Card 1"));

JPanel card2 = new JPanel();

card2.add(new JLabel("This is Card 2"));

cardPanel.add(card1, "Card 1");

cardPanel.add(card2, "Card 2");

JButton switchButton = new JButton("Switch Card");

switchButton.addActionListener(e -> cardLayout.next(cardPanel));

setLayout(new BorderLayout());

add(cardPanel, BorderLayout.CENTER);

add(switchButton, BorderLayout.SOUTH);

}

}

class TicTacToePanel extends JPanel {

private JButton[][] buttons = new JButton[3][3];

private boolean playerXTurn = true; // X goes first

private int playerXScore = 0;

private int playerOScore = 0;

private JLabel statusLabel;

private JLabel scoreLabel;

TicTacToePanel(JLabel statusLabel, JLabel scoreLabel) {

this.statusLabel = statusLabel;

this.scoreLabel = scoreLabel;

setLayout(new GridLayout(3, 3));

initializeBoard();

}

private void initializeBoard() {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

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

buttons[i][j].setFont(new Font("Arial", Font.PLAIN, 40));

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

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

buttons[i][j].addActionListener(e -> handleButtonClick((JButton) e.getSource()));

add(buttons[i][j]);

}

}

}

private void handleButtonClick(JButton button) {

if (!button.getText().equals("")) {

return; // Button already clicked

}

if (playerXTurn) {

button.setText("X");

button.setForeground(Color.blue);

statusLabel.setText("Player O's turn");

} else {

button.setText("O");

button.setForeground(Color.red);

statusLabel.setText("Player X's turn");

}

playerXTurn = !playerXTurn;

checkForWin();

}

private void checkForWin() {

String[][] board = new String[3][3];

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

board[i][j] = buttons[i][j].getText();

}

}

// Check rows, columns, and diagonals

if (checkLine(board[0][0], board[0][1], board[0][2]) ||

checkLine(board[1][0], board[1][1], board[1][2]) ||

checkLine(board[2][0], board[2][1], board[2][2]) ||

checkLine(board[0][0], board[1][0], board[2][0]) ||

checkLine(board[0][1], board[1][1], board[2][1]) ||

checkLine(board[0][2], board[1][2], board[2][2]) ||

checkLine(board[0][0], board[1][1], board[2][2]) ||

checkLine(board[0][2], board[1][1], board[2][0])) {

if (!playerXTurn) {

statusLabel.setText("Player X wins!");

playerXScore++;

} else {

statusLabel.setText("Player O wins!");

playerOScore++;

}

updateScore();

disableBoard();

} else if (isBoardFull()) {

statusLabel.setText("It's a tie!");

}

}

private boolean checkLine(String s1, String s2, String s3) {

return !s1.equals("") && s1.equals(s2) && s2.equals(s3);

}

private boolean isBoardFull() {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

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

return false;

}

}

}

return true;

}

private void disableBoard() {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

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

}

}

}

private void updateScore() {

scoreLabel.setText("Player X: " + playerXScore + " | Player O: " + playerOScore);

}

public void resetBoard() {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

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

buttons[i][j].setEnabled(true);

}

}

playerXTurn = true;

statusLabel.setText("Player X's turn");

}

}

// class ShowLayout extends JFrame {

// private JLabel statusLabel;

// private JLabel scoreLabel;

// private TicTacToePanel ticTacToePanel;

// private PanelGridLayout panelGrid;

// private PanelNullLayout panelNull;

// private PanelFlowLayout panelFlow;

// private PanelCardLayout panelCard;

// ShowLayout() {

// setTitle("Enhanced Layouts and Tic-Tac-Toe Game");

// setLayout(new BorderLayout());

// statusLabel = new JLabel("Player X's turn", SwingConstants.CENTER);

// statusLabel.setFont(new Font("Arial", Font.BOLD, 16));

// statusLabel.setOpaque(true);

// statusLabel.setBackground(Color.cyan);

// add(statusLabel, BorderLayout.NORTH);

// scoreLabel = new JLabel("Player X: 0 | Player O: 0", SwingConstants.CENTER);

// scoreLabel.setFont(new Font("Arial", Font.BOLD, 14));

// scoreLabel.setOpaque(true);

// scoreLabel.setBackground(Color.lightGray );

// add(scoreLabel, BorderLayout.SOUTH);

// // Creating instances of different panels

// ticTacToePanel = new TicTacToePanel(statusLabel, scoreLabel);

// panelGrid = new PanelGridLayout();

// panelNull = new PanelNullLayout();

// panelFlow = new PanelFlowLayout();

// panelCard = new PanelCardLayout();

// // Adding panels to a tabbed pane

// JTabbedPane tabbedPane = new JTabbedPane();

// tabbedPane.addTab("Tic-Tac-Toe", ticTacToePanel);

// tabbedPane.addTab("Grid Layout", panelGrid);

// tabbedPane.addTab("Null Layout", panelNull);

// tabbedPane.addTab("Flow Layout", panelFlow);

// tabbedPane.addTab("Card Layout", panelCard);

// add(tabbedPane, BorderLayout.CENTER);

// setSize(600, 600);

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

// setLocationRelativeTo(null);

// setVisible(true);

// }

// public static void main(String[] args) {

// SwingUtilities.invokeLater(() -> new ShowLayout());

// }

// }

public class ShowLayout extends JFrame {

private JLabel statusLabel;

private JLabel scoreLabel;

private TicTacToePanel ticTacToePanel;

private PanelGridLayout panelGrid;

private PanelNullLayout panelNull;

private PanelFlowLayout panelFlow;

private PanelCardLayout panelCard;

ShowLayout() {

setTitle("Enhanced Layouts and Tic-Tac-Toe Game");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new BorderLayout());

// Status Label

statusLabel = new JLabel("Player X's turn", SwingConstants.CENTER);

statusLabel.setFont(new Font("Arial", Font.BOLD, 16));

statusLabel.setOpaque(true);

statusLabel.setBackground(new Color(53, 140, 183)); // Dark cyan

statusLabel.setForeground(Color.WHITE); // White text

add(statusLabel, BorderLayout.NORTH);

// Score Label

scoreLabel = new JLabel("Player X: 0 | Player O: 0", SwingConstants.CENTER);

scoreLabel.setFont(new Font("Arial", Font.BOLD, 14));

scoreLabel.setOpaque(true);

scoreLabel.setBackground(new Color(235, 235, 235)); // Light gray

// scoreLabel.setBackground(new Color(200 ,200 ,200));

add(scoreLabel, BorderLayout.SOUTH);

// Creating instances of different panels

ticTacToePanel = new TicTacToePanel(statusLabel, scoreLabel);

panelGrid = new PanelGridLayout();

panelNull = new PanelNullLayout();

panelFlow = new PanelFlowLayout();

panelCard = new PanelCardLayout();

// Adding panels to a tabbed pane

JTabbedPane tabbedPane = new JTabbedPane();

tabbedPane.addTab("Tic-Tac-Toe", ticTacToePanel);

tabbedPane.addTab("Grid Layout", panelGrid);

tabbedPane.addTab("Null Layout", panelNull);

tabbedPane.addTab("Flow Layout", panelFlow);

tabbedPane.addTab("Card Layout", panelCard);

add(tabbedPane, BorderLayout.CENTER);

// Setting window size and visibility

setSize(600, 600);

setLocationRelativeTo(null); // Center window

setVisible(true);

}

public static void main(String[] args) {

SwingUtilities.invokeLater(() -> new ShowLayout());

}

}