Tic-Tac-Toe Game

Game Developer

Vũ Thành Đạt – ITITIU21041

Nguyễn Thiện Tín - IEIE

NAME	ID	CONTRIBUTION
Vũ Thành Đạt	ITITIU21041	Game Developer Write Report Presenter
Nguyễn Thiện Tín	IEIE	Game Developer

Tic-Tac-Toe Game Development Report

Objective:
Develop a simple Tic-Tac-Toe game using Java with a graphical user interface (GUI).

Overview:
The project involves creating a two-player Tic-Tac-Toe game where players take turns to mark X or O on a 3x3 grid. The game checks for winning conditions after each move and announces the winner. Additionally, a "Play Again" button is provided to reset the game board for a new round.

Key Components:

1. **JFrame**: Main window to hold all components.
2. **JPanel**: Containers for organizing the layout.
    - title_panel: Displays the game title and current player's turn.

- button_panel: Holds the 3x3 grid of buttons.
- bottom_panel: Contains the "Play Again" button.

3. **JLabel**: Displays game status and current player's turn.
4. **JButton**: Represents each cell in the Tic-Tac-Toe grid and the "Play Again" button.

Implementation Details:

1. **Frame Setup:**
   - Created a JFrame named frame with a size of 800x800 pixels.
   - Set the background color to dark gray (RGB: 50, 50, 50).
2. **Title Panel:**
   - Created a JPanel named title_panel with a BorderLayout.
   - Added a JLabel named textfield to display the game title and current turn.
3. **Button Panel:**
   - Created a JPanel named button_panel with a 3x3 GridLayout.
   - Added nine JButtons to the panel, each representing a cell in the Tic-Tac-Toe grid.
   - Set font and styles for the buttons to ensure visibility and aesthetics.
4. **Bottom Panel:**
   - Created a JPanel named bottom_panel with a BorderLayout.
   - Added a JButton named playAgainButton for resetting the game.

5. **Event Handling:**
    - Implemented the ActionListener interface to handle button clicks.
    - Defined actions for marking X or O, switching turns, and checking win conditions.
    - Added functionality to reset the game board when the "Play Again" button is pressed.
6. **Game Logic:**
    - Used a boolean variable player1_turn to track the current player.
    - Implemented the check method to verify winning conditions after each move.
    - Defined methods xWins and oWins to handle the win scenarios and disable further moves.
    - Added a resetGame method to clear the board and enable buttons for a new game.

Results:

- The game successfully alternates turns between two players.
- It correctly identifies win conditions and displays the winner.
- The "Play Again" button resets the game board for another round.

Future Improvements:

- Implement a scoreboard to keep track of wins for each player.
- Add a feature for draw conditions when all cells are filled without a winner.

- Enhance the UI with more polished graphics and animations.

Conclusion:

The Tic-Tac-Toe game project provides a simple yet effective example of GUI programming in Java. It demonstrates the use of basic components like JFrame, JPanel, JLabel, and JButton, combined with event handling and game logic implementation. The addition of a "Play Again" button enhances the user experience, allowing for multiple rounds of play without restarting the application.

Code Example:

```java
// Full code provided earlier, summarized here for reference


public class TicTacTe implements AncestorListener {
    // Fields and constructor setup...

    @Override
    public void actionPerformed(ActionEvent e) {
        // Handle button clicks and game logic...
    }

    public void firstTurn() {
        // Initialize first turn...
    }

    public void check() {
        // Check win conditions...
    }

    public void xWins(int a, int b, int c) {
        // Handle X win scenario...
    }

    public void oWins(int a, int b, int c) {
        // Handle O win scenario...
    }

    public void resetGame() {
        // Reset the game board...
    }

    Run | Debug
    public static void main(String[] args) {
        new TicTacToe();
    }
}
```