Tic-Tac-Toe Game Documentation

1.Introduction

The Tic-Tac-Toe game is a simple Java application that allows two players to take turns marking spaces in a 3x3 grid to achieve a winning pattern. This documentation provides an overview of the project structure, key functionalities, and technical aspects.

2.Abstraction

2.1 Overview

The project follows a simple model-view-controller (MVC) architecture, where the TicTacToe class serves as the controller handling user input and game logic. The GUI elements (view) are created using Swing components, and the game state is maintained through instance variables.

2.2 Key Abstractions

- TicTacToe Class: The main class that initializes the game, sets up the GUI, and manages player turns.
- buttons Array: Represents the 3x3 grid buttons for user input.
- player1_turn Boolean: Indicates whether it is Player 1's turn.

3. Key Functionalities

3.1 GUI Setup

- The game window is created using JFrame and organized with JPanel containers.
- A title panel contains a JLabel for the game title, and a button panel holds the 3x3 grid of buttons.

3.2 User Interaction

- Players take turns clicking on the buttons to mark their moves (X or O).
- The game checks for wins or draws after each move.

3.3 Win Detection

- The check() method checks for winning conditions after each move.
- If a player achieves a winning pattern, the corresponding xWins or oWins method is called to highlight the winning buttons and declare the winner.

4. Methodology / Technical Part

4.1 Event Handling

- **The ActionListener interface is implemented to handle button clicks using the actionPerformed method.**
- **Players' moves are processed based on the current turn and update the GUI accordingly.**

4.2 Random First Turn

- **The firstTurn() method uses Thread.sleep to introduce a delay before randomly determining which player starts.**

4.3 Win Checking

- **The check() method checks all possible winning combinations for both X and O after each move.**

4.4 GUI Styling

- **Fonts, colors, and button sizes are configured to enhance the visual appeal of the game.**

5. What Is Missing

5.1 User Feedback

- **Provide feedback to the user after a game is finished (e.g., option to restart).**

5.2 Scalability

- **The current implementation supports a 3x3 grid. Consider enhancing the code to handle different grid sizes.**