**Pygame Tic-Tac-Toe Project Documentation**

**Overview**

This documentation covers the enhanced features implemented in a Python Tic-Tac-Toe game using Pygame. The base code was extended with multiple interactive features to improve user experience and gameplay functionality.

**Implemented Features**

**1. Sound Effects System**

The sound effects feature provides audio feedback for different game actions, enhancing the overall player experience.

**Line 3:** pygame.mixer.init() - Initializes the pygame mixer module to enable sound functionality in the game.

**Lines 5-9:** Sound dictionary creation - Creates a dictionary containing three Sound objects that load different audio files:

* "move": Plays when a player makes a valid move
* "win": Plays when a player wins a round
* "draw": Plays when the game ends in a tie

**Line 158:** sound["move"].play() - Triggers the move sound effect whenever a player successfully places their mark (X or O) on the board.

**Line 243:** sound["win"].play() - Plays the victory sound when the check\_win() function returns True, indicating a player has won.

**Line 250:** sound["draw"].play() - Plays the draw sound when is\_board\_full() returns True and no winner is found, signaling a tie game.

**2. Theme System with Dropdown Menu**

This feature allows players to customize the visual appearance of the game through different color themes.

**Lines 25-28:** Theme dictionary definition - Creates a themes dictionary containing three predefined color schemes:

* "classic": Light gray background with dark gray lines
* "futuristic": Purple background with cyan lines
* "nature": Green background with yellow-green lines

**Lines 35-45:** Global theme variables and color application function - Defines color variables and the theme\_colors() function that applies the selected theme's colors to the game elements.

**Lines 48-56:** Theme menu constants and button setup - Defines font objects and creates the themes button rectangle for the dropdown menu interface.

**Lines 62-71:** Theme dropdown panel creation - The theme\_item\_buttons() function calculates the dropdown panel size and creates clickable rectangles for each theme option.

**Lines 73-95:** Menu bar drawing function - draw\_menu\_bar() renders the themes button and, when activated, displays the dropdown panel with all available theme options.

**Lines 225-234:** Theme selection event handling - Detects mouse clicks on theme buttons and applies the selected theme by calling theme\_colors() and restart().

**3. Score Tracking System**

Maintains running tallies of wins for both players throughout multiple games.

**Lines 99-100:** Score variables initialization - Creates x\_wins and o\_wins global variables to track each player's victory count.

**Lines 244-248:** Score increment logic - When a player wins, increments the appropriate win counter (x\_wins for player X, o\_wins for player O).

**Lines 104-109:** Score display formatting - Creates a formatted text string showing current turn and both players' scores, then renders it using the big font.

**4. Status Bar Implementation**

Provides a dedicated area at the bottom of the screen for displaying game information.

**Lines 97-98:** Status bar area definition - Creates a STATUS\_BAR rectangle at the bottom of the screen with calculated height and positioning.

**Lines 102-119:** Status bar drawing function - draw\_status\_bar() fills the status area with white background, displays turn indicator and scores, and includes a restart hint.

**Line 110-113:** Centered text positioning - Calculates proper x and y coordinates to center the status text within the status bar area.

**Lines 117-119:** Restart hint display - Shows small text in the bottom-right corner instructing players how to restart the game.

**5. Restart Functionality**

Allows players to reset the game state and start fresh without closing the application.

**Lines 194-200:** Restart function definition - restart() function resets the visual state by clearing the screen, redrawing menu elements, and clearing the game board.

**Lines 258-268:** Keyboard restart handling - Detects when the 'R' key is pressed and calls the restart function while resetting game variables to initial states.

**Lines 196-200:** Board state reset - Loops through all board positions and sets them to None, effectively clearing all player moves.

**6. Enhanced User Interface**

Improves the overall visual presentation and user interaction capabilities.

**Lines 73-95:** Interactive menu system - Creates a responsive themes button that toggles a dropdown menu when clicked.

**Lines 83-94:** Dynamic button highlighting - Adds visual feedback by changing button colors and displaying selection markers for the current theme.

**Lines 104-119:** Comprehensive status display - Shows multiple pieces of information including current player turn, win counts, and helpful instructions in an organized layout.

**Lines 225-234:** Responsive click handling - Manages different types of user interactions including theme selection and game board clicks with appropriate visual feedback.

**Technical Implementation Notes**

* The sound system requires audio files to be present in a "sounds" directory
* Theme colors are applied globally and affect all visual game elements
* Score tracking persists across game restarts but resets when the application is closed
* The status bar automatically adjusts text positioning based on content length
* Restart functionality preserves theme selection and score history

**Sources for Feature Implementation**

**Sound Effects:**

* Pygame Official Documentation: Audio Integration Guide
* Python Code Tutorials: "How to Add Sound Effects to your Python Game"
* GameDev Academy: "Pygame Sound Tutorial - Complete Guide"

**Theme System:**

* Pygame-menu Documentation: "Creating themes"
* Stack Overflow: Pygame color theme implementation discussions
* Real Python: "PyGame: A Primer on Game Programming in Python"

**Score Tracking:**

* GeeksforGeeks: "How can i add a Score in pygame?"
* YouTube Tutorials: "Adding Score Text" and "Creating a scoring system in Pygame"
* LeMaster Tech: "Python Pygame Tutorial - Creating and Tracking Score"

**Status Bar and UI:**

* CodersLegacy: "Pygame RPG - Status Bar"
* Pygame Tutorial Series: "Adding Text and Displaying Score"
* GeeksforGeeks: "Creating start Menu in Pygame"

**Restart Functionality:**

* Stack Overflow: "How Can I make the restart function in pygame?"
* Reddit: "How do add a 'restart' function in Pygame?"
* YouTube: "Pygame Tutorial - Lives and Restart Screen"