**Tic Tac Toe Game Documentation**

**Overview**

This Pygame-based Tic Tac Toe game allows two players to take turns placing "X" and "O" on a 3x3 grid. The game checks for a winner or a draw after each move and updates the display accordingly.

**Features**

* **Interactive GUI**: Provides a visual representation of the game board with clickable cells.
* **Dynamic Feedback**: Updates the display to show the current player's turn, winner, or a draw.
* **Reset Functionality**: Automatically resets the game after a win or draw.

**Requirements**

* Python 3.x
* Pygame library

**Installation of Required Libraries**

To install the Pygame library, use the following command:

pip install pygame

**Code Walkthrough**

**1. Importing Libraries**

import pygame as pg, sys

from pygame.locals import \*

import time

* **pygame**: The Pygame library for creating the graphical user interface.
* **sys**: Provides access to system-specific parameters and functions.
* **time**: Used for time-related functions such as delays.

**2. Initializing Global Variables**

XO = 'x'

winner = None

draw = False

width = 400

height = 400

white = (255, 255, 255)

line\_color = (10, 10, 10)

TTT = [[None]\*3, [None]\*3, [None]\*3]

* **XO**: Keeps track of the current player ("x" or "o").
* **winner**: Stores the winner of the game.
* **draw**: Indicates if the game is a draw.
* **width and height**: Dimensions of the game window.
* **white and line\_color**: Colors used in the game.
* **TTT**: A 3x3 list representing the game board.

**3. Initializing Pygame Window**

pg.init()

fps = 30

CLOCK = pg.time.Clock()

screen = pg.display.set\_mode((width, height + 100), 0, 32)

pg.display.set\_caption("Tic Tac Toe")

* **pg.init()**: Initializes all imported Pygame modules.
* **fps**: Frames per second for the game loop.
* **CLOCK**: Clock object to manage the frame rate.
* **screen**: Creates the game window.
* **pg.display.set\_caption()**: Sets the window title.

**4. Loading and Resizing Images**

opening = pg.image.load('tic tac opening.png')

x\_img = pg.image.load('x.png')

o\_img = pg.image.load('o.png')

x\_img = pg.transform.scale(x\_img, (80, 80))

o\_img = pg.transform.scale(o\_img, (80, 80))

opening = pg.transform.scale(opening, (width, height + 100))

* **pg.image.load()**: Loads images for the game.
* **pg.transform.scale()**: Resizes images to fit the game board.

**5. Functions**

* **game\_opening()**: Displays the opening screen, draws the game grid, and updates the status.

def game\_opening():

screen.blit(opening, (0, 0))

pg.display.update()

time.sleep(1)

screen.fill(white)

pg.draw.line(screen, line\_color, (width / 3, 0), (width / 3, height), 7)

pg.draw.line(screen, line\_color, (width / 3 \* 2, 0), (width / 3 \* 2, height), 7)

pg.draw.line(screen, line\_color, (0, height / 3), (width, height / 3), 7)

pg.draw.line(screen, line\_color, (0, height / 3 \* 2), (width, height / 3 \* 2), 7)

draw\_status()

**draw\_status()**: Updates the game status message (current player's turn, winner, or draw).

def draw\_status():

global draw

if winner is None:

message = XO.upper() + "'s Turn"

else:

message = winner.upper() + " won!"

if draw:

message = 'Game Draw!'

font = pg.font.Font(None, 30)

text = font.render(message, 1, (255, 255, 255))

screen.fill((0, 0, 0), (0, 400, 500, 100))

text\_rect = text.get\_rect(center=(width / 2, 500 - 50))

screen.blit(text, text\_rect)

pg.display.update()

* **check\_win()**: Checks for winning combinations in rows, columns, and diagonals.

def check\_win():

global TTT, winner, draw

for row in range(0, 3):

if ((TTT[row][0] == TTT[row][1] == TTT[row][2]) and (TTT[row][0] is not None)):

winner = TTT[row][0]

pg.draw.line(screen, (250, 0, 0), (0, (row + 1) \* height / 3 - height / 6), (width, (row + 1) \* height / 3 - height / 6), 4)

break

for col in range(0, 3):

if (TTT[0][col] == TTT[1][col] == TTT[2][col]) and (TTT[0][col] is not None):

winner = TTT[0][col]

pg.draw.line(screen, (250, 0, 0), ((col + 1) \* width / 3 - width / 6, 0), ((col + 1) \* width / 3 - width / 6, height), 4)

break

if (TTT[0][0] == TTT[1][1] == TTT[2][2]) and (TTT[0][0] is not None):

winner = TTT[0][0]

pg.draw.line(screen, (250, 70, 70), (50, 50), (350, 350), 4)

if (TTT[0][2] == TTT[1][1] == TTT[2][0]) and (TTT[0][2] is not None):

winner = TTT[0][2]

pg.draw.line(screen, (250, 70, 70), (350, 50), (50, 350), 4)

if all([all(row) for row in TTT]) and winner is None:

draw = True

draw\_status()

* **drawXO(row, col)**: Draws the "X" or "O" on the game board.

def drawXO(row, col):

global TTT, XO

if row == 1:

posx = 30

if row == 2:

posx = width / 3 + 30

if row == 3:

posx = width / 3 \* 2 + 30

if col == 1:

posy = 30

if col == 2:

posy = height / 3 + 30

if col == 3:

posy = height / 3 \* 2 + 30

TTT[row - 1][col - 1] = XO

if XO == 'x':

screen.blit(x\_img, (posy, posx))

XO = 'o'

else:

screen.blit(o\_img, (posy, posx))

XO = 'x'

pg.display.update()

* **userClick()**: Handles mouse clicks to place an "X" or "O" and checks for a win or draw.

def userClick():

x, y = pg.mouse.get\_pos()

if x < width / 3:

col = 1

elif x < width / 3 \* 2:

col = 2

elif x < width:

col = 3

else:

col = None

if y < height / 3:

row = 1

elif y < height / 3 \* 2:

row = 2

elif y < height:

row = 3

else:

row = None

if row and col and TTT[row - 1][col - 1] is None:

drawXO(row, col)

check\_win()

* **reset\_game()**: Resets the game board and restarts the game after a win or draw.

def reset\_game():

global TTT, winner, XO, draw

time.sleep(3)

XO = 'x'

draw = False

game\_opening()

winner = None

TTT = [[None] \* 3, [None] \* 3, [None] \* 3]

**6. Main Game Loop**

game\_opening()

while True:

for event in pg.event.get():

if event.type == QUIT:

pg.quit()

sys.exit()

elif event.type == MOUSEBUTTONDOWN:

userClick()

if winner or draw:

reset\_game()

pg.display.update()

CLOCK.tick(fps)

* **game\_opening()**: Calls the function to display the initial screen and grid.
* **while True**: Main game loop that handles events and updates the display.
* **userClick()**: Handles player input.
* **reset\_game()**: Resets the game if needed.

**Usage**

1. Run the script to start the game.
2. Click on the cells to place "X" or "O".
3. The game will automatically detect a winner or a draw and reset after 3 seconds.