TIC TAC TOE GAME IN PHYTON

 Using Pygame Library.

Code Initialization

```
# Importing necessary modules
import pygame
from pygame.locals import *

# Initializing Pygame
pygame.init()

# Setting up the game window dimensions
screen_height = 300
screen_width = 300
line_width = 6
screen = pygame.display.set_mode((screen_width, screen_height))
pygame.display.set_caption('Tic Tac Toe')
```

 The code begins by importing the Pygame library, which is used for game development. Pygame is initialized using ‘pygame.init()’. The game window dimensions, line width and window caption are set.

Colors and Font

```
# Defining colors
red = (255, 0, 0)
green = (0, 255, 0)
blue = (0, 0, 255)

# Defining font
font = pygame.font.SysFont(None, 40)
```

 RGB values are defined for three colors: red, green, and blue. A font obejct is created using ‘pygame.font.SysFont’ with a font size of 40.

Game Variables

```
# Game variables
clicked = False
player = 1
pos = (0, 0)
markers = []
game_over = False
winner = 0
```

 Variables are initialized to keep track of mouse clicks, current player, mouse position, game grid (markers), game over status, and winner.

Play Again Rectangle

```
# Setting up a rectangle for "Play Again" Option
again_rect = Rect(screen_width // 2 - 80, screen_height // 2, 160, 50)
```

 A rectangle (‘again_rect’) is defined to represent the “Play Again” button’s position and dimensions.

Creating the Game Grid

```
# Creating an empty 3×3 list to represent the grid
for x in range(3):
    row = [0] * 3
    markers.append(row)
```

 A 3x3 grid is created using a nested loop, and the **‘markers’** list is populated with rows of zeros, indicationg an empty grid.

Drawing the Game Board

```
def draw_board():
    # Drawing the game board
    bg = (255, 255, 210)
    grid = (50, 50, 50)
    screen.fill(bg)
    for x in range(1, 3):
        pygame.draw.line(screen, grid, (0, 100 * x), (screen_width, 10
        pygame.draw.line(screen, grid, (100 * x, 0), (100 * x, screen_
```

<p> The ‘draw_board’ function is defined to draw the game board on the screen. It fills the screen with a light bacground color (‘bg’) and draws grid lines using ‘pygame.draw.line’.p>

Drawing Game Markers

```
def draw_markers():
    # Drawing X and O markers
    x_pos = 0
    for x in markers:
        y_pos = 0
        for y in x:
            if y == 1:
                pygame.draw.line(screen, red, (x_pos * 100 + 15, y_pos
                pygame.draw.line(screen, red, (x_pos * 100 + 85, y_pos
            if y == -1:
                pygame.draw.circle(screen, green, (x_pos * 100 + 50, y
            y_pos += 1
        x_pos += 1
```

<p> The ‘draw_markers’ function is defined to draw X and O markers based on the values in the ‘markers’ grid. It uses ‘pygame.draw.line’ for X markers and ‘pygame.draw.circle’ for O markers. p>

Checking Game Over

```
def check_game_over():
    # Checking for win or tie
    global game_over
    global winner

    # Checking columns and rows
    x_pos = 0
    for x in markers:
        if sum(x) == 3:
            winner = 1
            game_over = True
        if sum(x) == -3:
            winner = 2
            game_over = True

        if markers[0][x_pos] + markers[1][x_pos] + markers[2][x_pos]
            winner = 1
            game_over = True
        if markers[0][x_pos] + markers[1][x_pos] + markers[2][x_pos]
            winner = 2
            game_over = True
        x_pos += 1

    # Checking diagonals
    if markers[0][0] + markers[1][
        winner = 1
```

<p> The ‘check_game_over’ function determines whether the game is over, and if so, who the winner is. Itchecks for winning conditions in rows, columns, and diagonals. If a player wins or of there’s a tie, it sets the ‘game_over’ and ‘winner’ variables accordingly. p>

THANK YOU!