**TIC -TAC-TOE GAME**

*Code:*

from tkinter import \*

import random

def next\_turn(row, column):

    global player

    if buttons[row][column]['text'] == "" and check\_winner() is False:

        if player == players[0]:

            buttons[row][column]['text'] = player

            if check\_winner() is False:

                player = players[1]

                label.config(text=(players[1]+" turn"))

            elif check\_winner() is True:

                label.config(text=(players[0]+" wins"))

            elif check\_winner() == "Tie":

                label.config(text="Tie!")

        else:

            buttons[row][column]['text'] = player

            if check\_winner() is False:

                player = players[0]

                label.config(text=(players[0]+" turn"))

            elif check\_winner() is True:

                label.config(text=(players[1]+" wins"))

            elif check\_winner() == "Tie":

                label.config(text="Tie!")

def check\_winner():

    for row in range(3):

        if buttons[row][0]['text'] == buttons[row][1]['text'] == buttons[row][2]['text'] != "":

            buttons[row][0].config(bg="green")

            buttons[row][1].config(bg="green")

            buttons[row][2].config(bg="green")

            return True

    for column in range(3):

        if buttons[0][column]['text'] == buttons[1][column]['text'] == buttons[2][column]['text'] != "":

            buttons[0][column].config(bg="green")

            buttons[1][column].config(bg="green")

            buttons[2][column].config(bg="green")

            return True

    if buttons[0][0]['text'] == buttons[1][1]['text'] == buttons[2][2]['text'] != "":

        buttons[0][0].config(bg="green")

        buttons[1][1].config(bg="green")

        buttons[2][2].config(bg="green")

        return True

    elif buttons[0][2]['text'] == buttons[1][1]['text'] == buttons[2][0]['text'] != "":

        buttons[0][2].config(bg="green")

        buttons[1][1].config(bg="green")

        buttons[2][0].config(bg="green")

        return True

    elif empty\_spaces() is False:

        for row in range(3):

            for column in range(3):

                buttons[row][column].config(bg="yellow")

        return "Tie"

    else:

        return False

def empty\_spaces():

    spaces = 9

    for row in range(3):

        for column in range(3):

            if buttons[row][column]['text'] != "":

                spaces -= 1

    if spaces == 0:

        return False

    else:

        return True

def new\_game():

    global player

    player = random.choice(players)

    label.config(text=player+" turn")

    for row in range(3):

        for column in range(3):

            buttons[row][column].config(text="",bg="#F0F0F0​")

window = Tk()

window.title("Tic-Tac-Toe")

players = ["x","o"]

player = random.choice(players)

buttons = [[0,0,0],

           [0,0,0],

           [0,0,0]]

label = Label(text=player + " turn", font=('consolas',40))

label.pack(side="top")

reset\_button = Button(text="restart", font=('consolas',20), command=new\_game)

reset\_button.pack(side="top")

frame = Frame(window)

frame.pack()

for row in range(3):

    for column in range(3):

        buttons[row][column] = Button(frame, text="",font=('consolas',40), width=5, height=2,

                                      command= lambda row=row, column=column: next\_turn(row,column))

        buttons[row][column].grid(row=row,column=column)

window.mainloop()

*OUTPUT:*

A screenshot of a cellphone

Description automatically generated\* A screenshot of a calculator

Description automatically generated A screenshot of a game

Description automatically generated A screenshot of a game

Description automatically generated-- A screenshot of a game

Description automatically generated A screenshot of a game

Description automatically generatedA screenshot of a game

Description automatically generated A screenshot of a game

Description automatically generated A screenshot of a game

Description automatically generated

Explanation:

**1. Imports and Initialization**

* from tkinter import \*: This line imports the tkinter library, which is used to create the graphical user interface (GUI) for the game.
* import random: This line imports the random module, which will be used to randomly select the starting player.
* players = ["x", "o"]: This creates a list containing the two possible players: "x" and "o".
* player = random.choice(players): This randomly selects one of the players from the players list to start the game.
* buttons = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]: This creates a 2D list to represent the game board. Each element in the list will hold a button object.

**2. GUI Setup**

* window = Tk(): This creates the main window for the game.
* window.title("Tic-Tac-Toe"): Sets the title of the window to "Tic-Tac-Toe".
* label = Label(...): Creates a label to display the current player's turn.
* frame = Frame(window): Creates a frame to hold the game board buttons.
* for row in range(3):: This loop iterates over each row of the game board.
  + for column in range(3):: This loop iterates over each column in the current row.
    - buttons[row][column] = Button(...): Creates a button object for the current cell and assigns it to the corresponding position in the buttons list. The command attribute of the button is set to the next\_turn function, which will be called when the button is clicked.
    - buttons[row][column].grid(...): Places the button in its correct position on the grid.

**3. Game Logic Functions**

* **next\_turn(row, column):**
  + Checks if the clicked cell is empty and if the game is not over.
  + Updates the cell with the current player's symbol.
  + Calls check\_winner() to determine if the game has been won.
  + If the game is not over, switches to the next player's turn and updates the label.
  + If the game is over, displays the winner or a tie message.
* **check\_winner():**
  + Checks for winning combinations in rows, columns, and diagonals.
  + If a winning combination is found, highlights the winning cells and returns True.
  + If the board is full and there is no winner, declares a tie and returns "Tie".
  + If no winner is found and the board is not full, returns False.
* **empty\_spaces():**
  + Counts the number of empty cells on the board.
  + Returns False if all cells are filled, otherwise returns True.
* **new\_game():**
  + Randomly selects a starting player.
  + Clears the game board and resets the label.

**4. Main Loop**

* window.mainloop(): Starts the GUI event loop, which listens for user interactions (e.g., button clicks) and updates the game accordingly.

This code provides a basic implementation of a Tic-Tac-Toe game with a graphical user interface.