REPORT FOR TIC-TAC-TOE GAME

As a project work for course

PYTHON PROGRAMMING (INT213)

Name: Prangon Kumar Das

Registration Number: 12000360

Program: B.Tech CSE

School: School of Computer Science and Engineering

Date of Submission: 20th November 2021

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

Table of Contents

1. Acknowledgement	3
2. Introduction	4-6
3. Steps for Creating Tic-Tac-Toe Game	7-16
4. Tic Tac Toe Python Project Output	17-18
5. Conclusion	19
6. References	20

ACKNOWLEDGEMENT

I would like to thanks my teacher Dr. Sagar Pande for guiding me to do this project very well on Tic-Tac-Toe Game.

And respectfully would like special thanks to Lovely Professional University for creating such amazing environment. And giving such project which will help us in future. I am learning so many things although it's in Online. Our python teacher is very cooperative with us.

INTRODUCTION

1. Context:

This project has been done as a part of my course for the INT213 at Lovely Professional University. Supervised by Dr. Sagar Pande, I have two months to fulfill the requirements in order to succeed the module.

2. Motivations:

Being particularly interested in everything related to the python programming, the project provided us with an excellent opportunity to learn and reinforce our enthusiasm for this domain. Tic tac toe game is GUI-based project. That’s why I decided to take this as my project.

3. Idea:

As a first experience, we intended to make the project as didactic as possible by approaching each stage of the programming process and attempting to comprehend it thoroughly. The purpose was to create a game with interesting features, which will be developed in the next sections.

What is Tic-Tac-Toe Game :

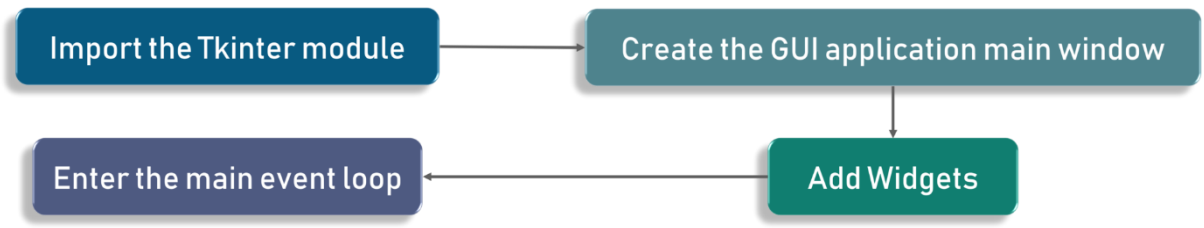
Tic-Tac-Toe is a very simple two-player game. This game is also known as Noughts and Crosses or Xs and Os game. In this game we have a board consisting of a 3X3 grid. To play this game we require two players to play one is X and the other is O and both players play by putting their marks in empty squares.

To win the game players have to get 3 of her marks in a row (up, down, across, or diagonally).

To build this game we use the tkinter module with the concept of python.

What is Tkinter in Python :

Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. It is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. This framework provides Python users with a simple way to create GUI elements using the widgets found in the Tk toolkit. Tk widgets can be used to construct buttons, menus, data fields, etc. in a Python application. Tkinter has a special role throughout this project.



Steps for Creating Tic-Tac-Toe Game

These are the step to build Tic-Tac-Toe game using python

- Import modules
- Initialize window
- Function to check result
- Function to check the winner
- Define labels and buttons

1. Import Modules:

```
from tkinter import *
import tkinter.messagebox as msg
```

In this step, we import tkinter and messsagebox module

2. Initialize window:

```
root= Tk()
root.title('TIC-TAC-TOE---Project Gurukul')
#labels
Label(root,text="player1 : X",font="times 15").grid(row=0,column=1)
Label(root,text="player2 : O",font="times 15").grid(row=0,column=2)

digits = [1,2,3,4,5,6,7,8,9]

#for player1 sign = X and for player2 sign= Y
mark = ''

#counting the no. of click
count = 0


panels = ["panel"]*10
```

- **Tk()** is use to initialize window
- **title()** used to set the title of the window

3. Function to check result:

```
def win(panels,sign):
    return ((panels[1] == panels[2] == panels [3] == sign)
            or (panels[1] == panels[4] == panels [7] == sign)
            or (panels[1] == panels[5] == panels [9] == sign)
            or (panels[2] == panels[5] == panels [8] == sign)
            or (panels[3] == panels[6] == panels [9] == sign)
            or (panels[3] == panels[5] == panels [7] == sign)
            or (panels[4] == panels[5] == panels [6] == sign)
            or (panels[7] == panels[8] == panels [9] == sign))
```

In this function, the result will be checked by checking which player makes three of their marks in a row (up, down, across, or diagonally).

4. Function to check the winner:

```
def checker(digit):
    global count, mark, digits
```

Check which button clicked

```
    if digit==1 and digit in digits:
        digits.remove(digit)
```

player1 will play if the value of count is even and for odd player2 will play

```
        if count%2==0:
            mark ='X'
            panels[digit]=mark
        elif count%2!=0:
            mark = 'O'
            panels[digit]=mark

        button1.config(text = mark)
        count = count+1
        sign = mark

        if(win(panels,sign) and sign=='X'):
            msg.showinfo("Result","Player1 wins")
            root.destroy()
        elif(win(panels,sign) and sign=='O'):
            msg.showinfo("Result","Player2 wins")
            root.destroy()
```

```
    if digit==2 and digit in digits:
        digits.remove(digit)

        if count%2==0:
            mark ='X'
            panels[digit]=mark
        elif count%2!=0:
            mark = 'O'
            panels[digit]=mark

        button2.config(text = mark)
        count = count+1
        sign = mark

        if(win(panels,sign) and sign=='X'):
            msg.showinfo("Result","Player1 wins")
            root.destroy()
        elif(win(panels,sign) and sign=='O'):
            msg.showinfo("Result","Player2 wins")
            root.destroy()
```

```
    if digit==3 and digit in digits:
        digits.remove(digit)

        if count%2==0:
            mark ='X'
            panels[digit]=mark
        elif count%2!=0:
            mark = 'O'
            panels[digit]=mark

        button3.config(text = mark)
        count = count+1
        sign = mark

        if(win(panels,sign) and sign=='X'):
            msg.showinfo("Result","Player1 wins")
            root.destroy()
        elif(win(panels,sign) and sign=='O'):
            msg.showinfo("Result","Player2 wins")
            root.destroy()
```

```
    if digit==4 and digit in digits:
        digits.remove(digit)

        if count%2==0:
            mark ='X'
            panels[digit]=mark
        elif count%2!=0:
            mark = 'O'
            panels[digit]=mark

        button4.config(text = mark)
        count = count+1
        sign = mark

        if(win(panels,sign) and sign=='X'):
            msg.showinfo("Result","Player1 wins")
            root.destroy()
        elif(win(panels,sign) and sign=='O'):
            msg.showinfo("Result","Player2 wins")
            root.destroy()
```

```
    if digit==5 and digit in digits:
        digits.remove(digit)

        if count%2==0:
            mark ='X'
            panels[digit]=mark
        elif count%2!=0:
            mark = 'O'
            panels[digit]=mark

        button5.config(text = mark)
        count = count+1
        sign = mark

        if(win(panels,sign) and sign=='X'):
            msg.showinfo("Result","Player1 wins")
            root.destroy()
        elif(win(panels,sign) and sign=='O'):
            msg.showinfo("Result","Player2 wins")
            root.destroy()
```

```
if digit==6 and digit in digits:
    digits.remove(digit)

    if count%2==0:
        mark ='X'
        panels[digit]=mark
    elif count%2!=0:
        mark = 'O'
        panels[digit]=mark

    button6.config(text = mark)
    count = count+1
    sign = mark

    if(win(panels,sign) and sign=='X'):
        msg.showinfo("Result","Player1 wins")
        root.destroy()
    elif(win(panels,sign) and sign=='O'):
        msg.showinfo("Result","Player2 wins")
        root.destroy()
```

```
if digit==7 and digit in digits:
    digits.remove(digit)

    if count%2==0:
        mark ='X'
        panels[digit]=mark
    elif count%2!=0:
        mark = 'O'
        panels[digit]=mark

    button7.config(text = mark)
    count = count+1
    sign = mark

    if(win(panels,sign) and sign=='X'):
        msg.showinfo("Result","Player1 wins")
        root.destroy()
    elif(win(panels,sign) and sign=='O'):
        msg.showinfo("Result","Player2 wins")
        root.destroy()
```

```
if digit==8 and digit in digits:
    digits.remove(digit)

    if count%2==0:
        mark ='X'
        panels[digit]=mark
    elif count%2!=0:
        mark = 'O'
        panels[digit]=mark

    button8.config(text = mark)
    count = count+1
    sign = mark

    if(win(panels,sign) and sign=='X'):
        msg.showinfo("Result","Player1 wins")
        root.destroy()
    elif(win(panels,sign) and sign=='O'):
        msg.showinfo("Result","Player2 wins")
        root.destroy()
```

```
if digit==9 and digit in digits:
    digits.remove(digit)

    if count%2==0:
        mark ='X'
        panels[digit]=mark
    elif count%2!=0:
        mark = 'O'
        panels[digit]=mark

    button9.config(text = mark)
    count = count+1
    sign = mark

    if(win(panels,sign) and sign=='X'):
        msg.showinfo("Result","Player1 wins")
        root.destroy()
    elif(win(panels,sign) and sign=='O'):
        msg.showinfo("Result","Player2 wins")
        root.destroy()
```

If count is greater then 8 then the match has been tied

```
    if(count>8 and win(panels,'X')==False and win(panels,'O')==False):
        msg.showinfo("Result","Match Tied")
        root.destroy()
```

Players have a total of 9 clicks to play the game. Each time the player clicked; one chance will decrease by increasing the value of count by 1 if the value of count is greater than 8 then the result of game is tie.

- If the value of count is even then player1 will play else player2 will play.
- **config()** used to mark the button with appropriate text
- **showinfo()** methods in the messagebox widget used to show some relevant information
- **destroy()** stop the mainloop to quit the program

5. Define labels and buttons:

Define labels

```
Label(root,text="player1 : X",font="times 15").grid(row=0,column=1)
Label(root,text="player2 : O",font="times 15").grid(row=0,column=2)
```

- **Label()** widget used to display text that users aren't able to modify.

Define buttons

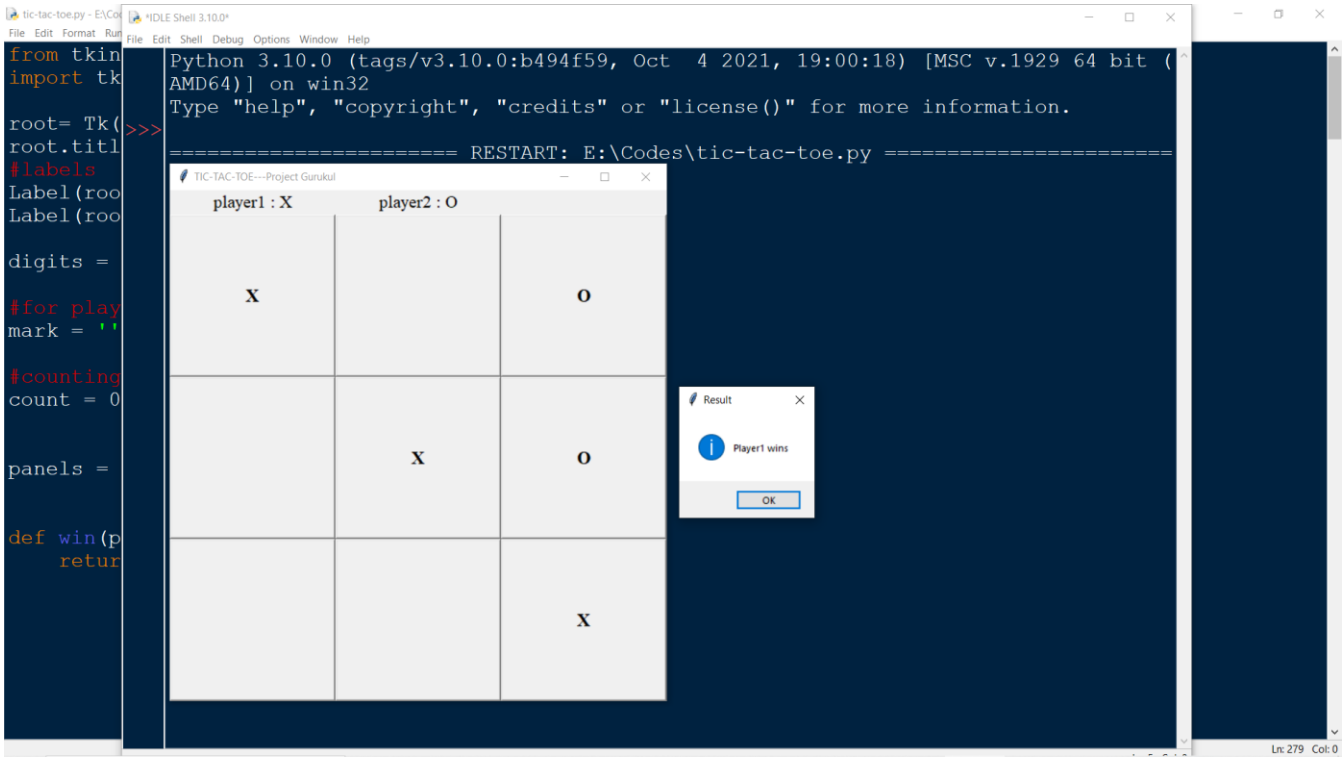
```
button1=Button(root,width=15,font=('Times 16 bold'),height=7,command=lambda:checker(1))
button1.grid(row=1,column=1)
button2=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda:checker(2))
button2.grid(row=1,column=2)

button3=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(3))
button3.grid(row=1,column=3)
button4=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(4))
button4.grid(row=2,column=1)

button5=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(5))
button5.grid(row=2,column=2)
button6=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(6))
button6.grid(row=2,column=3)

button7=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(7))
button7.grid(row=3,column=1)
button8=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(8))
button8.grid(row=3,column=2)

button9=Button(root,width=15,height=7,font=('Times 16 bold'),command=lambda: checker(9))
button9.grid(row=3,column=3)



root.mainloop()
```

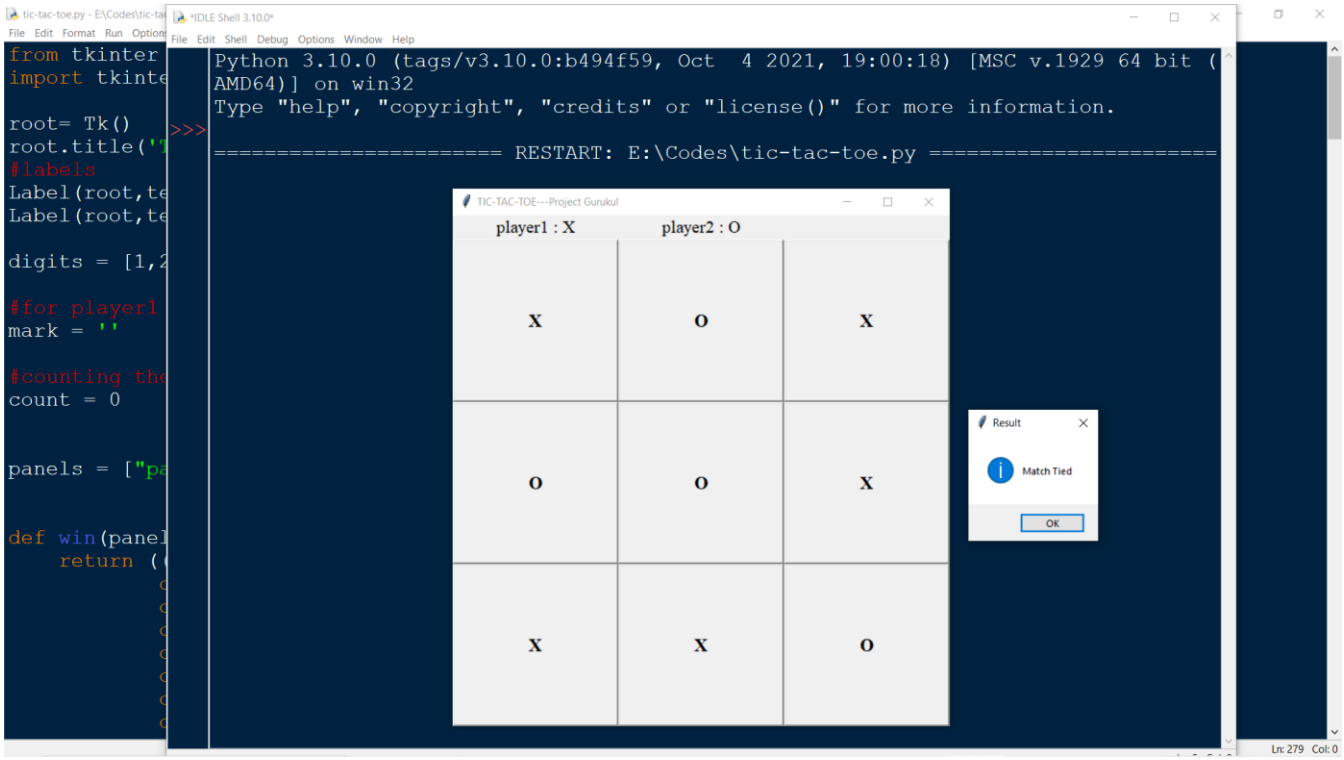
- **Button()** widget display button

- **root** is the name of window which we refered
- **text** stores the value which we display on the label
- **font** in which our text is written
- **command** will called when the button is clicked
- **lambda()** function used to send specific data to the callback function.

mainloop() method executes when we want to run our program.

Tic Tac Toe Python Project Output


IDLE Shell 3.10.0
File Edit Shell Debug Options Window Help
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: E:\Codes\tic-tac-toe.py ==========
TIC-TAC-TOE---Project Gurukul
player1 : X
player2 : O
X
O
X
O
X
Result
Player1 wins
OK


IDLE Shell 3.10.0
File Edit Shell Debug Options Window Help
Python 3.10.0 (tags/v3.10.0:b494f59, Oct 4 2021, 19:00:18) [MSC v.1929 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: E:\Codes\tic-tac-toe.py ==========
TIC-TAC-TOE---Project Gurukul
player1 : X
player2 : O
X
O
X
O
O
X
X
X
O
Result
Match Tied
OK

CONCLUSIONS

I have successfully developed the Tic-Tac-Toe game project in python. I use tkinter module for rendering graphics on a display window. I learn how to create buttons and config text on buttons and also how to use lambda functions to send specific values to callback functions.

In this way I successfully made a Tic-Tac-Toe game python project.

This hands-on project has taught us a lot. In the future, we would like to improve our accuracy drastically through our endeavors.

REFRENCES

To conduct this project the tool that has been used:

- Tkinter (Library): https://docs.python.org/3/library/tkinter.html

I also used the following sites for analysis and to solve various difficulties that arose throughout the making of this project:

- https://www.wikipedia.org/
- https://www.python.org/
- https://www.w3schools.com/python/