Universitatea Politehnica Timișoara

Game or App for Android in Python, TKinter

Student: Darius-Mihai, Ilies
Student: Razvan-Gabriel, Lazar

Coordinator: Radu Stefan, Ricman

Table of Contents

1 INTRODUCTION

Designed using Python and Tkinter, Tic Tac Toe for Android presents a simple yet captivating classical Tic Tac Toe game.. This project represents a paradigm shift in digital gaming, offering an intricately designed rendition of the timeless Tic Tac Toe game, exclusively optimized for Android platforms. By using libraries such as Tkinter and Sys the code encompasses a window containing a game board that is meant for the two players to test their wits, and upon the deciding of a victor a message box pops up to announce the winner.

By automating the process of the turns of the game and analyzing the moves and winning or draw conditions, the code offers simlpicity for it's purpose on creating the game board, reseting the board and announcing the winner. It has a simple and easy to use interface and real-time gameplay, reset and exit functions.

1.1 GENERAL INFORMATION

The primary objective of Tic Tac Toe for Android is to provide a polished and refined implementation of the classic Tic Tac Toe game, specifically optimized for Android platforms. By leveraging the capabilities of Python and Tkinter, the project aims to deliver an intuitive and user-friendly gaming experience that appeals to players of all ages.

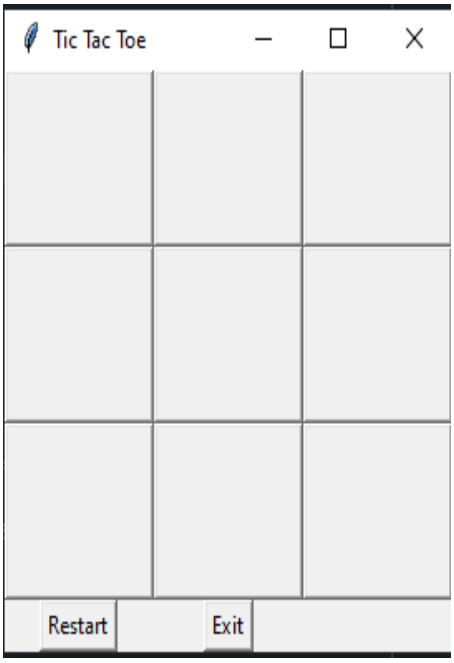
One of the key highlights of Tic Tac Toe for Android is its utilization of Python and Tkinter libraries, enabling seamless integration of essential functionalities and intuitive user interface elements. Leveraging the power of these libraries, the project effortlessly implements player-versus-player gameplay, ensuring that players can engage in exhilarating matches with friends or family members with utmost ease.

Furthermore, the project incorporates essential functionalities such as a reset board button and an exit button, providing users with the convenience of resetting the game board to its initial state or exiting the game at any time. The inclusion of emulation technology further enhances the user experience, ensuring smooth performance across a variety of Android devices.

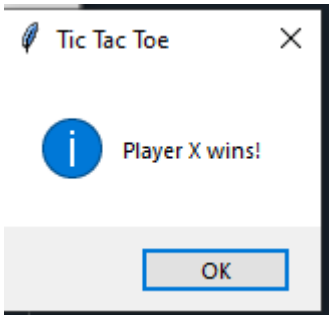
1.1 FIGURES AND IMAGES

When the game is running, a classic Tic Tac Toe is displayed in small window on the screen. This game board dynamically updates the position of the X's and O's.

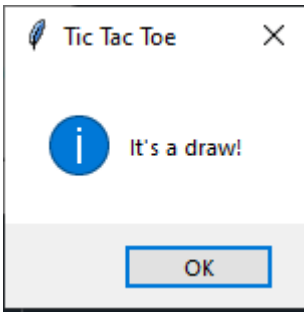
As the game goes on, the players can place their X's and O's on the board until a victor is decided or the game ends in a draw. The result of the match played is displayed in real-time, allowing the players to play another match or to exit the game.



The Game Board



Victor Announcement



Draw Announcement

2. USER-DEFINED FUNCTIONS

2.1 GENERAL INFORMATION

We use user-defined functions to implement a simple and easy to use and understand script for ther users. Such method is commonly used amongst coders, because it helps in 3 main categories:

1) Modularity

- By “saving” progress and specific features into separate functions, the code becomes modular. Each function has set and distinct purpose, making the code easy to understand.

2) Readability

- User-defined functions provide suggestive name that convey the usage of each part of the code
- This helps with readability by making it easier to what each function does without needing to dive into details
- It improves the code comprehension for both the author and the readers who may use the code in the future.

3) Maintainability

- It promotes code reusabilit

- When changes or updates happen, having well-defined functions makes it easier to modify specific locations of the code without affecting other unrelated parts of the code.

2.2 IMPLEMENTED USER-DEFINED FUNCTIONS

The Tic Tac Toe game is structured around several user-defined functions to facilitate interaction and control flow.

- The **`create_board()`** function initializes the game board by creating buttons for each cell and placing them on the Tkinter window. Additionally, it creates the restart and exit buttons, essential for managing game state and user interaction.
- When a player clicks a button on the game board, the **`button_click(row, col)`** function is invoked. This function handles the button click event, updating the board state, checking for a winner or a draw, and updating the current player accordingly.
- The **`check_winner(row, col)`** function examines the row, column, and diagonal where the latest move was made to determine if there's a winner. Similarly, the **`check_draw()`** function checks if all cells on the game board are filled, indicating a draw.
- To restart the game, the **`restart_game()`** function resets the game board to its initial state and sets the current player to 'X'. Conversely, the **`exit_game()`** function gracefully exits the game by destroying the Tkinter root window and exiting the application.
- Finally, the **`run_game()`** function serves as the entry point to the Tic Tac Toe game. It creates an instance of the TicTacToe class, which initializes the game and runs the main loop, ensuring smooth execution and user interaction.

3. CONCLUSIONS

In conclusion, the development journey of Tic Tac Toe for Android has been an exciting endeavor, blending careful planning, technical expertise, and a focus on user

enjoyment. Through the use of Python and Tkinter, we've crafted a version of the beloved game that feels right at home on Android devices.

Our primary goal was to ensure the game is accessible and enjoyable for players of all ages. To achieve this, we implemented features like player-versus-player gameplay and intuitive interface elements. Additionally, we included options for restarting or exiting the game, providing players with flexibility and control over their gaming experience.

The accompanying documentation serves as a helpful resource, offering clear instructions and insights into the game's mechanics. By providing detailed explanations and examples, we aim to empower users to understand and enjoy the game to its fullest.

As we look ahead, Tic Tac Toe for Android stands as a testament to our commitment to creating fun and engaging gaming experiences. We're excited to share this project with players around the world and hope it brings joy and entertainment to all who play it.

BIBLIOGRAPHY

[1] Descoperă întreaga poveste | Disney+ (youtube.com)

[2] Kivy Tutorial - GeeksforGeeks

[3] Tic Tac Toe game with GUI using tkinter in Python - GeeksforGeeks

[4] Python for Android | How To Run Python Programs On Android (youtube.com)

ANNEX

```python
# -*- coding: utf-8 -*-
"""
Created on Mon Apr 20 12:07:49 2024

@author: ilies
"""
import tkinter as tk
from tkinter import messagebox
import sys

class TicTacToe:
    def __init__(self):
        self.root = tk.Tk()
        self.root.title("Tic Tac Toe")

        self.current_player = "X"
        self.board = [["" for _ in range(3)] for _ in range(3)]

        self.buttons = [[None for _ in range(3)] for _ in range(3)]
        self.create_board()

    def create_board(self):
        for i in range(3):
            for j in range(3):
                self.buttons[i][j] = tk.Button(self.root, text="", font=("Helvetica", 20), width=5, height=2,
                                               command=lambda row=i, col=j: self.button_click(row, col))
                self.buttons[i][j].grid(row=i, column=j)

        restart_button = tk.Button(self.root, text="Restart", command=self.restart_game)
        restart_button.grid(row=3, column=0)

        exit_button = tk.Button(self.root, text="Exit", command=self.exit_game)
        exit_button.grid(row=3, column=1)

    def button_click(self, row, col):
        if self.board[row][col] == "":
```

```
            self.buttons[row][col].config(text=self.current_player)
            self.board[row][col] = self.current_player
            if self.check_winner(row, col):
                messagebox.showinfo("Tic Tac Toe", f"Player {self.current_player} wins!")
                self.restart_game()
            elif self.check_draw():
                messagebox.showinfo("Tic Tac Toe", "It's a draw!")
                self.restart_game()
            else:
                self.current_player = "O" if self.current_player == "X" else "X"

    def check_winner(self, row, col):
        # Check row
        if self.board[row][0] == self.board[row][1] == self.board[row][2] == self.current_player:
            return True
        # Check column
        if self.board[0][col] == self.board[1][col] == self.board[2][col] == self.current_player:
            return True
        # Check diagonal
        if (row == col and self.board[0][0] == self.board[1][1] == self.board[2][2] == self.current_player) or \
           (row + col == 2 and self.board[0][2] == self.board[1][1] == self.board[2][0] == self.current_player):
            return True
        return False

    def check_draw(self):
        for row in self.board:
            for cell in row:
                if cell == "":
                    return False
        return True

    def restart_game(self):
        for i in range(3):
            for j in range(3):
                self.buttons[i][j].config(text="")
                self.board[i][j] = ""
        self.current_player = "X"

    def exit_game(self):
        self.root.destroy()
        sys.exit()

    def run(self):
        self.root.mainloop()

if __name__ == "__main__":
    game = TicTacToe()
    game.run()
```