BUN LAB
Broadband Ubiquitous Networking Lab

通訊網路實驗

Android & Python Programming
Python GUI

Dept. of Electrical and Computer Engineering (ECE)

National Yang Ming Chiao Tung University

BUN LAB
Broadband Ubiquitous Networking Lab

評分標準 & 注意事項

- 出席　30%
- Demo 30%
- 結報　40%
  - e3上有學習單
  - 檔名：學號_姓名_Labx.pdf
  - 交pdf到對應的資料夾中 (期限一週)

BUN LAB
Broadband Ubiquitous Networking Lab

課程大綱

- Anaconda3 Spyder介紹
- Python GUI Tkinter 入門
- 圈圈叉叉小程式

BUN LAB
Broadband Ubiquitous Networking Lab

Demo項目

- Q1: 圈圈叉叉小程式

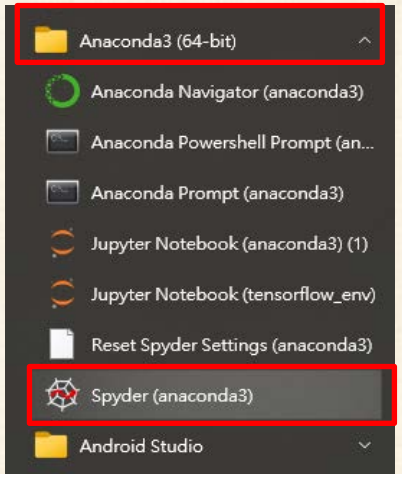
Anaconda

- 很受歡迎的Data Science、Machine Learning平台
- 具備多種流行的packages
- 適用於Windows、Linux、MacOS等作業系統
- 在安裝、執行上相對簡易

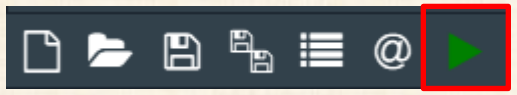
BUN LAB
Broadband Ubiquitous Networking Lab

Introduction

- 開始 > Anaconda3 > Spyder(anaconda3)

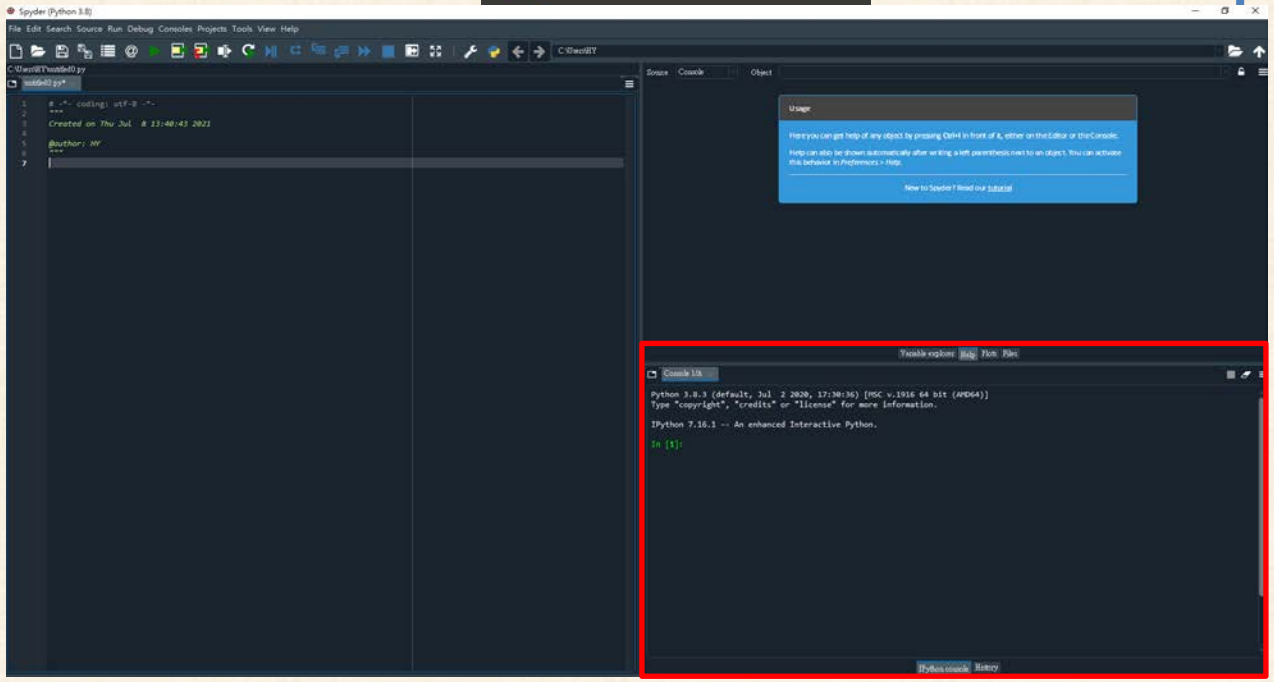


- 綠色箭頭可以debug



- Console裡面可以顯示
  變數值、執行結果等

但今天的執行結果會另外跳視窗出來



Introduction

- Python GUI
- PyQt
  - 整合度高、有圖形化介面可以用
- Tkinter
  - Python內建的package
  - 簡單

Python GUI Tkinter (1)

- import tkinter as tk
- 創造tkinter的物件主視窗
  - window = tk.Tk()
  - window.title('Lab3')

Python GUI Tkinter (2)

- 建立主視窗後即可創造其他widget物件，如Button、Label等
  - e.g. label = tk.Label(window, text='Hello World!')

所在視窗名稱　label顯示的文字

  - e.g. button = tk.Button(window, text='hit me', command='xxx')
  - e.g. frame = tk.Frame(window)

設定Button

- button = tk.Button(frame, text = "7" , borderwidth=5, width= 4, command = lambda: Click("7"))
  - 按鈕的邊框寬度
  - 按鈕的寬度
- e.g.
  - command = lambda: Click(“7”)
    當按下按鈕時傳值’7’給function Click
  - command = lambda: Clear()

建立一個臨時的、一次性的單行函式來傳遞引數

```
In [4]: lambda : 3
Out[4]: <function __main__.<lambda>()>

In [5]: (lambda : 3) ()
Out[5]: 3
```

設定Button

- button.config(bg='red') 設定button背景顏色
- button.config(text='hello') 設定button文字
- button.config(font='Arial') 設定button字型
- button.config(width='10',height = '20') 設定button寬高

Python GUI Tkinter (3)

- 將widget放在視窗上 e.g. frame.pack()
  - pack() / pack(side='left/right/top/bottom')
    - widget會直接放在視窗上(或指定的方位)
  - grid(column=0, row=1)
    - widget會放在指定的位置(行列)
  - place(x=10, y=30)
    - widget會放在指定的座標

三者不可在同一視窗中混用

Python GUI Tkinter (4)

- 將Tkinter物件放入等待迴圈，讓window不斷重新整理
  - window.mainloop()

Python GUI Tkinter (5)

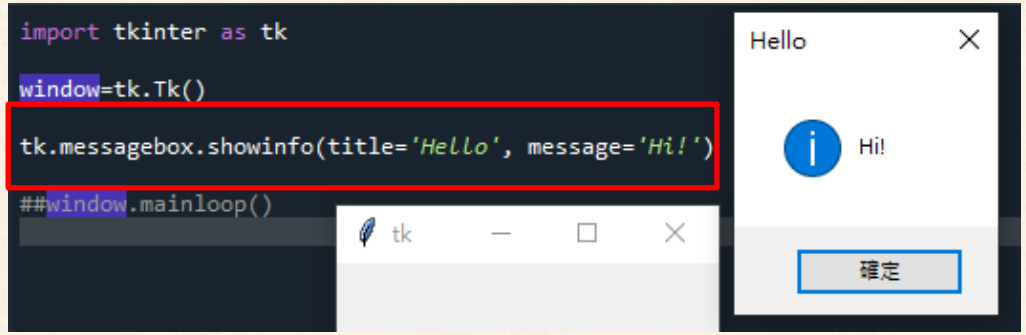
- 警示視窗(messagebox)

tk.messagebox.showinfo()

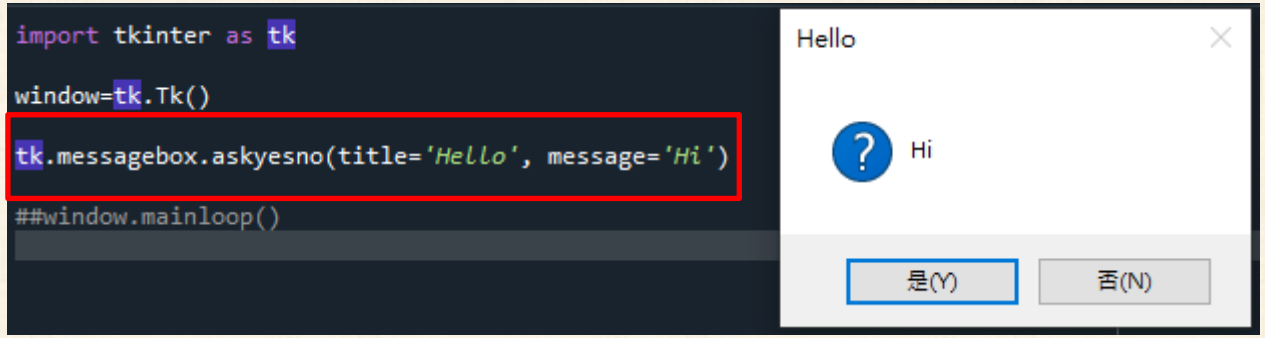
tk.messagebox.showwarning()

tk.messagebox.showerror()

tk.messagebox.askyesno() #回傳true或false



如果有module tkinter has no attribute messagebox錯誤，
在import tkinter as tk 下面新增
from tkinter import messagebox



Global variable

- Global variable 的使用

```
a = 999

def function2():
    a += 1
    print(a)

function2()
```

```
$ python3 global02.py
Traceback (most recent call last):
  File "global02.py", line 7, in <module>
    function2()
  File "global02.py", line 4, in function2
    a += 1
UnboundLocalError: local variable 'a' referenced before assignment
```

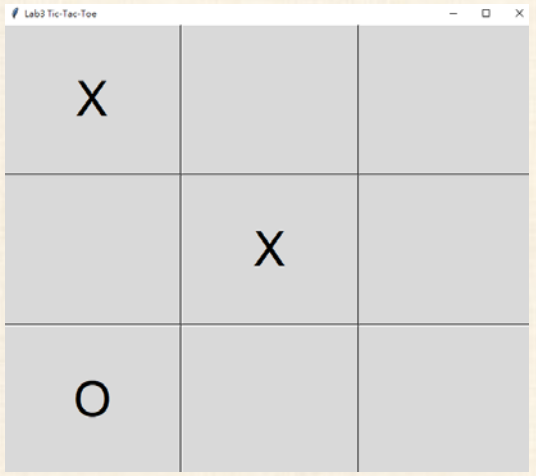
```
a = 999

def function3():
    global a
    a += 1
    print(a)

function3()
```

```
$ python3 global03.py
1000
```

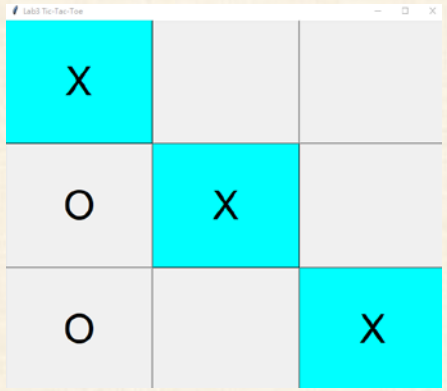
圈圈叉叉 Q1

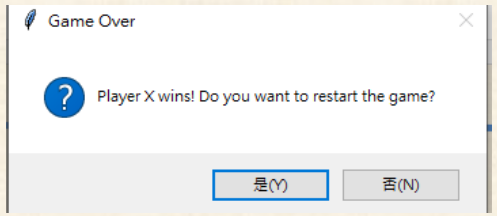
Demo標準:

1. 玩家可以在九宮格內用滑鼠點選，選擇要下圈或叉
2. 判斷勝利條件，且勝利時要顯示 'X' or 'O' 贏
3. 每次結束(平手或勝利)時，跳出詢問視窗，確認是否要重新一局
4. 點選'是'，會重新開始一局，點選'否'會將視窗關閉
5. 獲勝時，改變button背景顏色，標示贏家所下的連線
6. 平手時，將全部button的背景改顏色

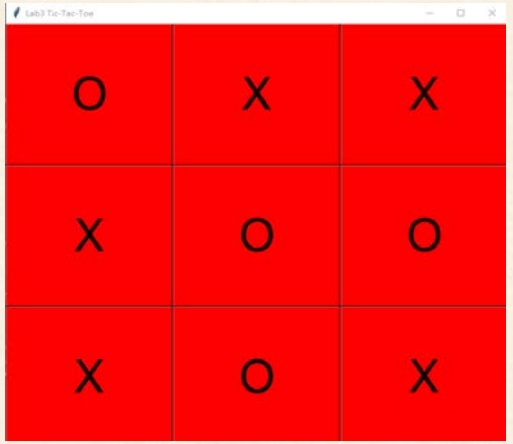


(c) 進行中的遊戲

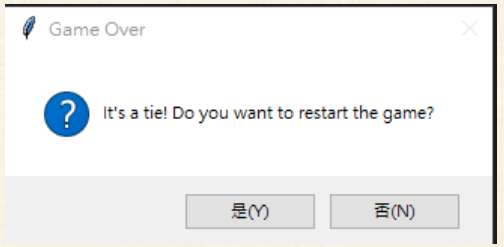




(a) 獲勝



(b) 平手



Q1

- **創建主視窗** > 建出button > 撰寫button的listener function

```
2  if __name__ == '__main__':
3      # create main window
4      window = tk.Tk()
5      window.title("Lab3 Tic-Tac-Toe")
6
7      # create game board
8      create_board()
9
10     # Initialize variables
11     board = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
12     current_player = 1
13
14     window.mainloop()
```

Q1

- 創建主視窗 > **建出button** > 撰寫button的listener function

```
9 # Create board
8 def create_board():
7     for i in range(3):
6         for j in range(3):
5             button = tk.Button(window, text="", font=("Arial", 50),
4                     height=2, width=6,
3                     command=lambda row=i, col=j: handle_click(row, col))
2             button.grid(row=i, column=j, sticky="nsew")
1
```

Q1

- 創建主視窗 > 建出button > **撰寫button的listener function**

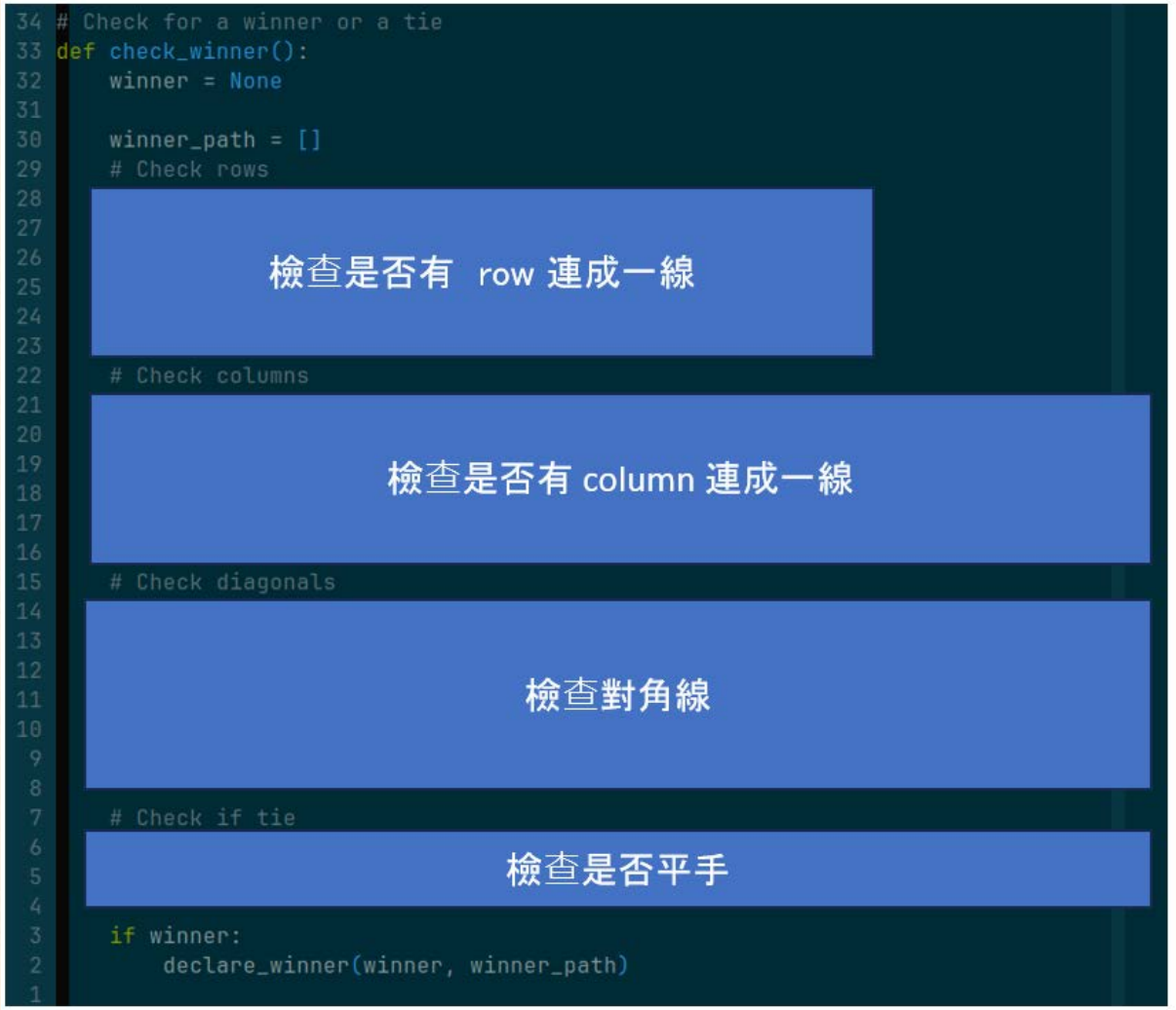
Q1

- 提示:若要設定特定row & column 的button，可以使用這個function

```
button = window.grid_slaves(row=row, column=col)[0]
```

Q1

- > **檢查遊戲使否結束** > 標出顏色並詢問是否繼續遊戲



Q1

- > 檢查遊戲使否結束 > **標出顏色並詢問是否繼續遊戲**

```
# Declare the winner and ask to restart the game
def declare_winner(winner, winner_path):
    if winner == "tie":
        遊戲平手、改變board顏色
    else:
        遊戲有勝負、標出勝利者的連線

    # ask if the player wants to continue
    answer = 視窗詢問玩家是否繼續遊戲

    if answer:
        # play another round
        global board
        board = [[0, 0, 0], [0, 0, 0], [0, 0, 0]]

        for i in range(3):
            for j in range(3):
                button = window.grid_slaves(row=i, column=j)[0]
                button.config(text="", bg=default_bg_color)

        global current_player
        current_player = 1
    else:
        window.destroy()
```