# GUI module in Python (GUI: graphical user interface)

授課老師：邱淑怡

# Graphical user interface (GUI)

- A GUI allows users to interact with the operating system and other programs using graphical elements such as icons, buttons, and dialog boxes.
- GUI module:
  - Tkinter
  - PyQt
- Difference
  - Pyqt: need to install
  - Tkinter: the standard Python interface to the Tcl/Tk GUI toolkit

# Tkinter

- Python does not have GUI programming features built into the language itself.
- The name “tkinter” is short for “Tk interface.”
- It is named this because it provides a way for
- Python programmers to use a GUI library named Tk.
- In Python you can use the tkinter module to create simple GUI programs.
- Tkinter module can only be executed on the computer and cannot run on colab

# Tkinter Widgets(元件)

- Label (標籤)
- Button (按鈕)
- Entry (文字方塊)
- Checkbutton (核取按鈕)
- Radiobutton (單選按鈕)
- ComboBox (下拉式選單)

**Table 14-1** `tkinter` Widgets

| Widget | Description |
|---|---|
| `Button` | A button that can cause an action to occur when it is clicked. |
| `Canvas` | A rectangular area that can be used to display graphics. |
| `Checkbutton` | A button that may be in either the "on" or "off" position. |
| `Entry` | An area in which the user may type a single line of input from the keyboard. |
| `Frame` | A container that can hold other widgets. |
| `Label` | An area that displays one line of text or an image. |
| `Listbox` | A list from which the user may select an item |
| `Menu` | A list of menu choices that are displayed when the user clicks a `Menubutton` widget. |
| `Menubutton` | A menu that is displayed on the screen and may be clicked by the user |
| `Message` | Displays multiple lines of text. |
| `Radiobutton` | A widget that can be either selected or deselected. `Radiobutton` widgets usually appear in groups and allow the user to select one of several options. |
| `Scale` | A widget that allows the user to select a value by moving a slider along a track. |
| `Scrollbar` | Can be used with some other types of widgets to provide scrolling ability. |
| `Text` | A widget that allows the user to enter multiple lines of text input. |
| `Toplevel` | A container, like a `Frame`, but displayed in its own window. |

# Test Tkinter module

```
import tkinter as tk
tk._test()
```

# Program

```
import tkinter as tk
window = tk.Tk()  #call Tk() to create window
…[statements, 程式區塊]
…
window.mainloop()
```

=

```
import tkinter as tk


def main():
        window = tk.Tk()
        window.mainloop()
main()
```

# Declare variables & get the value of the variable in Tkinter module

- Declare variable
  - radioValue = tk.IntVar()  # radioValue is int
  - num1=tkinter.DoubleVar() # num1 is float
  - radio1 = tk.BooleanVar()  # radio1 is Boolean
  - var1=tk.StringVar()  # var1 is string
- Get the value of variables
  - radioValue.get()
  - num1.get()
  - radio1.get()
  - var1.get()

# The first program: hello world

```
import tkinter as tk
window = tk.Tk()
window.title("Hello World!")
window.minsize(width=500, height=500)
window.resizable(width=False, height=False)
window.mainloop()
```

# Label (標籤)

```
import tkinter as tk

window = tk.Tk()

window.title('window')

window.geometry(‘500x100’)  #width*height

label_1 = tk.Label(window, text='Hello World', bg='yellow', fg='#bd34eb',
font=('Arial', 12))

#label_1.grid(column=0, row=0)

label_1.pack()  # default: ’top’ location

window.mainloop()
```

# Tkinter layout management

The three methods

1. pack()
2. grid()
3. place()

**The pack() and grid() cannot be used together in the same window, but pace() can used together with pack() or grid()

(視窗容器中不能同時使用 pack() 與 grid() , 但 place() 卻可以與 pack() 或 grid() 同時使用)

# pack()

1. pack(): organizes widgets in horizontal and vertical boxes.(流水式排版, 預設元件會依加入先後順序由上而下, 由左而右自行排列)

| 參數 | 說明 |
| --- | --- |
| side | 排列方向 : TOP (預設), BOTTOM, LEFT, RIGHT |
| Fill | 填滿所分配空間之方向 : NONE (預設), X, Y, BOTH |
| expand | 填滿容器 : True/False (預設) |
| padx/pady | 元件邊框與容器之距離 (px, 預設=0) |
| ipadx/ipady | 元件內容 (文字/圖像) 與其邊框之距離 (px, 預設=0) |
| anchor | 元件在容器中的錨定位置 : E, W, S, N, CENTER (預設), NE, SE, SW, NW |

# Example_1

```
import tkinter as tk
window =tk.Tk()
window.title("pack() Test")
window.geometry("200x250")
tk.Label(window, text="平日").pack()
tk.Label(window, text="周末假日").pack()
tk.Label(window, text="國定假日").pack()
window.mainloop()
```

```
import tkinter as tk
window =tk.Tk()
window.title("pack() Test_1")
window.geometry("200x250")
tk.Label(window, text="平日").pack(side=tk.BOTTOM)
tk.Label(window, text="周末假日").pack(side=tk.BOTTOM)
tk.Label(window, text="國定假日").pack(side=tk.BOTTOM)
window.mainloop()
```

# Example_2

由於 pack() 是流水式排版，元件是按照先後順序擺放在錨定位置

```
import tkinter as tk
root=tk.Tk()
root.title("pack() Test_2")
root.geometry("300x250")
tk. Label(root, text="東").pack(anchor=tk.E)
tk.Label(root, text="西").pack(anchor=tk.W)
tk.Label(root, text="南").pack(anchor=tk.S)
tk.Label(root, text="北").pack(anchor=tk.N)
tk.Label(root, text="中").pack(anchor=tk.CENTER)
tk.Label(root, text="東南").pack(anchor=tk.SE)
tk.Label(root, text="西北").pack(anchor=tk.NW)
tk.Label(root, text="西南").pack(anchor=tk.SW)
tk.Label(root, text="東北").pack(anchor=tk.NE)
root.mainloop()
```

# Example_3 (間隙距離 padx/pady)

```
import tkinter as tk
root=tk.Tk()
root.title("pack() Test_3")
#root.geometry("300x200")

tk.Label(root, text="平日").pack(padx=20, pady=10)
tk.Label(root, text="周末假日").pack()
tk.Label(root, text="國定假日").pack(padx=20, pady=10)
root.mainloop()
```

# grid()

1. 表格式排版, 該元件是依據所指定的索引位置, 如同二維陣列元素一般放入表格

| grid()參數 | 說明 |
| --- | --- |
| row | 列索引 |
| column | 行索引 |
| rowspan | 儲存格合併列數 |
| columnspan | 儲存格合併行數 |
| padx/pady | 元件邊框與容器之距離 (px, 預設=1) |
| ipadx/ipady | 元件內容 (文字/圖像) 與其邊框之距離 (px, 預設=1) |
| sticky | 元件於網格中的錨定位置 : E, W, S, N, CENTER (預設) |

# Example_4

grid() 時若不傳參數,
預設是以 n 列 1 行的網格來依序放置元件

```
import tkinter as tk


root=tk.Tk()
root.title("grid() 測試")
root.geometry("300x150")


tk.Label(root, text="平日").grid()
tk.Label(root, text="周末假日").grid()
tk.Label(root, text="國定假日").grid()
root.mainloop()
```

# Example_5

```
import tkinter as tk
root=tk.Tk()
root.title("grid() 測試")
root.geometry("300x150")

tk.Label(root, text="平日").grid(row=0, column=0)
tk.Label(root, text="周末假日").grid(row=0, column=1)
tk.Label(root, text="國定假日").grid(row=1, column=0)
tk.Label(window, text="國定假日and周末假日").grid(row=1,column=1)
root.mainloop()
```

# place()

1. place()

| place()參數 | 說明 |
|---|---|
| x | 相對於視窗左上角之 x 座標 |
| y | 相對於視窗左上角之 y 座標 |
| width | 指定元件寬度 (px) |
| height | 指定元件高度 (px) |
| relx | 相對於父容器寬度之比率 x 座標 (0~1) |
| rely | 相對於父容器高度之比率 y 座標 (0~1) |
| relwidth | 相對於父容器寬度之比率 (0~1) |
| relheight | 相對於父容器高度之比率 (0~1) |
| anchor | 元件在容器中的錨定位置 : E, W, S, N, CENTER (預設), NE, SE, SW, NW |

# place() 提供兩種定位方法

1. 絕對定位 :
   - 以 (x, y) 參數指定絕對座標
   - 以 (width, height) 指定絕對大小

2. 相對定位 :
   - 以 (relx, rely) 參數指定相對座標
   - 以 (relwidth, relheight) 參數指定相對大小

# place(): absolute location (絕對位置) vs relative location (相對位置)

```
import tkinter as tk


root=tk.Tk()
root.title("place() 測試")
root.geometry("250x100")


tk.Label(root, text="absolute location:").place(x=25, y=25)
tk.Label(root, text="relative location: ").place(relx=0.5,rely=0.5)
root.mainloop()
```

# place(): relative location(相對位置)元件之寬度與高度設定

```
import tkinter as tk


root=tk.Tk()
root.title("place() 測試")
root.geometry("250x100")
tk.Label(root, text=" absolute location: ").place(x=25, y=25)
tk.Label(root, text=" relative location: ").place(relx=0.5, rely=0.5, relwidth=0.5, relheight=0.5)
root.mainloop()
```

place() 的 x 和 y 參數表示放置的元件(element)在主視窗的絕對位置，視窗左上角為 (0,0)，往右為正，往下為正

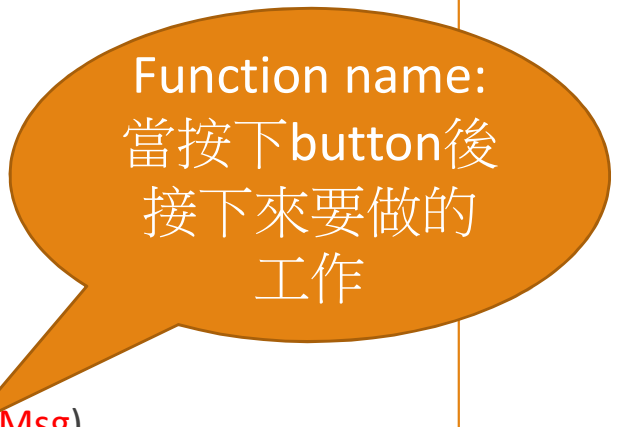
# Example_6

```
import tkinter as tk
root = tk.Tk()
root.title(‘test’)
root.geometry('200x200’)
a = tk.Label(root, text='AAA', background='#f90’)
b = tk.Label(root, text='BBB', background='#09c’)
c = tk.Label(root, text='CCC', background='#fc0’)
a.place(x=0, y=0) # put it on (0,0)
b.place(x=50, y=50)
c.place(x=100, y=100)
root.mainloop()
```

# Button (按鈕)

```
import tkinter as tk
def HelloMsg():
    label["text"] = "Hello, Python!"


win=tk.Tk()
win.geometry('500x100')
btn=tk.Button(win, text="Hello按鈕", command=HelloMsg)
label=tk.Label(win)
btn.pack()
label.pack()
win.mainloop()
```



# Entry (文字方塊)

```
import tkinter
def add_num():
    result.set(num1.get() + num2.get())
win = tkinter.Tk()
win.title('加法視窗程式')
num1=tkinter.DoubleVar()
num2=tkinter.DoubleVar()
result=tkinter.DoubleVar()
```

```
item1=tkinter.Entry(win, width=10, textvariable=num1)
label1=tkinter.Label(win, width=5, text='+')
item2=tkinter.Entry(win, width=10, textvariable=num2)
btn=tkinter.Button(win, width=5, text='=', command=add_num)
label2=tkinter.Label(win, width=10, textvariable=result)
item1.pack(side='left')
label1.pack(side='left')
item2.pack(side='left')
btn.pack(side='left')
label2.pack(side='left')
win.mainloop()
```

# The example for Entry

```
import tkinter as tk
win = tk.Tk()
win.title("Hello, use Entry")
win.geometry("200x200")
a = tk.StringVar() # 建立文字變數
a.set('') # 一開始設定沒有內容
tk.Label(win, textvariable=a).pack() # 放入 Label
tk.Entry(win, textvariable=a).pack() # 放入 Entry
win.mainloop()
```

# Radiobutton (單選按鈕): single choice

- Radio buttons normally appear in groups of two or more and allow the user to select one of several possible options.
- Check buttons, which may appear alone or in groups, allow the user to make yes/no or on/off selections.

1. Create Radio buttons of several possible options
2. Get the value of selected Radio button

# Example

- **同一組中的單選按鈕共享相同的變數** radioValue，且使用 value 選項賦給了不同的數值。
- 所選擇的單選按鈕的值自動地更新變數 radioValue，它是一個 tk.IntVar。
- 標籤文字在以下示例程式碼中自動顯示所選按鈕的值

```
import tkinter as tk
win = tk.Tk()
win.geometry('200x100')

radioValue = tk.IntVar()
radio_1 = tk.Radiobutton(win, text='January',variable=radioValue, value=0)
radio_2 = tk.Radiobutton(win, text='Febuary',variable=radioValue, value=1)
radio_3 = tk.Radiobutton(win, text='March',variable= radioValue, value=2)
radio_1.grid(column=0, row=0, sticky="W")
radio_2.grid(column=0, row=1, sticky="W")
radio_3.grid(column=0, row=2, sticky="W")
labelValue = tk.Label(win, textvariable=radioValue)
labelValue.grid(column=2, row=0, sticky="E", padx=40)
win.mainloop()
```

When you create a group of Radiobuttons, you associate them all with the same radioValue (variable name) object.
You also assign a unique integer value to each Radiobutton widget.
When one of the Radiobutton widgets is selected, it stores its unique integer value in the radioValue object.

# Example

```
import tkinter
from tkinter import messagebox
def showMsg():
    i=radio_v.get()
    messagebox.showinfo('選取結果', '您最想去的國家為：'+country[i])
win=tkinter.Tk()
win.title('最想要旅遊國家調查')
win.geometry('300x150')
label=tkinter.Label(win, text='請選取您最想要旅遊的國家：').pack()
country={0:'土耳其', 1:'英國', 2:'日本', 3:'埃及'}
radio_v = tkinter.IntVar()
radio_v.set(0)
for i in range(len(country)):
    tkinter.Radiobutton(win,text=country[i],variable=radio_v,value=i).pack()
tkinter.Button(win, text = "確定", command=showMsg).pack()
win.mainloop()
```

# Checkbutton (核取按鈕): multiple choice

- Checkbutton: 是提供使用者核取選項的按鈕，使用者可以多選或不選任何一個，**選項都是獨立的(變數也獨立建立)**

# Checkbutton

```
import tkinter as tk
win = tk.Tk()
win.geometry('200x100')

radioValue1 = tk.BooleanVar()
radioValue2 = tk.BooleanVar()
radioValue3 = tk.BooleanVar()
rdioOne = tk.Checkbutton(win, text='January',variable=radioValue1)
rdioTwo = tk.Checkbutton(win, text='Febuary',variable=radioValue2)
rdioThree = tk.Checkbutton(win, text='March',variable=radioValue3)
rdioOne.grid(column=0, row=0, sticky="W")
rdioTwo.grid(column=0, row=1, sticky="W")
rdioThree.grid(column=0, row=2, sticky="W")

labelValue1 = tk.Label(win, textvariable=radioValue1)
labelValue2 = tk.Label(win, textvariable=radioValue2)
labelValue3 = tk.Label(win, textvariable=radioValue3)
labelValue1.grid(column=2,row=0)
labelValue2.grid(column=2,row=1)
labelValue3.grid(column=2,row=2)

win.mainloop()
```

# Checkbutton(核取按鈕)

```
import tkinter
from tkinter import messagebox

def showMsg():
    result = ''
    for i in check_v:
        if check_v[i].get() == True:
            result = result + country[i] + ' '
    messagebox.showinfo('核取結果', '您想去的國家為：'+result)
win=tkinter.Tk()
win.title('想要旅遊國家調查')
win.geometry('300x150')
label=tkinter.Label(win, text='請選取您想要旅遊的國家：').pack()
country ={0:'土耳其', 1:'英國', 2:'日本', 3:'埃及'}
check_v ={}
for i in range(len(country)):
    check_v[i] = tkinter.BooleanVar()
    tkinter.Checkbutton(win, text=country[i], variable=check_v[i]).pack()
tkinter.Button(win, text='確定', command=showMsg).pack()
win.mainloop()
```

# Example

```
def mymsg():
    msgfood.set("您最喜歡的主食為"+choosefood.get()) #抓值
import tkinter as tk
yrwin = tk.Tk()
choosefood=tk.StringVar() #設定動態變數為字串
msgfood=tk.StringVar() #設定動態變數為字串
foodlabel=tk.Label(yrwin,text="請選擇你最喜歡的主食")
foodlabel.pack()
ch01=tk.Radiobutton(yrwin,text="飯",value="1",variable=choosefood,command=mymsg)
ch01.pack()
ch02=tk.Radiobutton(yrwin,text="麵",value="2",variable=choosefood,command=mymsg)
ch02.pack()
msglabel=tk.Label(yrwin,textvariable=msgfood,fg="blue")
msglabel.pack()
ch02.select() #預設選擇(一定要設定看是預設選飯還是選麵)
mymsg()
yrwin.mainloop()
```

# ComboBox (下拉式選單 )

1. 建立 Combobox
2. 設定 Combobox 預設的選項
3. 取得目前 Combobox 的選項
4. Combobox 綁定事件

# Combobox (下拉式選單)

```
import tkinter as tk
from tkinter import ttk

def show_value():
    label["text"]=var.get()       # 呼叫類別變數的 get() 方法取得被選取項目

win=tk.Tk()
win.title("Tkinter Combobox")
win.geometry("300x200")

var=tk.StringVar()          # 與 Combobox 綁定之類別變數
var.set("Python")           # 設定預設被選取選項
V1=[“Python”, “Javascript”, “R”, “Julia”, “PHP”]
Combobox=ttk.Combobox(win, values=v1, textvariable=var)    # 綁定類別變數
combobox.pack()
ttk.Button(win, text="確定", command=show_value).pack()
label=ttk.Label(win)
label.pack()
win.mainloop()
```

# listbox (列表框)

```
import tkinter as tk
root = tk.Tk()
root.title('my window')
root.geometry('200x180')
mylistbox = tk.Listbox(root)
mylistbox.insert(tk.END, 'apple')
mylistbox.insert(tk.END, 'banana')
mylistbox.insert(tk.END, 'orange')
mylistbox.insert(tk.END, 'lemon')
mylistbox.insert(tk.END, 'tomato')
mylistbox.pack()
root.mainloop()
```

# 類別

| 類別 | 介紹 |
| --- | --- |
| Frame | 視窗。 |
| Label | 文字標籤。 |
| Button | 按鈕。 |
| Canvas | 可以用來繪圖、文字等都可以，像我就會來拿放圖片。 |
| Checkbutton | 核取按鈕。 |
| Entry | 文字輸入欄。 |
| Listbox | 列表選單。 |
| Menu | 選單列的下拉式選單。 |
| LabelFrame | 文字標籤視窗。 |
| MenuButton | 選單的選項。 |
| Message | 類似 Label ，可多行。 |
| OptionMenu | 下拉式的選項選單 |
| PaneWindow | 類似 Frame ，可包含其他視窗元件。 |
| Radiobutton | 單選按鈕。 |
| Scale | 拉桿。 |
| Scrollbar | 捲軸。 |
| Spinbox | 微調器 |
| Text | 文字方塊。 |

https://www.rs-online.com/designspark/python-tkinter-cn#_Toc61529916

# Exercise 8

# Question

請設計一個圖形化使用者介面，可以讓使用者填寫生活健康狀況的問卷，最後顯示問卷的結果

題目:

1.請問是否抽菸習慣? 是、否 (value=0、1)

2.請問是否有飲酒習慣? 是、否(value=0、1)

3.請問每天睡眠時間多少小時? (下拉選單)

4.請問每天是否有均衡飲食? 是、否(value=1、0)

5. 最後，請問你的姓名: (請練習產生Entry)

6. “確認”按鈕

最後顯示結果如下:

➔若3題的總分>=2且睡眠時間>=6, 顯示”XXX, 你的健康狀況良好”；否則就顯示”XXX, 你的健康狀況不好”

# Review

Textbook: chapter 14