gui

April 10, 2024

0.1 18 exemples de codes pour faire le tour de Tkinter

Une bonne documentation https://tkdocs.com/tutorial/

L'exemple minimal

[2]:
```
from tkinter import Tk
root = Tk()
root.mainloop()
```

Les éléments s'emboitent les uns dans les autres

[10]:
```
from tkinter import Tk, Label, Frame
root = Tk()
mot = Label(root,text="world")
mot.grid(row=1,column=1)
root.mainloop()
```

Un premier exemple de gestion d'évènement : la fermeture de la fenêtre

[ ]:
```
from tkinter import Tk, Label, Frame
def stop(e):
    print("fini")

root = Tk()
mot1 = Label(root,text="world")
mot1.grid(row=1,column=1)
root.bind("<Destroy>", stop)
root.mainloop()
```

La lecture sur le clavier : la touche 'a'

[ ]:
```
from tkinter import Tk, Label, Frame
def un_a(e):
    print(e)

root = Tk()
root.geometry('600x400')
mot1 = Label(root,text="world")
mot1.grid(row=1,column=1)
```

```
root.bind("<a>",un_a)
root.mainloop()
```

Un bouton qui dit bonjour

[ ]:
```
from tkinter import Tk, Button, Frame
def bonjour():
    print("bonjour")

root = Tk()
root.geometry('600x400')
window = Frame(root)
window.grid()
but = Button(window,text="Dis bonjour", command=bonjour)
but.grid(row=1,column=1)
root.mainloop()
```

0.2 Des widgets

Quelques widgets classiques

[1]:
```
from tkinter import Tk,  Frame,  Text
from tkinter.ttk import Button, Label, Spinbox, Checkbutton, Radiobutton, Entry
root = Tk()
root.geometry('600x400')
window = Frame(root)
window.grid()
Button(window,text="Dis bonjour").grid(row=1,column=1)
Label(window,text="C'est moi").grid(row=2,column=1)
Spinbox(window,values=('One', 'Two', 'Three')).grid(row=3,column=1)
Checkbutton(window, text="Tu n'as rien oublié ?").grid(row=4,column=1)
Radiobutton(window, text="Option 1", value="1").grid(row=5,column=1)
Radiobutton(window, text="Option 2", value="2").grid(row=6,column=1)
Radiobutton(window, text="Option 3", value="4").grid(row=7,column=1)
Entry(window).grid(row=8,column=1)
Text(window).grid(row=1,column=2,rowspan=8)
root.mainloop()
```

Encore des widgets

[ ]:
```
from tkinter import Tk, Frame, Listbox,IntVar, Canvas
from tkinter.ttk import Combobox, Progressbar, Scale,Separator
import tkinter as tk
root = Tk()
root.geometry('600x400')
window = Frame(root)
window.grid()
cbox = Combobox(window, values=("Left","Right","Up","Down"))
```

```
cbox.grid(column=1, row=1)
lbox = Listbox(window,height=4)
lbox.grid(column=1, row=2)
lbox.insert(tk.END,"Left","Right","Up","Down")

Scale(window,length=200).grid(column=1,row=3)
Progressbar(window,orient="horizontal",length=120,maximum=10,variable=IntVar(root,3)).
 ↪grid(column=1,row=4)
sep = Separator(window, orient="vertical",)
sep.grid(row=1,column=2,rowspan=5,sticky=(tk.S,tk.N))
can = Canvas(window,background="yellow", width=200,height=200)
can.grid(row=1,column=3,rowspan=4)
can.create_oval(10,10,200,100,fill="red",outline="blue")
root.mainloop()
```

0.3 La position des widgets

On pose les éléments sur une grille

[ ]:
```
from tkinter import Tk, Label, Button
root = Tk()
root.geometry('600x400')
Button(root,text="Bonjour").grid(row=1,column=1)
Label(root,text="Ici").grid(row=2,column=1)
Button(root,text="Hello").grid(row=2,column=2)
Label(root,text="Here").grid(row=1,column=2)
root.mainloop()
```

Pour gérer la position interne dans la cellule, on dispose de sticky, padx, pady, ipadx,ipady

[ ]:
```
from tkinter import Tk, Canvas, NE
root = Tk()
root.geometry('600x400')
Canvas(root,width=100,height=100,background="yellow").grid(row=1,column=1)
Canvas(root,width=100,height=100,background="blue").
 ↪grid(row=1,column=2,ipadx=20)
Canvas(root,width=100,height=100,background="green").
 ↪grid(row=2,column=1,padx=30)
Canvas(root,width=100,height=100,background="red").grid(row=2,column=2)
Canvas(root,width=100,height=100,background="orange").
 ↪grid(row=3,column=1,sticky=NE)
Canvas(root,width=100,height=100,background="purple").grid(row=3,column=1)
Canvas(root,width=100,height=100,background="lightblue").
 ↪grid(row=3,column=2,pady=50)
root.mainloop()
```

On peut gérer des petits widgets et des gros (fusion de cellules)

[ ]:
```
from tkinter import Tk, Canvas
root = Tk()
root.geometry('600x400')
Canvas(root,width=100,height=200,background="yellow").
 ↪grid(row=1,column=1,rowspan=2)
Canvas(root,width=100,height=100,background="blue").grid(row=1,column=2)
Canvas(root,width=100,height=100,background="green").grid(row=2,column=2)
Canvas(root,width=200,height=100,background="red").
 ↪grid(row=3,column=1,columnspan=2)
root.mainloop()
```

0.4 Question de style

Il y a une notion de thème, qu'on peut changer. Attention, ça devient vite moche.

[ ]:
```
import tkinter as tk
from tkinter import ttk
root = tk.Tk()
root.geometry("400x200")
root.style = ttk.Style()
root.style.theme_use('classic') #'default' by default
tk.Button(root,text="Ici").grid(row=1,column=1)
#attention, le style ne porte que sur les widget ttk, pas tk
ttk.Button(root, text="Change Theme").grid(row=2,column=1)
root.mainloop()
```

On peut changer le style d'un objet en changeant ses propriétés

[ ]:
```
import tkinter as tk
import tkinter.font as tkfont
from tkinter import ttk
root = tk.Tk()
root.geometry("400x200")
ttk.Button(root, text="Moi", width=10).grid(row=1,column=1)
ttk.Button(root, text="Je").grid(row=2,column=1)
ttk.Entry(root,text="example",show="*").grid(row=3,column=1)
ttk.Entry(root,text="example").grid(row=4,column=1)
ttk.Label(root, text="original", foreground="red", font="Arial").
 ↪grid(row=5,column=1)
ttk.Label(root, text="original",
          font=tkfont.Font(family="Cochin",size=24,weight="normal")).
 ↪grid(row=6,column=1)
root.mainloop()
```

Mais pour une version plus uniforme, on utilise des styles

[ ]:
```
import tkinter as tk
import tkinter.font as tkfont
```

```
from tkinter import ttk
root = tk.Tk()
root.geometry("400x200")
s1 = ttk.Style()
s1.configure("B1.TButton", width=15, foreground='red')
s2 = ttk.Style()
s2.configure("B2.TButton",font =('calibri', 16, 'underline'))
ttk.Button(root, text="Moi", style="B1.TButton").grid(row=1,column=1)
ttk.Button(root, text="Je", style="B2.TButton").grid(row=2,column=1)
ttk.Button(root, text="Moi", style="B1.TButton").grid(row=3,column=1)
ttk.Button(root, text="Je", style="B1.TButton").grid(row=4,column=1)
ttk.Button(root, text="Moi", style="B2.TButton").grid(row=5,column=1)
root.mainloop()
```

0.5 Evènements

Il y en a de toutes sortes, pour chacun, il faut prévoir une fonction dite de callback. La fonction en question va dépendre du type d'évenement attendu. On commence par les évènements dépendant du widget.

[ ]:
```
import tkinter as tk
from tkinter import IntVar
from tkinter.ttk import Button,Scale,Checkbutton
def ici(): print("ici")
def etla(): print("et la")
def ouf(e): print(e)
def fou(): print(f"x = {x.get()}")

root = tk.Tk()
root.geometry("400x200")
Button(root, text="Hello", command=ici).grid(row=1,column=1)
Button(root, text="Halo", command=etla).grid(row=2,column=1)
s = Scale(root, orient='horizontal',value=0.3,variable=IntVar(root),command=ouf)
s.grid(row=3, column=1)
x = IntVar(root)
Checkbutton(root,text='verifie', command=fou,variable=x).grid(row=4,column=1)
root.mainloop()
```

D'autres évènements sont plus généraux : clic de souris, touche clavier, ouverture de fenêtre, etc. Pour une version exhaustive, on se ramènera à la documentation

https://tcl.tk/man/tcl/TkCmd/bind.htm

[ ]:
```
import tkinter as tk
from tkinter.ttk import Button

def ouille(): print("ouch")
def nexte(e): print(e)
```

```
def trait(e): print(e)
def rouge(e): canvas.configure(bg="red")
def vert(e): canvas.configure(bg="green")
root = tk.Tk()
root.geometry("400x400")
canvas = tk.Canvas(root, bg="yellow")
canvas.grid(row=1,column=1)
Button(root,text="Oups", command=ouille).grid(row=2,column=1)
root.bind("<Key>", nexte)
root.bind("<ButtonPress>",trait)
canvas.bind("<Leave>",rouge)
canvas.bind("<Enter>",vert)
root.mainloop()
```

Il n'y a pas un évènement pour le temps. En revanche, un widget peut demander une exécution après un temps donné

[ ]:
```
import tkinter as tk
import time

def refresh():
    horloge.config( text=time.strftime("%H:%M:%S", time.localtime()))
    horloge.after(1000, refresh)

root = tk.Tk()
horloge=tk.Label(root, font="Times 25")
horloge.grid(row=1,column=1)
horloge.after_idle(refresh)
root.mainloop()
```

0.6 De bonnes pratiques

La notation lambda, notation de Church

[ ]:
```
import tkinter as tk
from tkinter.ttk import Button
root = tk.Tk()
root.geometry("400x400")
canvas = tk.Canvas(root, bg="yellow")
canvas.grid(row=1,column=1)
Button(root,text="Oups", command= lambda : print("ouch")).grid(row=2,column=1)
root.bind("<Key>", lambda e : print(e))
root.bind("<ButtonPress>",lambda e : print(e))
canvas.bind("<Leave>",lambda e : canvas.configure(bg="red"))
canvas.bind("<Enter>",lambda e : canvas.configure(bg="green"))
root.mainloop()
```

Du dessin à la main, on utilise la notion d'état du système pour enregistrer l'historique.

[ ]:
```
import tkinter as tk
from tkinter.ttk import Button
class State:
    def __init__(self):
        self.mouse_pressed = False
        self.focus=False
    def press(self):
        self.mouse_pressed=True
    def release(self):
        self.mouse_pressed=False

def is_in(s, b): s.focus = b

def dessine(canvas, state, e):
    if state.focus and state.mouse_pressed:
        canvas.create_line(e.x,e.y,e.x+2,e.y+2)
    else:
        canvas.create_line(e.x,e.y,e.x+2,e.y+2,fill="red")

root = tk.Tk()
root.geometry("400x400")
canvas = tk.Canvas(root, bg="darkblue")
canvas.grid(row=1,column=1)
Button(root,text="Oups", command= lambda : print("ouch")).grid(row=2,column=1)
state = State()
root.bind("<ButtonPress>",lambda e : state.press())
root.bind("<ButtonRelease>",lambda e : state.release())
root.bind("<Motion>",lambda e : dessine(canvas, state, e))
canvas.bind("<Leave>",lambda e : is_in(state, False))
canvas.bind("<Enter>",lambda e : is_in(state, True))
root.mainloop()
```

[ ]:
```

```