Module TKINTER en Python

Concepts importants de *Tk*

widgets

Une interface utilisateur *Tkinter* est composée de *widgets* individuels. Chaque widget est représenté comme un objet Python, instancié à partir de classes comme `ttk.Frame`, `ttk.Label` et `ttk.Button`.

hiérarchie des widgets

Les widgets sont organisés dans une *hiérarchie*. L'étiquette et le bouton étaient contenus dans un cadre, qui à son tour était contenu dans la fenêtre racine. Lors de la création de chaque widget *enfant*, son widget *parent* est passé comme premier argument au constructeur du widget.

options de configuration

Les widgets ont des *options de configuration*, qui modifient leur apparence et leur comportement, comme le texte à afficher dans une étiquette ou un bouton. Différentes classes de widgets auront différents ensembles d'options.

gestion de la géométrie

Les widgets ne sont pas automatiquement ajoutés à l'interface utilisateur lorsqu'ils sont créés. Un *gestionnaire de géométrie* comme `grid` contrôle où ils sont placés dans l'interface utilisateur.

boucle d'événements

Tkinter réagit aux entrées de l'utilisateur, aux modifications de votre programme et même rafraîchit l'affichage uniquement lors de l'exécution active d'une *boucle d'événements*. Si votre programme n'exécute pas la boucle d'événements, votre interface utilisateur n'est pas mise à jour.

Exemple de base :

```
from tkinter import *
from tkinter import ttk
root = Tk()
frm = ttk.Frame(root, padding=10)
frm.grid()
ttk.Label(frm, text="Hello World!").grid(column=0, row=0)
ttk.Button(frm, text="Quit", command=root.destroy).grid(column=1, row=0)
root.mainloop()
```

Créé une instance de Tk

Créé un cadre

Met une grille

Créé une étiquette en 0,0

Place un bouton quitter en 1,0

Affichage et Bouclage d'actualisation

```
from tkinter import *

fenetre = Tk()

label = Label(fenetre, text="Hello World")
label.pack()

fenetre.mainloop()
```

Dans cette fenêtre on a un texte : le label et un bouton : Button

On peut y ajouter des case à cocher :

```
# checkbutton
bouton = Checkbutton(fenetre, text="Nouveau?")
bouton.pack()
```

Des boutons d'option :

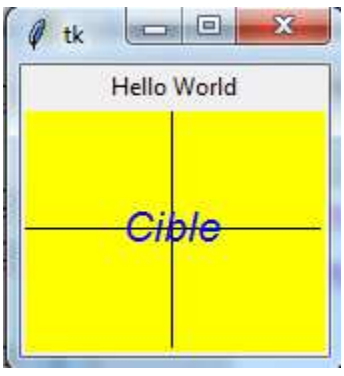
```
# radiobutton
value = StringVar()
bouton1 = Radiobutton(fenetre, text="Oui", variable=value, value=1)
bouton2 = Radiobutton(fenetre, text="Non", variable=value, value=2)
bouton3 = Radiobutton(fenetre, text="Peu être", variable=value, value=3)
bouton1.pack()
bouton2.pack()
bouton3.pack()
```

Des listes déroulantes :

```
# liste
liste = Listbox(fenetre)
liste.insert(1, "Python")
liste.insert(2, "PHP")
liste.insert(3, "jQuery")
liste.insert(4, "CSS")
liste.insert(5, "Javascript")

liste.pack()
```

Des canvas (zone de dessin ou texte) :

```
# canvas
canvas = Canvas(fenetre, width=150, height=120, background='yellow')
ligne1 = canvas.create_line(75, 0, 75, 120)
ligne2 = canvas.create_line(0, 60, 150, 60)
txt = canvas.create_text(75, 60, text="Cible", font="Arial 16 italic",
fill="blue")
canvas.pack()
```




```
create_arc()        :  arc de cercle
create_bitmap()     :  bitmap
create_image()      :  image
create_line()       :  ligne
```

```
create_oval()       :  ovale
create_polygon()    :  polygone
create_rectangle()  :  rectangle
create_text()       :  texte
create_window()     :  fenetre
```

```
modifier les coordonnées :
canvas.coords(élément, x0, y0, x1, y1)
```

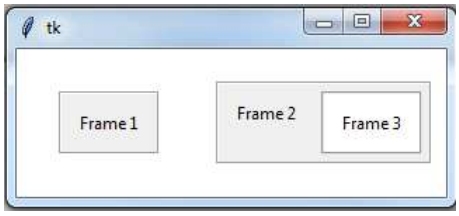
supprimer un élément avec la méthode **delete :**

```
canvas.delete(élément)
```

utiliser un curseur :

```
value = DoubleVar()
scale = Scale(fenetre, variable=value)
scale.pack()
```

Vous pouvez créer des cadres :




```
fenetre['bg']='white'

# frame 1
Frame1 = Frame(fenetre, borderwidth=2, relief=GROOVE)
Frame1.pack(side=LEFT, padx=30, pady=30)
# frame 2
Frame2 = Frame(fenetre, borderwidth=2, relief=GROOVE)
Frame2.pack(side=LEFT, padx=10, pady=10)
# frame 3 dans frame 2
Frame3 = Frame(Frame2, bg="white", borderwidth=2, relief=GROOVE)
Frame3.pack(side=RIGHT, padx=5, pady=5)

# Ajout de labels
Label(Frame1, text="Frame 1").pack(padx=10, pady=10)
Label(Frame2, text="Frame 2").pack(padx=10, pady=10)
Label(Frame3, text="Frame 3",bg="white").pack(padx=10, pady=10)
```

Spinbox : pour choisir un nombre :

```
s = Spinbox(fenetre, from_=0, to=10)
s.pack()
```

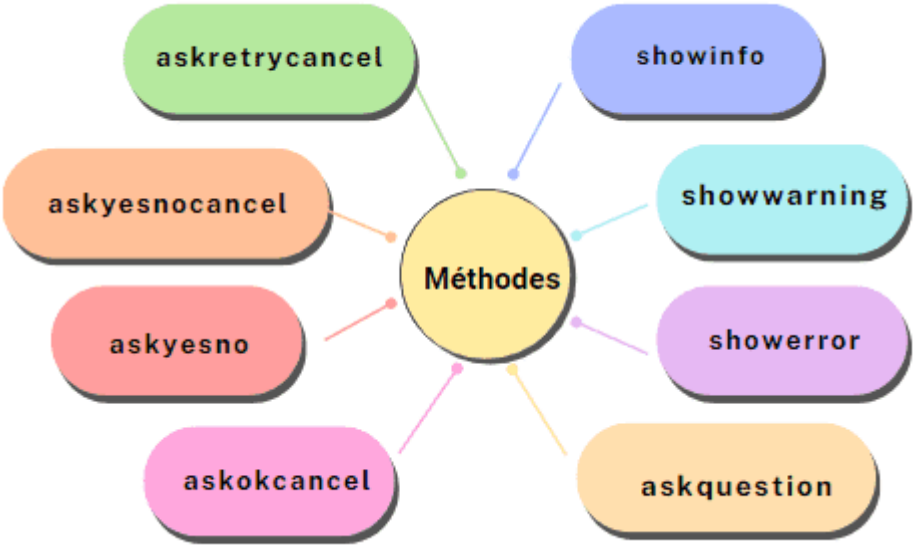
Le label Frame :

```
l = LabelFrame(fenetre, text="Titre de la frame", padx=20, pady=20)
l.pack(fill="both", expand="yes")

Label(l, text="A l'intérieure de la frame").pack()
```

Les fenetre d'alerte :

```
from tkinter.messagebox import *

def callback():
    if askyesno('Titre 1', 'Êtes-vous sûr de vouloir faire ça?'):
        showwarning('Titre 2', 'Tant pis...')
    else:
        showinfo('Titre 3', 'Vous avez peur!')
        showerror("Titre 4", "Aha")
```

```
Button(text='Action', command=callback).pack()
```



Tkinter showinfo : Afficher des infos

Tkinter showwarning : Avertir d'un problème

Tkinter showerror : Signaler une erreur

Tkinter askquestion : Question Oui/Non

Tkinter askokcancel : Confirmation OK/Annuler

Tkinter askyesno : Décision binaire Oui/Non

Barre de menu :

```
def alert():
    showinfo("alerte", "Bravo!")

menubar = Menu(fenetre)

menu1 = Menu(menubar, tearoff=0)
menu1.add_command(label="Créer", command=alert)
menu1.add_command(label="Editer", command=alert)
menu1.add_separator()
menu1.add_command(label="Quitter", command=fenetre.quit)
menubar.add_cascade(label="Fichier", menu=menu1)

menu2 = Menu(menubar, tearoff=0)
menu2.add_command(label="Couper", command=alert)
menu2.add_command(label="Copier", command=alert)
menu2.add_command(label="Coller", command=alert)
menubar.add_cascade(label="Editer", menu=menu2)

menu3 = Menu(menubar, tearoff=0)
menu3.add_command(label="A propos", command=alert)
menubar.add_cascade(label="Aide", menu=menu3)

fenetre.config(menu=menubar)
```

Pour connaître les options des commandes :

```
print dir(Button())
```

On peut placer les widgets :

```
side=TOP     :  haut
side=LEFT    :  gauche
side=BOTTOM  :  bas
side=RIGHT   :  droite
```

exemple : un bouton en bas à gauche l'autre en bas à droite,

```
Canvas(fenetre, width=250, height=100, bg='ivory').pack(side=TOP, padx=5,
pady=5)
Button(fenetre, text ='Bouton 1').pack(side=LEFT, padx=5, pady=5)
Button(fenetre, text ='Bouton 2').pack(side=RIGHT, padx=5, pady=5)
```

Les unités de dimensions : par défaut c'est le pixel mais on peut utiliser le mm : m ou le cm : c

Les options de dimensions :

height : Hauteur du widget.
width : Largeur du widget.
padx, pady : Espace supplémentaire autour du widget. X pour horizontal et V pour vertical.
borderwidth : Taille de la bordure.
highlightthickness : Largeur du rectangle lorsque le widget a le focus.

selectborderwidth : Largeur de la bordure tridimensionnel autour du widget sélectionné.
wraplength : Nombre de ligne maximum pour les widget en mode "word wrapping".

Les options de couleur :

background (ou bg) : couleur de fond du widget.
foreground (ou fg) : couleur de premier plan du widget.
activebackground : couleur de fond du widget lorsque celui-ci est actif.
activeForeground : couleur de premier plan du widget lorsque le widget est actif.
disabledForeground : couleur de premier plan du widget lorsque le widget est désactivé.
highlightbackground : Couleur de fond de la région de surbrillance lorsque le widget a le focus.
highlightcolor : couleur de premier plan de la région en surbrillance lorsque le widget a le focus.
selectbackground : Couleur de fond pour les éléments sélectionnés.
selectforeground : couleur de premier plan pour les éléments sélectionnés.

Le curseur : on peut modifier son aspect,

```
Button(fenetre, text ="arrow", relief=RAISED, cursor="arrow").pack()
Button(fenetre, text ="circle", relief=RAISED, cursor="circle").pack()
Button(fenetre, text ="clock", relief=RAISED, cursor="clock").pack()
Button(fenetre, text ="cross", relief=RAISED, cursor="cross").pack()
Button(fenetre, text ="dotbox", relief=RAISED, cursor="dotbox").pack()
Button(fenetre, text ="exchange", relief=RAISED, cursor="exchange").pack()
Button(fenetre, text ="fleur", relief=RAISED, cursor="fleur").pack()
Button(fenetre, text ="heart", relief=RAISED, cursor="heart").pack()
Button(fenetre, text ="man", relief=RAISED, cursor="man").pack()
Button(fenetre, text ="mouse", relief=RAISED, cursor="mouse").pack()
Button(fenetre, text ="pirate", relief=RAISED, cursor="pirate").pack()
Button(fenetre, text ="plus", relief=RAISED, cursor="plus").pack()
Button(fenetre, text ="shuttle", relief=RAISED, cursor="shuttle").pack()
Button(fenetre, text ="sizing", relief=RAISED, cursor="sizing").pack()
Button(fenetre, text ="spider", relief=RAISED, cursor="spider").pack()
Button(fenetre, text ="spraycan", relief=RAISED, cursor="spraycan").pack()
Button(fenetre, text ="star", relief=RAISED, cursor="star").pack()
Button(fenetre, text ="target", relief=RAISED, cursor="target").pack()
Button(fenetre, text ="tcross", relief=RAISED, cursor="tcross").pack()
Button(fenetre, text ="trek", relief=RAISED, cursor="trek").pack()
Button(fenetre, text ="watch", relief=RAISED, cursor="watch").pack()
```

Le relief des boutons : en creux en saillie, …

```
b1 = Button(fenetre, text ="FLAT", relief=FLAT).pack()
b2 = Button(fenetre, text ="RAISED", relief=RAISED).pack()
b3 = Button(fenetre, text ="SUNKEN", relief=SUNKEN).pack()
b4 = Button(fenetre, text ="GROOVE", relief=GROOVE).pack()
b5 = Button(fenetre, text ="RIDGE", relief=RIDGE).pack()
```

Image : intégrer une image en utilisant un canvas

```
photo = PhotoImage(file="ma_photo.png")

canvas = Canvas(fenetre,width=350, height=200)
canvas.create_image(0, 0, anchor=NW, image=photo)
canvas.pack()
```

Pour récupérer une valeur : méthode get,

```
def recupere():
    showinfo("Alerte", entree.get())

value = StringVar()
value.set("Valeur")
entree = Entry(fenetre, textvariable=value, width=30)
entree.pack()

bouton = Button(fenetre, text="Valider", command=recupere)
bouton.pack()
```

Charger une image :

Renvoie le chemin et le nom de fichier choisi

```
from tkinter.filedialog import *

filepath = askopenfilename(title="Ouvrir une image",filetypes=[('png
files','.png'),('all files','.*')])
photo = PhotoImage(file=filepath)
canvas = Canvas(fenetre, width=photo.width(), height=photo.height(),
bg="yellow")
canvas.create_image(0, 0, anchor=NW, image=photo)
canvas.pack()
```

charger un fichier texte :

```
filename = askopenfilename(title="Ouvrir votre document",filetypes=[('txt
files','.txt'),('all files','.*')])
fichier = open(filename, "r")
content = fichier.read()
fichier.close()

Label(fenetre, text=content).pack(padx=10, pady=10)
```

Les évènements

Vous pouvez récupérer les actions utilisateurs à travers les **events** (évènement en français).

```
def clavier(event):
    touche = event.keysym
    print(touche)

canvas = Canvas(fenetre, width=500, height=500)
canvas.focus_set()
canvas.bind("<Key>", clavier)
canvas.pack()
<Button-1>           : Click gauche
<Button-2>           : Click milieu
<Button-3>           : Click droit
<Double-Button-1>    : Double click droit
<Double-Button-2>    : Double click gauche
<KeyPress>           : Pression sur une touche
<KeyPress-a>         : Pression sur la touche A (minuscule)
<KeyPress-A>         : Pression sur la touche A (majuscule)
<Return>             : Pression sur la touche entrée
<Escape>             : Touche Echap
<Up>                 : Pression sur la flèche directionnelle haut
<Down>               : Pression sur la flèche directionnelle bas
<ButtonRelease>      : Lorsque qu'on relache le click
<Motion>             : Mouvement de la souris
<B1-Motion>          : Mouvement de la souris avec click gauche
<Enter>              : Entrée du curseur dans un widget
<Leave>              : Sortie du curseur dans un widget
<Configure>          : Redimensionnement de la fenêtre
<Map> <Unmap>        : Ouverture et iconification de la fenêtre
<MouseWheel>         : Utilisation de la roulette
```

Créez un court programme qui dessinera les 5 anneaux olympiques dans un rectangle de fond blanc. Un bouton « Quitter » doit permettre de fermer la fenêtre.

```python
# Créé par claude, le 09/11/2025 en Python 3.7
import tkinter as tk

def quitter():
    fenetre.destroy()

# Création de la fenêtre principale
fenetre = tk.Tk()
fenetre.title("Anneaux Olympiques")

# Création du canevas (zone de dessin)
canvas = tk.Canvas(fenetre, width=400, height=250, bg="white")
canvas.pack(pady=10)

# --- Coordonnées et couleurs des anneaux ---
anneaux = [
    (50, 50, "blue"),
    (150, 50, "black"),
    (250, 50, "red"),
    (100, 100, "yellow"),
    (200, 100, "green")
]

# --- Dessin des anneaux ---
for (x, y, couleur) in anneaux:
    canvas.create_oval(x, y, x + 80, y + 80, width=5, outline=couleur)

# --- Bouton Quitter ---
bouton_quitter = tk.Button(fenetre, text="Quitter", command=quitter, bg="lightgray")
bouton_quitter.pack(pady=10)

# --- Lancement de la boucle principale ---
fenetre.mainloop()
```