Tkinter en Python : Création d'interfaces graphiques

Par Robert DIASSÉ

1. Introduction à Tkinter

Qu'est-ce que Tkinter ?

Tkinter est la bibliothèque graphique incluse avec Python. Elle permet de créer des fenêtres, des boutons, des champs de saisie et bien d'autres éléments graphiques sans dépendances supplémentaires.

Installation et importation

Tkinter est inclus par défaut dans Python. Pour vérifier si Tkinter est installé :

```
import tkinter as tk
print("Tkinter est bien installé !")
```

Si vous obtenez une erreur, installez Tkinter via votre gestionnaire de paquets :

- **Windows** : installé par défaut avec Python.
- **Linux** (Debian/Ubuntu) : `sudo apt install python3-tk`
- **MacOS** : installé par défaut.

---

2. Création d'une fenêtre simple

Code minimal pour afficher une fenêtre Tkinter

```
import tkinter as tk

# Création de la fenêtre principale
fenetre = tk.Tk()
fenetre.title("Première fenêtre Tkinter")
fenetre.geometry("400x300")

# Lancer la boucle d'événements
fenetre.mainloop()
```

1. `tk.Tk()` : crée une fenêtre principale.
2. `.title("Texte")` : définit le titre de la fenêtre.
3. `.geometry("400x300")` : fixe la taille en pixels.

4. `.mainloop()` : maintient la fenêtre ouverte et écoute les événements (clics, saisies...).

---

3. Ajout de widgets (éléments d'interface)

Étiquettes (`Label`)

```
import tkinter as tk

fenetre = tk.Tk()
fenetre.title("Étiquette Tkinter")

# Création d'une étiquette
label = tk.Label(fenetre, text="Bienvenue dans Tkinter", font=("Arial", 14))
label.pack()

fenetre.mainloop()
```

- `tk.Label()` : crée une étiquette.
- `text="..."` : définit le texte affiché.
- `font=("Arial", 14)` : applique une police et une taille.
- `.pack()` : affiche l'élément (méthode d'affichage simple).

Champs de saisie (`Entry`)

```
import tkinter as tk

fenetre = tk.Tk()
fenetre.title("Champ de saisie")

# Zone de texte
saisie = tk.Entry(fenetre, width=30)
saisie.pack()

fenetre.mainloop()
```

- `tk.Entry()` : crée une zone de saisie.
- `width=30` : définit la largeur en caractères.

Boutons (`Button`) et gestion des événements

```
import tkinter as tk

fenetre = tk.Tk()
fenetre.title("Bouton interactif")
```

```
def cliquer():
    label.config(text="Bouton cliqué !")

label = tk.Label(fenetre, text="Cliquez sur le bouton", font=("Arial", 14))
label.pack()

# Bouton
bouton = tk.Button(fenetre, text="Cliquez-moi", command=cliquer)
bouton.pack()

fenetre.mainloop()
```

- `command=cliquer` : relie le bouton à la fonction `cliquer()`.
- `label.config(text="...")` : modifie dynamiquement le texte du label.

---

4. Organisation des widgets : Gestion des mises en page

Tkinter propose plusieurs méthodes pour organiser les widgets :

1. `.pack()` : Disposition automatique

Dispose les widgets verticalement (ou horizontalement).

```
label.pack(side="top")
bouton.pack(side="bottom")
```

2. `.grid()` : Placement en tableau

```
label.grid(row=0, column=0)
bouton.grid(row=1, column=0)
```

3. `.place()` : Position absolue

```
label.place(x=50, y=100)
```

---

5. Création d'une application interactive complète

Objectif : Un mini formulaire avec affichage dynamique

L'utilisateur saisit son nom dans un champ, et lorsqu'il appuie sur un bouton, un message personnalisé s'affiche.

Code complet :

```
import tkinter as tk

# Fonction pour afficher le message
def afficher_message():
    nom = saisie.get()
    label_message.config(text=f"Bonjour, {nom} !")

# Création de la fenêtre
fenetre = tk.Tk()
fenetre.title("Formulaire interactif")
fenetre.geometry("300x200")

# Étiquette
label = tk.Label(fenetre, text="Entrez votre nom :", font=("Arial", 12))
label.pack()

# Champ de saisie
saisie = tk.Entry(fenetre, width=20)
saisie.pack()

# Bouton
bouton = tk.Button(fenetre, text="Valider", command=afficher_message)
bouton.pack()

# Zone d'affichage du message
label_message = tk.Label(fenetre, text="", font=("Arial", 12))
label_message.pack()

fenetre.mainloop()
```

---

6. Exercice guidé : Création d'un convertisseur de devises

Objectif :
Créer une interface graphique qui permet de convertir une somme d'euros en dollars.

1. Décomposition du problème

- Un champ pour entrer la somme en euros.
- Un bouton pour déclencher la conversion.
- Une étiquette pour afficher le résultat.
- Un taux de conversion fixe (ex : 1€ = 1.1$).

2. Code étape par étape

Création de la fenêtre et des widgets

```
import tkinter as tk

fenetre = tk.Tk()
fenetre.title("Convertisseur Euro -> Dollar")
fenetre.geometry("300x200")
```

Ajout des éléments de l'interface

```
label = tk.Label(fenetre, text="Entrez un montant en euros :", font=("Arial", 12))
label.pack()

saisie = tk.Entry(fenetre, width=10)
saisie.pack()

label_resultat = tk.Label(fenetre, text="", font=("Arial", 12))
label_resultat.pack()
```

Définition de la fonction de conversion

```
def convertir():
    try:
        euros = float(saisie.get())
        dollars = euros * 1.1
        label_resultat.config(text=f"{euros} € = {dollars:.2f} $")
    except ValueError:
        label_resultat.config(text="Entrez un nombre valide")
```

Ajout du bouton

```
bouton = tk.Button(fenetre, text="Convertir", command=convertir)
bouton.pack()
```

Finalisation et affichage

```
fenetre.mainloop()
```

---

7. Exercices supplémentaires

1. **Ajouter un champ pour convertir en plusieurs devises (USD, GBP, JPY).**
2. **Créer une application de calculatrice simple avec Tkinter.**

3. **Créer un gestionnaire de tâches où l'utilisateur peut ajouter et supprimer des éléments d'une liste.**

Conclusion

Tkinter est une excellente première approche des interfaces graphiques en Python. Avec ces bases, vous pouvez créer des interfaces interactives et bien structurées. La prochaine étape serait d'explorer des concepts avancés comme la gestion des fichiers, la connexion à une base de données ou encore l'amélioration de l'ergonomie de vos applications.