Python pour les data scientistes

Chapitre 8 : Le package Tkinter

1. Introduction

Tkinter est un module de base intégré dans Python**,** normalement vous n'avez rien à faire pour pouvoir l'utiliser. L'un des avantages de **Tkinter** est sa portabilité sur les OS les plus utilisés par le grand public. Cette librairie permet de créer des interfaces graphiques.

La conception d'une interface graphique se déroule généralement selon deux étapes. La première consiste à dessiner l'interface, c'est-à-dire choisir une position pour les objets de la fenêtre (boutons, zone de saisie, liste déroulante, …). La seconde étape définit le fonctionnement de la fenêtre, c'est-à-dire associer à chaque objet des fonctions qui seront exécutées si un tel événement se réalise (pression d'un bouton, pression d'une touche, …).

2. Installation de Tkinter

Tkinter est installé par défaut, si ce n'est pas le cas, lancez la commande suivante:

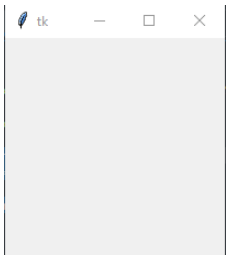
pip install tkinter

3. La fenêtre principale

Pour créer un logiciel graphique vous devez ajouter dans une fenêtre des éléments graphiques que l'on nomme widget. Ce widget peut être tout aussi bien une liste déroulante que du texte.

Exemple : Une fenêtre

```
import tkinter
fenetre = tkinter.Tk()
#  On met ici le code qui définit les objets
#  et leur positionnement (bouton, one de texte,  ..etc.
fenetre.mainloop ()
```



La première ligne permet d'obtenir un identificateur relié à la fenêtre principale. La seconde ligne, outre le fait qu'elle affiche cette fenêtre, lance ce qu'on appelle une boucle de messages.

En utilisant la classe **Tk** importée du module **tkinter**, on crée un objet qui va représenter la fenêtre principale de notre application :

Exemple : L'importation de l'ensemble des éléments du paquet tkinter

```
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# En utilisant l'instruction « from tkinter import * », on peut appeler les éléments du module
directement.
```

Ou bien (pour illustrer la différence des deux syntaxes de la partie précédente) :

Exemple : L'importation de l'ensemble des éléments du paquet tkinter

```
import tkinter
# Création d'une fenêtre avec la classe Tk :
fenetre = tkinter.Tk()
```

En utilisant l'instruction « import tkinter », on définit le module avant d'appeler un de ses éléments.

- On peut changer le titre de la fenêtre avec :

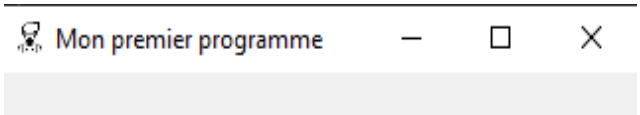
```
fenetre.title("Mon premier programme")
```

Ici fenêtre représente le nom de l'objet fenêtre que nous avons créée.

- Pour définir un logo personnalisé à la fenêtre principale, on utilise la méthode **iconbitmap()** :

```
fenetre.iconbitmap("monlogo.ico")
```

Notez bien qu'il faut convertir le logo qu'on veut définir en extension **.ico** , et l'ajouter dans le dossier du projet.



- Pour configurer la couleur de l'arrière-plan, on utilise la méthode *config()*, contenant comme argument, le paramètre *bg* ou *background* qu'on lui affecte la valeur de la couleur tel que *bg = 'red'* ou *bg = 'blue'* ou le code hexadécimal de la couleur en 4, 8, 12 ou 16 bits tel que : *bg = '#808000'* pour la couleur jaune ou *bg = '#87CEEB'* pour la couleur bleu ciel :

```
fenetre.config(bg = "#87CEEB")
```

- Pour personnaliser la taille d'affichage par défaut de la fenêtre principale, on utilise la méthode *geometry()*, en ayant comme attribut la largeur x la hauteur en pixels :

```
fenetre.geometry("640x480")
```

Le module tkinter comporte plusieurs widgets qui sont des composants graphiques qu'on peut ajouter à la fenêtre tels que : des boutons Button, des cases à cocher Checkbutton , des boutons radio Radiobutton , des textes ou images Label , des listes Listbox , des cadres Frame, un … etc. Chacun de ces widgets comporte des paramètres permettant de personnaliser leurs couleurs de l'arrière-plan ou du texte, leurs dimensions, la police de leurs caractères... etc. Ainsi, pour les organiser et les placer, on utilise une des méthodes suivantes :

- pack() qui permet d'organiser et placer des widgets en blocs.
- grid() qui permet d'organiser et placer des widgets en étant comme des cases de tableau.
- place() qui permet d'organiser et placer des widgets dans un endroit précisé.

4. Les boutons

Un bouton a pour but de faire le lien entre une fonction et un clic de souris. Un bouton correspond à la classe **Button**. Le code suivant permet d'afficher un bouton dans une fenêtre avec un titre :

Exemple : Un bouton dans une fenêtre

```
import tkinter
fenetre = tkinter.Tk()
bouton = tkinter.Button (text = "valider")
bouton.pack()
fenetre.mainloop ()
```

5. Les labels

Les labels sont des espaces prévus pour écrire du texte. Les labels servent souvent à décrire un widget comme un input.

Exemple : Un label pour afficher un titre

```
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Ajout d'un texte dans la fenêtre :
texte1 = Label (fenetre, text = "Ceci est un Label")
texte1.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```

6. Les frames

Le widget « Frame » sert à créer un cadre ou une zone sur laquelle on peut placer, grouper et

```
Cadre1 = Frame (fenetre, les options du widget Frame)
```

Exemple : Un frame avec des boutons

```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création d'un cadre dans la fenêtre :
cadre1 = Frame(fenetre)
cadre1.pack()
# Ajout de boutons dans le cadre :
bouton1 = Button (cadre1, text = "bouton1")
bouton2 = Button (cadre1, text = "bouton2")
bouton3 = Button (cadre1, text = "bouton3")
bouton1.pack()
bouton2.pack()
bouton3.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```

7. Les champs de saisie

Le widget « Entry » sert à implémenter un champ sur lequel l'utilisateur peut saisir un texte, ou des données, sa syntaxe s'écrit sous la forme :

entre = Entry (fenetre, les options du widget Entry)

Exemple : une zone de saisie

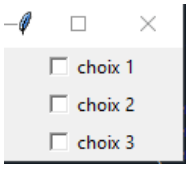
```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création d'un champ de saisie de l'utilisateur dans la fenêtre :
entrée1 = Entry (fenetre)
entrée1.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```

8. Case à cocher

Le widget « Checkbutton » sert à afficher un certain nombre d'options sous forme de cases à cocher que l'utilisateur peut en sélectionner plusieurs, et sa syntaxe s'écrit sous la forme :

Exemple : des cases à cocher

```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création des cases à cocher "Checkbutton" dans la fenêtre :
case_cocher1 = Checkbutton (fenetre, text = "choix 1")
case_cocher2 = Checkbutton (fenetre, text = "choix 2")
case_cocher3 = Checkbutton (fenetre, text = "choix 3")
case_cocher1.pack()
case_cocher2.pack()
case_cocher3.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```

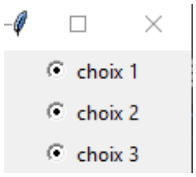


9. Bouton radio

Les boutons radio sont des cases à cocher qui sont dans un groupe et dans ce groupe seul un élément peut être sélectionné.

Exemple : des boutons radio

```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création des cases à cocher "Radiobutton" dans la fenêtre :
case_cocher1 = Radiobutton (fenetre, text = "choix 1")
case_cocher2 = Radiobutton (fenetre, text = "choix 2")
case_cocher3 = Radiobutton (fenetre, text = "choix 3")
case_cocher1.pack()
case_cocher2.pack()
case_cocher3.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```



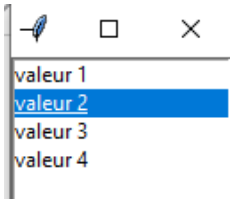
10. Les Listes

Les listes servent à fournir à l'utilisateur une liste d'options, sa syntaxe :

```
liste1 = Listbox (fenetre, les options du widget Listbox)
```

Exemple : Une liste

```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création d'une liste dans la fenêtre :
# la méthode insert() sert à inserer des valeur à la liste :
liste1 = Listbox (fenetre)
liste1.insert(1, "valeur 1")
liste1.insert(2, "valeur 2")
liste1.insert(3, "valeur 3")
liste1.insert(4, "valeur 4")
liste1.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```



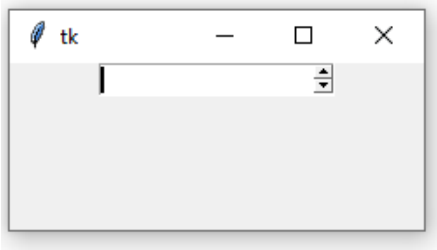
11. Les Spinbox

Le widget « Spinbox » sert à ajouter un champ de saisie d'utilisateur, ou bien, à ajouter un bouton sur lequel l'utilisateur peut choisir une valeur dans une intervalle précise, et sa syntaxe s'écrit sous la forme :

spin1 = Spinbox (fenetre, les options du widget Spinbox)

Exemple : Un Spinbox

```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création d'un Spinbox dans la fenêtre :
liste1 = Spinbox (fenetre)
liste1.pack()
# Affichage de la fenêtre créée :
fenetre.mainloop()
```



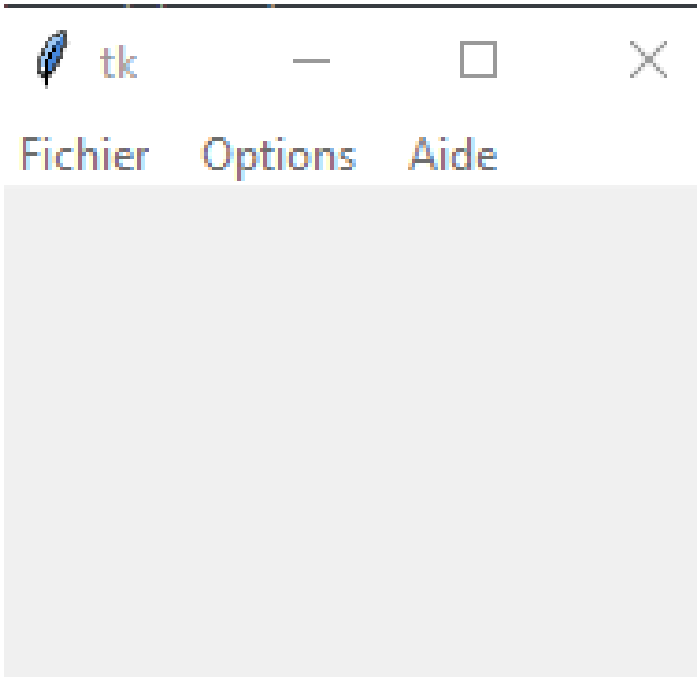
12. Les menus

Le widget « Menubutton » sert à configurer une barre de menu sur laquelle l'utilisateur peut choisir des options, et sa syntaxe s'écrit sous la forme :

menu1 = Menu (fenetre, les options du widget Menubutton)

Exemple : Un menu

```
# L'importation de l'ensemble des éléments du paquet tkinter :
from tkinter import *
# Création d'une fenêtre avec la classe Tk :
fenetre = Tk()
# Création d'une barre de menu dans la fenêtre :
menu1 = Menu(fenetre)
menu1.add_cascade(label="Fichier")
menu1.add_cascade(label="Options")
menu1.add_cascade(label="Aide")
# Configuration du menu dans la fenêtre
fenetre.config(menu = menu1)
# Affichage de la fenêtre créée :
fenetre.mainloop()
```


tk
Fichier
Options
Aide