Premier pas avec Tkinter

Le module Tkinter est un puissant outil permettant, dans un script python, d'insérer des éléments graphiques et dynamiques.
La réalisation d'éléments graphiques va se faire à l'aide de widgets, permettant d'intégrer des boutons, des textes , des cadres, des listes …
La structure d'un bout de code sera toujours la même :

```
from tkinter import *
fen=Tk()    # on ouvre une fenêtre
……
fen.mainloop()  # cette instruction permet de faire tourner le script et de réceptionner les instructions qui sont données dans le programme.
```

Tout d'abord, on peut personnaliser notre fenêtre *fen*,
Il existe des méthodes pour cela : Donner un titre (title), définir des dimensions (geometry)
Une méthode s'utilise avec la syntaxe suivante :

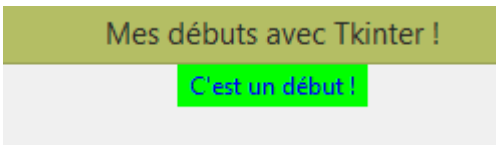
```
fen.title ('Mes débuts avec Tkinter !')
fen.geometry('400*300')
```

Alors, que met-on dans cette fenêtre graphique ? Pleins de choses !
Des boutons, des textes, des listes déroulantes, des images … On va utiliser des widget , En voici quelques-uns.

Quelques exemples :

Un texte : Utilisation de la classe Label
Voici la structure qui permet de créer un texte dans une fenêtre graphique :

```
l=Label(fen,text=" C'est un début ! ",bg='green',fg='blue')
l.pack()
```

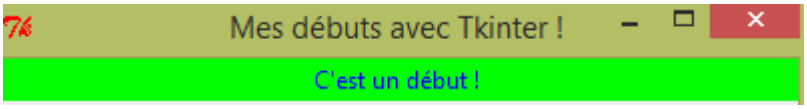


Explications :
Ligne l, c'est le widget , il appartient à la classe Label. Ce widget appartiendra à la fenêtre fen . Ces attributs seront le texte (text=), la couleur du texte (fg) et la couleur de fond (bg), On aurait pu aussi avoir des attributs comme la taille de la police utilisée, la largeur, la hauteur....

Ligne 2 : On utilise la méthode pack pour positionner le widget dans la fenêtre
Par défaut, les widgets se positionnent les uns en dessous des autres.
On peut utiliser des attributs pour modifier le positionnement par défaut :

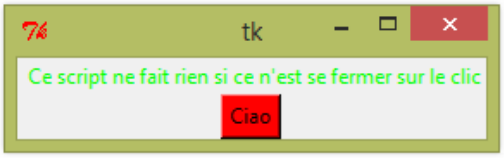
```
side=TOP     :  haut
side=LEFT    :  gauche
side=BOTTOM  :  bas
side=RIGHT   :  droite
```

Il y a d'autres attributs comme **padx et pady** qui permettent de laisser une marge à l'intérieur du widget.
fill permet de remplir toute la fenêtre avec le widget : fill= BOTH, on remplit horizontalement et verticalement, fill=X, horizontalement, fill= Y verticalement



Un bouton : Utilisation de la classe **Button**
Le principe de fonctionnement est le même que pour la classe Label, Ici, lorsque l'on appuiera sur le bouton , une action s'exécutera, la plus simple étant de fermer la fenêtre. On utilise la méthode destroy , Comme toutes les méthodes , elle est appliquée à un objet, ici la fenêtre, On verra plus loin qu'un bouton peut servir à autre chose.

```
from tkinter import *
fen=Tk()
tex=Label(fen, text=" Ce script ne fait rien si ce n'est se fermer sur le clic",fg='green')
tex.pack()
bou= Button(fen, text ="Ciao",bg='red', command = fen.destroy)
bou.pack()
fen.mainloop()
```



Une zone de saisie: Utilisation de la classe **Entry**
Ici, on va créer une zone où l'on pourra saisir des informations
e= Entry(fen)
Il sera bien sur intéressant de récupérer ce que l'on a entré dans la zone de saisie,Pour cela on utilisera la méthode get (appliquée au widget « e »)

```
from tkinter import *
fen=Tk()
tex=Label(fen, text=" Entrez votre prénom",fg='green')
tex.pack(side=LEFT)
e=Entry(fen)
e.pack(side=RIGHT)

fen.mainloop()
```

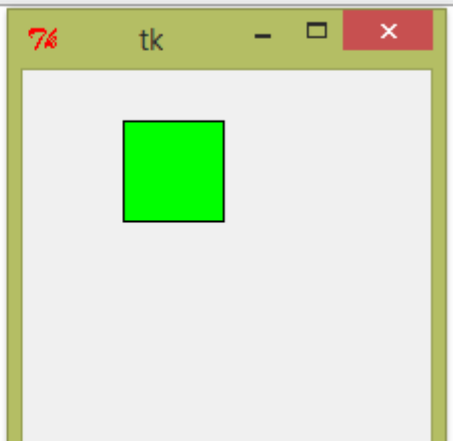
Une classe pour afficher des images, faire des dessins : Canvas

Ici, il y a beaucoup de possibilités et quantité de méthodes sont disponibles pour obtenir le dessin voulu.

Voici un exemple pour créer un rectangle

```
from tkinter import *
fen=Tk()

w=Canvas(fen,width='200',height='200')
w.pack()
w.create_rectangle(50,25,100,75,fill='green')
fen.mainloop()
```



Les coordonnées données sont celles des points en haut à gauche et en bas à droite. A noter que le (0,0) correspond au point en haut à gauche du cadre.