

OBJECTIFS : Les objectifs de ce TP sont de manipuler la structure python des dictionnaires et de découvrir les fonctionnalités graphiques de la bibliothèque *Tkinter* .

L'évaluation de ce travail est basée sur le rendu du fichier *.py* qui sera constitué. **On ne demande pas** de constituer de fichier texte contenant des copies d'écran.

⇒ Pour débiter, télécharger le dossier *tp11.zip* contenant les 27 fichiers images qui seront utilisés. Le dézipper dans votre espace personnel sur *U:* . Ouvrir un nouveau fichier *.py* dans le dossier contenant ces images et le sauvegarder sous le nom ***tp11.py*** .

1. PREPARER SON FICHER .py :

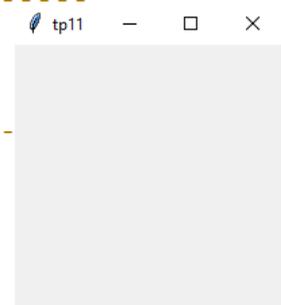
⇒ Organiser son fichier en créant 3 compartiments (*Modules, Fonctions et Main*), comme sur la figure ci-dessous. On y importera :

- A partir de la bibliothèque Tkinter, les classes :
 - **Tk** : pour pouvoir créer une fenêtre Tkinter,
 - **Canvas** : pour pouvoir créer dans cette fenêtre, une zone contenant des images, formes géométriques,
 - **Label** : pour pouvoir créer dans cette fenêtre, des zones textes,
 - **Text** : pour pouvoir y créer des champs de saisie,
 - **Button** : pour pouvoir y créer des boutons.
- A partir de la bibliothèque *PIL*, les classes *Image* et *ImageTk* pour pouvoir manipuler des images,
- A partir de la bibliothèque *random*, la fonction *randint* pour pouvoir générer des nombres aléatoires.

```
# Modules -----
from tkinter import Tk , Canvas , Label , Text , Button
from PIL import Image, ImageTk # pip install pillow
from random import randint

# Fonctions -----

# Main -----
```



2. CREER UNE FENETRE tkinter :

⇒ Définir la fonction *creer_fenetre()* . L'appeler dans la partie *Main*.

```
# Fonctions -----
def creer_fenetre() :
    fenetre = Tk()
    fenetre.title("tp11")
    return fenetre

# Main -----
fenetre = creer_fenetre()

fenetre.mainloop()
```

On crée un objet tkinter (Tk) que l'on appelle ici : *fenetre*

Permet de rajouter un titre en haut de la fenêtre

On retourne cet objet afin qu'il puisse être utilisé dans le programme principal

Permet de maintenir la fenêtre ouverte

⇒ Exécuter ce fichier. La fenêtre créée apparaît. Exécuter ensuite dans le shell : `>>> type(fenetre)` pour vous apercevoir que la variable *fenetre* correspond à un objet. Cet objet est définie dans le programme principal par retour de la fonction *créer_fenetre()*. Cette variable *fenetre* devient ainsi une **variable GLOBALE** et on pourra y accéder n'importe où dans le script, **même à l'intérieur des fonctions**.

3. CREER DES WIDGETS dans cette fenêtre tkinter :

⇒ Définir la fonction *créer_widgets()* donnée ci-dessous. L'appeler dans la partie *Main* :

```
# Fonctions -----
def créer_fenetre() :
    fenetre = Tk()
    fenetre.title("tp11")
    return fenetre

def créer_widgets() :
    zone_graphique = Canvas(fenetre, width=1000, height=600 , bg = 'black')
    zone_graphique.grid(row = 0 , column = 0 , columnspan = 3 )

    return zone_graphique

# Main -----

fenetre = créer_fenetre()
zone_graphique = créer_widgets()

fenetre.mainloop()
```

La variable *fenetre* ayant le statut de **variable GLOBALE**, elle peut être lue et modifiée à l'intérieur des fonctions, sans que l'on ait à la passer en argument.

⇒ Exécuter ce fichier.

Pour l'instant on crée seulement une zone graphique de 1000 px de large par 600 px de haut, avec un fond d'écran noir. Cette zone est un objet de la classe *Canvas* qui est stocké ici dans la variable nommée *zone_graphique*. Cet objet est positionné dans la fenêtre *Tkinter* avec la méthode *grid()* sur la 1^{ère} colonne (column = 0) et la 1^{ère} ligne (row = 0). On indique que cette colonne sera large et qu'elle occupera la largeur des 3 colonnes qui seront placées ensuite (columnspan = 3).

⇒ On continue ci-dessous, à compléter le script de la fonction `creer_widgets()`, en insérant sur la seconde ligne (`row = 1`), le texte « *Entre un mot :* » qui est un objet de la classe *Label*. Cet objet est stocké dans la variable `mon_texte`.

```
def creer_widgets() :
    zone_graphique = Canvas(fenetre, width=1000, height=600 , bg = 'black')
    zone_graphique.grid(row = 0 , column = 0 , columnspan = 3 )

    mon_texte = Label(fenetre, text = "Entre un mot : ")
    mon_texte.grid(row = 1 , column = 0)

    return zone_graphique, mon_texte
```

```
fenetre = creer_fenetre()
zone_graphique, mon_texte = creer_widgets()
```

⇒ Exécuter ce fichier

⇒ Compléter encore le script de cette fonction `creer_widgets()` en insérant sur la seconde ligne (`row = 1`) et la seconde colonne (`column = 1`) un champ de saisie qui permettra de **saisir** un texte sur le clavier. Ce champ de saisie est un objet de la classe *Text*, qui est stocké ici dans la variable `champ_saisie`.

```
def creer_widgets() :
    zone_graphique = Canvas(fenetre, width=1000, height=600 , bg = 'black')
    zone_graphique.grid(row = 0 , column = 0 , columnspan = 3 )

    mon_texte = Label(fenetre, text = "Entre un mot : ")
    mon_texte.grid(row = 1 , column = 0)

    champ_saisie = Text(fenetre , height = 1 , width = 14)
    champ_saisie.grid(row = 1 , column = 1)

    return zone_graphique, mon_texte, champ_saisie
```

```
fenetre = creer_fenetre()
zone_graphique, mon_texte, champ_saisie = creer_widgets()
```

⇒ Exécuter ce fichier

⇒ Compléter une dernière fois le script de la fonction `creer_widgets()`, en insérant sur la seconde ligne (`row = 1`) et la troisième colonne (`column = 2`) un bouton qui permettra à l'utilisateur de créer un **évènement**. Ce champ de saisi est un objet de la classe `Button`, qui sera stocké dans la variable `bouton_valider`.

Créer aussi la fonction nommée `debut()` qui est appelé à chaque fois que l'on clique sur le bouton. Cette fonction est une « callback » en anglais. Elle est associée à un évènement.

```
def creer_widgets() :
    zone_graphique = Canvas(fenetre, width=1000, height=600 , bg = 'black')
    zone_graphique.grid(row = 0 , column = 0 , columnspan = 3 )

    mon_texte = Label(fenetre, text = "Entre un mot : ")
    mon_texte.grid(row = 1 , column = 0)

    champ_saisie = Text(fenetre , height = 1 , width = 14)
    champ_saisie.grid(row = 1 , column = 1)

    bouton_valider = Button(fenetre, text = "Valider" , width = 12 , command = debut)
    bouton_valider.grid(row = 1, column = 2)

    return zone_graphique, mon_texte, champ_saisie, bouton_valider
```

```
def debut():
    print('tu as cliqué sur le bouton')
    mot = champ_saisie.get("1.0", "end-1c")
    print('le texte entré est : ', mot)
```

On applique la méthode `get()` à l'objet `champ_saisie` qui retourne le texte saisi par l'utilisateur. Ce texte est stocké ici dans la variable `mot`.

Main -----

```
fenetre = creer_fenetre()
zone_graphique, mon_texte, champ_saisie, bouton_valider = creer_widgets()
```

⇒ Exécuter ce fichier et tester le bon fonctionnement du champs de saisie et du bouton.

⇒ Si on fait un bilan des variables utilisées pour l'instant, dans le programme principal, on a :

- la variable `fenetre` qui contient un objet de la classe `Tk` : `>>> type(fenetre)`
`<class 'tkinter.Tk'>`

- la variable `zone_graphique` qui contient un objet de la classe `Canvas` : `>>> type(zone_graphique)`
`<class 'tkinter.Canvas'>`

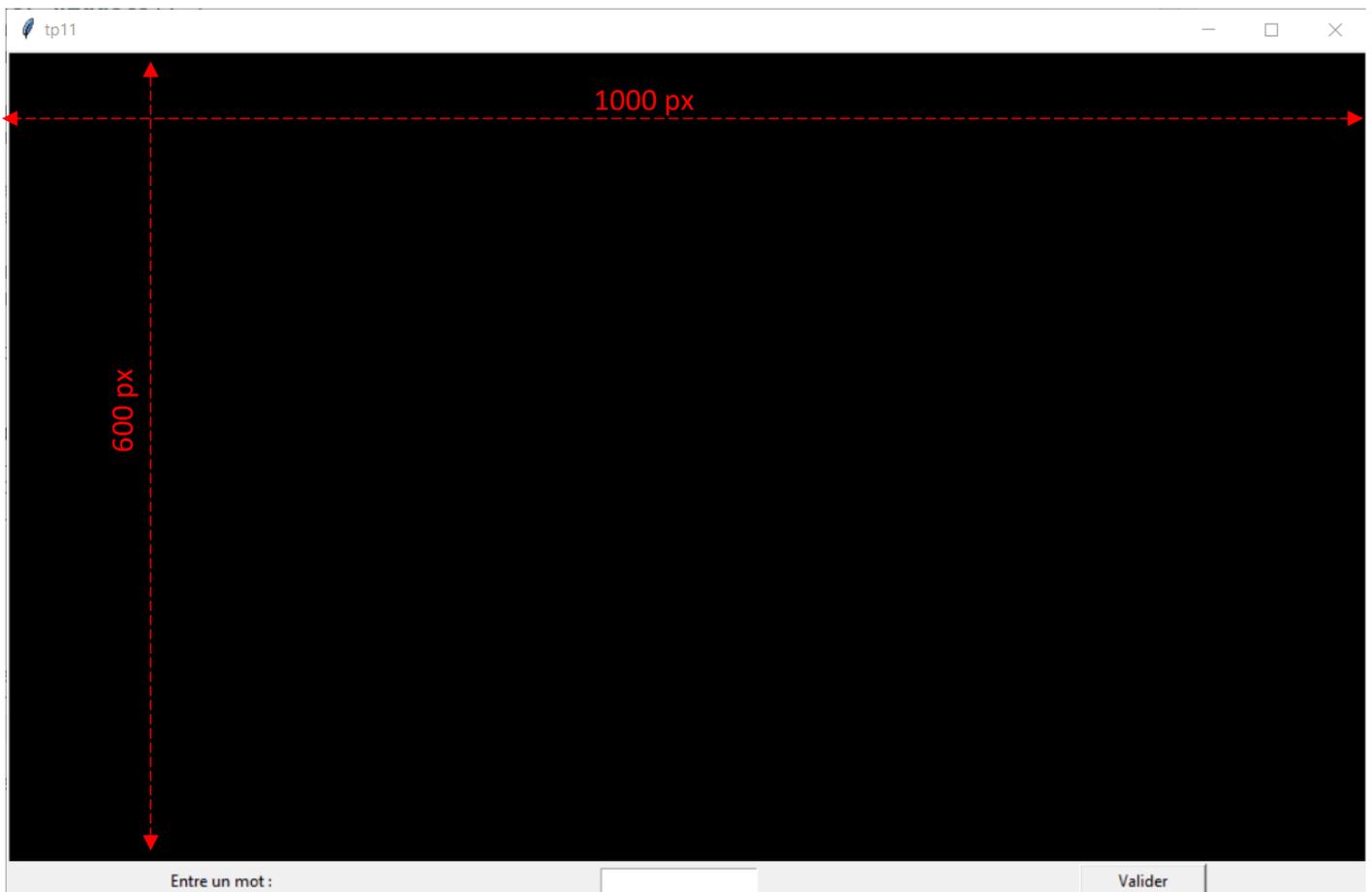
- la variable `mon_texte` qui contient un objet de la classe `Label` : `>>> type(mon_texte)`
`<class 'tkinter.Label'>`

- la variable `champ_saisie` qui contient un objet de la classe `Text` : `>>> type(champ_saisie)`
`<class 'tkinter.Text'>`

- la variable `bouton_valider` qui contient un objet de la classe `Button` :

```
>>> type(bouton_valider)
<class 'tkinter.Button'>
```

La fenêtre Tkinter contient ainsi 4 widgets. Le widget principal est le canvas qui permettra dans le tp suivant, d'accueillir des éléments graphiques. Ce canvas a une largeur de 1000 px et une hauteur de 600 px :



INFOS importantes :

- Les objets **définis ou retournés dans le programme principal** ont le statut de **variables GLOBALES**. Ils peuvent ainsi être lus et modifiés à l'intérieur des fonctions, sans que l'on ait à les passer en argument.
- Pour **créer** un objet widget, on utilise la syntaxe : `nom_objet = nom_classe(attributs)`
- Pour **positionner** cet objet widget dans le fenêtre Tk, on utilise la méthode `grid()` :
`nom_objet.grid(arguments)`
- Pour **modifier** un objet widget déjà créé, on utilise la méthode `configure()` :
`nom_objet.configure(attributs à modifier)`
- Pour **supprimer** un objet widget, on utilise la méthode `destroy()` : `nom_objet.destroy()`

----- FIN de cette 1^{ère} partie -----

DOCUMENT A RENDRE :

Cette 1^{ère} partie vous a permis de découvrir la bibliothèque Tkinter et de créer une fenêtre comprenant plusieurs widgets :

- 1 canvas (objet de la classe Canvas) qui pourra ultérieurement accueillir des éléments graphiques
- 1 widget de type texte (objet de la classe Label)
- 1 widget champ de saisie (objet de la classe Text)
- 1 widget bouton (objet de la classe Button) dont l'action est liée à un évènement.

Transférer le fichier tp11.py **par l'intermédiaire de l'onglet transfert** du site <https://nsibranly.fr> en utilisant le code : **tp11** .

Le tp11_B en est la suite. Il vous permettra d'apprendre à insérer des images dans le canvas et à les supprimer. Il vous donnera aussi l'occasion de créer des évènements claviers.