Interface graphique avec le module tkinter (Python)

Table des matières

Prérequis :
- le langage Python a déjà été découvert.

Objectif :
- utiliser le module **tkinter** pour gérer une interface graphique.

Outils nécessaires :
- une plateforme de développement du langage Python 3.5 (Idle ou Edupython) ;
- le module **tkinter** ;
- les fichiers suivants :
  - ⋆ 1-boum-Prof.py ;
  - ⋆ 2-Frames-Prof.py ;
  - ⋆ 3-Canvas-LignesObliques-Prof.py.

I) Introduction

Le module **tkinter** permet de créer des interfaces graphiques (GUI : **G**raphical **U**ser **I**nterface) pour une intéraction conviviale entre l'ordinateur et l'utilisateur, par exemple pour obtenir des fenêtres qui s'ouvrent et qui indiquent un message d'erreur, une demande d'autorisation, des jeux, des fenêtres du système d'exploitation pour rechercher des fichiers...
Pour pouvoir utiliser le module **tkinter**, il faut tout d'abord importer la librairie **tkinter** :

```
from tkinter import *
```

II) Création de widgets « esclaves » dans une fenêtre « maitre ».

☞ **Exercice 1 (à faire valider par le professeur)**

① Ouvrir le programme « 1-boum-Prof.py », l'enregistrer sous le nom « 1-boum-votrenom.py » puis l'exécuter plusieurs fois pour le comprendre.

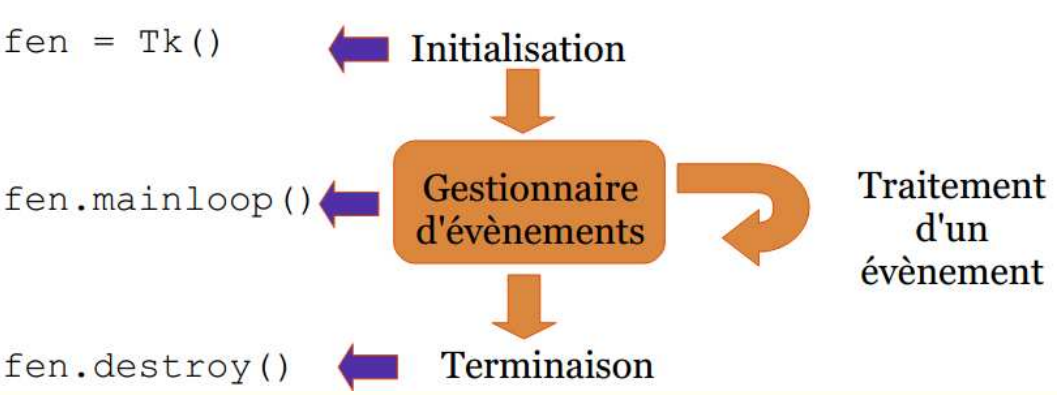
Bien observer la structure du programme :
- noter que la fonction est définie au début puis utilisée par la suite, dans le programme principal ;
- observer comment des lignes de commentaires bien organisées (lignes 4 et 1) permettent de séparer la partie définition de fonctions du corps principal du programme ;
- le programme a été « toiletté » après sa mise au point.

Cette activité est fortement inspirée d'un document élaboré par Christine Agbanrin et Matthieu Gaudré du lycée Alain, Le Vésinet.

② Lire les informations encadrées ci-dessous puis commenter le programme (lignes : 2, 7, 14, 18, 22, 27).

À retenir : le gestionnaire d'évènements

fen=Tk() crée la fenêtre qui s'appellera **fen**.
fen.mainloop() : provoque le démarrage du **gestionnaire d'évènements** associé à la fenêtre **fen**. Cette instruction est nécessaire pour que l'application soit « à l'affût » des clics de souris, des pressions exercées sur les touches du clavier etc. C'est donc cette instruction qui la met en marche.
fen.destroy : provoque la fermeture de la fenêtre.



À retenir (voir aussi le mini-lexique en annexe)

① Widgets avec paramètres (widget : window + gadget)

Label : permet un affichage simple de texte.

Button : définit un bouton sur lequel on peut cliquer et qui exécute une commande quelconque (une fonction existante dans Python ou une fonction que vous avez codée).

Paramètres de ces widgets :

- **text** = texte à afficher ;
- **fg** = couleur du texte ;
- **bg** = couleur de fond.

Paramètres spécifiques au widget **Button** :

- **command** permet de préciser la fonction à lancer lors d'un clic sur le bouton ;
- **destroy** permet de fermer la fenêtre lorsque le bouton est cliqué ;
- attention : le paramètre **command** attend un nom de fonction sans argument (donc sans parenthèse).

② Gestion de l'espace dans la fenêtre

La méthode **pack()** (**tex1.pack()** ou **bou1.pack()**) : la fenêtre s'ajuste automatiquement au contenu si on ne précise pas ses dimensions.

Options de la méthode **pack** :

- l'option **side** peut accepter les valeurs **TOP**, BOTTOM, **LEFT** ou **RIGHT**, pour « pousser » le widget du côté correspondant dans la fenêtre ;
- les options **padx** et **pady** permettent de réserver un petit espace autour du widget (exprimé en nombre de pixels),

  padx : espace à gauche et à droite du widget,

  pady : espace au-dessus et au-dessous du widget.

Autres méthodes (voir le mini-lexique en annexe) :

- méthode **grid()** : la fenêtre est découpée en un quadrillage virtuel ;
- méthode **place()** : on place un repère sur la fenêtre (origine en haut à gauche).

III) Utilisation de Frame

☞ **Exercice 2 (à faire valider par le professeur)**

① Ouvrir le programme « 2-Frames-Prof.py », l'enregistrer sous le nom « 2-Frames-votrenom.py puis l'exécuter plusieurs fois pour le comprendre.

② Le modifier en ajoutant une quatrième widget Frame, dont le parent est le widget frame2 et contenant un texte « Envoyer mail au prof » et un bouton effacer.

IV) Tracés dans un Canvas

Le canevas représente une surface pour dessiner. On ne dessine pas directement dans la fenêtre, il nous faut un « container » intermédiaire. Avec **tkinter**, on ne peut pas dessiner ni insérer d'images en dehors d'un canevas (ces dessins pourront être animés dans le prochain TP).

☞ **Exercice 3 (à faire valider par le professeur)**

① Ouvrir le programme « 3-Canvas-LignesObliques-Prof.py », l'exécuter.

Bien comprendre les deux fonctions.

Sur un support papier à conserver, expliquer la différence entre une variable locale à une fonction et une variable globale.

Puis ne pas hésiter à prendre des notes sur les observations suivantes :

Observer comment les deux fonctions sont commentées :

- lignes 9, 10, 13 puis 17, 18, 19, une explication générale sur la fonction ;
- ligne 13, un commentaire pour la ligne suivante ;
- lignes 22, 24, 25, des commentaires en fin de ligne pour plus de précision.

Observer comment dans la ligne 14, les variables **y1** et **y2** sont modifiées en une seule instruction.

Avez-vous bien compris les lignes 21, 22, 24 ?

Ne pas hésiter à insérer une instruction : (**print(c)** afin de comprendre les valeurs prises par la variable **c**).

Le programme a été « toiletté » et bien commenté .

② Modifier ce programme (3-Canvas-LignesParalleles-votrenom.py) afin que toutes les lignes tracées soient horizontales et parallèles.

③ Modifier ce programme (3-Canvas-LignesViseur-votrenom.py) afin,

- d'agrandir le canevas de manière à lui donner une largeur de 500 unités et une hauteur de 650.
- de modifier la taille des lignes, afin que leurs extrémités se confondent avec les bords du canevas.
- d'ajouter une fonction **drawcross** qui tracera deux lignes rouges en croix au centre du canevas : l'une horizontale et l'autre verticale.
- d'ajouter un bouton portant l'indication « viseur ». Un clic sur ce bouton devra provoquer l'affichage de la croix.

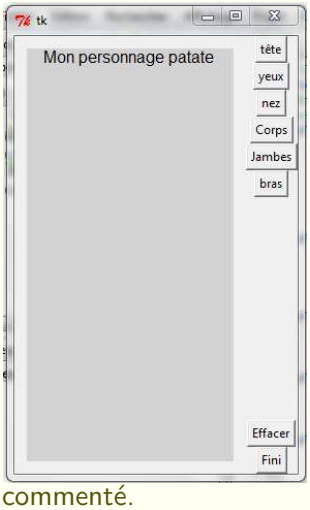
Pour plus d'information sur les canevas, voir le mini-lexique en annexe.

V) Monsieur patate

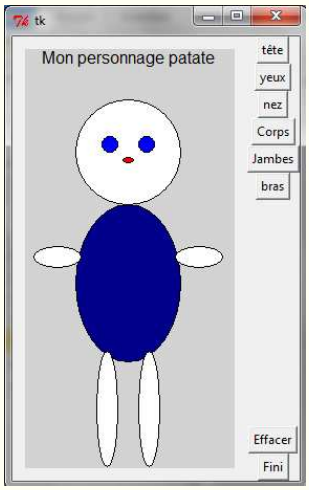
☞ **Exercice 4 (à faire valider par le professeur)**

Mini projet en binôme (4-Patate-nom1-nom2.py)

Ecrire un programme qui fait apparaitre la fenêtre ci-contre.



Lorsqu'on clique sur les boutons, les différentes parties du corps apparaissent.
Le bouton « Effacer » efface le dessin et le bouton « Fini » détruit la fenêtre.



Le programme sera toiletté et commenté.

VI) Insertion d'images

Le module **tkinter** permet d'insérer une image au format .gif (le programme python et l'image doivent se trouver dans le même dossier).

☞ **Exercice 5 (à faire valider par le professeur)**

① Une image peut être insérée dans un widget **Label**, tester le script ci-dessous (5-ImageLabel-votrenom.py) :

```
from tkinter import *
fen1 = Tk()
dessin=PhotoImage(file='python.gif') #charger l'image depuis un fichier
boite=Label(fen1,image=dessin) #copie l'image dans le label boite
boite.pack()
fen1.mainloop()
```

② Une image peut aussi être insérée dans un widget canvas, tester le script ci-dessous (5-ImageCanvas-votrenom.py) :

```
from tkinter import *
#=========Programme principal===============
fen1 = Tk() #création de la fenêtre maître
dessin=PhotoImage(file='python.gif') #charge l'image depuis un fichier
#.....commenter les 4 lignes ci-dessous.geometry : voir le mini-lexique en annexe.....
largeur=dessin.width()
hauteur=dessin.height()
dimension=str(largeur+100)+"x"+str(hauteur+100)
fen1.geometry(dimension)
#.....commenter les 2 lignes ci-dessous.....
can1 = Canvas(fen1,bg='blue',width=largeur+10, height=hauteur+10)
can1.pack()
#.....commenter la ligne ci-dessous (notamment anchor).....
img=can1.create_image(6,6,anchor=NW,image=dessin) #copie l'image sur le canevas
fen1.mainloop()
```

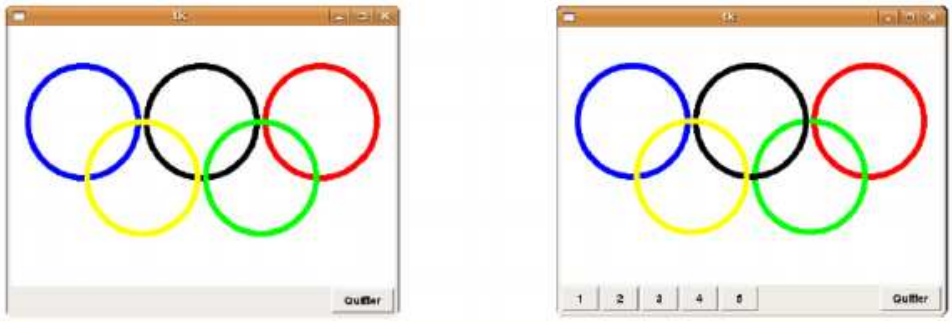
VII) Tests des autres widgets

☞ **Exercices**

① Tester les boutons **Checkbutton**, **Entry**, **Radiobutton** commentés en annexe.

② a) Créer un court programme qui dessinera les 5 anneaux olympiques dans un rectangle de fond blanc (**white**). Un bouton « Quitter » doit permettre de fermer la fenêtre.

b) Modifier ce programme en y ajoutant 5 boutons. Chacun de ces boutons provoquera le tracé d'un des 5 anneaux.



VIII) Sitographie :

- Divers widgets prêts à l'emploi :
  http://www.fil.univ-lille1.fr/˜marvie/python/chapitre6.html
- Objets graphiques **Canvas** :
  http://physique.umontreal.ca/˜mousseau/phy1234/notes/notes_52.html
- Sources d'inspiration de ce TP :
  http://python.developpez.com/cours/TutoSwinnen/?page=Chapitre8
  http://fsincere.free.fr/isn/python/cours_python_tkinter.php
- Pour s'exercer sur d'autres exemples :
  http://softdmi.free.fr/tuto_python/tuto-python.php?chapitre=2&page=1
- Une documentation **tkinter** :
  http://tkinter.fdex.eu/index.html

Pour un mini-lexique **tkinter** en annexe, cliquer ici.