TD3 : INTRODUCTION A TKINTER POUR CREER DES INTERFACES GRAPHIQUES EN PYTHON

A. BUT DU TD

L'objectif de ce TD est d'apprendre à manipuler quelques composants du module Python "Tkinter" permettant de créer des interfaces graphiques.

B. Quelques éléments de Tkinter

Le module Tkinter ("Tk interface") de Python permet de créer des interfaces graphiques. Il est composé de nombreux composants graphiques (ou widgets) dont des fenêtres (classe Tk), des boutons (classe Button), des cases à cocher (classe Checkbutton), des étiquettes (classe Label), des zones de texte simple (classe Entry), des menus (classe Menu), des zone graphiques (classe Canvas), des cadres (classe Frame), etc...

Comme pour toute interface graphique, il est possible gérer des événements : clic avec la souris, déplacement de la souris, appui sur une touche du clavier... Cela répond aux principes de la programmation événementielle : tous les programmes que vous avez écrits jusqu'à présent sont typiquement organisés sous la forme suivante : une phase d'initialisation, suivie d'une boucle de traitement jusqu'à ce que l'utilisateur souhaite terminer, une phase de terminaison éventuelle.

Les programmes dans les systèmes de fenêtrage (Windows, Mac OS X, ...) suivent une autre logique : après une phase d'initialisation (qui affiche des objets d'interfaces sur l'écran), le programme se met en attente d'événements (ou de messages) et doit réagir à ces événements envoyés par le système et par l'utilisateur. De tels événements sont par exemple « réafficher le contenu de la fenêtre », « appui sur le bouton gauche de la souris à la position x, y », « souris déplacée », « touche clavier appuyée », « fenêtre déplacée », « fenêtre iconisée », etc… Un programme d'interface graphique doit donc répondre à tous ces événements de manière ad hoc pour être complet. Dans ce TD, nous verrons et traiterons que quelques événements parmi les plus importants.

La documentation en ligne du module "Tkinter" peut être trouvée à l'adresse suivante :

https://docs.python.org/3.4/library/tkinter.html

Afin de se familiariser avec quelques uns des composants proposés par le module Tkinter, il vous est demandé d'étudier les exemples suivants.

Exemple 1 : Application tkinter minimaliste

```
from tkinter import *

# Création de la fenêtre principale (main window)
mafenetre = Tk()

# Création d'un widget Label (texte 'Bonjour ECL !')
label = Label(mafenetre, text = 'Bonjour ECL !', fg = 'blue')
```

```
# Positionnement du widget avec la méthode pack()
label.pack()

# Création d'un widget Button (bouton Quitter)
bouton = Button(mafenetre, text = 'Quitter', command = mafenetre.destroy)
bouton.pack()

# Lancement du gestionnaire d'événements
mafenetre.mainloop()
```

Exemple2 : Mise en oeuvre des widgets Button et Label

```
from tkinter import *
import random

def tirage():
    nb = random.randint(1,100)
    texte.set('Nombre : ' + str(nb))

# Création de la fenêtre principale (main window)
mafenetre = Tk()
mafenetre.title('Tirage aléatoire')
mafenetre.geometry('300x100+400+400')

# Création d'un widget Button (bouton Lancer)
boutonLancer = Button(mafenetre, text ='Tirage', command = tirage)

# Positionnement du widget avec la méthode pack()
boutonLancer.pack(side = LEFT, padx = 5, pady = 5)

texte = StringVar()
tirage()

# Création d'un widget Label (texte 'Résultat -> x')
labelResultat = Label(mafenetre, textvariable = texte, fg ='red', bg ='white')
labelResultat.pack(side = LEFT, padx = 5, pady = 5)

# Création d'un widget Button (bouton Quitter)
boutonQuitter = Button(mafenetre, text ='Quitter', command = mafenetre.destroy)
boutonQuitter.pack(side = LEFT, padx = 5, pady = 5)

mafenetre.mainloop()
```

Exemple 3 : Mise en oeuvre des widgets Frame, Button et Label

```
from tkinter import *

# Création de la fenêtre principale
mafenetre = Tk()
mafenetre.title('Pense bête')
mafenetre['bg']='blue' # couleur de fond

# création d'un widget Frame dans la fenêtre principale
frame1 = Frame(mafenetre,borderwidth=2,relief=GROOVE)
frame1.pack(side=LEFT,padx=10,pady=10)

# création d'un second widget Frame dans la fenêtre principale
frame2 = Frame(mafenetre,borderwidth=2,relief=GROOVE)
frame2.pack(side=LEFT,padx=10,pady=10)

# création d'un troisieme widget Frame dans la fenêtre principale
frame3 = Frame(mafenetre,bg="white",borderwidth=2,relief=GROOVE)
frame3.pack(side=LEFT,padx=10,pady=10)
```

```
# création d'un widget Label et d'un widget Button dans un widget Frame
Label(frame1,text="Acheter de l'eau").pack(padx=10,pady=10)
Button(frame1,text="Effacer",fg='navy',command=frame1.destroy).pack(padx=10,pady=10)

Label(frame2,text="Appeler le médecin").pack(padx=10,pady=10)
Button(frame2,text="Effacer",fg='navy',command=frame2.destroy).pack(padx=10,pady=10)

Label(frame3,text="Entrainement de tennis à 18h",bg="white").pack(padx=10,pady=10)
Button(frame3,text="Effacer",fg='navy',command=frame3.destroy).pack(padx=10,pady=10)

mafenetre.mainloop()
```

Exemple 4 : Mise en oeuvre des widgets Entry, Label, Button et boîte de dialogue MessageBox

```
from tkinter import *
from tkinter.messagebox import * # boîte de dialogue
from random import randint

def verification():
    if nombre.get() == str(nb):
        # le nombre a été deviné : on affiche une boîte de dialogue puis on ferme la
fenêtre
        showinfo('Résultat','Bravo, c\'est gagné.\nAu revoir !')
        mafenetre.destroy()
    else:
        # le nombre n'a pas été deviné : on affiche une boîte de dialogue
        showwarning('Résultat','Perdu...\nVeuillez recommencer !')
        nombre.set('')

# Création de la fenêtre principale (main window)
mafenetre = Tk()
mafenetre.title('Deviner un nombre entre 1 et 10')

# Création d'un widget Label (texte 'Mot de passe')
label = Label(mafenetre, text = 'Nombre : ')
label.pack(side = LEFT, padx = 5, pady = 5)

# Création d'un widget Entry (champ de saisie)
nombre= StringVar()
texte = Entry(mafenetre, textvariable= nombre, bg ='bisque', fg='maroon')
texte.focus_set()
texte.pack(side = LEFT, padx = 5, pady = 5)

# Génération d'un nombre aléatoire entre 1 et 10
nb = randint(1,10)
print('Valeur :',nb)

# Création d'un widget Button (bouton Valider)
bouton = Button(mafenetre, text ='Tester', command = verification)
bouton.pack(side = LEFT, padx = 5, pady = 5)

mafenetre.mainloop()
```

Exemple 5 : Mise en oeuvre des widgets Canvas et Button

```
from tkinter import *
import random

def dessinerCarre():
    """ Dessine un carre de centre (x,y) et de cote 2*c """
    x = random.randint(0,largeur)
    y = random.randint(0,hauteur)
    c = 20
```

```
    canevas.create_rectangle(x-c, y-c, x+c, y+c, outline='blue', fill='red')

def effacer():
    """ Efface le contenu de la zone d'affichage """
    canevas.delete(ALL)

# Création de la fenêtre principale (main window)
mafenetre = Tk()
mafenetre.title('Affichage des carrés')

# Création d'un widget Canvas (zone graphique)
largeur = 480
hauteur = 320
canevas = Canvas(mafenetre, width = largeur, height = hauteur, bg ='white')
canevas.pack(padx =5, pady =5)

# Création d'un widget Button (bouton Afficher)
boutonAfficher = Button(mafenetre, text ='Dessiner', command = dessinerCarre)
boutonAfficher.pack(side = LEFT, padx = 10, pady = 10)

# Création d'un widget Button (bouton Effacer)
boutonEffacer = Button(mafenetre, text ='Effacer', command = effacer)
boutonEffacer.pack(side = LEFT, padx = 5, pady = 5)

# Création d'un widget Button (bouton Quitter)
boutonQuitter = Button(mafenetre, text ='Quitter', command = mafenetre.destroy)
boutonQuitter.pack(side = LEFT, padx = 5, pady = 5)

mafenetre.mainloop()
```

Exemple 6 : Mise en oeuvre des widgets Canvas et Button ainsi que de la gestion de la souris

```
from tkinter import *

def cliquer(event):
    """ Gestion de l'événement Clic gauche sur la zone graphique """
    # position du pointeur de la souris
    x = event.x
    y = event.y
    # on dessine un cercle
    r = 20
    canevas.create_oval(x-r, y-r, x+r, y+r, outline='blue',fill='black')

def effacer():
    """ Efface la zone graphique """
    canevas.delete(ALL)

# Création de la fenêtre principale
mafenetre = Tk()
mafenetre.title('Affichage des cercles')

# Création d'un widget Canvas
largeur = 480
hauteur = 320
canevas = Canvas(mafenetre, width = largeur, height = hauteur, bg ='white')

# La méthode bind() permet de lier un événement avec une fonction :
# un clic gauche sur la zone graphique provoquera l'appel de la fonction utilisateur
# cliquer()
canevas.bind('<Button-1>', cliquer)
canevas.pack(padx =5, pady =5)

# Création d'un widget Button (bouton Effacer)
Button(mafenetre, text ='Effacer', command = effacer).pack(side=LEFT,padx= 5,pady =5)
```

```
# Création d'un widget Button (bouton Quitter)
Button(mafenetre,text='Quitter',command=mafenetre.destroy).pack(side=LEFT,padx=5,pady
=5)
mafenetre.mainloop()
```

Exemple 7 : Reprise de l'exemple précédent en utilisant des classes

```
from tkinter import *

class ZoneAffichage(Canvas):
    def __init__(self,parent, w, h, c):
        Canvas.__init__(self, width = w, height = h, bg = c)

class FenPrincipale(Tk):
    def __init__(self):
        Tk.__init__(self)
        self.title('Affichage des cercles')
        self.__zoneAffichage = ZoneAffichage(self,480,320,'white')

        # La méthode bind() permet de lier un événement avec une fonction :
        # un clic gauche sur la zone graphique provoquera l'appel de la fonction
utilisateur cliquer()
        self.__zoneAffichage.bind('<Button-1>',self.cliquer)
        self.__zoneAffichage.pack(padx=5, pady=5)

        # Création d'un widget Button (bouton Effacer)
        self.__boutonEffacer    =    Button(self,    text    ='Effacer',    command    =
self.effacer).pack(side=LEFT,padx = 5,pady = 5)

        # Création d'un widget Button (bouton Quitter)
        self.__boutonQuitter    =    Button(self,    text    ='Quitter',    command    =
self.destroy).pack(side=LEFT,padx=5,pady=5)

    def cliquer(self,event):
        """ Gestion de l'événement Clic gauche sur la zone graphique """
        # position du pointeur de la souris
        x = event.x
        y = event.y
        # on dessine un cercle
        r = 20
        self.__zoneAffichage.create_oval(x-r,y-r,x+r,y+r,outline='blue',fill='black')

    def effacer(self):
        """ Efface la zone graphique """
        self.__zoneAffichage.delete(ALL)

# Création de la fenêtre principale
fen = FenPrincipale()
fen.mainloop()
```

C. Programme à réaliser

Ecrire un programme permettant d'afficher un dessin constitué de formes géométriques (réutiliser les classes du TD1). En cliquant avec le bouton gauche de la souris sur une des formes, ses propriétés (coordonnées, périmètre, surface) doivent s'afficher dans une fenêtre de type messagebox (exemple 4).

Quelques indications :

- Repartir de l'exemple 7 et compléter la classe "ZoneAffichage" pour lui ajouter un attribut qui sera un objet "Dessin" et dont les formes (carrés, rectangles, cercles) devront lui être ajoutées dans le constructeur de "ZoneAffichage".

- Deux méthodes seront ajoutées à la classe “ZoneAffichage” : une méthode “afficher” pour afficher les formes du dessin dans ce canvas, et une méthode “selection_formes” pour gérer le clic de la souris sur les formes.
- La méthode “cliquer” de la classe “FenPrincipale” sera modifiée pour gérer le clic de la souris sur les formes.
- Une méthode “afficher” sera ajoutée à la classe “FenPrincipale” pour permettre l’affichage du contenu de son objet “ZoneAffichage,” et sera associée au clic sur un bouton “Afficher” à ajouter également à la fenêtre “FenPrincipale”.
- Dans chacune des classes des formes géométriques, une méthode “affiche” sera ajoutée permettant l’affichage de cette forme dans le canvas “ZoneAffichage” (polymorphisme).
- Une méthode “selection” sera également ajoutée à toutes les formes pour tester si le clic de la souris avec le bouton gauche a été réalisé sur la forme. Cette méthode renverra “True” si c’est le cas, et “False” sinon.
- Une méthode “affiche_formes” sera ajoutée à la classe “Dessin” permettant l’affichage dans le canvas “ZoneAffichage” de toutes les formes qui la composent.
- Une méthode “selection_formes” sera ajoutée à la classe “Dessin” permettant de tester si le clic de la souris avec le bouton gauche a été réalisé sur une des formes qui la compose. Si c’est le cas, un message sera affiché dans une fenêtre de type “messagebox” indiquant toutes les propriétés de la forme sélectionnée.

Une capture d’écran d’un exemple d’application attendue est donné ci-dessous.