Chapitre 3 : IHM et objet

Audras

Lycée Saint-Just

année 2018-2019 ISN

1) Les Objets

Définition

En Python, tout est objet, il est donc util de comprendre comment ils sont fait. Un objet possède des propriétés, c'est à dire des variables qui lui sont propres, et des méthodes, c'est à dire des fonctions qui lui sont propres. En général, lorsqu'on peut modifier une propriété d'objet, cela se fait en passant par une méthode et en utilisant la syntaxe objet.méthode() ou la syntaxe méthode(objet) ou d'autres syntaxes pour les méthodes spéciales __methodeSpeciale__()
Par exemple liste.append(3) appelle la méthode append() de l'objet liste qui va modifier sa propriété d'élément en ajoutant 3 en fin de liste.
liste+[3] est définie par la méthode spéciale liste.__add__(3) qui renvoie une nouvelle liste où 3 est ajouté en fin de liste.

Propriété

Si par exemple liste=[1,2], on peut connaître toutes les méthodes d'un objet avec help(liste) et en lisant la documentation officiel à l'adresse `https://docs.python.org/fr/3/index.html`

```
>>> help(liste)
Help on list object:

class list(object)
 |  list() -> new empty list
 |  list(iterable) -> new list initialized from iterable's items
 |
 |  Methods defined here:
 |
 |  __sizeof__(...)
 |      L.__sizeof__() -- size of L in memory, in bytes
 |
 |  append(...)
 |      L.append(object) -> None -- append object to end
 |
 |  clear(...)
 |      L.clear() -> None -- remove all items from L
```

Exemple

La méthode format() du type str permet de mettre en forme texte et variable.

```
>>> x=5
>>> print("le carré de {0} est {1}".format(x,x*x))
le carré de 5 est 25
>>> print("{1} est le carré de {0}".format(x,x*x))
25 est le carré de 5
>>> print("le carré de {} est {}".format(x,x*x))
le carré de 5 est 25
```

2) Interface Homme Machine

Définition

En générale l'homme interagit avec la machine via le couple clavier/souris, il faut programmer une interface textuelle en console ou graphique. Il existe aussi d'autre moyen d'interaction comme la caméra, le micro, une kinect, un casque de réalité virtuel, ...
La méthode format() et des motifs d'entré du texte permettent de façonner une interface textuelle. Pour une interface graphique, on va utiliser la bibliothèque tkinter par from tkinter import *.
Le principe d'une interface graphique ou GUI est de générer un gestionnaire d'évènement qui fonctionne en parallèle du programme, dans un thread, et qui sera attentif au clavier et à la souris.
`http://tkinter.fdex.eu/`
`http://infohost.nmt.edu/tcc/help/pubs/tkinter/tkinter.pdf`
`http://ateliers.mse.free.fr/tkinter/page_tkinter.html`

Propriété

ihm est le process parent dans lequel sont placés tous les éléments. myFrame est le cadre de gauche, il est placé dans la case (0,0) de ihm. menu est celui de droite, il est en (0,1) et poussé en haut (Nord). On peut placer du texte ou une image dans un Label. Le Canvas drawing permet de dessiner des figures. La variable valEntry est modifiée dès que l'utilisateur change la valeur de l'entrée. Pour tracer le cercle on donne les coordonnées des sommets en haut à gauche et en bas à droite du rectangle dans lequel est circonscrit le cercle, l'origine du repère est le coin en haut à gauche et l'axe des ordonnées est dirigé vers le bas. La fonction changeRadius() permet de modifier le rayon du cercle tracé. La fonction changeRadiusEvent(e) prends en argument l'évènement <KeyPress-Return>. myEntry.bind(...) permet le liéer l'évènement "touche entrée" avec le widget myEntry et la fonction changeRadiusEvent.

Algorithme

Lorsque l'évènement est le temps on crée une animation. Il faut créer une boucle sans fin à l'aide d'une fonction qui s'appelle avec un délai. Ainsi le programme s'exécutera normalement mais il appellera la fonction régulièrement.
L'instruction est ihm.after(temps,fonction) et l'unité de temps du délai est la milliseconde.
Ajouter ces lignes au programme précédent, avant ihm.mainloop(), et en ajoutant la ligne :
"from random import randint" en début du fichier.

```
24 def changeTime():
25     valEntry.set(randint(0,10)*10)
26     changeRadius()
27     ihm.after(1000*2,changeTime)
28 ihm.after(1000*5,changeTime)
```

Ici le délai est de 5s au départ puis le changement de rayon s'effectue toutes les 2s.

```
1 from tkinter import *
2 ihm=Tk()
3 myFrame=Frame(ihm,bd=2,relief=RIDGE)
4 myFrame.grid(row=0,column=0)
5 menu=Frame(ihm,bd=2,relief=RIDGE)
6 menu.grid(row=0,column=1,sticky=N)
7 Label(myFrame,text="mon dessin").grid(row=0,column=0)
8 Label(menu,text="ma fenêtre 2").grid(row=0,column=0)
9 drawing=Canvas(myFrame,width=600,height=500)
10 drawing.grid(row=1,column=0)
11 valEntry=IntVar(menu, 50)
12 circle = drawing.create_oval(50,50,50+2*valEntry.get()
13                                   ,50+2*valEntry.get())
14 def changeRadius():
15     r=valEntry.get()
16     drawing.coords(circle,50,50,50+2*r,50+2*r)
17 def changeRadiusEvent(e): changeRadius()
18 myEntry = Entry(menu,textvariable=valEntry)
19 myEntry.grid(row=1,column=0)
20 myEntry.bind('<KeyPress-Return>',changeRadiusEvent)
21 Button(menu,width=10,command=changeRadius,text='valide rayon'
22                                   ).grid(row=2,column=0)
23 ihm.mainloop()
```

3) La conduite de projet

Définition

Il faut un groupe de trois, ni trop homogène pour une meilleurs créativité, ni trop hétérogène pour que la communication se fasse bien entre les membres. Il faut une idée de départ sur un thème relatif aux enseignements présents au lycée ou à l'organisation du travail d'un lycéen.
Exemple : gestion d'un planning, d'une base de fiches de révision, calculs de math ou de physique. Autour de la gravité, de la propagation d'une épidémie, du codage d'un message. Créer des notes de musique à partir de courbes ou d'échantillons. Modifier une image ou un son. Vie artificiel : automate cellulaire, fractale, arbres, simulations de trafic,...
Méthode de travail : Il faut faire une analyse descendante du projet, c'est à dire le découper en sous-tâches puis recommencer avec chaque sous-tâche et ainsi de suite jusqu'à n'avoir plus que des tâches élémentaires. Sélectionner une base qui constitura un mini-projet.

Propriété

Chaque fonction devra être documentée : il faut indiquer les spécifications pour pouvoir l'utiliser et dire quelle sera son effet. Il est possible d'écrire cette documentation directement entre des guillements sous le def f() :

Respecter deux des principes SOLIDE de la programmation orientée objet. Single Responsibility Principle et Open Close Principle. Chaque fonction ne doit avoir la responsabilitée que d'une seule tâche. Le projet doit être ouvert à de nouvelles fonctionnalités mais les fonctions doivent être fermée aux mofifications. Lorsque le mini-projet fonctionne, on doit pouvoir l'améliorer, le compléter en un projet sans modifier le code des fonctions qui font leur travail correctement, de façon à ne pas risquer de tous perdre par manque de temps ou à cause d'une difficulté mal évaluée.