Création de Threads et Interface Graphique

Joan RAVALOMANDA, L3 informatique

10 octobre 2017

Résumé

Dans ce rapport , je vais vous montrer comment créer un thread , son fonctionnement , la création d'une interface graphique (tKinter ou Swing), et les problèmes rencontrés lors de la réalisation des exercices en python et Java .

1 Introduction

Un processus est un programme (fichier exécutable) en cours d'exécution sur un processeur. Il est caractérisé par : son code , son espace d'adressage , son ID, sa priorité. Il est crée lorsqu'un autre processus lance son exécution. Les états successifs d'un processus sont généralement représentées par un diagramme d'état. Tout processus dispose d'un thread principal depuis lequel d'autres threads peuvent être lancés.

Un thread est donc une portion de code capable de s'exécuter en parallèle à d'autres traitements. Ils sont utiles dans bien des cas et parfois même nécessaires comme nous le verrons plus loin dans la section à propos de Swing (Interface Graphique)

Dans un premier temps, nous allons voir la notion de Thread. Ensuite , nous parlerons d'interface graphique et puis nous finirons par voir ces notions dans 2 exercices en Java et en Python.

2 Thread

2.1 Qu'est-ce que c'est ?

Un Thread est un fil d'exécution de code à l'intérieur d'un processus et qui a la possibilité d'être ordonnancé (sous-processus). Le principal avantage des threads est de pouvoir répartir différents traitements d'un même programme en plusieurs unités distinctes pour permettre leurs exécutions "simultanées".

2.2 JAVA

Un thread en Java est un objet, qui doit étendre la classe Thread. Comme Java ne supporte pas l'heritage multiple, il n'est pas toujours possible d'étendre cette classe. Tout programme Java comporte au moins un thread principal correspondant a la methode main. Ci dessous ,on a le code courant du thread de la classe Main.

```
public class Main {
   public static void main(String[] args) {
      System.out.println("Hello World!");
   }
}
```

Ce code affiche tout simplement un message : Hello World !

2.3 Python

Pareil en Python, il comporte au moins un thread principal correspondant au thread de contrôle initial du programme.

Le module threading fournit la classe Thread pour creer et gerer des threads.

Pour créer un thread, il suffit de surcharger la méthode run de la classe Thread. Si le thread a besoin de données lors de son exécution, il faut surcharger son constructeur sans oublier d'appeler le constructeur de la classe mère. L'exécution de thread commence par la création d'une instance et l'appel à la méthode start. En résumé, il faut :

— surcharger la classe `threading.Thread`,
— surcharger le constructeur sans oublier d'appeler le constructeur `threading.Thread.__init__`
— surcharger la méthode `run`, c'est le code que devra exécuter le thread,
— créer une instance de la nouvelle classe et appeler la méthode `start` pour lancer le thread secondaire qui formera le second fil d'exécution.

```
class MonThread (threading.Thread):
    def __init__(self, jusqua):      # jusqua = donnée supplémentaire
        threading.Thread.__init__(self)  # ne pas oublier cette ligne
        # (appel au constructeur de la classe mère)
        self.jusqua = jusqua           # donnée supplémentaire ajoutée à la classe

    def run(self):
        for i in range(0, self.jusqua):
            print("thread ", i)
            time.sleep(0.08)   # attend 100 millisecondes sans rien faire
            # facilite la lecture de l'affichage


m = MonThread(10)          # crée le thread
m.start()                  # démarre le thread,
# l'instruction est exécutée en quelques millisecondes
# quelque soit la durée du thread

for i in range(0, 10):
    print("programme ", i)
    time.sleep(0.1)           # attend 100 millisecondes sans rien faire
```

Le programme affiche des lignes qui proviennent du thread principal et du thread secondaire dont les affichages diffèrent, ce qui donne :

Sortie

```
thread  0
programme  0
programme  1
thread  1
thread  2
programme  2
thread  3
```

3 Interface Graphique

Une interface graphique (GUI : Graphical User Interface) ou un environnement graphique est un dispositif de dialogue homme-machine, dans lequel les objets à manipuler sont dessinés sous forme de pictogrammes à l'écran, de sorte que l'usager peut utiliser en imitant la manipulation physique de ces objets avec un dispositif de pointage, le plus souvent une souris.

3.1 Java

Swing fait partie de la bibliothèque Java Foundation Classes (JFC). C'est une API dont le but est similaire à celui de l'API AWT mais dont les modes de fonctionnement et d'utilisation sont complètement différents. Swing a été intégré au JDK.

La bibliothèque JFC contient :

— L'API Swing : de nouvelles classes et interfaces pour construire des interfaces graphiques
— Accessibility API
— 2D API : support du graphisme en 2D
— API pour l'impression et le cliquer/glisser

Les composants Swing forment un nouvelle hiérarchie parallèle à celle de l'AWT. L'ancêtre de cette hiérarchie est le composant JComponent. La procédure à suivre pour utiliser un composant Swing est identique à celle des composants de la bibliothèque AWT : créer le composant en appelant son constructeur, appeler les méthodes du composant si nécessaire pour le personnaliser et l'ajouter dans un conteneur.

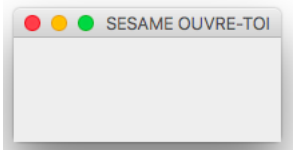
Swing utilise la même infrastructure de classes qu'AWT, ce qui permet de mélanger des composants Swing et AWT dans la même interface.

La classe de base d'une application est la classe JFrame. Son rôle est équivalent à la classe Frame de l'AWT et elle s'utilise de la même façon.

Code d'une simple fenêtre en Java :

```
import javax.swing.*;
import java.awt.event.*;

public class Swing1 extends JFrame {

   public Swing1() {
      super("SESAME OUVRE-TOI");

      WindowListener l = new WindowAdapter() {
         public void windowClosing(WindowEvent e){
            System.exit(0);
         }
      };

      addWindowListener(l);
      setSize(200,100);
      setVisible(true);
   }

   public static void main(String [] args){
      JFrame frame = new Swing1();
   }
}
```

Résultat : Une fenêtre s'ouvre...



3.2 Python

Comment créer des interfaces graphiques à l'aide d'un module présent par défaut dans Python : Tkinter ?

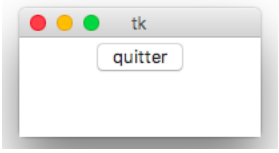
Tkinter (Tk interface) est un module intégré à la bibliothèque standard de Python, bien qu'il ne soit pas maintenu directement par les développeurs de Python. Il permet de créer des interfaces graphiques.

De nombreux composants graphiques (ou widgets) sont disponibles :

— fenêtre (classe Tk),
— bouton (classe Button),
— case à cocher (classe Checkbutton),
— étiquette (classe Label),
— zone de texte simple (classe Entry),
— menu (classe Menu),
— zone graphique (classe Canvas),
— cadre (classe Frame)...

Code d'une simple fenêtre en Python :

```
from tkinter import *

Fenetre = Tk()

bouton=Button(Fenetre, text="quitter", command=Fenetre.destroy)

bouton.pack()

Fenetre.mainloop()
```



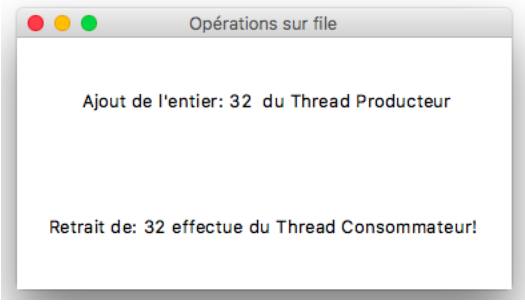
Particularité de cette fenêtre : Ce code crée une fenêtre contenant un bouton "Quitter". Lorsqu'on va cliquer dessus, la fenêtre va se fermer.

4 File entiers

Ce qui nous ai demandé dans cet exercice est de crée une file d'entiers , un thread producteur et un thread consommateur se partageant cette file et répétant indéfinement les actions :

— Producteur : Ajout d'un entier choisi aleatoirement entre 1 et 100 puis repos. Si la file est pleine, ce thread est mis en attente jusqu'a ce qu'une place soit disponible.
— Consommateur : Retrait d'un entier, affichage de cet entier puis repos. Si la file est vide, le thread est mis en attente jusqu'a ce qu'un entier soit disponible.

Résultat obtenu :



Il était demandé aussi que les threads soient interrompus lorsqu'on appuie sur la touche return. J'ai eu du mal à le faire. Généralement, sauf erreur de ma part, on utilise input("") pour python3, dans le main, pour qu'on stoppe les threads en cliquer sur la touche entrée mais ca ne fonctionner pas pour moi.

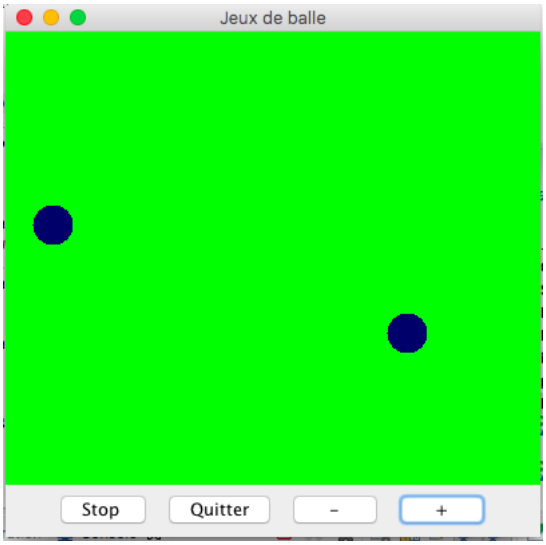
Du coup, il fallait que je crée une fonction Clavier en passant par un évènement

Résultat obtenu :

```
def Clavier(event):
    touche = event.keysym
    print(touche)
    if touche == 'Return':
        window.destroy()
```

5 Balle En Mouvement

Ici, il était demandé d'afficher des balles en mouvements dans une fenêtre avec plusieurs caractéristiques en plus (start, stop, ajout de balle, suppression de balles, collision des balles, affichage du score, d'une horloge). J'ai pas pu finir cet exercice. Voila ce que j'ai pu réalisé :



Il me manque les collision, l'horloge et le score.

Références

[1] Wikipedia : https://fr.wikipedia.org/wiki/Interface_graphique

[2] Notion d'évènement : http://tkinter.fdex.eu/doc/event.html

[3] Thread, Fenetre, GUI : http://openclassrooms.com/