Compte Rendu : DM sur les Threads

Kévin HOARAU, L3 informatique

10 octobre 2017

Résumé

Dans ce rapport , je vais vous montrer comment créer un thread , la création d'une interface graphique (tKinter ou Swing) en python et Java et les difficultés rencontrées au cours des exercices 2.3 et 3.2.

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 identifiant
— Sa priorité

Dans chaque processus, Il y a des threads. Il est créé lorsqu'un autre processus lance son exécution. Un processus dispose d'un thread principal depuis lequel on peut lancés d'autres threads parallèlement(thread démon). Lorsqu'on en crée plusieurs, on peut exécuter des tâches simultanément.Il est préférable de séparer les tâches dans plusieurs threads pour optimiser le programme.

En premier lieu, nous allons voir la notion de Thread. En deuxième lieu, nous parlerons d'interface graphique et puis nous finirons par voir ces notions et les difficultés rencontrées dans les exercices 2.3 et 3.2.

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 (c'est un **"sous-processus"**). Les threads d'un même processus partagent le **même espace d'adressage**, le **même segment de code** et le **même segment de données** mais possèdent chacun leur **propre compteur de programme** et **propre pile d'exécution**

2.2 Python

En Python, pour créer et gérer des threads on utilise le module threading qui fournit la classe Thread.

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 retenir les éléments suivants :

— 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.

Code :

```
class ThreadProducteur(Thread):
    def __init__(self,texte, q, tempsSommeil, nom):
        Thread.__init__(self)
        self.texte = Label(fen, text="Thread Producteur : Ajout de l'entier .")
        self.texte.pack()
        self.message = texte
        self.q = q
        self.n=0
        self.tempsSommeil = tempsSommeil
        self.name = nom
        self.daemon = True

    def run(self):
        while True:
            self.ajout = randint(1,100)
            self.q.put(self.ajout, block=True,timeout=None)
            self.texte["text"] = "Thread Producteur : Ajout de l'entier {} .".format(self.ajout)
            self.ajout=0
            self.texte.pack(expand=True, timeout=None)
            time.sleep(self.tempsSommeil)

  p = ThreadProducteur(texte,q, 1, "p")
  p.start()
```

2.3 Java

En Java, on va utiliser la classe `Thread` qui se trouve dans le package `java.lang`. Pour créer un thread et donc avoir la tâche exécutée en parallèle, il faut appeler la méthode `start` de la classe `Thread`. Cette méthode fait deux choses : elle crée tout d'abord un nouveau thread d'exécution et elle y exécute ensuite la méthode `run`.

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 !

3 Interface Graphique

Une interface graphique (GUI : Graphical User Interface) ou un environnement graphique est un dispositif de dialogue homme-machine

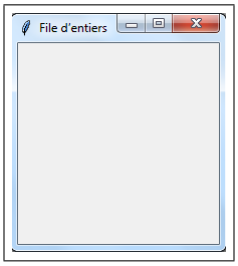
3.1 Python

Pour créer une interface graphique en python, on utilise la bibliothèque `Tkinter`. `Tkinter` a été renommé `tkinter` en python 3.

Voici un exemple de code en python :

```
from tkinter import *

fenetre = Tk()
fenetre.title('File d'entiers')
fenetre.mainloop()
```



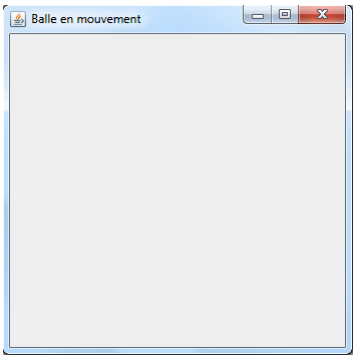
3.2 Java

Pour créer une interface graphique en Java, on utilise différentes bibliothèques mais essentiellement les packages `javax.swing, java.awt`.

Voici un exemple de code en Java :

```
import javax.swing.*;

public class Fenetre extends JFrame {
public Fenetre() {
      super("Balle en mouvement");
      setSize(350, 350);
      setLocationRelativeTo(null);
      setDefaultCloseOperation(EXIT_ON_CLOSE);
      setVisible(true);
   }
public static void main(String[] args) { new Fenetre(); }
}
```



4 Difficultés rencontrées

Maintenant je vais parlé des difficultés rencontrées au cours des exercices 2.3 et 3.2.

4.1 Exercice 2.3

Pour cette exercice, j'ai décidé de le faire sous Python. Au début, le problème que j'ai eu c'était qu'on devait appuyer sur la touche **Return** pour que les threads s'arrête. J'avais essayé avec **input()** mais ça lancé le thread producteur après je devais cliquer sur **Return** pour lancer le thread Consommateur. Pour résoudre ce problème j'ai utilisé un événement de clavier qui détecte les touches pressés du clavier.

Voici le code que j'ai utilisé :

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

4.2 Exercice 3.2

Pour cette exercice, j'ai décidé de le faire sous Java. Il fallait faire une fenêtre avec des balles en mouvements dedans. En ajoutant des collisions, l'affichage du score et de l'horloge. Je n'est pas pu finir cet exercice, il me manque les collisions, l'affichage du score et de l'horloge. Mais aussi la balle n'apparait pas(mais le tout dans un seul fichier ça fonctionne) et quelques détails dans le programme à régler(pause, quand on ajoute une balle c'est tjrs à la même position etc).

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/