TP1 Threads

1 message

ali lasfar <ali.lasfar@gmail.com>
To: bouiknaneMohammed.pro@gmail.com

Sat, 24 Dec 2022 at 11:31

EST Salé

Département Informatique

LP BigData

Programmation Concurrente en Java

TP N°1 Thread

Un thread est une partie des instructions du processus en cours d'exécution.

Un processus peut contenir un ou plusieurs thread (applications dites multi-threadées) s'exécutant en quasi-simultanéité ou simultanément sur les processeurs multi-cœurs.

Exercice 1

Étudier le programme suivant **:**

```
public class E1_Thread1  extends Thread {

private int nombre_max;


//constructeur du thread

E1_Thread1 (int nombre_max) {this.nombre_max=nombre_max;}


//code à exécuter par le thread
```

```
public void run() {

System.out.println(" Je suis le thread. Mes tâches consistent à compter de 1 à "+this.nombre_max+" "+ "en faisant une pause de 1 seconde entre chaque valeur");


for (int nombre=1; nombre <=nombre_max;nombre++) {

 System.out.println("Le compteur est à " + nombre + ".");

  try {

    Thread.sleep(1000); // une pause de 1 seconde

    } catch (InterruptedException e) {return; }

  }

          System.out.println(" Fin du thread ");

}// fin run


public static void main(String args[]) {

System.out.println(" Début  de Main  ");

E1_Thread1 lecompteur = new E1_Thread1(5);

  lecompteur.start();

  int n = 0;

  while (lecompteur.isAlive ()) {

  System.out.println("en train de compter (n=" + n + ")" );

  n = n + 1;

  try {

      Thread.sleep(3000); // une pause de 3 secondes

       } catch(InterruptedException e) {}

          }System.out.println(" Fin  de Main  ");

    }// fin main

    } // fin E1_Thread1
```

1- Quels sont les résultats affichés que l'on pourra observer ? (Tester plusieurs exécutions)

2- On varie la valeur du délai dans la méthode run à 6000 ms.
Quels sont les résultats affichés ?

Exercice 2

Étudier le programme suivant :

```
public class E2_thread2  extends Thread {

   private static int numThread = 0; // nombre de threads crées dans Main

   private int numero; // numéro de ce thread


   // constructeur

   public E2_thread2() {

      numero = numThread;

      numThread = numThread + 1;

      System.out.println("Thread numero " + numero + " cree." );

   }


   public void run() {

      System.out.println("Thread numero " + numero + " demarre." );

       try {

          Thread.sleep(1000); // pause de 1 seconde

       } catch (InterruptedException e) {return; }

      System.out.println("Thread numero " + numero + " termine." );

   }//fin run


   public static void main(String args[]) {

      System.out.println("Programme demarre..");

      // Creation de 5 threads
```

```
        for (int i=0; i < 5; i++) {
            Thread unThread = new E2_thread2();
            unThread.start();}
        System.out.println("Programme principal termine.");
    }// fin main
} // fin E2_thread2
```

1- Combien de threads s'exécutent ?.

2- Quelle est la première ligne affichée? Et la dernière?

3- L'argument de l'appel sleep() est un nombre de millisecondes. Supprimer cet appel (commenter tout le bloc try/catch). Pourquoi le résultat affiché est-il si différent ?

4- Ajouter une variable de classe (static) de type int nommée partage et initialisée à zéro. Modifier la méthode run() pour qu'elle incrémente cette variable en suivant l'algorithme suivant:

```
int acc = partage;
attendre 1 ms
acc = acc + 1;
partage = acc;
```

5- Faire afficher la valeur de la variable partage à la fin du main().Que s'affiche-t-il ? Pourquoi?

Exercice 3

Soit la classe suivante :

```
public class Alphabet {
    public void affiche() {
        for (char a = 'A'; a <= 'Z'; a++) {
            System.out.print(a);
            try { Thread.sleep(10); // ms
            } catch (InterruptedException e) {}
```

```
    }
        System.out.print("\n");
    }
    public static void main(String args[]) {
        Alphabet A = new Alphabet();
        A.affiche();
    }
}
```

Modifier cette classe en utilisant l'interface Runnable pour que l'affichage se fasse dans un thread séparé. On pourra donc écrire le

programme de test suivant :

```
AlphabetThread A1 = new AlphabetThread();
Thread T1 = new Thread(A1);
AlphabetThread A2 = new AlphabetThread();
Thread T2 = new Thread(A2);
T1.start();
T2.start();
```

Exercice 4 : Des threads indépendants

Un "compteur" a un nom (*Programmation Concurrente* par exemple) et il compte de 1 à n (nombre entier positif quelconque). Il marque une pause aléatoire entre chaque nombre (de 0 à 5000 millisecondes par exemple).

Un compteur affiche chaque itération (Toto affichera par exemple, "Toto : 3") et il affiche un message du type "*** Toto a fini de compter jusqu'à 10" quand il a fini.

Ecrivez la classe compteur et testez-la en lançant plusieurs compteurs qui comptent jusqu'à 10. Voyez celui qui a fini le plus vite.

Exercice 5 : Des threads un peu dépendants

Modifiez la classe Compteur pour que chaque compteur affiche son ordre d'arrivée : le message de fin est du type : "Toto a fini de compter jusqu'à 10 en position 3".

Un thread possède une priorité et un nom. Si aucun nom particulier n'est donné dans le constructeur du thread, un nom par défaut composé du préfixe "Thread-" suivi d'un numéro séquentiel incrémenté automatiquement lui est attribué.

Abdelali LASFAR - Enseignant Chercheur
Département Informatique
Ecole Supérieure de Technologie - Salé
Avenue Prince Héritier, B.P 227 Salé Maroc
Tél (212) 5 37 88 15 61
Fax (212) 5 37 88 15 64
abdelai.lasfar@um5.ac.ma