Algorithmique parallèle et distribuée

Threads

Pour se faire la main

Exercice 1

1. Ecrire une classe *UnThread*
   – héritant de *java.lang.Thread* ;
   – affichant 10 fois "hello world".
   Essayez les différentes méthodes vues en cours pour l'implémentation d'un thread.

2. Ecrire un programme principal
   – créant 5 objets de type UnThread ;
   – lançant leur exécution (start).

3. Transformez le programme de façon à ce que chaque thread
   – soit identifié par un entier par le biais d'une variable qui lui est propre ;
   – affiche cette identité en complément de chaque "hello world".

4. Lancez le programme a plusieurs reprise et observez le résultat. Que se passe t il ? Peut on prédire l'ordre d'entrelacement des threads ?

5. Modifiez à nouveau le programme pour ne plus utiliser de variables supplémentaire pour identifier les threads (setName() et getName() sont deux méthodes de la classe thread permettant cette modification).

Des compteurs

Exercice 2

1. Un "compteur" a un nom (Toto 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) ;
   – `Random r = new Random();`
   crée un générateur de nb aléatoire ;
   – `r.nextBoolean() ;`
   retourne un booléen choisi aléatoirement ;

  - `int r.nextInt(int max) ;`
    retourne un entier choisi aléatoirement entre 0 et max
  - `Thread.sleep();`
    méthode de classe statique permettant l'endormissement d'un thread
- il affiche chaque nombre (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 ;
- écrivez la classe compteur et testez-la en lançant plusieurs compteurs qui comptent jusqu'à 10. Renouvelez l'opération à plusieurs reprises, et voyez celui qui a fini le plus vite.

2. 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". Lancez le programme a plusieurs reprise et observez le résultat. Que se passe t il ?

3. Modifiez maintenant la classe Compteur afin que chaque instance incrémente un même compteur partagé. A nouveau, celui qui incrémente le compteur à la valeur 10 affiche un message prévenant qu'il a terminé. Lancez le programme à plusieurs reprises. Que se passe t il ?

Une chaine partagée

Exercice 3

1. Créez une classe thread dont chaque instances :
   - possède un nom qui lui est propre (reçu par le constructeur) ;
   - partage une chaine de caractère commune ;
   - concatène son nom à la chaine commune lorsqu'il est executé.

2. La méthode principale lancera quelques instances de ces threads (avec des noms de longueur différentes) puis affichera la chaine commune.

3. Lancez le programme a plusieurs reprise et observez le résultat. Que se passe t il ?