Télécom Saint-Étienne
FISE2 2017-2018
Java INFO 4.1
Lab #10 - 3h00

This year's teachers: C. Gravier, F. Laforest, J. Subercaze

Goals of the session:

Basic Multithreading

- Thread synchronization
- Understand data synchronization issues
- Inter-thread communication

Official documentation and tutorial https://docs.oracle.com/javase/tutorial/essential/concurrency/.

Rules

You must read carefully this document. Your time is valuable, as is the time of your teacher. Every question to the teacher whose answer is to be found in this document will lead to a lock of the resource teacher for 10 minutes. This means that nobody in the group will be able to any question to the teacher.

1 Exercise 1: Synchronization

The code for this exercise is located in the package `fr.tse.fise2.info4.lab10.exercise1`. The unit test is: `fr.tse.fise2.info4.lab10.exercise1.Exercise1Test` in the test folder.

There are three classes in the `src` folder:

Counter Implements a counter that can be incremented

ThreadModifier Thread that increments a given counter

Exercise1 The main class

Question 1.1 - Observe Run the the unit test several times, what do you see? What is the expected value, what is/are the actual(s)? How is this error called?

Question 1.2: Understand Draw (on a piece of paper, really) a timeline of the execution using 2 threads plus one for the counter values. To represent the execution draw three vertical lines. The middle one is the counter, and the right and left lines are the threads. Represent calls to the counter using arrows and display the internal values of the counter. Local variables (where the scope is a function), are local to each thread. This will help you understanding why the program is incorrect.

Question 1.3: Fix You will fix this algorithm using three diferrent solutions You will copy the package content into three different packages, one for each solution. Do not forget to copy the unit tests accordingly.

You will use:

- The keyword synchronized. Use it only once in your code.

- Using Atomic Objects
- Using Locks

Exercise 2 - Parallel computation

The main application of multithreaded application is to parallelize the execution of some algorithms, in order to boost performance. In this exercise you will setup a parallel word counter, using Queue as a mechanism for thread communications in a producer/consumers situation.

The producer generate random sentences (Lorem Ipsum ...) and the consumers must count the number of words in these sentences.

Questions

1. Complete the `Consumer` to count word in sentences. You must pass the unit test.
2. Complete the `Producer` to produce sentences and send them to the queue. Detailed instructions are in the Javadoc and in the comments
3. Complete the `Consumer` to read data from the Queue.
4. Create a main file to execute the producer and a number of threads equal to the number core available on your machine. You must get this number from the JVM. Then display the number of words.
5. Write a integration test based on the above question. Ensure that the count done by the consumers is equal to the number of word produced.
6. Does your test hang? Google what a poison pill is.

Online resources

Multithreading questions are very popular in technical interviews. Here are some that you can try:

- http://www.techbeamers.com/java-multithreading-quiz-with-20-interview-questions/
- http://www.javamadesoeasy.com/2015/10/threadmulti-threading-quiz-in-java-mcq.html