



Instituto Tecnológico Superior de Jerez

Jerez, Zacatecas

20 de marzo de 2020

Felipe de Jesús Márquez Olague

flype360360@gmail.com

S18070182

Ingeniería en Sistemas Computacionales

Cuarto Semestre

Tópicos avanzados de Programación

Mapa conceptual

M.T.I. Salvador Acevedo Sandoval

Exercises

1. Fill in the blanks in each of the following statements:

- a) A thread enters the *terminated* state when method ends.
- b) To pause for a designated number of milliseconds and resume execution, a thread should call method sleep of class Thread.
- c) Method signal of class Condition moves a single thread in an object's *waiting* state to the *runnable* state.
- d) Method signalAll of class Condition moves every thread in an object's *waiting* state to the *runnable* state.
- e) A(n) Runnable thread enters the terminated state when it completes its task or otherwise terminates.
- f) A *runnable* thread can enter the timed waiting state for a specified interval of time.
- g) At the operating-system level, the *runnable* state actually encompasses two separate states, ready and running.
- h) Runnables are executed using a class that implements the Executor interface.
- i) ExecutorService method shutdown ends each thread in an ExecutorService as soon as it finishes executing its current Runnable, if any.
- j) A thread can call method await on a Condition object to release the associated Lock and place that thread in the *waiting* state.
- k) In a(n) producer/consumer relationship, the producer generates data and stores it in a shared object, and the consumer reads data from the shared object.
- l) Class ArrayBlockingQueue implements the BlockingQueue interface using an array.
- m) Keyword Synchronized indicates that only one thread at a time should execute on an object.

2. State whether each of the following is *true* or *false*. If *false*, explain why.

- a) A thread is not *runnable* if it has terminated.
- b) Some operating systems use timeslicing with threads. Therefore, they can enable threads to preempt threads of the same priority.
- c) When the thread's quantum expires, the thread returns to the *running* state as the operating system assigns it to a processor.
- d) On a single-processor system without timeslicing, each thread in a set of equal-priority threads (with no other threads present) runs to completion before other threads of equal priority get a chance to execute.

3. (True or False) State whether each of the following is *true* or *false*. If *false*, explain why.

- a) Method `sleep` does not consume processor time while a thread sleeps.
Is True
- b) Declaring a method synchronized guarantees that deadlock cannot occur.
Is false, because Timeslicing allows a thread to execute until its timeslice expires. Then other threads of equal priority can execute.
- c) Once a `ReentrantLock` has been obtained by a thread, the `ReentrantLock` object will not allow another thread to obtain the lock until the first thread releases it.
Is False because when a thread's quantum expires, the thread returns to the ready state and the operating system assigns to the processor another thread.
- d) Swing components are thread safe.
Is true

4. (Multithreading Terms) Define each of the following terms.

- a) thread.** - An individual execution context of a program
- b) multithreading.** - The ability of more than one thread to execute concurrently.
- c) runnable state.** - A state in which the thread is capable of running
- d) timed waiting state.** - A state in which the thread cannot use the processor because it is waiting for a time interval to expire or a notification from another thread.
- e) preemptive scheduling.** - A thread of higher priority enters a running state and is assigned to the processor. The thread preempted from the processor is placed back in the ready state according to its priority.
- f) Runnable interface.** - An interface that provides a run method. By implementing the Runnable interface, any class can be executed as a separate thread.
- g) notifyAll method.** - Transitions all threads waiting on an object's monitor to the runnable state.
- h) producer/consumer relationship.** - A relationship in which a producer and a consumer share common data. The producer typically wants to "produce" (add information) and the consumer wants to "consume" (remove information).
- i) quantum.** - A small amount of processor time, also called a time slice.

5. (Multithreading Terms) Define each of the following terms in the context of Java's threading mechanisms:

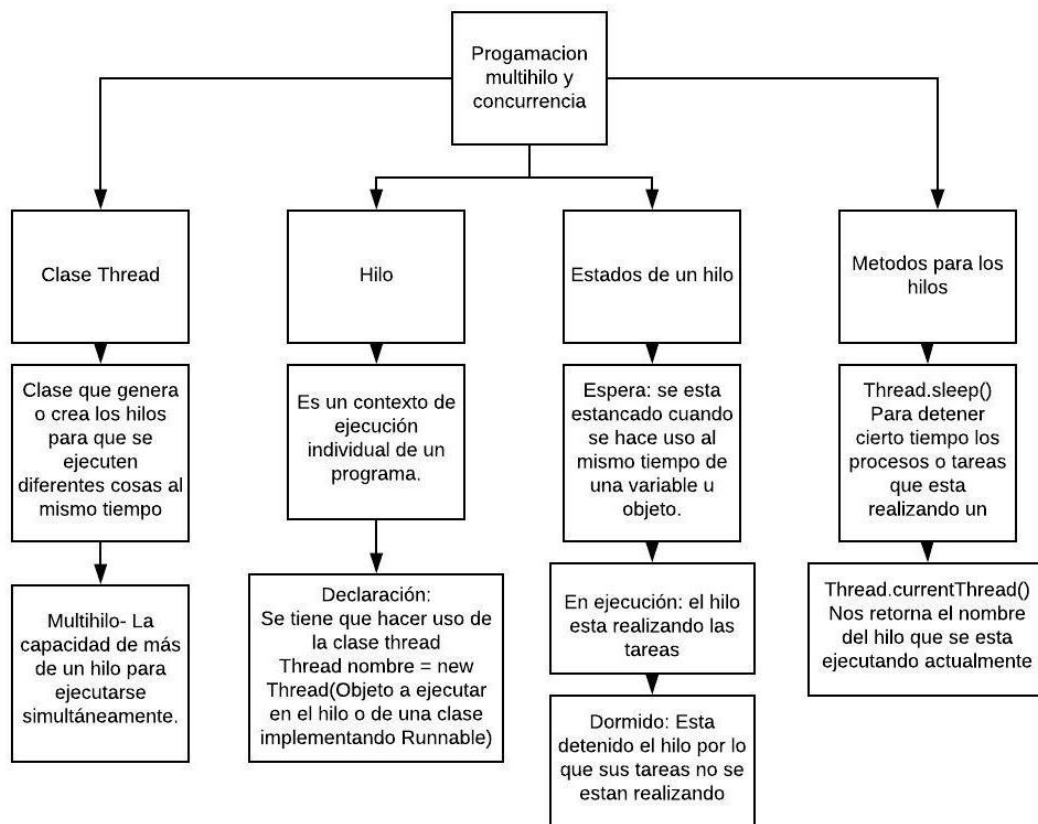
- a) synchronized.** - When a method or block is declared synchronized and it is running, the object is locked. Other threads cannot access the other synchronized methods of the object until the lock is released.
- b) producer.** - A thread that writes data to a shared memory resource.
- c) consumer.** - A thread that reads data from a shared memory resource.

d) wait. - Places a thread in the waiting state until another thread call notify or notifyAll on the same object or until a specified amount of time elapses.

e) notify. - Wake a thread currently waiting on the given object.

f) Lock. - An interface implemented by objects that control access to a resource shared among multiple threads.

g) Condition. - Objects of this interface represent condition variables that can be used with Locks to manage access to a shared resource



Referencias

Paul Deitel. (1996). Multithreading. En Java How To Program(1045-1115). Harvey Deitel Deitel & Associates, Inc.