Institución: Instituto tecnológico superior de jerez

Lugar: Libramiento Fresnillo-Tepetongo, Fracc. Los Cardos, 99863 Jerez de García Salinas, Zac.

Fecha: 26 de marzo del 2021

Alumno: Bryan Ezequiel Valdez Calderón

No. de control: S19070118

Correo electrónico: bryan501th@gmail.com

Carrera: Ingeniería en sistemas computacionales

Materia: Tópicos Avanzados de Programación

Semestre: 4

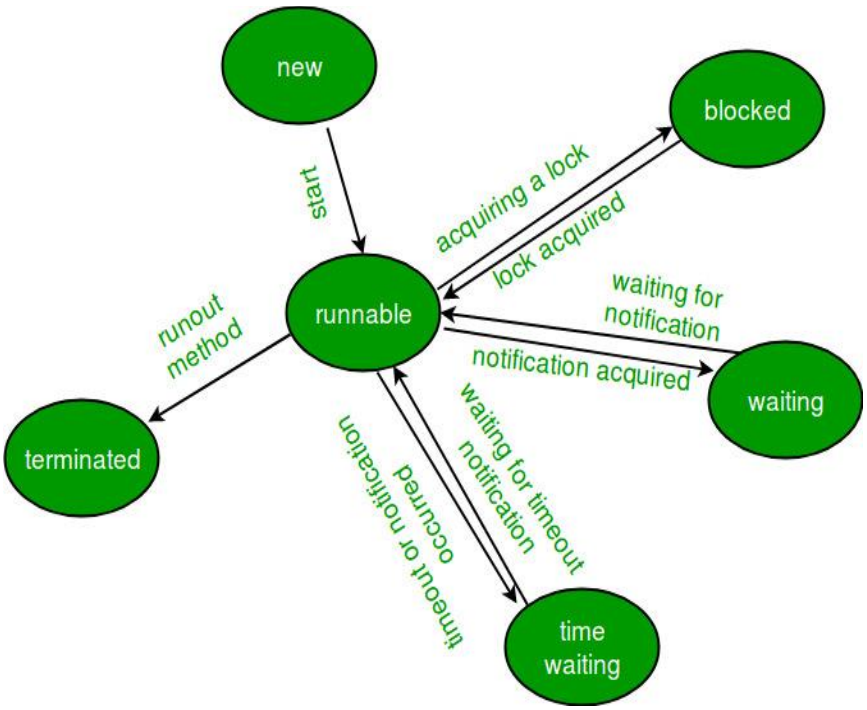
Actividad: Actividad 1 - Programas en Java [Concurrencia]

Docente: Acevedo Sandoval Salvador

Un hilo en Java existe en cualquier punto del timpo en cualquiera de los siguientes estados, un hilo está solo en uno de los siguientes estados en un instante:

1. New
2. Runnable
3. Blocked
4. Waiting
5. Timed Waiting
6. Terminated

El diagrama de abajo representa varios estados de un hilo en cualquier instante de tiempo;



Fuente de la imagen: Core Java Vol 1, 9th Edition, Horstmann, Cay S. & Cornell, Gary 2013

Ciclo de vida de un hilo

1. **Nuevo hilo:**
   Cuando un nuevo hilo es creado, está en el estado nuevo. El hilo no se ha empezado a ejecutar cuando está en este estado. Cuando un hilo está en este estado, su código aún debe ser ejecutado y no se ha empezado a ejecutar.

2. **Estado ejecutable:**
   Un hilo que está listo para ejecutarse es movido al estado ejecutable. En este estado, un hilo puede estar corriéndose o bien puede estar listo en cualquier instante del tiempo. Es la responsabilidad del organizador de hilos de darle, el tiempo para ejecutarse

   Un programa multi-hilo asigna una cantidad de tiempo para cada hilo individual. Todos y cada uno de los hilos corre en un corto tiempo mintras y pausa y

renuncia el hilo a otro thread, para que otros hilos tengan una oportunidad de ejecutarse. Cuando pasa, todos los hilos están listos para ejecutarse, esperando que el CPU y que los hilos en ejecución estén en ejecución.

3. **Estado bloqueado/espera:**
   Cuando un hilo está temporalmente inactivo, entonces está en uno de los siguientes estados:
- Blocked
- Waiting

   Por ejemplo, cuando un hilo está esperando que un I/O terminé, está en el estado blocked. Es la responsabilidad del administrador de hilos reactivar y administrar un hilo en estado blocked/waiting. Un hilo en este estado no puede continuar su ejecución más lejos hasta que es movido a un estado ejecutable. Cualquier hilo en esos estados no consume ningún ciclo del procesador.

   Un hilo está en el estado bloqueado cuando intenta acceder a unasección protegida de código que e está bloqueada por algún otro hilo. Cuando la sección protegida es desbloqueada, el administrador escoje uno de los hilos que es bloqueado para esa sección y lo mueve al estado ejecutable. En donde, un hilo que está en el estado de espera cuando espera para otro hilo en una condición. Cuando su condición es cumplida, el administrador es notificado y el hilo de espera es movido al estado ejecutable.

   Si un hilo que está en ejecución es movido al estado blocked/waiting, otro hilo en el estado ejecutable es administrado por el administrador de hilos para ejecutarse. Es la responsabilidad del administrador de hilos determiner cuál hilo correr.

4. **Timed Waiting:**
   Un hilo está en estado de tiempo esperado cuando llama un método con un parámetro de tiempo de espera. Un hilo está en este estado hasta que el tiempo de espera se completa o hasta que una notificación es recibida. Por ejemplo, cuando un hilo llama a sleep o una espera condicional, es movida a un estado de tiempo de espera.
5. **Estado terminado:**
   Un hilo se termina por cualquiera de las siguientes razones:
- Existe normalmente porque. Esto pasa cuando el código del hilo se ha ejecutado enteramente por el programa.
- Porque ha ocurrido un evento inusualmente erronéo, como la segmentación o una excepción no manejada.

   Un hilo que está en un estado terminado ya no consume ningún ciclo del CPU

Implementando estados de Hilos en Java

In Java, to get the current state of the thread, use **Thread.getState()** method to get the current state of the thread. Java provides **java.lang.Thread.State** class

that defines the ENUM constants for the state of a thread, as summary of which is given below:

En Java, para conseguir el estado actual de un hilo, se usa el método **Thread.getState()** para conseguir el estado actual del hilo. Java le otorga a la clase **java.lang.Thread.State** que define las constantes ENUM para el estado de un hilo, un resumen de cuál se da abajo:

1. **Tipo de constante:** New
   `Declaración: public static final Thread.State NEW`

   Descripción: El estado del hilo para un hilo que no ha sido iniciado aún.
2. **Tipo de constante:** Runnable
   `Declaración: public static final Thread.State RUNNABLE`

   Descripción:
   El estado de un hilo para un hilo ejecutable. Un hilo en el estado ejecutable se está ejecutando en la máquina vitual de Java pero puede estar esperando para otros recursos desde el sistema operativo como el procesador.
3. **Tipo de constante:** Blocked
   `Declaración: public static final Thread.State BLOCKED`

   Descripción: Thread state for a thread blocked waiting for a monitor lock. A thread in the blocked state is waiting for a monitor lock to enter a synchronized block/method or reenter a synchronized block/method after calling Object.wait().
4. **Tipo de constante:** Waiting
   `Declaración: public static final Thread.State WAITING`

   Descripción:
   Estado de hilo para un hilo en espera. Un hilo está en el estado de epsera debido a que está llamando a uno de los siguientes métodos:

- Object.wait with no timeout
- Thread.join with no timeout
- LockSupport.park

  Un hilo en el estado de espera está esperando por otro hilo para hacer una acción particular.

5. **Tipo de constante:** Tiempo esperando
   `Declaración: public static final Thread.State TIMED_WAITING`

   Descripción:
   Estado de hilo para un hilo en espera con un tiempo especificado de espera. Un hilo está en el estado de espera debido a que está llamando uno de los siguientes métodos con un tiempo positivo de espera específico:

- Thread.sleep
- Object.wait with timeout
- Thread.join with timeout

- LockSupport.parkNanos
- LockSupport.parkUntil

6. **Tipo de constante:** Terminated
   `Declaración: public static final Thread.State TERMINATED`

   Descripción:
   Estado de hilo para un hilo terminado. El hilo ha completado su ejecución.

   La llave synchronized es usada para hacer que la clase o método hilo-seguro lo cuál significa que solo un hilo puede tener el bloqueo del método synchronized y usarlo, otros hilos tienen que esperar hasta que el bloqueo se libera y cualquiera de ellos adquiere ese bloqueo. Es importante usar si nuestro programa está ejecutándose en un entorno multihilo donde

Figure - 1

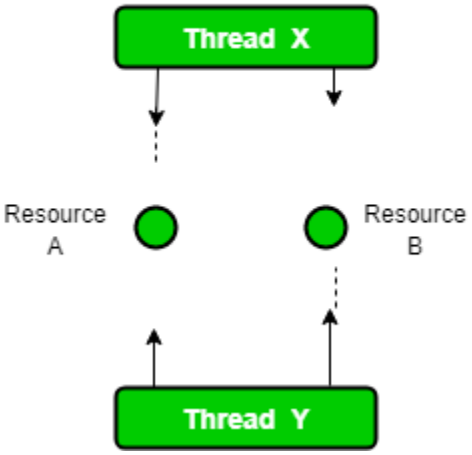
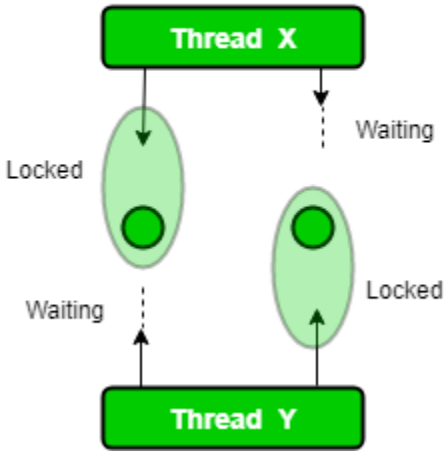


Figure - 2



No es recomendado ejecutar el programa de arriba con un IDE en línea. podemos copiar el código fuente y correrlo en nuestra maquina local. Podemos ver que corre por un tiempo indefinido, porque los hilos están en la condición deadlock y no deja que el código se ejecute.

También podemos detectar el deadlock corriendo el programa a través del cmd. Debemos recolectar el vertedero de hilos. El comando de recolectar depende del tipo del sistema operativo. Si estamos usand Windows y Java 8, el comando es jcmd $PID Thread.print.
Podemos conseguir el PID al correr el comando jps.

Mapa Conceptual