Instituto Tecnológico Superior De Jerez.

Ingeniería en Sistemas Computacionales

Tópicos Avanzados de Programación

4to semestre

Tema 3. Programación concurrente

Mapa Conceptual

Oscar Vargas Garcia

OscarVarGar@outlook.es

Jerez de García Salinas

MTI Salvador Acevedo Sandoval

13 de junio de 2024

Ciclo de Vida y Estados de un Hilo en Java

En cualquier momento, un hilo en Java existe en uno de los siguientes estados. Un hilo se encuentra solo en uno de los estados mostrados en un instante determinado:

Estado Nuevo

Estado Runnable (Ejecutable)

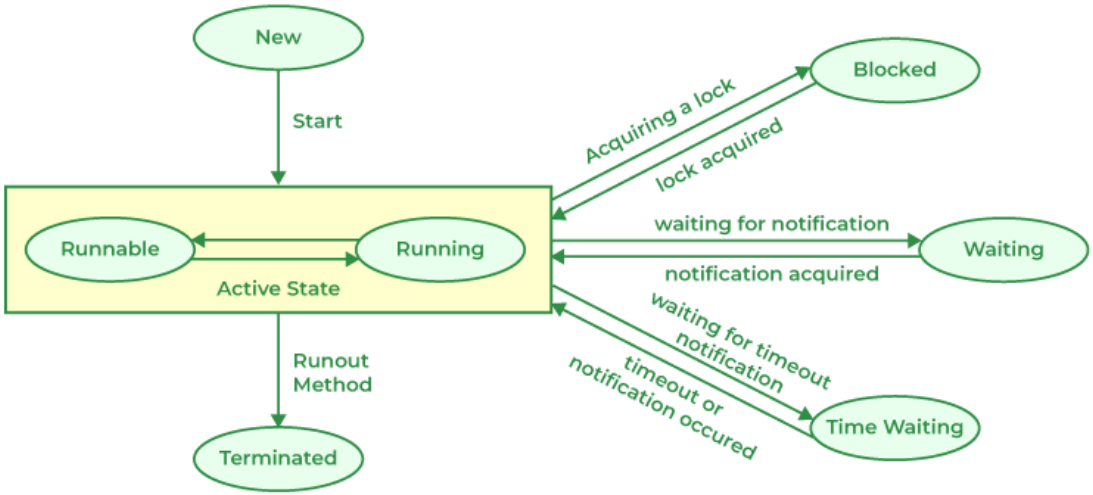
Estado Bloqueado

Estado de Espera

Estado de Espera con Tiempo

Estado Terminado

El diagrama que se muestra a continuación representa los diversos estados de un hilo en cualquier instante en el tiempo.



Ciclo de Vida de un Hilo

Hay múltiples estados de un hilo en su ciclo de vida, como se menciona a continuación:

- Nuevo Hilo: Cuando se crea un nuevo hilo, se encuentra en el estado nuevo. El hilo aún no ha comenzado a ejecutarse cuando se encuentra en este estado. Cuando un hilo está en el estado nuevo, su código aún no se ha ejecutado y no ha comenzado a ejecutarse.

- Estado Runnable (Ejecutable): Un hilo que está listo para ejecutarse se mueve a un estado runnable. En este estado, un hilo puede estar realmente ejecutándose o puede estar listo para ejecutarse en cualquier momento. Es responsabilidad del planificador de hilos darle al hilo el tiempo para ejecutarse. Un programa multihilo asigna una cantidad fija de tiempo a cada hilo individual. Cada hilo se ejecuta por un corto tiempo y luego se pausa y cede la CPU a otro hilo para que otros hilos puedan tener la oportunidad de ejecutarse. Cuando esto sucede, todos los hilos que están listos para ejecutarse, esperando la CPU y el hilo que actualmente se está ejecutando se encuentran en un estado

runnable.

- Bloqueado: El hilo estará en estado bloqueado cuando esté intentando adquirir un bloqueo pero actualmente el bloqueo está adquirido por otro hilo. El hilo se moverá del estado bloqueado al estado runnable cuando adquiera el bloqueo.

- Estado de Espera: El hilo estará en estado de espera cuando llame al método wait() o al método join(). Se moverá al estado runnable cuando otro hilo lo notifique o cuando ese hilo termine.

- Espera con Tiempo: Un hilo se encuentra en un estado de espera con tiempo cuando llama a un método con un parámetro de tiempo de espera. Un hilo está en este estado hasta que se complete el tiempo de espera o hasta que se reciba una notificación. Por ejemplo, cuando un hilo llama a sleep o a una espera condicional, se mueve a un estado de espera con tiempo.

- Estado Terminado: Un hilo termina por alguna de las siguientes razones:

  Porque sale normalmente. Esto sucede cuando el código del hilo ha sido completamente ejecutado por el programa.

  Porque ocurrió algún evento erróneo inusual, como una falla de segmentación o una excepción no manejada.

Implementación de los Estados de Hilo en Java

En Java, para obtener el estado actual del hilo, usa el método Thread.getState() para obtener el estado actual del hilo. Java proporciona la clase java.lang.Thread.State que define las constantes ENUM para el estado de un hilo, un resumen de las cuales se da a continuación:

Nuevo (New)
Estado del hilo para un hilo que aún no ha comenzado.

```
public static final Thread.State NEW
```

Runnable (Ejecutable)

Estado del hilo para un hilo runnable. Un hilo en el estado runnable se está ejecutando en la máquina virtual de Java, pero puede estar esperando otros recursos del sistema operativo, como un procesador.

```
public static final Thread.State RUNNABLE
```

Bloqueado (Blocked)
Estado del hilo para un hilo bloqueado esperando un bloqueo de monitor. Un hilo en el estado bloqueado está esperando un bloqueo de monitor para entrar en un bloque/método sincronizado o para volver a entrar en un bloque/método sincronizado después de llamar a Object.wait().

```
public static final Thread.State BLOCKED
```

Esperando (Waiting)
Estado del hilo para un hilo en espera. Un hilo está en el estado de espera debido a la llamada a uno de los siguientes métodos:

- Object.wait sin tiempo de espera
- Thread.join sin tiempo de espera
- LockSupport.park

```
public static final Thread.State WAITING
```

Espera con Tiempo (Timed Waiting)
Estado del hilo para un hilo en espera con un tiempo de espera especificado. Un hilo está en el estado de espera con tiempo debido a la llamada a uno de los siguientes métodos con un tiempo de espera positivo especificado:

- Thread.sleep
- Object.wait con tiempo de espera
- Thread.join con tiempo de espera
- LockSupport.parkNanos

- LockSupport.parkUntil

```
public static final Thread.State TIMED_WAITING
```

Terminado (Terminated)

Estado del hilo para un hilo terminado. El hilo ha completado su ejecución.

```
Declaration: public static final Thread.State TERMINATED
```

Bloqueo en la Programación Multihilo en Java

El modificador synchronized se utiliza para hacer que una clase o método sea seguro para subprocesos, lo que significa que solo un hilo puede tener el bloqueo del método sincronizado y usarlo, mientras que otros hilos deben esperar hasta que se libere el bloqueo y uno de ellos lo adquiera.

Es importante utilizarlo si nuestro programa se ejecuta en un entorno multihilo donde dos o más hilos pueden ejecutarse simultáneamente. Sin embargo, a veces también puede causar un problema conocido como "Deadlock" (bloqueo mutuo). A continuación, se muestra un ejemplo simple de la condición de Deadlock:

Figure - 1

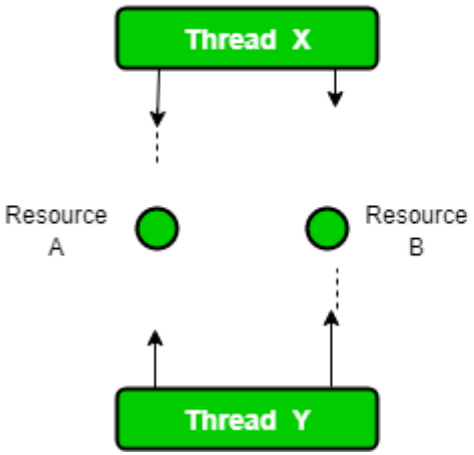
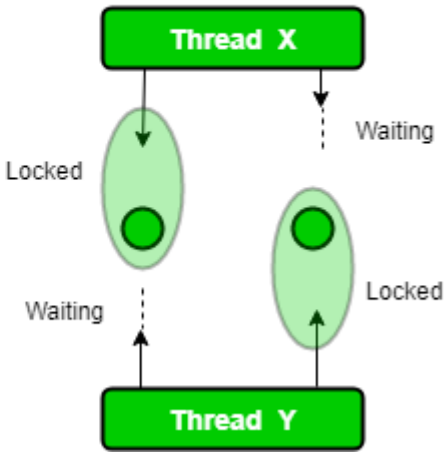


Figure - 2



Evitar la condición de bloqueo mutuo

Podemos evitar la condición de bloqueo mutuo al conocer sus posibilidades. Es un proceso muy complejo y no es fácil de detectar. Sin embargo, si lo intentamos, podemos evitar esto. Hay algunos métodos mediante los cuales podemos evitar esta condición. No podemos eliminar completamente su posibilidad, pero podemos reducirla.

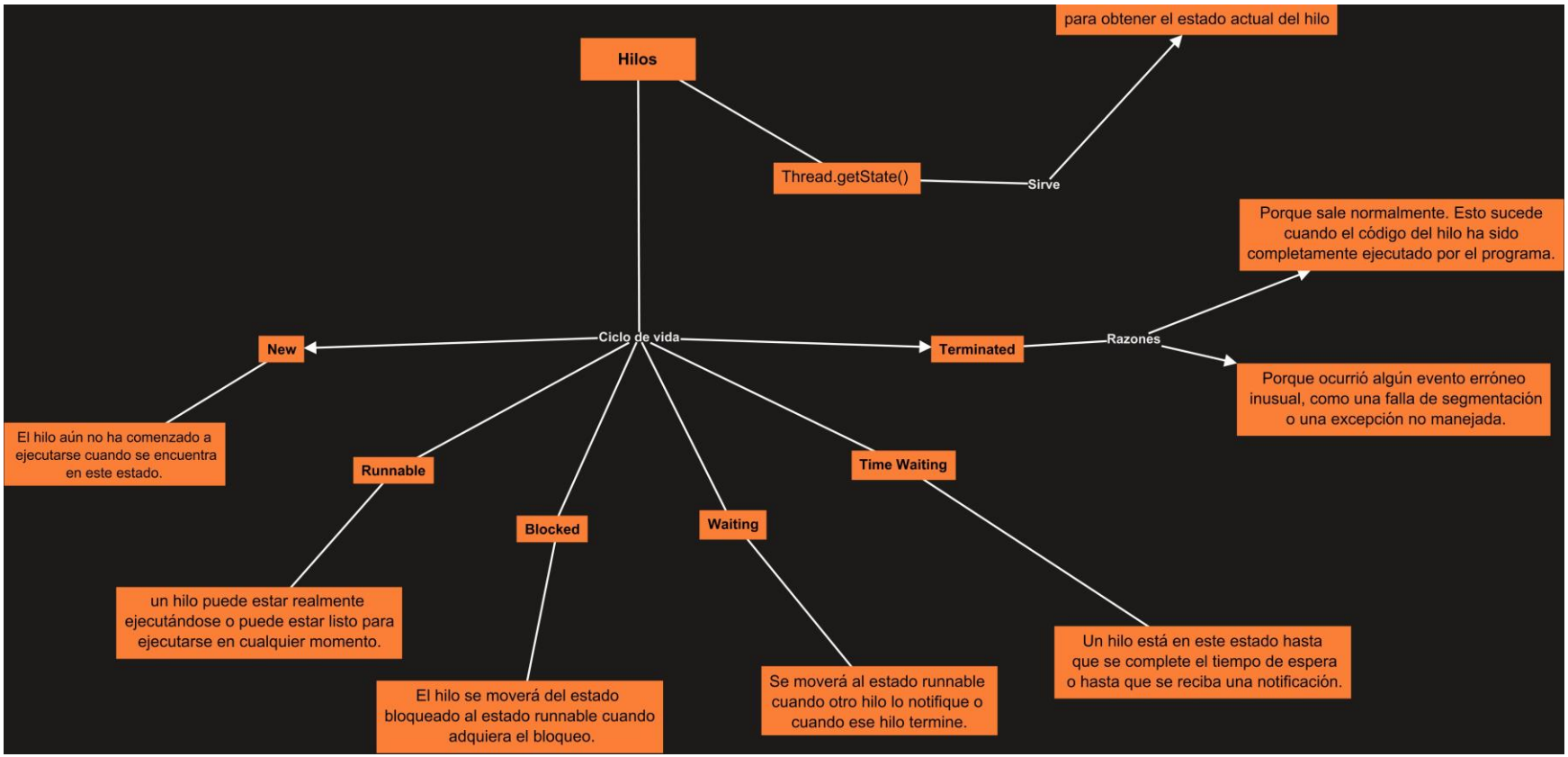
Evitar Bloqueos Anidados: Esta es la principal razón del bloqueo mutuo. El bloqueo mutuo ocurre principalmente cuando otorgamos bloqueos a múltiples hilos. Evita otorgar bloqueos a múltiples hilos si ya se ha otorgado a uno.

Evitar Bloqueos Innecesarios: Deberíamos bloquear solo aquellos miembros que sean necesarios. Tener un bloqueo innecesario puede llevar al bloqueo mutuo.

Uso de Thread.join: La condición de bloqueo mutuo ocurre cuando un hilo está esperando a que otro termine. Si ocurre esta condición, podemos usar Thread.join con el tiempo máximo que creemos que tomará la ejecución.

Puntos Importantes:

Si los hilos están esperando que los demás terminen, entonces se conoce como bloqueo mutuo. El bloqueo mutuo es una condición compleja que ocurre únicamente en el caso de múltiples hilos. El bloqueo mutuo puede romper nuestro código en tiempo de ejecución y puede destruir la lógica empresarial. Deberíamos evitar esta condición tanto como sea posible.


Hilos
Thread.getState()
Sirve
para obtener el estado actual del hilo
Ciclo de vida
New
El hilo aún no ha comenzado a ejecutarse cuando se encuentra en este estado.
Runnable
un hilo puede estar realmente ejecutándose o puede estar listo para ejecutarse en cualquier momento.
Blocked
El hilo se moverá del estado bloqueado al estado runnable cuando adquiera el bloqueo.
Waiting
Se moverá al estado runnable cuando otro hilo lo notifique o cuando ese hilo termine.
Time Waiting
Un hilo está en este estado hasta que se complete el tiempo de espera o hasta que se reciba una notificación.
Terminated
Razones
Porque sale normalmente. Esto sucede cuando el código del hilo ha sido completamente ejecutado por el programa.
Porque ocurrió algún evento erróneo inusual, como una falla de segmentación o una excepción no manejada.