Apéndice A: Métodos de la Clase `Thread`

`Thread()`
Realiza la reserva de memoria necesaria para la creación de un nuevo objeto `Thread`.

`Thread(Runnable` *`target`*`)`
Realiza la reserva de memoria necesaria para la creación de un nuevo objeto `Thread`.

`Thread(Runnable target, String` *`name`*`)`
Realiza la reserva de memoria necesaria para la creación de un nuevo objeto `Thread`.

`Thread(String` *`name`*`)`
Realiza la reserva de memoria necesaria para la creación de un nuevo objeto `Thread`.

`Thread(ThreadGroup` *`group`*`, Runnable` *`target`*`)`
Realiza la reserva de memoria necesaria para la creación de un nuevo objeto `Thread`.

`Thread(ThreadGroup` *`group`*`, Runnable` *`target`*`, String` *`name`*`)`
Crea un nuevo objeto `Thread` con un objeto de ejecución concreto y un nombre concreto, y se une al grupo de hilos especificado.

`Thread(ThreadGroup` *`group`*`, String` *`name`*`)`
Crea un nuevo objeto `Thread` como miembro de un grupo de hilos concreto.

`static int activeCount()`
Devuelve el número actual de hilos activos en el grupo de hilos de este hilo.

`void checkAccess()`
Determina si el hilo actualmente en ejecución tiene permiso para modificar este hilo.

`static Thread currentThread()`
Devuelve una referencia al objeto hilo que se está ejecutando actualmente.

`void destroy()`
Destruye este hilo, sin realizar ningún tipo de limpieza.

`static void dumpStack()`
Imprime una traza de pila del hilo actual.

`static int enumerate(Thread[]` *`tarray`*`)`
Copia dentro del array especificado todos los hilos activos del grupo y subgrupos de hilos del hilo en cuestión.

`ClassLoader getContextClassLoader()`
Devuelve el contexto `ClassLoader` de este `Thread.`

`String getName()`
Devuelve el nombre del hilo.

`void setName(String name)`
Cambia el nombre de este hilo, asignándole el especificado como argumento.

`int getPriority()`
Devuelve la prioridad del hilo.

`ThreadGroup getThreadGroup()`
Devuelve el grupo de hilos al cual pertenece el hilo.

`void interrupt()`
Interrumpe la ejecución del hilo.

`static boolean interrupted()`
Comprueba si el hilo actual ha sido interrumpido.

`boolean isAlive()`
Comprueba si el hilo está vivo.

`boolean isDaemon()`
Comprueba si el hilo es un hilo demonio.

`void setDaemon(boolean on)`
Establece este hilo como hilo daemon, o como hilo de usuario.

`void join()`
Espera a que este hilo muera.

`void join(long millis)`
Espera, como mucha *`millis`* milisegundos a que este hilo muera.

`void run()`
Si este hilo se construyó utilizando un objeto `Runnable` de ejecución independiente, entonces el método `run` de ese objeto es invocado; en otro caso, este método no hace nada y vuelve.

`static void sleep(long millis)`
Hace que el hilo actualmente en ejecución pase a dormir temporalmente durante el número de milisegundos especificado.

`void start()`
Hace que este hilo comience la ejecución; la Máquina Virtual de Java llama al método `run` de este hilo.

`String toString()`
Devuelve una representación en formato cadena de este hilo, incluyendo el nombre del hilo, la prioridad, y el grupo de hilos.

`static void yield()`
Hace que el hilo actual de ejecución, pare temporalmente y permita que otros hilos se ejecuten (útil en sistemas con planificación de hilos no preventiva).
Se aconseja consultar las referencia citada anteriormente para consultar la descripción de métodos cuyo uso directo es inherentemente inseguro, como son `stop`, `suspend` y `resume`.