**Difference between sleep() and wait()?**

**Sleep()** is a method which is used to hold the process for few seconds or the time you wanted but in case of **wait()** method thread goes in waiting state and it won’t come back automatically until we call the notify() or notifyAll().

The ***major difference*** is that wait() releases the lock or monitor while sleep() doesn’t releases any lock or monitor while waiting. Wait is used for inter-thread communication while sleep is used to introduce pause on execution, generally.

Thread.sleep() sends the current thread into the “Not Runnable” state for some amount of time. The thread keeps the monitors it has acquired — i.e. if the thread is currently in a synchronized block or method no other thread can enter this block or method. If another thread calls t.interrupt() it will wake up the sleeping thread. Note that sleep is a static method, which means that it always affects the current thread (the one that is executing the sleep method). A common mistake is to call t.sleep() where t is a different thread; even then, it is the current thread that will sleep, not the t thread.

object.wait() sends the current thread into the “Not Runnable” state, like sleep(), but with a twist. Wait is called on an object, not a thread; we call this object the “lock object.” Before lock.wait() is called, the current thread must synchronize on the lock object; wait() then releases this lock, and adds the thread to the “wait list” associated with the lock. Later, another thread can synchronize on the same lock object and call lock.notify(). This wakes up the original, waiting thread. Basically, wait()/notify() is like sleep()/interrupt(), only the active thread does not need a direct pointer to the sleeping thread, but only to the shared lock object.

synchronized(LOCK) {

Thread.sleep(1000); // LOCK is held

}

synchronized(LOCK) {

LOCK.wait(); // LOCK is not held

}

Let categorize all above points :

**Call on:**

* wait(): Call on an object; current thread must synchronize on the lock object.
* sleep(): Call on a Thread; always currently executing thread.

**Synchronized:**

* wait(): when synchronized multiple threads access same Object one by one.
* sleep(): when synchronized multiple threads wait for sleep over of sleeping thread.

**Hold lock:**

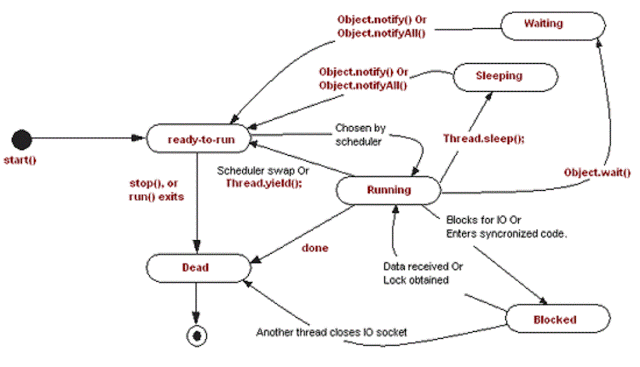
* wait(): release the lock for other objects to have chance to execute.
* sleep(): keep lock for at least t times if timeout specified or somebody interrupt.

**Wake-up condition:**

* wait(): until call notify(), notifyAll() from object
* sleep(): until at least time expire or call interrupt().

**Usage:**

* sleep(): for time-synchronization and;
* wait(): for multi-thread-synchronization.



# [Difference between Wait and Sleep, Yield in Java](http://javarevisited.blogspot.in/2011/12/difference-between-wait-sleep-yield.html)

**The difference between wait and sleep** or the **difference between sleep and yield in Java** are one of the popular [core Java interview questions](http://javarevisited.blogspot.com/2015/10/133-java-interview-questions-answers-from-last-5-years.html) and asked on [multi-threading interviews](http://javarevisited.blogspot.com/2011/07/java-multi-threading-interview.html). Out of three methods which can be used to pause a thread in Java, sleep() and yield() methods are defined in thread class while wait() is defined in the Object class, which is another interview question. The key difference between wait() and sleep() is that former is used for [inter-thread communication](http://javarevisited.blogspot.com/2013/12/inter-thread-communication-in-java-wait-notify-example.html) while later is used to introduced to pause the current thread for a short duration. This difference is more obvious from the fact that, when a thread calls the wait() method, it releases the monitor or lock it was holding on that object, but when a thread calls the sleep() method, it never releases the monitor even if it is holding.   
  
  
Coming back to yield(), it's little different than wait() and sleep(), it just releases the CPU hold by Thread to give another thread an opportunity to run though it's not guaranteed who will get the CPU. It totally depends upon thread scheduler and it's even possible that the thread which calls the yield() method gets the CPU again. Hence, it's not reliable to depend upon yield() method, it's just on best effort basis.

## Wait vs Sleep vs Yield in Java

In this Java tutorial, we will learn *what is sleep in Java*, important points of sleep in java *and difference between Wait and sleep in Java*.

### Difference between Wait and Sleep in Java

[difference between sleep, wait and yield in Java example](http://javarevisited.blogspot.com/2011/09/spring-interview-questions-answers-j2ee.html)Main *difference between wait and sleep is that wait()* method **release the acquired monitor** when thread is waiting while Thread.sleep() method **keeps the lock** or monitor even if thread is waiting. Also, wait for method in Java should be called from synchronized method or block while there is no such requirement for sleep() method.   
  
Another difference is **Thread.sleep()** method is a static method and **applies on current thread**, while [wait()](http://javarevisited.blogspot.com/2015/07/how-to-use-wait-notify-and-notifyall-in.html) is an instance specific method and only got wake up if some other thread calls notify method on same object. also, in the case of sleep, sleeping thread immediately goes to Runnable state after waking up while in the case of wait, waiting for a thread first acquires the lock and then goes into Runnable state. So based upon your need, if you want to [pause a thread](http://java67.blogspot.sg/2015/06/how-to-pause-thread-in-java-using-sleep.html) for specified duration then use sleep() method and if you want to implement inter-thread communication use wait method.

Here is the list of difference between wait and sleep in Java :  
  
1) wait is called from synchronized context only while sleep can be called without synchronized block. see [Why to wait and notify needs to call from synchronized method](http://javarevisited.blogspot.com/2011/05/wait-notify-and-notifyall-in-java.html) for more detail.  
  
2) waiting thread can be awake by calling notify and notifyAll while sleeping thread can not be awakened by calling notify method.  
  
3) wait is normally done on condition, Thread wait until a condition is true while sleep is just to put your thread on sleep.

4) wait for release lock on an object while waiting while sleep doesn’t release lock while waiting.  
  
5) The wait() method  is called on an Object on which the synchronized block is locked, while sleep is called on the Thread. See [Core Java, Volume 1 9th Edition](http://www.amazon.com/Volume-II-Advanced-Features-Edition-Series/dp/013708160X?tag=javamysqlanta-20) by Cay S. Horstmann for more details on how to use wait() and notify method in Java.

## Difference between yield and sleep in Java

*The major difference between yield and sleep in Java* is that yield() method **pauses the currently executing** thread temporarily for giving a chance to the remaining waiting threads of the same priority to execute. If there is no waiting thread or all the waiting threads have a lower priority then the same thread will continue its execution. The yielded thread when it will get the chance for execution is decided by the thread scheduler whose behavior is vendor dependent. Yield method doesn’t guarantee  that current thread will pause or stop but it guarantee that CPU will be relinquished by current Thread as a result of a call to Thread.yield() method in java. See [Java Concurrency in Practice](http://www.amazon.com/dp/0321349601/?tag=javamysqlanta-20) for more details.

Sleep method in Java has two variants one which takes millisecond as sleeping time while other which takes both mill and nanosecond for sleeping duration.

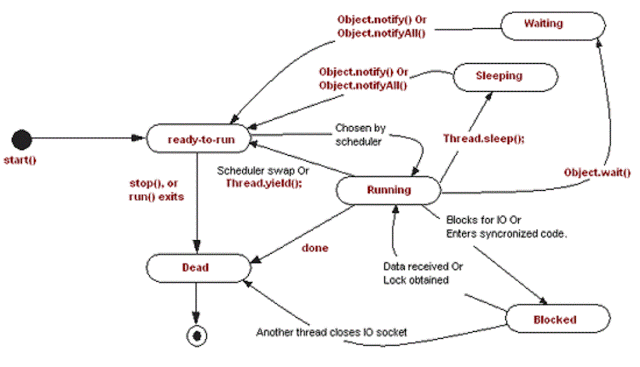
**sleep(long millis)**

or

**sleep(long millis,int nanos)**

Causes the **currently executing thread to sleep** for the specified number of milliseconds plus the specified number of nanoseconds.

Here is nice diagram which shows how thread transition happen to different thread states by calling the wait(), sleep() and yield() methods.

[](https://3.bp.blogspot.com/-SNJgCtjH9HI/VvTEb0u9RXI/AAAAAAAAFUg/Z_uW1Z8j8SEg3A1GigzIZY2UNh8_QH6-A/s1600/Sleep%2Bvs%2Bwait%2Bvs%2Byield%2Bmethod%2Bjava.gif)

### Example of Thread.sleep() method in Java

Here is sample code example of Sleep [Thread in Java](http://javarevisited.blogspot.com/2011/02/how-to-implement-thread-in-java.html). In this example, we have put Main thread in Sleep for 1 second.

/\*

 \* Example of Thread Sleep method in Java

 \*/

**public** **class** SleepTest {

**public** **static** **void** main(String... args){

              System.*out*.println(Thread.*currentThread*().getName() + " is going to sleep for 1 Second");

**try** {

                     Thread.*currentThread*().*sleep*(1000);

              } **catch** (InterruptedException e) {

                     // **TODO** Auto-generated catch block

                     e.printStackTrace();

              }

              System.*out*.println("Main Thread is woken now");

       }

}

**Output:**

**main is going to sleep for 1 Second**

**Main Thread is woken now**

### 10 points about Thread sleep() method in Java

I have listed down some important and worth to remember points about Sleep() method of Thread [Class in Java](http://javarevisited.blogspot.com/2011/10/class-in-java-programming-general.html):

1) Thread.sleep() method is used to **pause the execution, relinquish the CPU and return it to thread** scheduler.

2) Thread.The sleep() method is a [static method](http://javarevisited.blogspot.com/2011/11/static-keyword-method-variable-java.html) and always *puts the current thread to sleep*.

3) Java has two variants of sleep method in Thread class one with one argument which takes milliseconds as the duration of sleep and another method with two arguments one is millisecond and other is the nanosecond.

4) Unlike [wait() method in Java](http://javarevisited.blogspot.com/2011/05/wait-notify-and-notifyall-in-java.html), sleep() method of Thread class **doesn't relinquish the lock** it has acquired.

5) sleep() method **throws Interrupted Exception** if another thread interrupts a sleeping thread in java.

6) With sleep() in Java it's not guaranteed that when sleeping thread woke up it will definitely get CPU, instead it will go to Runnable state and fight for CPU with other thread.

7) There is a **misconception** about sleep method in Java that calling t.sleep() will put Thread "t" into sleeping state, that's not true because Thread.sleep method is a static method it always put the current thread into Sleeping state and not thread "t".

That’s all on **Sleep method in Java**. We have seen the **difference between sleep and wait** along with **sleep and yield in Java**. In Summary, just keep in mind that both sleep() and yield() operate on the current thread.

Read more: <http://javarevisited.blogspot.com/2011/12/difference-between-wait-sleep-yield.html#ixzz44saVLJ6l>