**wait**

Object wait methods has three variance, one which waits indefinitely for any other thread to call notify or notifyAll method on the object to wake up the current thread.

**notify**

notify method wakes up only one thread waiting on the object and that thread starts execution.

**notifyAll**

notifyAll method wakes up all the threads waiting on the object, although which one will process first depends on the OS implementation.





[*https://raw.githubusercontent.com/vsaravanan/java22/master/src/main/java/com/saravanjs/java22/console/multithreading/WaitNotifyTest.java*](https://raw.githubusercontent.com/vsaravanan/java22/master/src/main/java/com/saravanjs/java22/console/multithreading/WaitNotifyTest.java)

[Java Thread wait, notify and notifyAll Example | DigitalOcean](https://www.digitalocean.com/community/tutorials/java-thread-wait-notify-and-notifyall-example)

*class* Message {
 *private* String msg;

 *public* Message(String str){
 *this*.msg=str;
 }

 *public* String getMsg() {
 *return* msg;
 }

 *public void* setMsg(String str) {
 *this*.msg=str;
 }

}

*class* Waiter *implements Runnable*{

 *private* Message msg;

 *public* Waiter(Message m){
 *this*.msg=m;
 }

 @Override
 *public void* run() {
 String name = Thread.*currentThread*().getName();
 *synchronized* (msg) {
 *try*{
 System.*out*.println(name+" waiting to get notified at time:"+System.*currentTimeMillis*());
 msg.wait();
 }*catch*(InterruptedException e){
 e.printStackTrace();
 }
 System.*out*.println(name+" waiter thread got notified at time:"+System.*currentTimeMillis*());
 *//process the message now* System.*out*.println(name+" processed: "+msg.getMsg());
 }
 }

}

*class* Notifier *implements Runnable* {

 *private* Message msg;

 *public* Notifier(Message msg) {
 *this*.msg = msg;
 }

 @Override
 *public void* run() {
 String name = Thread.*currentThread*().getName();
 System.*out*.println(name+" started");
 *try* {
 Thread.*sleep*(1000);
 *synchronized* (msg) {
 msg.setMsg(name+" Notifier work done");
 msg.notify();
*// msg.notifyAll();* }
 } *catch* (InterruptedException e) {
 e.printStackTrace();
 }

 }

}

*public class* WaitNotifyTest {

 *public static void* main(String[] args) {
 Message msg = *new* Message("process it");
 Waiter waiter = *new* Waiter(msg);
 *new* Thread(waiter,"waiter").start();

 Waiter waiter1 = *new* Waiter(msg);
 *new* Thread(waiter1, "waiter1").start();

 Notifier notifier = *new* Notifier(msg);
 *new* Thread(notifier, "notifier").start();
 System.*out*.println("All the threads are started");
 }

}

for msg.notify();

All the threads are started

notifier started

waiter waiting to get notified at time:1718699314333

waiter1 waiting to get notified at time:1718699314342

waiter waiter thread got notified at time:1718699315335

waiter processed: notifier Notifier work done

for msg.notifyAll();

All the threads are started

notifier started

waiter waiting to get notified at time:1718699488848

waiter1 waiting to get notified at time:1718699488860

waiter waiter thread got notified at time:1718699489854

waiter processed: notifier Notifier work done

waiter1 waiter thread got notified at time:1718699489858

waiter1 processed: notifier Notifier work done