



SNS COLLEGE OF TECHNOLOGY

(An Autonomous Institution)

Approved by AICTE, New Delhi, Affiliated to Anna University, Chennai

Accredited by NAAC-UGC with 'A++' Grade (Cycle III) &

Accredited by NBA (B.E - CSE, EEE, ECE, Mech&B.Tech.IT)

COIMBATORE-641 035, TAMIL NADU



UNIT V

MULTITHREADING IN JAVA

Inter-thread Communication in Java

Inter-thread communication or **Co-operation** is all about allowing synchronized threads to communicate with each other.

Cooperation (Inter-thread communication) is a mechanism in which a thread is paused running in its critical section and another thread is allowed to enter (or lock) in the same critical section to be executed. It is implemented by following methods of **Object class**:

- wait()
- notify()
- notifyAll()

1) wait() method

The wait() method causes current thread to release the lock and wait until either another thread invokes the notify() method or the notifyAll() method for this object, or a specified amount of time has elapsed.

The current thread must own this object's monitor, so it must be called from the synchronized method only otherwise it will throw exception.

Method	Description
public final void wait()throws InterruptedException	It waits until object is notified.
public final void wait(long timeout)throws InterruptedException	It waits for the specified amount of time.

2) notify() method

The notify() method wakes up a single thread that is waiting on this object's monitor. If any threads are waiting on this object, one of them is chosen to be awakened. The choice is arbitrary and occurs at the discretion of the implementation.

Syntax:

```
public final void notify()
```

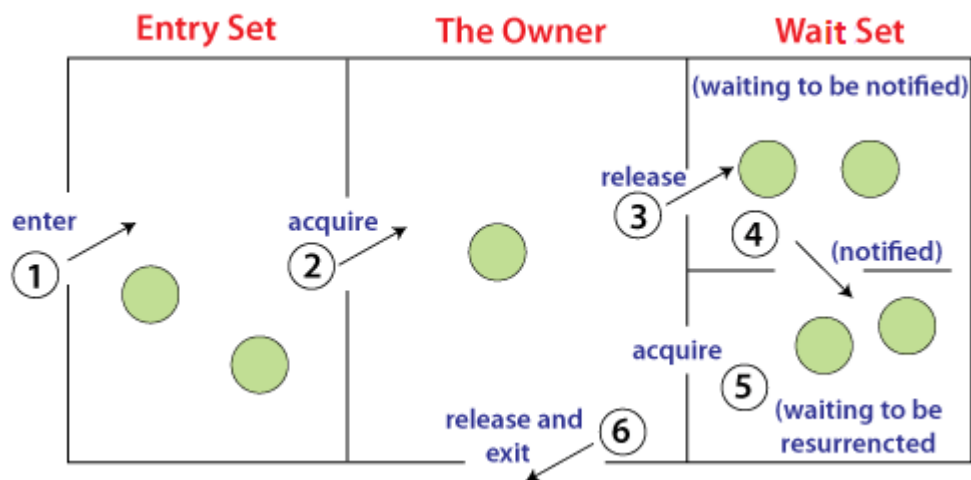
3) notifyAll() method

Wakes up all threads that are waiting on this object's monitor.

Syntax:

```
public final void notifyAll()
```

Understanding the process of inter-thread communication



The point to point explanation of the above diagram is as follows:

1. Threads enter to acquire lock.
2. Lock is acquired by one thread.
3. Now thread goes to waiting state if you call wait() method on the object. Otherwise it releases the lock and exits.
4. If you call notify() or notifyAll() method, thread moves to the notified state (runnable state).
5. Now thread is available to acquire lock.

6. After completion of the task, thread releases the lock and exits the monitor state of the object.

Why wait(), notify() and notifyAll() methods are defined in Object class not Thread class?

It is because they are related to lock and object has a lock.

Difference between wait and sleep?

wait()	sleep()
The wait() method releases the lock.	The sleep() method doesn't release the lock.
It is a method of Object class	It is a method of Thread class
It is the non-static method	It is the static method
It should be notified by notify() or notifyAll() methods	After the specified amount of time, sleep is completed.

Example of Inter Thread Communication in Java

Let's see the simple example of inter thread communication.

Test.java

1. **class** Customer{
2. **int** amount=10000;
- 3.
4. **synchronized void** withdraw(**int** amount){
5. System.out.println("going to withdraw...");
- 6.
7. **if**(**this**.amount<amount){
8. System.out.println("Less balance; waiting for deposit...");
9. **try**{ wait();}**catch**(Exception e){ }
10. }
11. **this**.amount-=amount;
12. System.out.println("withdraw completed...");
13. }
- 14.

```
15. synchronized void deposit(int amount){
16. System.out.println("going to deposit...");
17. this.amount+=amount;
18. System.out.println("deposit completed... ");
19. notify();
20. }
21. }
22.
23. class Test{
24. public static void main(String args[]){
25. final Customer c=new Customer();
26. new Thread(){
27. public void run(){c.withdraw(15000);}
28. }.start();
29. new Thread(){
30. public void run(){c.deposit(10000);}
31. }.start();
32.
33. }}
```

Output:

```
going to withdraw...
Less balance; waiting for deposit...
going to deposit...
deposit completed...
withdraw completed
```