Sri SAI RAM ENGINEERING COLLEGE
INSTITUTE OF TECHNOLOGY
West Tambaram, Chennai - 44

SAIRAM DIGITAL RESOURCES

YEAR II SEM III

CS8392

OBJECT ORIENTED PROGRAMMING
(Common to CSE, EEE, EIE, ICE, IT)

UNIT NO 4

MULTITHREADING AND GENERIC PROGRAMMING

Inter- Thread Communication

COMPUTER SCIENCE & ENGINEERING

Inter thread Communication

- Inter-thread communication is allows synchronized threads to communicate with each other.
- Cooperation or 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.
- Polling is usually implemented by a loop that is used to check some condition repeatedly. Once the condition is true, appropriate action is taken. This wastes CPU time.
- To avoid polling, Java includes an elegant interprocess communication mechanism via the wait( ), notify( ), and notifyAll( ) methods of Object class.

Thread methods

Method	Syntax	Description
wait()	final void wait( ) throws InterruptedException	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.
notify()	final void notify( )	wakes up a thread that called wait( ) on the same object
notifyAll()	final void notify All( )	wakes up all the threads that called wait( ) on the same object. One of the threads will be granted access

Example

```
class Customer {
   int amount=10000;
    synchronized void withdraw(int amount) {
        System.out.println("Amount to be withdrawn….");
        if(this.amount<amount)  {
           System.out.println("Less balance; waiting for deposit...");
           try  {        wait();              }
          catch(Exception e){}
        }
        this.amount-=amount;
        System.out.println("withdrawal completed...");
    }
   synchronized void deposit(int amount)  {
        System.out.println("Deposit Amount...");
        this.amount+=amount;
        System.out.println("Amount deposit completed... ");
        notify();
   }
}
class withdrawthread extends Thread {
         Customer C;
         withdrawthread(Customer C)  {
              this.C = C;
         }
         public void run()   {  C.withdraw(15000);  }
}
```

```
class depositthread extends Thread {
         Customer C;
         depositthread(Customer C)  { this.C = C;}
         public void run()      { C.deposit(10000);  }
}
class Test {
    public static void main(String args[]) {
        Customer c=new Customer();
         withdrawthread wt = new withdrawthread(c);
         wt.start();
         depositthread dt = new depositthread(c);
         dt.start();
 }
}
```

Output

Amount to be withdrawn…

Less balance; waiting for deposit...

Deposit Amount..

Amount deposit completed...

withdrawal completed