**Synchronization**

At times when more than one thread try to access a shared resource, we need to ensure that resource will be used by only one thread at a time. The process by which this is achieved is called **synchronization**. The synchronization keyword in java creates a block of code referred to as critical section.

Every Java object with a critical section of code gets a lock associated with the object. To enter critical section a thread need to obtain the corresponding object's lock.

**General Syntax :**

synchronized (object)

{

//statement to be synchronized

}

**Example**

***class*** *First*

*{*

***public******void*** *display(String msg)*

*{*

*System.****out****.print ("["+msg);*

***try***

*{*

*Thread.sleep(1000);*

*}*

***catch****(InterruptedException e)*

*{*

*e.printStackTrace();*

*}*

*System.****out****.println ("]");*

*}*

*}*

***class*** *Second* ***extends*** *Thread*

*{*

*String msg;*

*First fobj;*

*Second (First fp,String str)*

*{*

*fobj = fp;*

*msg = str;*

*start();*

*}*

***public******void*** *run()*

*{*

*fobj.display(msg);*

*}*

*}*

***public******class*** *Syncro*

*{*

***public******static******void*** *main (String[] args)*

*{*

*First fnew =* ***new*** *First();*

*Second ss =* ***new*** *Second(fnew, "Welcome");*

*Second ss1=* ***new*** *Second (fnew,"TO");*

*Second ss2 =* ***new*** *Second(fnew, "Netparam");*

*}*

*}*

***OutPut***

*[welcome[Netparam[TO]*

*]*

*]*