**Multithreading – Part-07 – Synchronization part-01**

* **Introduction:**

Synchronized is a modifier applicable only for methods and blocks but not for classes and variables.

If multiple threads are trying to operate simultaneously on the same Java object then there may be a chance of data inconsistency problem. To overcome this problem, we should go for “synchronized” keyword.

If a method or block declared as synchronized then at a time only one thread is allowed to execute. That method or block on the given object. So that data inconsistency problem will be resolved.

The main advantage of synchronized keyword is we can resolve data inconsistency problems, but the main disadvantage of synchronized keyword is it increases the waiting time of threads and creates performance problems. Hence if there is no specific requirement then it is not recommended to use synchronized keyword.

Recall the dog having fight with briyani.

* **Lock**

Internally synchronization concept is implemented by using lock.

Every object in Java has a unique lock.

Whenever we are using synchronized keyword then only lock concept will come into the picture.

If a thread wants to execute synchronized method, on the given object first it has to get lock of that object.

Once thread got the lock then it is allowed to execute, any synchronized method on that object.

Once method execution completes automatically thread releases lock.

Acquiring and releasing lock internally taken care by JVM. And programmer not responsible for this activity.

While a thread executing synchronized method on the given object the remaining threads are not allowed to execute any synchronized method simultaneously on the same object. But remaining threads are allowed to execute non-synchronized methods simultaneously.

Example:

class X{

synchronized m1();

synchronized m2();

void m3();

Object X 🡪 t1 got a lock and it is executing the method m1()

Meanwhile t2 and t3 trying to access the m1 or m2, it has to wait until t1 completes the work and releases the lock.

But other threads can access the m3() directly as it is not a synchronized method.

Note: Lock concept is implemented based on Object not on method level.

* **Synchronized and non-synchronized area in an Object:**

Non-Synch Area:

This area can be accessed by any number of threads simultaneously.

Synch Area:

This area can be accessed by only one thread at a time.

Example:

class X{

synchronized Area{

Wherever we are performing update operations [add/remove/delete/replace] that is where state of object changing.

}

Non Synchronized Area{

Wherever object state won’t be changed, like read() operation.

}

}

Example:

class ReservationSystem{

non-synchrinized checAvailability(){

// Just read operation

}

synchronized bookTicket(){

// Update operation

}

}

* **Synchronization Example:**

class Display{

public synchronized void wish(String name){

for(int i = 0; I < 10; i++){

System.out.print(“Good Morning”);

try{

Thread.sleep();

} catch(InterruptedException ie){

}

System.out.println(name);

}

}

}

class MyThread extends Thread{

Display d;

String name;

MyThread(Display d, String name){

this.d = d;

this.name = name;

}

public void run(){

d.wish(name);

}

}

class SynchronizedDemo{

public static void main(String[] args){

Display d = new Display();

MyThread t1 = new MyThread(d, “Dhoni”);

MyThread t2 = new MyThread(d, “Yuvraj”);

t1.start();

t2.start();

}

}

If we are not declaring wish method as synchronized then both threads will be executed simultaneously and hence we will get irregular output.

Example\_Non-Synchronized:

Good morning : Good morning: Yuvraj

Good morning: Dhoni

Good morning: Yuvraj

If we declare wish method as synchronized then at a time only one thread is allowed to execute wish method on the given display object. Hence, we will get regular output.

Example\_Synchronized:

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Dhoni

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj

Good morning: Yuvraj