Synchronized Block in Java

Synchronized block can be used to perform synchronization on any specific resource of the method.

Suppose we have 50 lines of code in our method, but we want to synchronize only 5 lines, in such cases, we can use synchronized block.

If we put all the codes of the method in the synchronized block, it will work same as the synchronized method.

Points to Remember

- Synchronized block is used to lock an object for any shared resource.
- Scope of synchronized block is smaller than the method.
- A Java synchronized block doesn't allow more than one JVM, to provide access control to a shared resource.
- The system performance may degrade because of the slower working of synchronized keyword.
- Java synchronized block is more efficient than Java synchronized method.

Syntax

```
synchronized (object reference expression) {
  //code block
}
```

Example of Synchronized Block

Let's see the simple example of synchronized block.

TestSynchronizedBlock1.java

```
class Table
{
 void printTable(int n){
  synchronized(this){//synchronized block
                    :i++){
                    .tln(n*i);
```

⇧ SCROLL TO TOP

```
      try{
       Thread.sleep(400);
      }catch(Exception e){System.out.println(e);}
     }
   }
 }//end of the method
}

class MyThread1 extends Thread{
Table t;
MyThread1(Table t){
this.t=t;
}
public void run(){
t.printTable(5);
}

}
class MyThread2 extends Thread{
Table t;
MyThread2(Table t){
this.t=t;
}
public void run(){
t.printTable(100);
}
}

public class TestSynchronizedBlock1{
public static void main(String args[]){
Table obj = new Table();//only one object
MyThread1 t1=new MyThread1(obj);
MyThread2 t2=new MyThread2(obj);
t1.start();
t2.start();
}
}
```

Output:

```
5
10
```

⇧ SCROLL TO TOP

```
25
100
200
300
400
500
```

Synchronized Block Example Using Anonymous Class

TestSynchronizedBlock2.java

```
// A Sender class
class Sender
{
  public void SenderMsg(String msg)
  {
    System.out.println("\nSending a Message: "  + msg);
    try
    {
      Thread.sleep(800);
    }
    catch (Exception e)
    {
      System.out.println("Thread interrupted.");
    }
    System.out.println("\n" +msg+ "Sent");
  }
}
// A Sender class for sending a message using Threads
class SenderWThreads extends Thread
{
  private String msg;
  Sender sd;

  // Receiver method to receive a message object and a message to be sent
  SenderWThreads(String m, Sender obj)
  {
    msg = m;
    sd = obj;
  }

  public void run()
  {
```

⇧ SCROLL TO TOP ly one thread sends a message at a time.

```
    synchronized(sd)
    {
      // synchronizing the sender object
      sd.SenderMsg(msg);
    }
  }
}
// Driver Code
public class ShynchronizedMultithreading
{
  public static void main(String args[])
  {
    Sender sender = new Sender();
    SenderWThreads sender1 = new SenderWThreads( "Hola " , sender);
SenderWThreads sender2 =  new SenderWThreads( "Welcome to Javatpoint website ", sender);

    // Start two threads of SenderWThreads type
    sender1.start();
    sender2.start();

    // wait for threads to end
    try
    {
      sender1.join();
      sender2.join();
    }
    catch(Exception e)
    {
      System.out.println("Interrupted");
    }
  }
}
```

Output:

```
Sending a Message: Hola
Hola Sent
Sending a Message: Welcome to Javatpoint website
Welcome to Javatpoint website Sent
```

⇧ SCROLL TO TOP

Next →