**Multithreading in Java**

Multithreading in java is a process of executing multiple threads simultaneously. The aim of multithreading is to achieve the concurrent execution.

**Thread**

* Thread is a lightweight components and it is a flow of control.
* In other words a flow of control is known as thread.

**State or Life cycle of thread**

State of a thread are classified into five types they are

1. New State
2. Ready State
3. Running State
4. Waiting State
5. Halted or dead State

## Lifecyle of Thread Class

## New State

* If any new thread class is created that represent **new state of a thread,**
* In new state thread is created and about to enter into main memory.
* No memory is available if the thread is in new state.

## Ready State

* In ready state thread will be entered into main memory,
* memory space is allocated for the thread and 1st time waiting for the CPU.

## Running State

Whenever the thread is under execution ,it is known as running state.

## Halted or dead State

* If the thread execution is stopped permanently then it comes under dead state,
* no memory is available for the thread if it comes to dead state.

## Achieve multithreading in java

In java language multithreading can be achieve in two different ways.

1. Using thread class
2. Using Runnable interface

In java language multithreading program can be created by following below rules.

1. Create any user defined class and make that one as a derived class of thread class.

**class** Class\_Name **extends** Thread

{

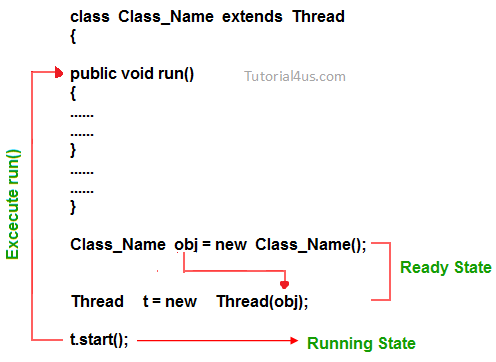
........

}

1. Override run() method of Thread class (It contains the logic of perform any operation)
2. Create an object for user-defined thread class and attached that object to predefined thread class object.

Class\_Name obj=new Class\_Name Thread t=new Thread(obj);

1. Call start() method of thread class to execute run() method.
2. Save the program with filename.java



import java.util.\*;

public class Multithreading

{

public static void main(String[] args)

{

A obj=new A();

obj.start();

B obj1=new B();

obj1.start();

}

}

class A extends Thread

{

public void show()

{

for(int i=1;i<=5;i++)

{

try {

System.out.println("Hi");

Thread.sleep(1000);

}

catch (InterruptedException ex)

{

Logger.getLogger(Multithreading.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

public void run()

{

show();

}

}

class B extends Thread

{

public void show()

{

for(int i=1;i<=5;i++)

{

try {

System.out.println("Hello");

Thread.sleep(1000);

}

catch (InterruptedException ex)

{

Logger.getLogger(Multithreading.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

public void run()

{

show();

}

}

Output:

Hi

Hello

Hi

Hello

Hi

Hello

Hi

Hello

Hi

Hello

Logic:

The thread has two methods used thread.sleep(),and thread.run() to execute the thread.