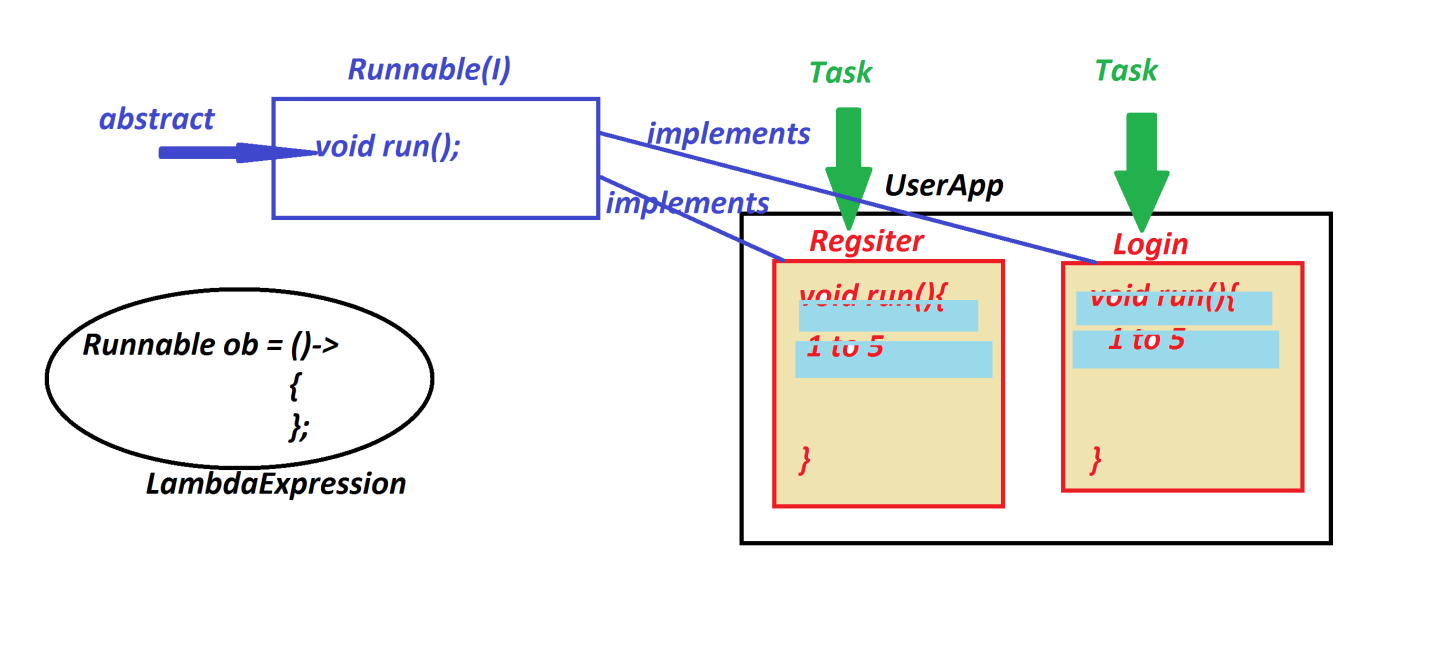
***DT : 19/11/2022***

******

***Ex-application:***

***Register.java***

***package test;***

***public class Register implements Runnable{***

***@Override***

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

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

***System.out.println("Registration for "+Thread.currentThread().getName());***

***try {***

***Thread.sleep(2000);***

***}catch(Exception e) {e.printStackTrace();}***

***}//end of loop***

***}***

***}***

***Login.java***

***package test;***

***public class Login implements Runnable{***

***@Override***

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

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

***System.out.println("Login for "+Thread.currentThread().getName());***

***try {***

***Thread.sleep(2000);***

***}catch(Exception e) {e.printStackTrace();}***

***}//end of loop***

***}***

***}***

***DemoThread1.java(MainClass)***

***package maccess;***

***import test.\*;***

***public class DemoThread1 {***

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

***Register ob1 = new Register();***

***Login ob2 = new Login();***

***Thread t1 = new Thread(ob1);***

***Thread t2 = new Thread(ob2);***

***t1.setName("User-1");***

***t2.setName("User-2");***

***t1.setPriority(Thread.MAX\_PRIORITY-7);//3***

***t2.setPriority(Thread.NORM\_PRIORITY-1);//6***

***t1.start();***

***t2.start();***

***}***

***}***

***o/p:***

***Registration for User-1***

***Login for User-2***

***Registration for User-1***

***Login for User-2***

***Registration for User-1***

***Login for User-2***

***Registration for User-1***

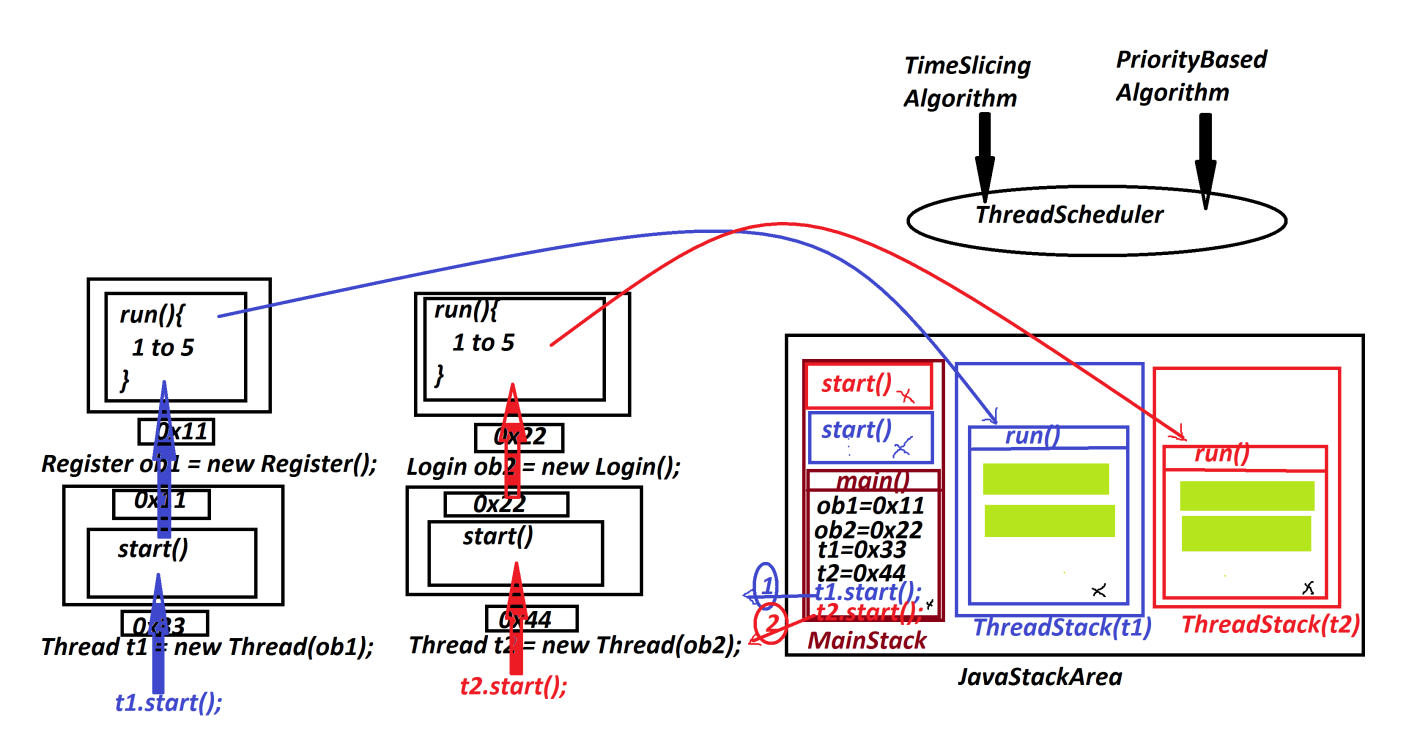
***Login for User-2***

***Registration for User-1***

***Login for User-2***

***==================================================================***

***Execution flow of above program:***

******

***==============================================================***

***Note:***

***=>In the process of executing multiple threads,Multiple-Thread stacks are created***

***and these Multiple-Thread stacks are executed Simultaneously.***

***define Thread-Scheduler?***

***=>Thread-Scheduler is a pre-defined algorithm to control Multiple threads for***

***execution.***

***=>This Thread-Scheduler will use the following algorithms:***

***(i)Time-Slicing Algorithm***

***(ii)Priority based Algorithm***

***(i)Time-Slicing Algorithm:***

***=>In Time-Slicing Algorithm all multiple threads are executed based on defined***

***time-slice.***

***(ii)Priority based Algorithm:***

***=>In priority based algorithm the threads are executed based on***

***thread-priorities.***

***Note:***

***=>Thread priorities from 1 to 10***

***1 - Least Priority***

***5 - Normal Priority***

***10 - Max Priority***

***=>The following fields of "java.lang.Thread" class represent priorities:***

***public static final int MIN\_PRIORITY;***

***public static final int NORM\_PRIORITY;***

***public static final int MAX\_PRIORITY;***

***==========================================================================***

***\*imp***

***Creating Threads using LambdaExpression:(Java8)***

***package maccess;***

***public class DemoThread2 {***

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

***new Thread(()->***

***{***

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

***System.out.println("Registration for "+Thread.currentThread().getName());***

***try {***

***Thread.sleep(2000);***

***}catch(Exception e) {e.printStackTrace();}***

***}//end of loop***

***}).start();***

***new Thread(()->***

***{***

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

***System.out.println("Login for "+Thread.currentThread().getName());***

***try {***

***Thread.sleep(2000);***

***}catch(Exception e) {e.printStackTrace();}***

***}//end of loop***

***}).start();***

***new Thread(()->***

***{***

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

***System.out.println("View products "+Thread.currentThread().getName());***

***try {***

***Thread.sleep(2000);***

***}catch(Exception e) {e.printStackTrace();}***

***}//end of loop***

***}).start();***

***}***

***}***

***============================================================================***