Lab 9 Thread

Ex 1.1

ThreadClassTest0.java

public class ExtendsThreadClassTest0 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args)

{

// Create object instance of a class that is subclass of Thread class

System.out.println("Creating PrintNameThread object instance..");

PrintNameThread pnt1 = new PrintNameThread("A");

// Start the thread by invoking start() method

System.out.println("Calling start() method of " + pnt1.getName() + " thread");

pnt1.start();

System.out.println("Creating PrintNameThread object instance..");

PrintNameThread pnt2 =

new PrintNameThread("B");

System.out.println("Calling start() method of " + pnt2.getName() + " thread");

pnt2.start();

System.out.println("Creating PrintNameThread object instance..");

PrintNameThread pnt3 =

new PrintNameThread("C");

System.out.println("Calling start() method of " + pnt3.getName() + " thread");

pnt3.start();

PrintNameThread pnt4 = new PrintNameThread("MyOwn");

System.out.println("Calling start() method of " + pnt4.getName() + " thread");

pnt4.start();

}

}

PrintNameThread.java

public class PrintNameThread extends Thread

{

public PrintNameThread(String name)

{

super(name);

}

public void run()

{

System.out.println("run() method of the " + this.getName() + " thread is called");

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

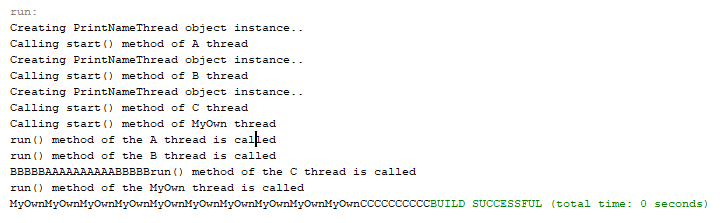
{

System.out.print(this.getName());

}

}

}



Ex1.2

ExtendThreadClassTest.java

public class ExtendThreadClassTest2 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

PrintNameThread pnt1 =

new PrintNameThread("A");

PrintNameThread pnt2 =

new PrintNameThread("B");

PrintNameThread pnt3 =

new PrintNameThread("C");

PrintNameThread pnt4 =

new PrintNameThread("MyOwn");

}

}

PrintNameThread.java





Ex 2.1

RunnableThreadTest1.java

public class RunnableThreadTest1 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

PrintNameRunnable pnt1 = new PrintNameRunnable("A");

Thread t1 = new Thread(pnt1);

t1.start();

PrintNameRunnable pnt2 = new PrintNameRunnable("B");

Thread t2 = new Thread(pnt2);

t2.start();

PrintNameRunnable pnt3 = new PrintNameRunnable("C");

Thread t3 = new Thread(pnt3);

t3.start();

MyOwnRunnableClass mrc = new MyOwnRunnableClass("MyOwn1");

Thread t4 = new Thread(mrc);

t4.start();

MyOwnRunnableClass mrc2 = new MyOwnRunnableClass("MyOwn2");

Thread t5 = new Thread(mrc2);

t5.start();

}

}

PrintNameRunnable.java

public class PrintNameRunnable implements Runnable {

String name;

PrintNameRunnable(String name) {

this.name = name;

}

// Implementation of the run() defined in the

// Runnable interface.

public void run() {

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

System.out.print(name);

}

}

}

MyOwnRunnableThread.java

public class MyOwnRunnableClass implements Runnable

{

private String name;

public MyOwnRunnableClass(String name)

{

this.name = name;

}

public void run()

{

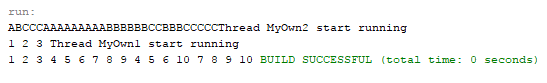
System.out.println("Thread " + this.name + " start running");

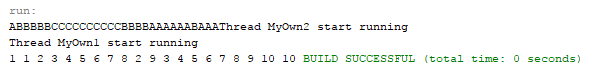
for(int i=1;i<=10;i++)

System.out.print(i+ " ");

}

}





Ex 3.1

ThreadGroupTest.java

public class ThreadGroupTest {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// Start three threads first. They should belong

// to a same ThreadsGroup.

new SimpleThread("Boston").start();

new SimpleThread("New York").start();

new SimpleThread("Seoul").start();

new SimpleThread("Bankok").start();

// Get ThreadGroup of the current thread and display

// the number of active threads that belong to the

// ThreadGroup.

ThreadGroup group = Thread.currentThread().getThreadGroup();

System.out.println("Number of active threads in this thread group = "

+ group.activeCount());

// Display the names of the threads in the current

// ThreadGroup.

Thread[] tarray = new Thread[10];

int actualSize = group.enumerate(tarray);

for (int i=0; i<actualSize;i++){

System.out.println("Thread " + tarray[i].getName()

+ " in thread group " + group.getName());

}

}

}

SimpleThread.java

public class SimpleThread extends Thread{

public SimpleThread(String str) {

super(str);

}

public void run() {

for (int i = 0; i < 5; i++) {

// System.out.format("%d %s%n", i, getName());

try {

sleep((long)(Math.random() \* 1000));

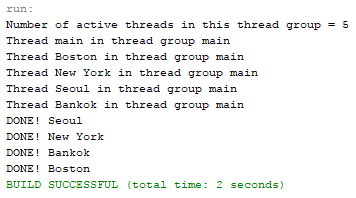
} catch (InterruptedException e) {}

}

System.out.format("DONE! %s%n", this.getName());

}

}



Ex3.2

DisplayAllThread.java

public class DisplayAllThreads {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

new SimpleThread("Boston").start();

new SimpleThread("New York").start();

new SimpleThread("Seoul").start();

new SimpleThread("Bangkok").start();

Thread[] tarray = findAllThreads();

for (int i=0; i<tarray.length;i++){

System.out.println("Thread " + tarray[i].getName()

+ " in thread group " + tarray[i].getThreadGroup().getName());

}

}

public static Thread[] findAllThreads() {

ThreadGroup group = Thread.currentThread().getThreadGroup();

ThreadGroup topGroup = group;

while (group != null) {

topGroup = group;

group = group.getParent();

}

int estimatedSize = topGroup.activeCount() \* 2;

Thread[] slackList = new Thread[estimatedSize];

int actualSize = topGroup.enumerate(slackList);

Thread[] list = new Thread[actualSize];

System.arraycopy(slackList, 0, list, 0, actualSize);

return list;

}

}

SimpleThread.java

public class SimpleThread extends Thread {

public SimpleThread(String str) {

super(str);

}

public void run() {

for (int i = 0; i < 5; i++) {

// System.out.format("%d %s%n", i, getName());

try {

sleep((long)(Math.random() \* 1000));

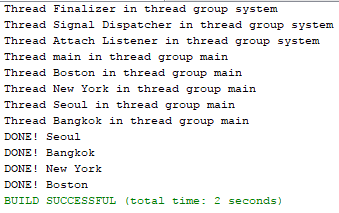
} catch (InterruptedException e) {}

}

System.out.format("DONE! %s%n", getName());

}

}



Ex 4

SynchronizedExample.java

public class SynchronizedExample {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

TwoStrings ts = new TwoStrings();

new PrintStringsThread("Hello ", "there.",ts);

new PrintStringsThread("How are ", "you?",ts);

new PrintStringsThread("Thank you ", "very much!",ts);

}

}

PrintStringThread.java

public class PrintStringsThread implements Runnable

{

Thread thread;

String str1, str2;

TwoStrings ts;

public PrintStringsThread(String str1, String str2,TwoStrings ts) {

this.str1 = str1;

this.str2 = str2;

this.ts = ts;

thread = new Thread(this);

thread.start();

}

public void run() {

synchronized(ts)

{

ts.print(str1,str2);

}

}

}

TwoStrings.java

public class TwoStrings {

public static void print(String str1, String str2) {

System.out.print(str1);

try {

Thread.sleep(500);

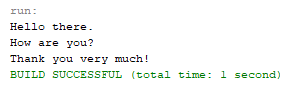
} catch (InterruptedException ie) {

}

System.out.println(str2);

}

}



Ex 5

ProducerConsumer.java

public class ProducerConsumer {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

CubbyHole c = new CubbyHole();

Producer p1 = new Producer(c, 1);

Consumer c1 = new Consumer(c, 1);

p1.start();

c1.start();

}

}

CubbyHole.java

public class CubbyHole {

private int contents;

private boolean available = false;

public synchronized int get(int who) {

while(available==false)

{

try{

wait();

}catch(InterruptedException e){}

}

available = false;

System.out.format("Consumer %d got: %d%n", who, contents);

notifyAll();

return contents;

}

public synchronized void put(int who,int value) {

while(available == true)

{

try{

wait();

}catch(InterruptedException e){}

}

contents = value;

available = true;

System.out.format("Producer %d put: %d%n", who, contents);

notifyAll();

}

}

Consumer.java

public class Consumer extends Thread {

private CubbyHole cubbyhole;

private int number;

public Consumer(CubbyHole c, int number) {

cubbyhole = c;

this.number = number;

}

public void run() {

int value = 0;

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

value = cubbyhole.get(number);

System.out.println("Consumer #" + this.number

+ " got: " + value);

}

}

}

Producer.java

public class Producer extends Thread {

private CubbyHole cubbyhole;

private int number;

public Producer(CubbyHole c, int number) {

cubbyhole = c;

this.number = number;

}

public void run() {

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

cubbyhole.put(number,i);

System.out.println("Producer #" + this.number

+ " put: " + i);

try {

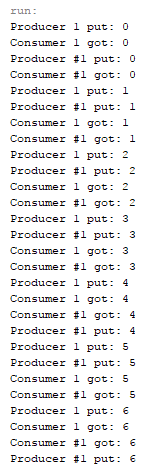
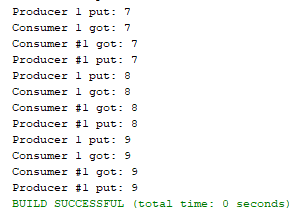
sleep((int)(Math.random() \* 100));

} catch (InterruptedException e) { }

}

}

}

Ex 6

TimerReminder.java

import java.util.Timer;

import java.util.TimerTask;

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

/\*\*

\*

\* @author 5835512048

\*/

public class TimerReminder {

/\*\*

\* @param args the command line arguments

\*/

Timer timer;

public TimerReminder(int seconds) {

timer = new Timer();

timer.schedule(new RemindTask(), seconds\*1000);

}

class RemindTask extends TimerTask {

public void run() {

System.out.println("Time's up!");

timer.cancel(); //Terminate the timer thread

}

}

public static void main(String args[]) {

System.out.println("About to schedule Reminder task in 5 seconds");

new TimerReminder(5);

System.out.println("Task scheduled.");

}

}