Ministerul Educaţiei, al Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Departamentul Informatică și Ingineria Sistemelor

**RAPORT**

Lucrarea de laborator nr.2

ASO

A efectuat:

st. gr. C-171 D. Melniciuc

A verificat:

dr., conf.univ. L.Rotaru

Chişinău 2020

Varianta:4

4. Un timer pentru a determina reapariţia procesului la fiecare minut.

Listingul programului:

// 4. Un timer pentru a determina reapariţia procesului la fiecare minut.

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

import java.util.Random;

import java.util.Scanner;

import java.time.Duration;

import java.time.Instant;

import java.util.Timer;

import java.util.TimerTask;

import java.io.FileReader;

import java.io.IOException;

import java.io.BufferedReader;

import java.nio.charset.StandardCharsets;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

class Counter {

long start;

boolean b = false;

int count = 0;

public synchronized int check (String str) {

if (b == true) {

// System.out.println("\tsleep");

try { Thread.sleep(7000); }

catch (InterruptedException e) { e.printStackTrace(); }

System.exit(1);

}

if (str == "th")

count++;

if (str == "time") {

b = true;

return count;

}

return count;

}

public int get (String str, int i) {

int cnt = i;

if (str == "put")

count++;

if (str == "get") {

return cnt;

}

return cnt;

}

}

class Task extends TimerTask {

public void run() {

System.out.println("\*in progress\*");

}

public void get(int i) {

try { Thread.sleep(i-1000); }

catch (InterruptedException exc) { }

}

}

public class lab2GUI\_2 extends JFrame {

static boolean area = false;

public lab2GUI\_2 () {

JFrame frame = new JFrame("labu 2");

frame.setPreferredSize(new Dimension(400,300));

JButton exit = new JButton("exit");

exit.setSize(new Dimension(100,30));

exit.setLocation(150,110);

exit.setForeground(Color.black);

exit.setFocusPainted(true);

exit.setContentAreaFilled(true);

final JButton start = new JButton("start");

start.setSize(new Dimension(100,30));

start.setLocation(150,60);

start.setForeground(Color.black);

final JLabel time = new JLabel(" -\_-");

time.setSize(new Dimension(190,60));

time.setLocation(155,150);

time.setForeground(Color.blue);

final JLabel time2 = new JLabel("");

time2.setSize(new Dimension(190,60));

time2.setLocation(155,180);

time2.setForeground(Color.red);

start.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

System.out.println("ca cit timp verificam? (ms)");

Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

System.out.println("x " + x);

Counter c = new Counter();

Thread t1 = new Thread(new Runnable() {

public void run() {

int sum = 0;

int count = 0;

while (true) {

Random rd = new Random();

for (int i = 0; i < 50; i++) {

i = rd.nextInt(100);

sum = sum + i;

System.out.println("nr " + i);

count++;

}

c.check("th");

c.get("put", count);

String str1 = Integer.toString(sum);

if (area == false) {

time.setText("suma " + "kebab");

}

else

time.setText("suma " + str1);

}

}

});

Task task = new Task();

Timer timer = new Timer();

Thread t2 = new Thread(new Runnable() {

public void run() {

int cnt = 0;

System.out.println("x2 " + x);

timer.schedule(task, x);

task.get(x);

timer.cancel();

System.out.println("End of task.");

cnt = c.check("time");

String str2 = Integer.toString(cnt);

time2.setText(cnt + " adunari");

// System.out.println( c.get("get", 0) + " adunari");

}

});

t1.start();

t2.start();

}

});

exit.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

System.exit(1);

}

});

frame.setLocation(500,250);

frame.setLayout(null);

frame.setResizable(false) ;

frame.setAlwaysOnTop(true);

frame.add(start);

frame.add(exit);

frame.add(time);

frame.add(time2);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.pack();

frame.setVisible(true);

}

public static void main(String[] args) throws NoSuchAlgorithmException {

try { new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor(); }

catch (Exception e1) { System.out.println(e1); }

String pass = "";

BufferedReader objReader = null;

try {

String strCurrentLine;

objReader = new BufferedReader(new FileReader("pass2"));

while ((strCurrentLine = objReader.readLine()) != null) {

pass += strCurrentLine;

}

} catch (IOException e) { e.printStackTrace(); } finally {

try {

if (objReader != null)

objReader.close();

} catch (IOException ex) { ex.printStackTrace(); }

}

MessageDigest md = MessageDigest.getInstance("MD5");

byte[] hashInBytes = md.digest(pass.getBytes(StandardCharsets.UTF\_8));

StringBuilder sb = new StringBuilder();

for (byte b : hashInBytes) {

sb.append(String.format("%02x", b));

}

String pass2 = sb.toString();

String pass2check = "1b12646d90503358dbb5f75a496febd5";

if ( pass2.equals(pass2check) ) {

area = true;

System.out.println("\t[+] Authorized access");

try { Thread.sleep(700); }

catch (InterruptedException e) { e.printStackTrace(); }

} else {

System.out.println("\nPassword is not: " + pass2 + " or password file not found");

System.out.println("Unauthorized copy\nPlease contact Dima with a kebab");

System.out.println("Press Any Key To Continue...");

new java.util.Scanner(System.in).nextLine();

System.exit(1);

}

try { new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor(); }

catch (Exception e1) { System.out.println(e1); }

lab2GUI\_2 frame = new lab2GUI\_2 ();

}

}

Output:

