**Ministerul Educaţiei, Culturii și Cercetării al Republicii Moldova**

**Universitatea Tehnică a Moldovei**

**Departamentul Informatică și Ingineria Sistemelor**

**Disciplina: Sisteme de Operare**

**RAPORT**

**Lucrarea de laborator nr. 1**

Au efectuat:

st. gr. CR-211 L. Mihail  
M.Victor

P.Tudor

A verificat:

Rotaru Lilia

Chişinău – 2024

* 1. **Probleme propuse spre realizare:**

Creaţi o aplicaţie cu unu sau mai multe taimere utilizînd clasele Timer și TimerTask în diferite moduri:

**1)Să reacționeze la un anumit interval de timp.**

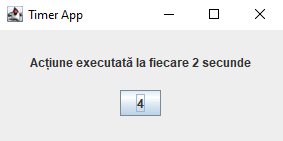
**2)Să reacționeze la un anumit timp.**

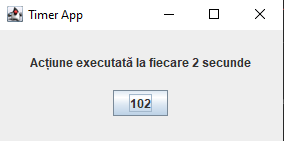
**3)Să reacționeze cu o perioadă indicată.**

Utilizați clasă **anonimă**.

**3)Să reacționeze cu o perioadă indicată**

import java.awt.\*;  
import java.util.Timer;  
import java.util.TimerTask;  
import javax.swing.\*;  
  
public class Main {  
 public static void main(String[] args) {  
 // Creating a Swing frame  
 JFrame frame = new JFrame("Timer App");  
 frame.setSize(300, 150); // Reduced height to make it look better  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
  
 // Creating a panel to display messages and the button  
 JPanel panel = new JPanel(new GridBagLayout());  
 GridBagConstraints constraints = new GridBagConstraints();  
 constraints.gridx = 0;  
 constraints.gridy = 0;  
 constraints.insets = new Insets(10, 10, 10, 10); // Adding padding  
  
 JLabel label = new JLabel("Acțiune executată la fiecare 2 secunde");  
 panel.add(label, constraints);  
  
 constraints.gridy = 1;  
 JButton button = new JButton("Adaugă secunde");  
 panel.add(button, constraints);  
  
 // Centering the panel  
 frame.add(panel, BorderLayout.*CENTER*);  
  
 // Displaying the frame in the center of the screen  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
  
 // Creating a Timer object to update the button  
 Timer buttonTimer = new Timer();  
  
 // Scheduling the task to update the button every 2 seconds  
 buttonTimer.schedule(new TimerTask() {  
 int seconds = 2; // Number of seconds  
  
 @Override  
 public void run() {  
 button.setText("" + seconds + "");  
 seconds += 2;  
 }  
 }, 0, 2000); // Scheduling updates every 2 seconds  
  
 // Adding an event listener for the button  
 button.addActionListener(e -> {  
 String text = label.getText();  
 String[] parts = text.split(" ");  
 int currentSeconds = Integer.*parseInt*(parts[parts.length - 2]);  
 currentSeconds += 2;  
 label.setText("Acțiune executată la fiecare 2 secunde");  
 });  
  
 // Creating and starting a separate execution thread to keep the application active  
 Thread thread = new Thread(() -> {  
 while (true) {  
 try {  
 Thread.*sleep*(100); // Waiting for 100 milliseconds  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 });  
 thread.start();  
 }  
}





1. **Întrebări de verificare:**
2. Dați definiția unui Timer.
3. Pentru ce este folosită metoda schedule() și din ce clasă Java ea vine?
4. Enumerați pașii care trebuie urmați pentru crearea unui timer.
5. Care este diferența dintre metoda schedule() și scheduleAtFixedRate()?
6. Cînd se oprește executarea unui timer?
7. Ce metode se folosesc pentru oprirea forțată a unui timer?
   1. **Definiția unui Timer:**

*În* ***Java****,* ***Timer*** *este o clasă din pachetul java.util care permite programatorilor să programeze sarcini (****tasks****) pentru a fi executate în viitor la intervale specificate sau la o dată și oră specifică.*

* 1. **Pentru ce este folosită metoda schedule() și din ce clasă Java ea vine?**

*Metoda* ***schedule()*** *este folosită pentru programarea executării unei sarcini (****task****) la o anumită dată și oră specificată sau la un interval de timp specificat între executări. Această metodă este definită în clasa* ***java.util.Timer****.*

* 1. **Enumerați pașii care trebuie urmați pentru crearea unui timer:**

*Importarea clasei* ***java.util.Timer.***

*Crearea unei instanțe de* ***Timer****.*

*Programarea unei sarcini (task) folosind metodele* ***schedule()*** *sau* ***scheduleAtFixedRate()*** *ale obiectului* ***Timer****.*

*Implementarea logicii sarcinii care va fi executată la momentul specificat.*

* 1. **Care este diferența dintre metoda schedule() și scheduleAtFixedRate()?**

***schedule():*** *Această metodă programează execuția unei sarcini la un anumit moment în timp sau după un anumit interval. Dacă o sarcină este întârziată din cauza încărcării sistemului, sarcina următoare este planificată pentru a fi executată după un interval de timp de la finalizarea sarcinii precedente.*

***scheduleAtFixedRate():*** *Această metodă programează execuția unei sarcini la un anumit moment în timp și apoi repetă sarcina la intervale fixe. Nu se ține cont de durata execuției sarcinii. Sarcinile se vor executa la intervale fixe, indiferent de durata execuției sau încărcarea sistemului.*

* 1. **Cînd se oprește executarea unui timer?**

*Execuția unui* ***timer*** *se oprește atunci când:*

*Sarcina programată a fost anulată sau terminată.*

*Timerul este oprit explicit folosind metoda* ***cancel()*** *a obiectului* ***Timer****.*

*Programul care a creat timerul se încheie sau thread-ul care execută timerul este oprit.*

* 1. **Ce metode se folosesc pentru oprirea forțată a unui timer?**

*Pentru a opri un timer forțat, se folosește metoda* ***cancel()*** *a obiectului* ***Timer****. Această metodă oprește programarea oricăror sarcini viitoare și eliberează toate resursele asociate cu timerul.*