**CFGS - Desenvolupament d’aplicacions multiplataforma**

**Mòdul 9 – Programació de serveis i processos**

**UF2 – Processos i fils**

Per l’activitat heu de lliurar un PDF amb el següent contingut:

* Nom o noms dels membres que fan l’activitat (màxim 2 alumnes).
* URL del repositori on hi ha els fonts (github, bitbucket,...).
* Noms d’usuari utilitzats per cada membre del grup del repositori del núvol.
* Explicacions, línies de codi creades/modificades i bolcats de pantalla de l’execució dels diferents algoritmes que es demanen en la tasca.
* La claredat de la documentació lliurada (a banda del codi) serà directament proporcional a la nota obtinguda.

El pdf s’ha d’anomenar seguint el format NomCognom1Cognom2.pdf del membre que fa el lliurament. Si un grup està format per dos membres NOMÉS UN l’ha de lliurar.

Tasques entregades fora de termini i/o on els fonts del repositori del núvol hagin estat creats/modificats fora de termini tindran, com a màxim un 5 sempre que el lliurament i/o la modificació del repositori sigui de, com a molt, 3 dies després del venciment. Passats 3 dies no s’avaluarà la tasca i es qualificarà com a 0.

No s’acceptarà tampoc cap tasca que no es lliure en el format especificat i/o que no contingui de forma clara els punts que es demanen.

**Enllaços d'interès:**

[**https://docs.oracle.com/javase/7/docs/api/index.html?java/lang/Thread.html**](https://docs.oracle.com/javase/7/docs/api/index.html?java/lang/Thread.html)

**Objectius:**

Identificar i utilitzar les paraules reservades del llenguatge Java per gestió de processos/fils

**Activitat 6**

Tingues present que el metode repaint() de la classe JPanel crida al mètode paintComponent() que totes les classes que heretin de JPanel han d’implementar (no has de fer res, només tenir present això).

**NOMS**

Raúl Santos Espino

Tiffany Fernández Anguera

**URL**

<https://github.com/TiffanyFA/m9uf2/tree/master/Activitat4>

**NOMS USUARIS**

RaulSE98

TiffanyFA

**EVIDÈNCIES**

A partir del codi que es subministra:

a) Identifica classes(dibuixa diagrama uml amb classes, atributs i mètodes).

b) Identifica les classes que contenen codi que es pot executar mitjançant fils.

c) Executa el codi i afegeix en el projecte l’arxiu necessari per a que funcioni.

e) Canvia el nombre de naus que poden aparèixer en el joc. Prova diferents quantitats i mira d’esbrinar la quantitat màxima.

f) Amb tres naus, quants fils d’execució hi ha i que fa cada un d’ells?

g) Tenint en compte els fils i el que fan, quin mètode o mètodes o parts de codi haurien d’estar codificats mitjançant la paraula **synchronized**? Per que? Implementa-ho.

AMPLIACIÓ. Afegeix una nau pròpia a sota de tot i que pugis dirigir a dreta i esquerra.

package uf2.practica;

import java.awt.\*;

import java.util.\*;

import javax.swing.\*;

public class NauEspaial extends javax.swing.JFrame {

public NauEspaial() {

initComponents();

}

@SuppressWarnings("unchecked")

private void initComponents() {

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT\_ON\_CLOSE);

setBackground(new java.awt.Color(255, 255, 255));

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

getContentPane().setLayout(layout);

layout.setHorizontalGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 400, Short.MAX\_VALUE));

layout.setVerticalGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGap(0, 300, Short.MAX\_VALUE));

pack();

}

public static void main(String args[]) {

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

}

catch (Exception ex) {

java.util.logging.Logger.getLogger(NauEspaial.class.getName()).log(

java.util.logging.Level.SEVERE, null, ex);

}

NauEspaial f = new NauEspaial();

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

f.setTitle("Naus Espaials");

f.setContentPane(new PanelNau());

f.setSize(480, 560);

f.setVisible(true);

}

}

class PanelNau extends JPanel implements Runnable{

private int numNaus=3;

Nau[] nau;

public PanelNau(){

nau = new Nau[numNaus];

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

Random rand = new Random();

int velocitat=(rand.nextInt(3)+5)\*10;

int posX=rand.nextInt(100)+30;

int posY=rand.nextInt(100)+30;

int dX=rand.nextInt(3)+1;

int dY=rand.nextInt(3)+1;

nau[i]= new Nau(i,posX,posY,dX,dY,velocitat);

}

Thread n = new Thread(this);

n.start();

}

public void run() {

System.out.println("Inici fil repintar");

while(true) {

try { Thread.sleep(100);} catch(Exception e) {} // espero 0,1 segons

System.out.println("Repintant");

repaint();

}

}

public void paintComponent(Graphics g) {

super.paintComponent(g);

for(int i=0; i<nau.length;++i) nau[i].pinta(g);

}

}

class Nau extends Thread {

private int numero;

private int x,y;

private int dsx,dsy,v;

private int tx = 10;

private int ty = 10;

private String img = "/images/nau.jpg";

private Image image;

public Nau(int numero, int x, int y, int dsx, int dsy, int v ) {

this.numero = numero;

this.x=x;

this.y=y;

this.dsx=dsx;

this.dsy=dsy;

this.v=v;

image = new ImageIcon(Nau.class.getResource("nau.png")).getImage();

Thread t = new Thread(this);

t.start();

}

public int velocitat (){

return v;

}

public void moure (){

x=x + dsx;

y=y + dsy;

// si arriva als marges ...

if ( x>= 450 - tx || x<= tx) dsx = - dsx;

if ( y >= 500 - ty || y<=ty ) dsy = - dsy;

}

public void pinta (Graphics g) {

Graphics2D g2d = (Graphics2D)g;

g2d.drawImage(this.image, x, y, null);

}

public void run() {

while (true) {

System.out.println("Movent nau numero " + this.numero);

try { Thread.sleep(this.v); } catch (Exception e) {}

moure();

}

}

}