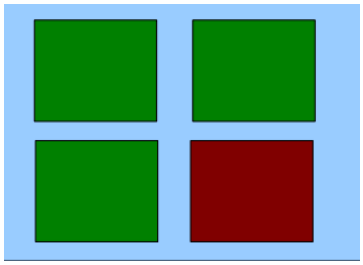
TP2

Le chenillard UDP

L'objectif de cet exercice est la réalisation d'un chenillard UDP. Sur votre PC, 4 fenêtres sont

ouvertes et sont positionnées ainsi :



Chaque fenêtre représente une instance de votre programme « chenillard UDP ». Votre programme positionne la couleur de la fenêtre à rouge, attend une seconde, puis passe la main au programme 2. La fenêtre 2 devient rouge pendant une seconde , puis passe la main au programme 3 ... La communication entre vos programmes doit se faire en UDP. Vous êtes libres dans la conception de votre programme, le format des données échangées, les paramètres en entrée du programme...

```java
import java.awt.Color;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.InetSocketAddress;
import java.net.Socket;

import javax.swing.JFrame;



public class ex2 {


    public static void main (String[] args) throws IOException, InterruptedException {

        int DEFAULT_PORT = 3000;
        int WINDOWS = Integer.parseInt(args[0]);
        int WINDOWSIZE = 300;

        int port = DEFAULT_PORT + WINDOWS;
        int nextPort = WINDOWS == 4 ? DEFAULT_PORT + 1 : port + 1;

        JFrame frame = new JFrame("Chenillard "+WINDOWS);
        frame.setSize(300,300);
        frame.setLocation(WINDOWS*WINDOWSIZE,WINDOWS);
        frame.getContentPane().setBackground(Color.GREEN);
        frame.setVisible(true);
        Thread.sleep(1000);

        DatagramSocket socket = new DatagramSocket(null);

        while(true) {

            // Reception

            socket.bind(new InetSocketAddress(port));

            byte[] bufR = new byte[2048];
            DatagramPacket dpR = new DatagramPacket(bufR, bufR.length);
            socket.receive(dpR);
            String message = new String(bufR, dpR.getOffset(), dpR.getLength());
            System.out.println("message " + message);

            socket.close();
            frame.getContentPane().setBackground(Color.RED);
            frame.setVisible(true);

            Thread.sleep(1000);


            // Emission au prochain
            socket = new DatagramSocket();
            InetSocketAddress adrDest = new InetSocketAddress("127.0.0.1",nextPort);

            byte[] bufE = new String("hello").getBytes();
            DatagramPacket dpE = new DatagramPacket(bufE, bufE.length,adrDest);
            socket.send(dpE);
            System.out.println("message envoyé");

            socket.close();
            frame.getContentPane().setBackground(Color.GREEN);
            frame.setVisible(true);
            Thread.sleep(1000);

        }

    Runtime.getRuntime().addShutdownHook(new Thread() {
        @Override
        public void run() {
            socket.close();
            System.out.println("Socket fermé");
        }
    });

    }
}
```

Le chenillard UDP auto adaptatif

Dans l'exercice précédent, vous avez noté une certaine lourdeur dans l'utilisation de votre chenillard : impossibilité de rajouter facilement un cinquième écran, si l'écran 3 s'arrête, alors tout le système s'arrête, ...

Dans cet exercice, vous allez réaliser deux programmes :

• un programme serveur ordonnanceur

• un programme client

Le programme client sera lancé N fois et correspond à un terminal clignotant, le programme serveur sera lancé une seule fois.

Le fonctionnement du programme client sera le suivant :

• démarrage du programme

• il s'enregistre auprès du serveur

• il attend ensuite les ordres du serveur : un ordre pour passer en rouge, un ordre pour passer en vert

Le fonctionnement du programme serveur sera le suivant :

• il gère l'enregistrement des clients les uns après les autres

• le dernier client se connecte (pour cela, un paramètre sur la ligne de commande permet d'indiquer au programme client qu'il est le dernier, le dernier client s'enregistre avec un message spécifique)

• le programme serveur envoie ensuite les ordres aux clients pour gérer correctement

l'affichage des couleurs