Langage *Java*™

Correction du contrôle de connaissances - DESS et IST3/SETI

22 février 2002

Thomas LEDUC

1. Sources du fichier *Tree.java*

```
1  import java.io.File;
2  import javax.swing.*;
3  import java.awt.Dimension;
4  import java.util.Hashtable;
5
6  public class Tree extends JFrame
7  {
8      Tree(String répertoire)
9      {
10         super("Commande Tree : " + répertoire);
11         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12
13         File racine = new File(répertoire);
14         if (!racine.exists()) {
15             System.err.println("Le fichier " + répertoire + " n'existe pas !");
16             System.exit(1);
17         }
18
19         JTree arbreGraphique = new JTree(new Arbre(racine));
20         arbreGraphique.putClientProperty("JTree.lineStyle", "Angled");
21
22         JScrollPane ardoise = new JScrollPane(arbreGraphique);
23         ardoise.setPreferredSize(new Dimension(300,400));
24
25         setContentPane(ardoise);
26         pack();
27         setVisible(true);
28     }
29
30     public static void main(String[] arguments)
31     {
32         new Tree((arguments.length == 0) ? "." : arguments[0]);
33     }
34 }
35
36 class Arbre extends Hashtable
37 {
38     Arbre(File racine)
39     {
40         parcourir(racine, this);
41     }
42
43     private void parcourir(File fichierCourant, Hashtable arborescenceAssociée)
44     {
45         if (fichierCourant.isDirectory()) {
46             File[] lesFichiers = fichierCourant.listFiles();
47             Hashtable arborescenceFille = new Hashtable(lesFichiers.length);
48             for(int i=0; i<lesFichiers.length; i++)
49                 parcourir(lesFichiers[i], arborescenceFille);
50
51             arborescenceAssociée.put(fichierCourant.getName(), arborescenceFille);
52         }
53         else arborescenceAssociée.put(fichierCourant.getName(), fichierCourant);
54     }
55 }
```

2. Sources du fichier *TalkMulticast.java*

```
1  import java.net.*;
2  import java.awt.*;
3  import java.awt.event.*;
```

```
4   import javax.swing.*;
5
6   public class TalkMulticast extends JFrame
7   {
8       TalkMulticast()
9       {
10          super("Un \" talk \" multicast ..." );
11          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12          setContentPane(new EmetteurEtRecepteur());
13          pack();
14          setVisible(true);
15      }
16
17      public static void main(String[] arguments)
18      {
19          new TalkMulticast();
20      }
21  }
22
23  class EmetteurEtRecepteur extends JPanel implements Runnable,ActionListener
24  {
25      private final static int portXcast = 8084;
26      private final static int tailleMessage = 60;
27      private InetAddress groupeXcast = null;
28      private MulticastSocket socketMulticast = null;
29
30      private JTextField saisieEnEmission = new JTextField(tailleMessage);
31      private JTextField texteEnRéception = new JTextField(tailleMessage);
32
33      EmetteurEtRecepteur()
34      {
35          setLayout(new BoxLayout(this,BoxLayout.Y_AXIS));
36
37          add(new JLabel("Le message à envoyer (\" Entrée \" pour valider ) :" ));
38          add(saisieEnEmission);
39          saisieEnEmission.addActionListener(this);
40
41          add(new JLabel("Message reçu :" ));
42          texteEnRéception.setEditable(false);
43          add(texteEnRéception);
44
45          try {
46              groupeXcast = InetAddress.getByName("239.255.8.7");
47              socketMulticast = new MulticastSocket(portXcast);
48              socketMulticast.joinGroup(groupeXcast);
49          }
50          catch(Exception uneException) {
51              System.err.println(uneException);
52          }
53
54          (new Thread(this)).start();
55      }
56
57      public void run()
58      {
59          DatagramPacket leDatagramme = null;
60          byte[] leContenuDuMessage = null;
61
62          try{
63              while(true) {
64                  leContenuDuMessage = new byte[tailleMessage];
65                  leDatagramme = new DatagramPacket(leContenuDuMessage,tailleMessage);
66                  socketMulticast.receive(leDatagramme);
67                  texteEnRéception.setText(new String(leContenuDuMessage).trim());
68              }
69          }
70          catch(Exception uneException) {
71              System.err.println(uneException);
72          }
73      }
74
75      public void actionPerformed(ActionEvent événement)
76      {
77          try{
78              byte[] leContenuDuMessage = saisieEnEmission.getText().getBytes();
79              DatagramPacket leDatagramme =
80                  new DatagramPacket(leContenuDuMessage,leContenuDuMessage.length,
81                                     groupeXcast,portXcast);
82              socketMulticast.send(leDatagramme);
83          }
84          catch(Exception uneException) {
85              System.err.println(uneException);
86          }
87      }
88  }
```