```java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package chat.server;

import java.net.InetAddress;

/**
 * Classe representant un serveur d'un chat utilisant les sockets.
 * @author Nico
 */
public class Server {

    /**
     * Nom du serveur.
     */
    private String serverName;

    /**
     * Port de communication.
     */
    private int serverPort;

    /**
     * Etat du serveur. True : connecté, false sinon.
     */
    private boolean state;

    /**
     * Thread d'écoute permettant la connexion d'utilisateurs et de transmettre
     * les messages à tous les utilisateurs.
     */
    private ServerMultiThread serverMultiThread;

    /**
     * Vue du serveur.
     */
    private ServerRunningView serverRunningView;

    /**
     * Constructeur.
     */
    public Server() {
       initialize("server",1099);
    }

    /**
     * Getter.
```

```
51          * @return Port de communication.
52          */
53         public int getServerPort() {
54             return serverPort;
55         }
56 
57         /**
58          * Getter.
59          * @return Nom du serveur.
60          */
61         public String getServerName() {
62             return serverName;
63         }
64 
65         /**
66          * Methode permettant d'initialiser le serveur.
67          * @param sName Nom du serveur.
68          * @param sPort Port de communication.
69          */
70         private void initialize(String sName, int sPort) {
71             serverName = sName;
72             serverPort = sPort;
73             state = false;
74             serverMultiThread = null;
75             serverRunningView = new ServerRunningView(this);
76             serverRunningView.setLocationRelativeTo(null);
77             serverRunningView.setVisible(true);
78         }
79 
80         /**
81          * Methode permettant de lancer le serveur.
82          * @param port
83          */
84         public void start(String port)
85         {
86             try {
87                 serverPort = Integer.parseInt(port);
88                 state = true;
89                 serverMultiThread = new ServerMultiThread(this);
90                 serverMultiThread.start();
91 
                   serverRunningView.diplayServerLocalAdress(InetAddress.getL
                   ocalHost()
92                         .getHostAddress());
93 
94             } catch (Exception e) {
95                 System.err.println("Server exception : " + e.toString());
96             }
97         }
98 
99         /**
100         * Methode permettant d'arreter le serveur.
```

```
101         */
102        public void stop() {
103            state = false;
104            if (serverMultiThread!=null) {
105                serverMultiThread.interrupt();
106            }
107            refreshClientsNumber();
108        }
109
110        /**
111         * Methode permettant de raffraichir le nombre de client sur la
            vue.
112         */
113        public void refreshClientsNumber() {
114            if(serverMultiThread!=null) {
115                serverRunningView.setClientNumber(serverMultiThread.
116                        getClientsNumber());
117            } else {
118                serverRunningView.setClientNumber(0);
119            }
120        }
121
122        /**
123         * Methode permettant de savoir si le serveur est en train de
            tourner.
124         * @return True si oui, false sinon.
125         */
126        public boolean isRunning() {
127            return state;
128        }
129
130    }
131
```