Instituto Politécnico de Viana do Castelo

Escola Superior de Tecnologia e Gestão

FICHA PRÁTICA n.º 2

Integração de Sistemas de Informação

Unidade Curricular:	**Integração de Sistemas de Informação**

Tema da Ficha Pratica:

Utilização de Sockets

Objectivos:

Pretende-se com esta ficha prática que os alunos interajam com o conceito de sockets.

Bibliografia:

Para apoio a esta ficha os alunos devem consultar os apontamentos teóricos e práticos da disciplina bem como de outros recursos online.

Índice

1. Utilização do conceito de Sockets usando a linguagem de programação JAVA

Para apoio à elaboração do exemplo seguinte consulte a documentação disponível em:

[1] http://java.sun.com/developer/onlineTraining/Programming/BasicJava2/socket.html

[2] http://java.sun.com/docs/books/tutorial/networking/sockets/index.html

1.1 tcpServer.java

```
// tcpServer.java by fpont 3/2000

// usage : java tcpServer <port number>.
// default port is 1500.
// connection to be closed by client.
// this server handles only 1 connection.

import java.net.*;
import java.io.*;

public class tcpServer {

    public static void main(String args[]) {

       int port;
       ServerSocket server_socket;
       BufferedReader input;

       try {
           port = Integer.parseInt(args[0]);
       }
       catch (Exception e) {
           System.out.println("port = 1500 (default)");
           port = 1500;
       }

       try {

           server_socket = new ServerSocket(port);
           System.out.println("Server waiting for client on port " +
                            server_socket.getLocalPort());

           // server infinite loop
           while(true) {
              Socket socket = server_socket.accept();
              System.out.println("New connection accepted " +
                               socket.getInetAddress() +
                               ":" + socket.getPort());
              input       =       new       BufferedReader(new       InputStreamRead-
er(socket.getInputStream()));
              // print received data
              try {
                  while(true) {
                     String message = input.readLine();
                     if (message==null) break;
                     System.out.println(message);
                  }
```

```
                }
                catch (IOException e) {
                    System.out.println(e);
                }

                // connection closed by client
                try {
                    socket.close();
                    System.out.println("Connection closed by client");
                }
                catch (IOException e) {
                    System.out.println(e);
                }

            }


        }

        catch (IOException e) {
            System.out.println(e);
        }
    }
}
```

1.2 tcpClient.java

```
// tcpClient.java by fpont 3/2000

// usage : java tcpClient <server> <port>
// default port is 1500

import java.net.*;
import java.io.*;

public class tcpClient {



    public static void main(String[] args) {

        int port = 1500;
        String server = "localhost";
        Socket socket = null;
        String lineToBeSent;
        BufferedReader input;
        PrintWriter output;
        int ERROR = 1;

        // read arguments
        if(args.length == 2) {
            server = args[0];
            try {
                port = Integer.parseInt(args[1]);
            }
            catch (Exception e) {
                System.out.println("server port = 1500 (default)");
                port = 1500;
            }
        }



        // connect to server
        try {
            socket = new Socket(server, port);
            System.out.println("Connected with server " +
                                    socket.getInetAddress() +
                                    ":" + socket.getPort());
        }
        catch (UnknownHostException e) {
            System.out.println(e);
            System.exit(ERROR);
        }
        catch (IOException e) {
            System.out.println(e);
            System.exit(ERROR);
        }



        try {
            input = new BufferedReader(new InputStreamReader(System.in));
            output = new PrintWriter(socket.getOutputStream(),true);

            // get user input and transmit it to server
            while(true) {
                lineToBeSent = input.readLine();
                // stop if input line is "."
                if(lineToBeSent.equals(".")) break;
                output.println(lineToBeSent);
            }
        }
        catch (IOException e) {
            System.out.println(e);
        }

        try {
            socket.close();
        }
        catch (IOException e) {
            System.out.println(e);
        }
    }
}
```

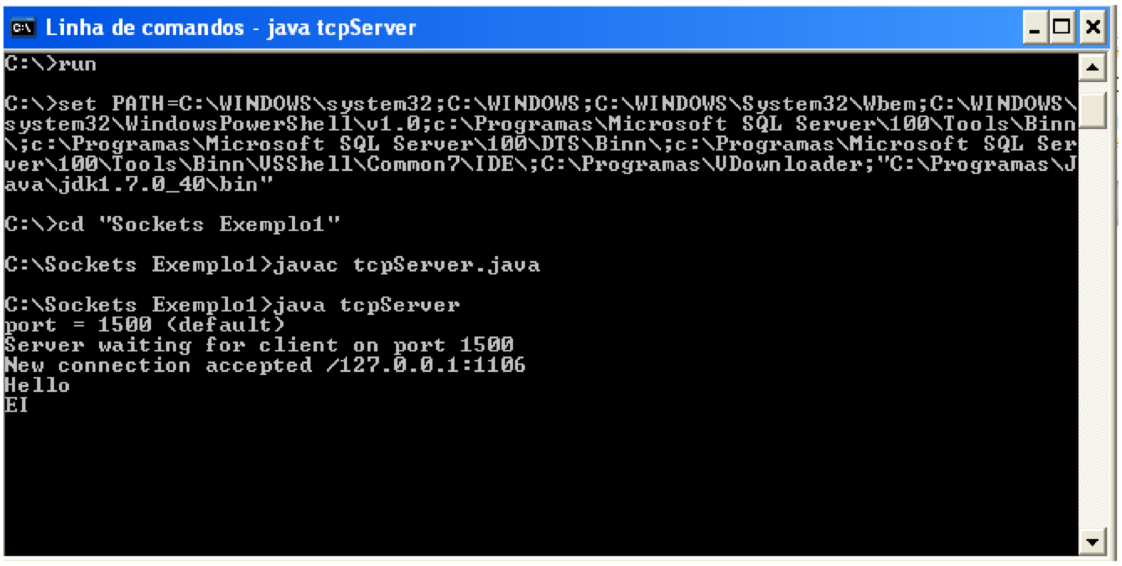
Com base no actual cliente e servidor de sockets descrito em cima, apenas é efectuada a ligação entre o programa servidor e o cliente. Para permitir a ligação de múltiplos clientes, o programa servidor deve ser convertido para um servidor multithread conforme o exemplo da classe seguinte:

```
import java.awt.Color;
import java.awt.BorderLayout;
import java.awt.event.*;
import javax.swing.*;

import java.io.*;
import java.net.*;
```

```
class ClientWorker implements Runnable {
  private Socket client;
  private JTextArea textArea;

  ClientWorker(Socket client, JTextArea textArea) {
   this.client = client;
   this.textArea = textArea;
  }

  public void run(){
    String line;
    BufferedReader in = null;
    PrintWriter out = null;
    try{
      in = new BufferedReader(new InputStreamRead-
er(client.getInputStream()));
      out = new PrintWriter(client.getOutputStream(), true);
    } catch (IOException e) {
      System.out.println("in or out failed");
      System.exit(-1);
    }

    while(true){
      try{
        line = in.readLine();
//Send data back to client
         out.println(line);
         textArea.append(line);
       } catch (IOException e) {
         System.out.println("Read failed");
         System.exit(-1);
       }
    }
  }
}

class SocketThrdServer extends JFrame{

   JLabel label = new JLabel("Text received over socket:");
   JPanel panel;
   JTextArea textArea = new JTextArea();
   ServerSocket server = null;

   SocketThrdServer(){ //Begin Constructor
     panel = new JPanel();
     panel.setLayout(new BorderLayout());
     panel.setBackground(Color.white);
     getContentPane().add(panel);
     panel.add("North", label);
     panel.add("Center", textArea);
   } //End Constructor

  public void listenSocket(){
    try{
      server = new ServerSocket(4444);
    } catch (IOException e) {
      System.out.println("Could not listen on port 4444");
      System.exit(-1);
    }
    while(true){
      ClientWorker w;
      try{
        w = new ClientWorker(server.accept(), textArea);
        Thread t = new Thread(w);
        t.start();
      } catch (IOException e) {
        System.out.println("Accept failed: 4444");
        System.exit(-1);
      }
    }
  }

  protected void finalize(){
//Objects created in run method are finalized when
//program terminates and thread exits
     try{
        server.close();
    } catch (IOException e) {
        System.out.println("Could not close socket");
        System.exit(-1);
    }
  }

  public static void main(String[] args){
        SocketThrdServer frame = new SocketThrdServer();
        frame.setTitle("Server Program");
        WindowListener l = new WindowAdapter() {
                public void windowClosing(WindowEvent e) {
                        System.exit(0);
                }
        };
        frame.addWindowListener(l);
        frame.pack();
        frame.setVisible(true);
        frame.listenSocket();
  }
}
```

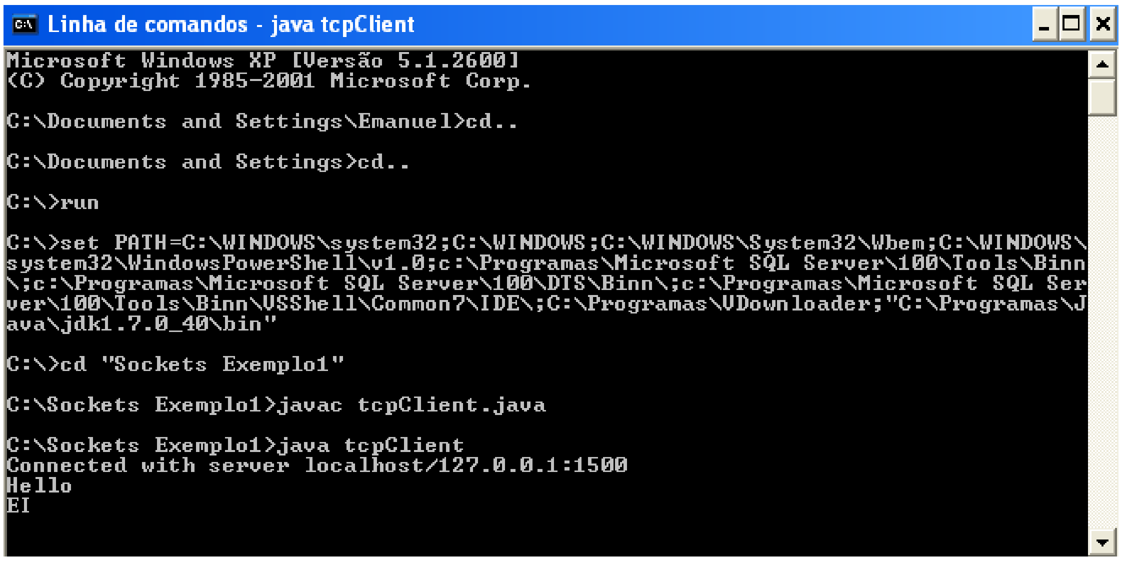
1.3 Execução

Janela do Servidor



```
C:\>run

C:\>set PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\
system32\WindowsPowerShell\v1.0;c:\Programas\Microsoft SQL Server\100\Tools\Binn
\;c:\Programas\Microsoft SQL Server\100\DTS\Binn\;c:\Programas\Microsoft SQL Ser
ver\100\Tools\Binn\VSShell\Common7\IDE\;C:\Programas\VDownloader;"C:\Programas\J
ava\jdk1.7.0_40\bin"

C:\>cd "Sockets Exemplo1"

C:\Sockets Exemplo1>javac tcpServer.java

C:\Sockets Exemplo1>java tcpServer
port = 1500 (default)
Server waiting for client on port 1500
New connection accepted /127.0.0.1:1106
Hello
EI
```

Janela do Cliente



```
Microsoft Windows XP [Versão 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Emanuel>cd..

C:\Documents and Settings>cd..

C:\>run

C:\>set PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\
system32\WindowsPowerShell\v1.0;c:\Programas\Microsoft SQL Server\100\Tools\Binn
\;c:\Programas\Microsoft SQL Server\100\DTS\Binn\;c:\Programas\Microsoft SQL Ser
ver\100\Tools\Binn\VSShell\Common7\IDE\;C:\Programas\VDownloader;"C:\Programas\J
ava\jdk1.7.0_40\bin"

C:\>cd "Sockets Exemplo1"

C:\Sockets Exemplo1>javac tcpClient.java

C:\Sockets Exemplo1>java tcpClient
Connected with server localhost/127.0.0.1:1500
Hello
EI
```

2. Utilização do conceito de Sockets – Envio de Mensagens

2.1 Server.java

```
import java.lang.*;
import java.io.*;
import java.net.*;

class Server {
   public static void main(String args[]) {
      String data = "Integração de Sistemas";
      try {
         ServerSocket srvr = new ServerSocket(1234);
         Socket skt = srvr.accept();
         System.out.print("Servidor conectado!\n");
         PrintWriter out = new PrintWriter(skt.getOutputStream(), true);
         System.out.print("Enviando string: '" + data + "'\n");
         out.print(data);
         out.close();
         skt.close();
         srvr.close();
      }
      catch(Exception e) {
         System.out.print("Whoops! Não funciona!\n");
      }
   }
}
```

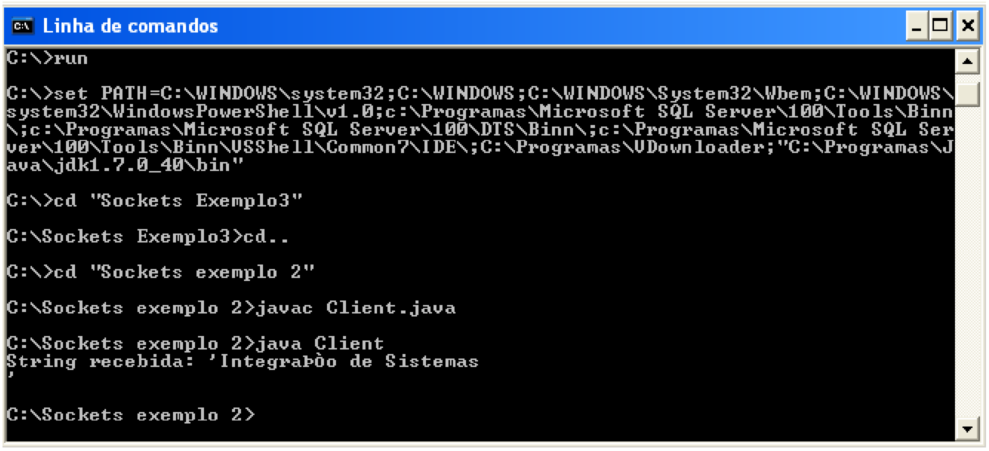
2.2 Client.java

```
import java.lang.*;
import java.io.*;
import java.net.*;

class Client {
   public static void main(String args[]) {
      try {
         Socket skt = new Socket("localhost", 1234);
         BufferedReader in = new BufferedReader(new
            InputStreamReader(skt.getInputStream()));
         System.out.print("String recebida: '");

         while (!in.ready()) {}
         System.out.println(in.readLine()); // Read one line and output it

         System.out.print("'\n");
         in.close();
      }
      catch(Exception e) {
         System.out.print("Whoops! Não funciona!\n");
      }
   }
}
```

2.3 Execução

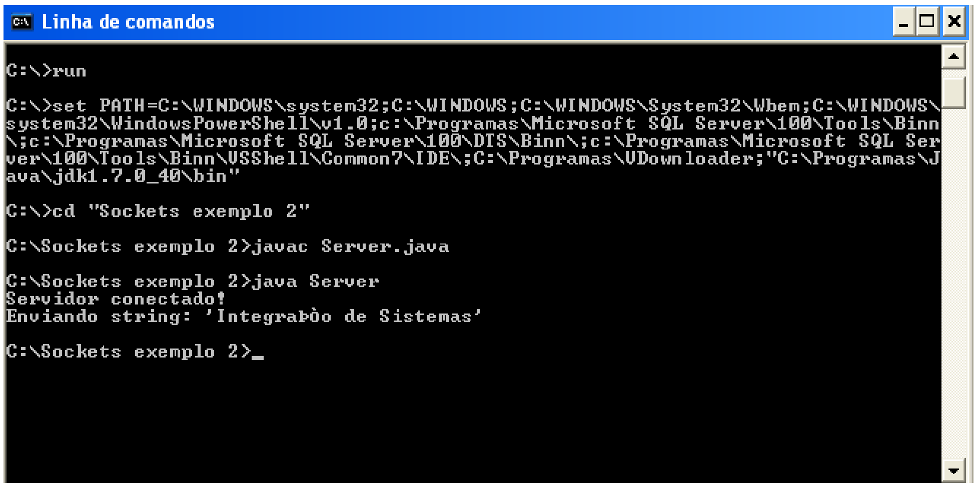
Neste exemplo quando se liga o cliente é enviada uma mensagem do servidor para o cliente , com a string definida no ficheiro java.

Janela do Cliente



```
C:\>run

C:\>set PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\
system32\WindowsPowerShell\v1.0;c:\Programas\Microsoft SQL Server\100\Tools\Binn
\;c:\Programas\Microsoft SQL Server\100\DTS\Binn\;c:\Programas\Microsoft SQL Ser
ver\100\Tools\Binn\VSShell\Common7\IDE\;C:\Programas\VDownloader;"C:\Programas\J
ava\jdk1.7.0_40\bin"

C:\>cd "Sockets Exemplo3"

C:\Sockets Exemplo3>cd..

C:\>cd "Sockets exemplo 2"

C:\Sockets exemplo 2>javac Client.java

C:\Sockets exemplo 2>java Client
String recebida: 'Integrapòo de Sistemas
'

C:\Sockets exemplo 2>
```

Janela do Servidor



```
C:\>run

C:\>set PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\
system32\WindowsPowerShell\v1.0;c:\Programas\Microsoft SQL Server\100\Tools\Binn
\;c:\Programas\Microsoft SQL Server\100\DTS\Binn\;c:\Programas\Microsoft SQL Ser
ver\100\Tools\Binn\VSShell\Common7\IDE\;C:\Programas\VDownloader;"C:\Programas\J
ava\jdk1.7.0_40\bin"

C:\>cd "Sockets exemplo 2"

C:\Sockets exemplo 2>javac Server.java

C:\Sockets exemplo 2>java Server
Servidor conectado!
Enviando string: 'Integrapòo de Sistemas'

C:\Sockets exemplo 2>_
```

3. Utilização do conceito de Sockets – Execução de Cálculos no Servidor

3.1 Server.java

```
import java.io.*;
import java.net.*;
import java.util.Hashtable;

public class Server
{
	public static void main(String []args)
	{
		try
		{
			// Create a socket on server
			ServerSocket ss = new ServerSocket(555);

			// hashtable to manage list of online and offline users
			Hashtable tOnlineUsers = new Hashtable(10);
			Hashtable tOfflineUsers = new Hashtable(10);

			// ---------------------------------------------------------------
			// Now start accepting connections from clients in a while loop
			// The server should run in an infinite loop
			while(true)
			{
				Socket socket = ss.accept();	// accept connection from 
client
				System.out.println("A new client is connected.");

				// to get data to and from server
				InputStream in = socket.getInputStream();
				BufferedReader br = new BufferedReader(new InputStreamRead-
er(in));
				OutputStream out = socket.getOutputStream();
				PrintWriter pr = new PrintWriter(out, true);

				// read user name from the client and store in table
				// in the format username + socket
				String strUserName = br.readLine();
				System.out.println("Username: " + strUserName + "\n");
				tOnlineUsers.put(strUserName, socket);

				// create a thread to allow simultaneous connections
				Worker w = new Worker(socket, tOnlineUsers, tOfflineUsers,
strUserName);
				w.start();
			}	// End of while

		}	// End of try
		catch(Exception e)
		{
			System.out.println("Some kind of error has occurred.");
		}	// End of exception

	}	// End of main()
}	// End of class
```

3.2 Client.java

```
import java.io.*;
import java.net.*;
import java.util.Hashtable;

public class Client
{

       public static void main(String []args)
       {
              // Declarations to get input from keyboard
              // --------------------------------------
              int port=555;             // server port
              String strPort="",        // server port
                        ip="",                 // IP of server
                        strUserName="";  // User name of client

              BufferedReader input = new BufferedReader(new InputStreamRead-
er(System.in));

              // Get ip, port & user name from the client
              try
              {
                     // instructions
                     System.out.println("Instructions to connect to the server.\n\n" +
                            "-> If the server is running on the same computer," +
                            "just press enter key or enter \"127.0.0.1\".\n\n" +
                            "-> Do not enter anything when it asks for port unless" +
                            "you don't edit the code in Server.java and edit it." +
                            "Just leave it blank by pressing the enter key.\n\n" +
                            "-> Enter the UserName of your choice. It can be you own
name.\n");

                     // get IP from the user
                     System.out.print("\n\nEnter IP of the server: ");
                     ip = input.readLine();
                     if (ip.equals(""))
                            ip = "127.0.0.1";   // default IP

                     // get port from user
                     System.out.print("Port Number: ");
                     strPort = input.readLine();
                     if (strPort.equals(""))
                            port = 555;              // default port
                     else
                            port = Integer.parseInt(strPort);

                     // get user name from the client
                     strUserName = "temp";
                     do
                     {
                            System.out.print("Enter User Name: ");
                            strUserName = input.readLine();
                     }
                     while (strUserName.equals(""));  // repeat until valid user name
is given


                     // -----------------------------------------------------------------
                     // IP, port and username is complete at this point
                     // Now, create a socket to connect to server.
                     // After that manage the connection in a while loop
                     // until user wants to exit on his/her will
```

```
                    // ------------------------------------------------------------

                    // create a new socket
                    Socket socket = new Socket(ip, port);

                    // Connection successfull at this point, so inform user about this
                    System.out.print("\n\n\t\tConnection successful.\n\t\t------------
----------");

                    // Declarations to manage connection
                    // --------------------------------
                    String strAnother = "z",  // looping variable
                             strInt1 = "45",        // First integer
                             strInt2 = "2",         // Second integer
                             strOp = "-",                 // Operator
                             strResult="";
                    // to get data to and from server
                    InputStream in = socket.getInputStream();
                    BufferedReader br = new BufferedReader(new InputStreamReader(in));
                    OutputStream out = socket.getOutputStream();
                    PrintWriter pr = new PrintWriter(out, true);

                    // send user name to the server
                    pr.println(strUserName);

                    // The while loop
                    // --------------
                    while (strAnother.charAt(0) != 'd')
                    {
                          // giver user a menu
                          System.out.println("\n\nEnter (d) to disconnect from serv-
er.\n" +
                                "      (s) to see other online people.\n" +
                                "      any other key to solve a simple expression.
");
                          strAnother = input.readLine();
                          if (strAnother.equals(""))
                                strAnother = "z";
                          switch(strAnother.charAt(0))
                          {
                                case 'd':
                                      pr.println("d");
                                      break;
                                case 's':    // send notification to server to see
online people
                                      pr.println("s");    // write (s) to server
                                      strResult = br.readLine();
                                      System.out.println(strResult);
                                      break;
                                default:     // send expression to the server
                                      pr.println("z");
                                      // get first number
                                      System.out.print("Enter First Number: ");
                                      strInt1 = input.readLine();
                                      if (strInt1.equals(""))
                                            strInt1 = "45";

                                      // get second number
                                      System.out.print("Enter Second Number: ");
                                      strInt2 = input.readLine();
                                      if (strInt2.equals(""))
                                            strInt2 = "2";

                                      // get operator
```

```
                                                System.out.print("Enter Operator: ");
                                                strOp = input.readLine();
                                                if (strOp.equals(""))
                                                        strOp = "-";

                                                // write 2 integers and ooperator to the serv-
er
                                                pr.println(strInt1);
                                                pr.println(strInt2);
                                                pr.println(strOp);

                                                // get result from the server
                                                strResult = br.readLine();
                                                System.out.println(strResult);
                                                break;
                                }       // End of switch
                        }       // End of the while loop

                        // At this point client wants to disconnect from the server,
                        // so close the connection
                        socket.close();

                }       // End of try
                catch(Exception e)
                {
                        System.out.println("Some kind of error has occurred.");
                        System.exit(0);
                }       // End of exception

        }       // End of main()
}       // End of class
```

3.3 Worker.java

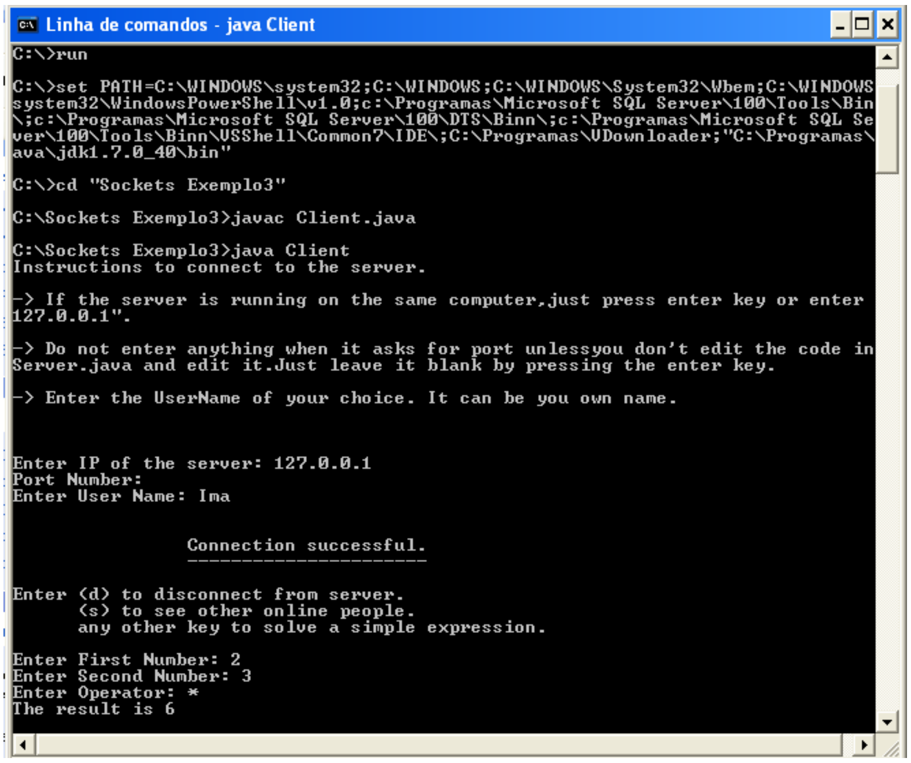
```
import java.io.*;
import java.net.*;
import java.util.*;

public class Worker extends Thread
{
        Socket socket;
        Hashtable tOnlineUsers;
        Hashtable tOfflineUsers;
        String strUserName;

        // -----------------constructor------------------------
        public Worker(Socket s, Hashtable online, Hashtable offline, String userName)
        {
                socket = s;
                tOnlineUsers = online;
                tOfflineUsers = offline;
                strUserName = userName;
        }       // End of constructor

        // ------------------run------------------------------
        public void run()
        {
                try
                {
                        // to get data to and from server
                        InputStream in = socket.getInputStream();
                        BufferedReader br = new BufferedReader(new InputStreamReader(in));
                        OutputStream out = socket.getOutputStream();
                        PrintWriter pr = new PrintWriter(out, true);
```

```
                    BufferedReader input = new BufferedReader(new InputStreamRead-
er(System.in));
                    String strAnother="", strResult="";

                    while(!strAnother.equals("d"))
                    {
                          strResult = "";
                          strAnother = br.readLine();      // read option
                          System.out.println("User option is " + strAnother);
                          switch(strAnother.charAt(0))
                          {
                                case 'd':
                                      System.out.println("Request for disconnect.");
                                      break;
                                case 's':           // print a list of online people
                                      System.out.println("Sending list of online
people....\n");
                                      Enumeration e = tOnlineUsers.keys();
                                      while (e.hasMoreElements())
                                            strResult += e.nextElement() + ", ";
                                      pr.println(strResult);
                                      break;
                                case 'z':
                                      System.out.println("Calculating result and
sending answer....\n");
                                      String strInt1 = br.readLine();
                                      String strInt2 = br.readLine();
                                      String strOp = br.readLine();
                                      int int1 = Integer.parseInt(strInt1);
                                      int int2 = Integer.parseInt(strInt2);
                                      char chOp = strOp.charAt(0);
                                      String strCalcResult = "";
                                      switch(chOp)
                                      {
                                            case '+':
                                                  strCalcResult = "The result is
"+(int1+int2); break;
                                            case '-':
                                                  strCalcResult = "The result is
"+(int1-int2); break;
                                            case '*':
                                                  strCalcResult = "The result is
"+(int1*int2); break;
                                            case '/':
                                                  strCalcResult = "The result is
"+(int1/int2); break;
                                            default:
                                                  strCalcResult = "The Operator is
invalid."; break;
                                      }
                                      pr.println(strCalcResult);

                                      break;
                          }      // End of switch
                    }      // End of while

                    // Now, close the socket after deleting that socket from online
list
                    Socket s = (Socket)tOnlineUsers.remove(strUserName);
                    tOfflineUsers.put(strUserName, s);
                    socket.close();

              }      // End of try
```

```
                catch(Exception e)
                {
                        System.out.println("Error has occurred in Worker.");
                }       // End of exception

        }       // End of run()

}       // End of class
```

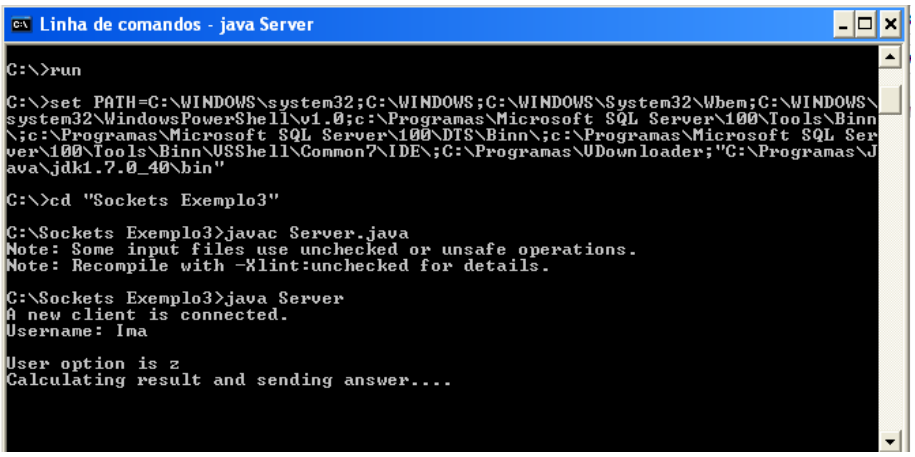
3.4 Execução

Janela do Cliente


Linha de comandos - java Client

```
C:\>run

C:\>set PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\
system32\WindowsPowerShell\v1.0;c:\Programas\Microsoft SQL Server\100\Tools\Bin
\;c:\Programas\Microsoft SQL Server\100\DTS\Binn\;c:\Programas\Microsoft SQL Se
ver\100\Tools\Binn\VSShell\Common7\IDE\;C:\Programas\VDownloader;"C:\Programas\
ava\jdk1.7.0_40\bin"

C:\>cd "Sockets Exemplo3"

C:\Sockets Exemplo3>javac Client.java

C:\Sockets Exemplo3>java Client
Instructions to connect to the server.

-> If the server is running on the same computer,just press enter key or enter
127.0.0.1".

-> Do not enter anything when it asks for port unlessyou don't edit the code in
Server.java and edit it.Just leave it blank by pressing the enter key.

-> Enter the UserName of your choice. It can be you own name.


Enter IP of the server: 127.0.0.1
Port Number:
Enter User Name: Ima


                Connection successful.
                ----------------------

Enter (d) to disconnect from server.
      (s) to see other online people.
      any other key to solve a simple expression.

Enter First Number: 2
Enter Second Number: 3
Enter Operator: *
The result is 6
```

Janela do Servidor


Linha de comandos - java Server

```
C:\>run

C:\>set PATH=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\
system32\WindowsPowerShell\v1.0;c:\Programas\Microsoft SQL Server\100\Tools\Binn
\;c:\Programas\Microsoft SQL Server\100\DTS\Binn\;c:\Programas\Microsoft SQL Ser
ver\100\Tools\Binn\VSShell\Common7\IDE\;C:\Programas\VDownloader;"C:\Programas\J
ava\jdk1.7.0_40\bin"

C:\>cd "Sockets Exemplo3"

C:\Sockets Exemplo3>javac Server.java
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

C:\Sockets Exemplo3>java Server
A new client is connected.
Username: Ima

User option is z
Calculating result and sending answer....
```