Artigo:

Sockets programming in Java: A tutorial

Writing your own client/server applications can be done seamlessly using Java

Qusay H. Mahmoud

http://www.javaworld.com/javaworld/jw-12-1996/jw-12-sockets.html

smtpClient.java

```
// Cliente simplificado usando o protocolo SMTP
// Lembrar do protocolo SMTP para entender o cliente abaixo

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

public class smtpClient {
public static void main(String[] args) {

// declarações:
// smtpClient: client socket
// os: output stream
// is: input stream

Socket smtpSocket = null;
DataOutputStream os = null;
DataInputStream is = null;

// Inicio:
// Tenta abrir socket na porta 25
// Tenta abrir input-output streams

try {
  smtpSocket = new Socket("hostname", 25);
  os = new DataOutputStream(smtpSocket.getOutputStream());
  is = new DataInputStream(smtpSocket.getInputStream());
}

catch (UnknownHostException e) {
  System.err.println("Don't know about host: hostname");
}

catch (IOException e) {
  System.err.println("Couldn't get I/O for the connection to: hostname");
}

// Se tudo foi bem inicializado, podemos enviar os dados
// para a porta 25 do servidor
if (smtpSocket != null && os != null && is != null) {
  try {
    // Os commando em letra maiúscula, são os commando do SMTP
    // Detalhes na RFC1822/3

    os.writeBytes("HELO\n");
    os.writeBytes("MAIL From: maria@ime.usp.br\n");
    os.writeBytes("RCPT To: pedro@harvard.univ.com\n");
    os.writeBytes("DATA\n");
    os.writeBytes("From: maria@ime.usp.br\n");
    os.writeBytes("Subject: testing\n");
    os.writeBytes("\n");  // linha em branco do protocolo
    os.writeBytes("Exemplo de mensagem\n"); // corpo da mensagem
    os.writeBytes("\n.\n"); // linha com um ponto (veja o protocolo)
    os.writeBytes("QUIT");

    // keep on reading from/to the socket till we receive the "Ok" from SMTP,
    // once we received that then we want to break.

    String responseLine;
    while ((responseLine = is.readLine()) != null) {
      System.out.println("Server: " + responseLine);
      if (responseLine.indexOf("Ok") != -1) {    // ?????? OK ????
         break;
      }
    }

    // clean up:
    // close the output stream
    // close the input stream
    // close the socket

    os.close();
    is.close();
    smtpSocket.close();
  }

  catch (UnknownHostException e) {
    System.err.println("Trying to connect to unknown host: " + e);
  }

  catch (IOException e) {
    System.err.println("IOException: " + e);
  }
} // do if
} // do main
} // da classe
```

echo3.java

```
// servidor de eco
// recebe uma linha e ecoa a linha recebida.

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

```java
public class echo3 {
public static void main(String args[]) {

// declaration section:
// declare a server socket and a client socket for the server
// declare an input and an output stream

ServerSocket echoServer = null;
String line;
DataInputStream is;
PrintStream os;
Socket clientSocket = null;

// Try to open a server socket on port 9999
// Note that we can't choose a port less than 1023 if we are not
// privileged users (root)

try {
  echoServer = new ServerSocket(9999);
}
catch (IOException e) {
  System.out.println(e);
}

// Create a socket object from the ServerSocket to listen and accept
// connections.
// Open input and output streams

try {
  clientSocket = echoServer.accept();
  is = new DataInputStream(clientSocket.getInputStream());
  os = new PrintStream(clientSocket.getOutputStream());

  // As long as we receive data, echo that data back to the client.

  while (true) {
    line = is.readLine();
    os.println(line);
  }
}

catch (IOException e) {
  System.out.println(e);
}
} // do main
} // da classe
```