Sockets Example

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | |  | This example introduces you to Java socket programming. The server listens for a connection. When a connection is established by a client. The client can send data. In the current example the client sends the message "Hi my server". To terminate the connection, the client sends the message "bye". Then the server sends the message "bye" too. Finally the connection is ended and the server waits for an other connection. The two programs should be runned in the same machine. however if you want to run them in two different machines, you may simply change the adress "localhost" by the IP adress of the machine where you will run the server.  **The server**  import java.io.\*;  import java.net.\*;  public class Provider{  ServerSocket providerSocket;  Socket connection = null;  ObjectOutputStream out;  ObjectInputStream in;  String message;  Provider(){}  void run()  {  try{  //1. creating a server socket  providerSocket = new ServerSocket(2004, 10);  //2. Wait for connection  System.out.println("Waiting for connection");  connection = providerSocket.accept();  System.out.println("Connection received from " + connection.getInetAddress().getHostName());  //3. get Input and Output streams  out = new ObjectOutputStream(connection.getOutputStream());  out.flush();  in = new ObjectInputStream(connection.getInputStream());  sendMessage("Connection successful");  //4. The two parts communicate via the input and output streams  do{  try{  message = (String)in.readObject();  System.out.println("client>" + message);  if (message.equals("bye"))  sendMessage("bye");  }  catch(ClassNotFoundException classnot){  System.err.println("Data received in unknown format");  }  }while(!message.equals("bye"));  }  catch(IOException ioException){  ioException.printStackTrace();  }  finally{  //4: Closing connection  try{  in.close();  out.close();  providerSocket.close();  }  catch(IOException ioException){  ioException.printStackTrace();  }  }  }  void sendMessage(String msg)  {  try{  out.writeObject(msg);  out.flush();  System.out.println("server>" + msg);  }  catch(IOException ioException){  ioException.printStackTrace();  }  }  public static void main(String args[])  {  Provider server = new Provider();  while(true){  server.run();  }  }  }    **The client**  import java.io.\*;  import java.net.\*;  public class Requester{  Socket requestSocket;  ObjectOutputStream out;  ObjectInputStream in;  String message;  Requester(){}  void run()  {  try{  //1. creating a socket to connect to the server  requestSocket = new Socket("localhost", 2004);  System.out.println("Connected to localhost in port 2004");  //2. get Input and Output streams  out = new ObjectOutputStream(requestSocket.getOutputStream());  out.flush();  in = new ObjectInputStream(requestSocket.getInputStream());  //3: Communicating with the server  do{  try{  message = (String)in.readObject();  System.out.println("server>" + message);  sendMessage("Hi my server");  message = "bye";  sendMessage(message);  }  catch(ClassNotFoundException classNot){  System.err.println("data received in unknown format");  }  }while(!message.equals("bye"));  }  catch(UnknownHostException unknownHost){  System.err.println("You are trying to connect to an unknown host!");  }  catch(IOException ioException){  ioException.printStackTrace();  }  finally{  //4: Closing connection  try{  in.close();  out.close();  requestSocket.close();  }  catch(IOException ioException){  ioException.printStackTrace();  }  }  }  void sendMessage(String msg)  {  try{  out.writeObject(msg);  out.flush();  System.out.println("client>" + msg);  }  catch(IOException ioException){  ioException.printStackTrace();  }  }  public static void main(String args[])  {  Requester client = new Requester();  client.run();  }  } |