TCP - Transmission Control Protocol

ТCP е транспортен протокол, който предоставя потвърдени услуги - изпращачът е уведомен дали получателят е получил данните (пакета).

За изпращането на каквито и да било данни, предварително е задължително установяване на връзката между двата компютъра. Тук всеки

един от компютрите има своя собствена роля: клиент и сървър.

Сървър -( от англ. server) е компютърна система или програма, която предоставя услуга (service). Която, за да бъде използвана от другите компютри те трябва да станат клиенти на този сървър. Сървърът има ролята да следи за входящи заяки за свързване ("слуша"). Когато такава заяка пристигне, сървърът започва обмяната на данни със свързалия се клиент. Редът на предаване на данните, съдържанието и типа на данните се нарича протокол.

Клиент - компютърна система или програма, която достъпва услуга на отдалечен сървър. Клиентът инициара връзката за обмен, докато сървърът само чака за входящи връзки. Клиентът-сокет може да поддържа връзка само към един сървър в един момент, докато сървърът може да обработва няколко клиенти наведжнъж.

//Реализация в Java

//Клиентът се създава, чрез създаване на инстанция от клас Soцket (намиращ се в пакета java.net.\*)

Socket client = new Socket("remote\_host",отдалечен\_порт); //Още при създаването си обекта от клас Socket осъществява връзка с отдалечената услуга, като за целта се изискват името (hostname) или IP адреса на машината и услугата(порта), към която желае да се свърже.

//Данните се се обменят посредство потоци. Във всеки обект от тип Socket има два потока: входен и изходен. Потоците се достъпват, чрез аксесорите getInputStream() и getOutputStream(). Като към тези потоци могат да бъдат закачени различни "четци", "записвачи" или "скенери".

InputStream inputClient = client.getInputStream();

OutputStream outputClient = client.getOutputStream();

PrintWriter printToServer = new PrintWriter(outputClient);

Scanner inputScanner = new Scanner(inputClient);

//Начините на четене и запис са абсолютно идентични на методите за четене и запис при файловете или стандартните вход и изход (клавиатура, монитор);

//Сървър със средствата на Java се създава на два етапа: първо - създава се обект от клас ServerSocket, който има за цел да следи за входящи връзки. При наличието на такава методът на обекта връща обект от клас Socket. Това представлява осъществената връзка с отдалечения клиент. Входът и изходът тук са идентични при разгледаните в клиента.

//Пример - да се създаде сървър, на който клиентите дават число, а той им изпраща толкова на брой съощения, колкото е числото. Сървърът да бъде многонишков.

//Сървър

package net.pxpress.networking.tcp\_example;

import java.io.IOException;

import java.io.PrintWriter;

import java.net.\*;

import java.util.Scanner;

class ClientServiceThread extends Thread {

 private Socket socket;

 public ClientServiceThread(Socket socket) {

 this.socket = socket;

 }

 @Override

 public void run() {

 Scanner input = null;

 PrintWriter output = null;

 try {

 input = new Scanner(socket.getInputStream());

 output = new PrintWriter(socket.getOutputStream(),true);

 int N = input.nextInt();

 for (int i = 0; i < N; i++) {

 output.println((i + 1) + " My message for you!");

 try {

 Thread.sleep(1000);

 } catch (InterruptedException e) {

 // Nyam da se prekasva rabotata na nishkata

 }

 }

 } catch (IOException e) {

 System.err

 .println("Problem pri otvaryaneto na potocite za vhod i izhod");

 } finally {

 if (input != null)

 input.close();

 if (output != null)

 output.close();

 try {

 socket.close();

 } catch (IOException e) {

 System.err.println("Problem pri zatvarqneto na soketa");

 }

 }

 }

}

public class TCPExampleServer {

 public static void main(String[] args) {

 ServerSocket server = null;

 try {

 server = new ServerSocket(3456);

 while (true) {

 Socket clientConnection = server.accept();

 System.out.println("New Connection >> "

 + clientConnection.getRemoteSocketAddress());

 ClientServiceThread incomingClient = new ClientServiceThread(

 clientConnection);

 incomingClient.start();

 }

 } catch (IOException e) {

 System.err.println("Problem pri startiraneto na servera");

 }

 }

}

//Клиент

package net.pxpress.networking.tcp\_example;

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class TCPExampleClient {

 public static void main(String[] args) {

 Socket connection =null;

 Scanner input = null;

 PrintWriter output = null;

 try

 {

 connection = new Socket("localhost",3456);

 input = new Scanner(connection.getInputStream());

 output = new PrintWriter(connection.getOutputStream(),true);

 Random rand = new Random();

 output.println(rand.nextInt(15));

 while(input.hasNextLine())

 System.out.println(input.nextLine());

 }

 catch (UnknownHostException e) {

 System.err.println("Ne posnat hostname");

 }

 catch(IOException e)

 {

 System.err.println("Problem pri svarzvaneto");

 }

 finally

 {

 if(input!=null)

 input.close();

 if(output!=null)

 output.close();

 try

 {

 if(connection!=null)

 connection.close();

 }

 catch(IOException e)

 {

 System.err.println("Problem pri zatvaryaneto na soketa");

 }

 }

 }

}