TCP/IP通讯传输流程图：

` 客户端： 服务端：

应用层 （http服务端）

链路层（网络）

网络层（ip）

应用层 （http客户端）

http协议

传输层（tcp)

传输层（tcp)

Tcp协议

网络层（ip

Ip协议 ip协议

链路层（网络）

以太网协议 以太网协议

InetAddress类的用法（表示互联网的ip）：

练1：

主要方法：static getByname()获取InetAddress对象.

**package** cn.com.inetAddress

**import** java.net.InetAddress;

**import** java.net.UnknownHostException;

**public** **class** InetAddressTest {

**public** **static** **void** main(String[] args) {

**try** {

InetAddress address = InetAddress.*getLocalHost*();

System.*out*.println(address.getHostAddress());//获得主机的ip

System.*out*.println(address.getHostName());//获得主机名

} **catch** (UnknownHostException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**try** {

InetAddress address1 = InetAddress.*getByName*("hao.360.cn");

System.*out*.println(address1.getHostAddress());//获得域名所对应的地址

System.*out*.println(address1.getHostName());//获得域名名

} **catch** (UnknownHostException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

// byte [] b = new byte[]{125,88,(byte) 193,(byte) 178};

// try {

// InetAddress address2 = InetAddress.getByAddress("hao.360.cn",b);

// System.out.println(address2.getHostAddress());

// System.out.println(address2.getHostName());

// byte[] by = address2.getAddress();

// int [] a = new int[by.length];

// for (int i = 0; i < b.length; i++) {

// System.out.println(by[i]);

// a[i] = (int)by[i];

// System.out.println(a[i]);

// }

//

//

// } catch (UnknownHostException e) {

// // **TODO** Auto-generated catch block

// e.printStackTrace();

// }

/\*

\* 用ip字符串形式来代替域名

\*/

**try** {

InetAddress address3 = InetAddress.*getByName*("125.88.193.178");

System.*out*.println(address3.getHostAddress());

System.*out*.println(address3.getHostName());

} **catch** (UnknownHostException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

InetSocketAddress的用法：常用方法如下：get

**package** cn.com.inetAddress;

**import** java.net.InetAddress;

**import** java.net.InetSocketAddress;

**import** java.net.UnknownHostException;

**public** **class** InetSocketAddressTest {

// private InetSocketAddress address = new InetSocketAddress("www.taobao.com",7899);

**public** **static** **void** main(String[] args) {

InetSocketAddress address = **null**;

**try** {

address = **new** InetSocketAddress(InetAddress.*getByName*("www.taobao.com"),7899);

} **catch** (UnknownHostException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

System.*out*.println(address.getHostName());

System.*out*.println(address.getPort());

InetAddress add =(address.getAddress());//获得InetAddress对象

System.*out*.println(add.getHostName());

System.*out*.println(add.getHostAddress());

}

}

Url（uniform resource locator）的用法：

练1：

**package** cn.com.url;

**import** java.io.IOException;

**import** java.io.InputStream;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**public** **class** URLTest {

**public** **static** **void** main(String[] args) {

URL u=**null**;

**try** {

u = **new** URL("https://www.google.cn:80/webhp#aa?canhu=33");

} **catch** (MalformedURLException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

System.*out*.println(u.getHost());

System.*out*.println(u.getPort());

System.*out*.println(u.getFile());

System.*out*.println(u.getProtocol());

System.*out*.println(u.getQuery());

System.*out*.println(u.getRef());

System.*out*.println(u.getPath());

// InputStream is = u.openStream();

// byte[] b = new byte[1024];

// int len = -1;

// while ((len = is.read(b)) != -1) {

// System.out.println(new String(b, 0, len));

//

// }

// if (is != null) {

// is.close();

// }

//

// } catch (MalformedURLException e) {

// // **TODO** Auto-generated catch block

// e.printStackTrace();

// } catch (IOException e) {

// // **TODO** Auto-generated catch block

// e.printStackTrace();

// }

}

}

练2读出：相应域名所对应网页的html源码：

**package** cn.com.url;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.io.OutputStreamWriter;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**public** **class** URLTest2 {

/\*\*

\* **@param** args

\*/

**public** **static** **void** main(String[] args) {

URL u = **null**;

BufferedReader br = **null**;

BufferedWriter bw = **null**;

**try**{

u = **new** URL("https://www.baidu.com/");

br = **new** BufferedReader(**new** InputStreamReader(u.openStream()));

bw = **new** BufferedWriter(**new** OutputStreamWriter(**new** FileOutputStream("333.htm"),"utf-8"));//swithch流主意编码与解码。

String line = **null**;

**while**((line=br.readLine())!=**null**){

bw.append(line);

//bw.write(line);

bw.newLine();

//bw.flush();

//System.out.println(line);

}

}

**catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**if**(br!=**null**){

**try** {

br.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**if**(bw!=**null**){

**try** {

bw.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

Udp 编程：

Udp特点：是非面向连接（面向传输的）的，不安全，效率高。

//飞秋数据格式：version：时间：发送人：ip：32： 发送内容：

**package** cn.com.udp;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**import** java.net.SocketException;

**import** java.net.UnknownHostException;

/\*\*

\*

\* <p>author:adopted</p>

\* **@tiltle** FeiQiuUDP.java

\* <p>Company: ck</p>

\* <p>Description:用udp协议给飞秋发短信</p>

\* **@date**:2017-6-5

\*/

**public** **class** FeiQiuUDP {

**public** **static** **void** main(String[] args) **throws** IOException {

//创建udp服务

DatagramSocket socket = **new** DatagramSocket();

//准备数据

String data=*getData*("feiq你好！！！");

//创建数据包对象,并将数据放入包中

DatagramPacket packet= **new** DatagramPacket(data.getBytes(), data.getBytes().length, InetAddress.*getByName*("192.168.191.1"), 2425);

//调用udp服务,发送数据包

socket.send(packet);

//关闭资源

socket.close();

}

**private** **static** String getData(String str) {

StringBuffer sb = **new** StringBuffer();

sb.append("1.0:");

sb.append(System.*currentTimeMillis*()+":");

sb.append("你班长:");

sb.append("192.168.191.255:");

sb.append("32:");

sb.append(str);

**return** sb.toString();

}

}

Udp练习1：

**package** cn.com.udp;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetSocketAddress;

**import** java.net.SocketException;

**public** **class** UDPClient {

/\*\*

\* **@param** args

客户端：

\*/

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**try** {

DatagramSocket client = **new** DatagramSocket(5555);

//创建客户端对象：

String data = "sdffddfs";

**byte**[] b = data.getBytes();

//准备数据：

DatagramPacket packet = **new** DatagramPacket(b, b.length,**new** InetSocketAddress("localhost",8887));

//将数据打包

client.send(packet);

// 发送数据

client.close();

} **catch** (SocketException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

**package** cn.com.udp;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.SocketException;

**public** **class** UDPService {

/\*\*

\* **@param** args服务端

\*/

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**try** {

//创建服务端

DatagramSocket service = **new** DatagramSocket(8887);

//准备容器接收数据

**byte** [] container = **new** **byte** [1024];

DatagramPacket packet = **new** DatagramPacket(container, container.length);

//接受数据

service.receive(packet);

//解析数据

**byte** [] data = packet.getData();

**int** len = packet.getLength();

System.*out*.println(**new** String (data,0,len));

service.close();

} **catch** (SocketException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

练习2：

传输八大原始型数据：以传输double型为例

**package** cn.com.udp;

**import** java.io.ByteArrayInputStream;

**import** java.io.DataInputStream;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.SocketException;

**public** **class** UDPService1 {

/\*\*

\* **@param** args

服务端：

\*/

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**try** {

DatagramSocket server = **new** DatagramSocket(7890);

**byte** [] container = **new** **byte** [1024];

DatagramPacket dp = **new** DatagramPacket(container, container.length);

server.receive(dp);

**byte** [] data = dp.getData();

System.*out*.println(*getData*(data));

System.*out*.println(**new** String());

server.close();

} **catch** (SocketException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**public** **static** **double** getData(**byte** [] b){

DataInputStream dis = **null**;

dis = **new** DataInputStream(**new** ByteArrayInputStream(b));

**double** d=9;

**try** {

d = dis.readDouble();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**finally**{

**if**(dis!=**null**){

**try** {

dis.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

**return** d;

}

}

**package** cn.com.udp;

**import** java.io.ByteArrayOutputStream;

**import** java.io.DataOutputStream;

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetSocketAddress;

**import** java.net.SocketException;

客户端：

**public** **class** UDPClient1 {

**public** **static** **void** main(String[] args) {

**try** {

DatagramSocket client = **new** DatagramSocket(5675);

**double** d =12.5;

DatagramPacket packet = **new** DatagramPacket(*getbyte*(d), *getbyte*(d).length, **new** InetSocketAddress("localhost",7890));

client.send(packet);

client.close();

} **catch** (SocketException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**public** **static** **byte** [] getbyte(**double** d){

DataOutputStream dos = **null**;

ByteArrayOutputStream baos = **null**;

baos = **new** ByteArrayOutputStream();

dos = **new** DataOutputStream(baos);

**byte** [] data = **null**;

**try** {

dos.writeDouble(d);

data = baos.toByteArray();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**finally**{

**if**(baos!=**null**){

**try** {

baos.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**if**(dos!=**null**){

**try** {

dos.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

**return** data;

}

}

Tcp编程：

Tcp:是面向连接传输的，安全，效率较低。

Tcp 练习一：多发多收消息程序。

//接受线程：

**package** cn.com.tcp3;

**import** java.io.DataInputStream;

**import** java.io.IOException;

**import** java.net.Socket;

**public** **class** Recive **implements** Runnable{

**private** DataInputStream dis;

**private** **boolean** flag = **true**;

**public** Recive (Socket socket) **throws** IOException{

**try** {

dis = **new** DataInputStream(socket.getInputStream());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

dis.close();

}

}

**public** String recive(){

String inf = "";

**try** {

inf = dis.readUTF();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

**try** {

dis.close();

} **catch** (IOException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

}

**return** inf;

}

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

**while**(flag){

System.*out*.println(recive());

}

}

}

//发送线程：

**package** cn.com.tcp3;

**import** java.io.BufferedReader;

**import** java.io.DataOutputStream;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.net.Socket;

**public** **class** Send **implements** Runnable{

**private** BufferedReader br;

**private** DataOutputStream dos;

String msg;

**private** **boolean** flag = **true**;

**public** Send(){

}

**public** Send(Socket socket){

br = **new** BufferedReader(**new** InputStreamReader(System.*in*));

**try** {

dos = **new** DataOutputStream(socket.getOutputStream());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**try** {

dos.close();

} **catch** (IOException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

}

}

**public** String getConsole(){

**try** {

msg = br.readLine();

**return** msg;

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

**return** "";

}

}

**public** **void** send(){

msg = getConsole();

**try** {

**if**(msg!=**null**&&!msg.equals("")){

dos.writeUTF(msg);

dos.flush();

}

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

**if**(dos!=**null**){

**try** {

dos.close();

} **catch** (IOException ew) {

// **TODO** wAuto-generated catch block

ew.printStackTrace();

}

}

}

}

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

**while**(flag){

send();

}

}

}

客户端；

**package** cn.com.tcp3;

**import** java.io.IOException;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**public** **class** TCPClient {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**try** {

Socket client = **new** Socket("localhost",7758);

**new** Thread(**new** Send(client)).start();

**new** Thread(**new** Recive(client)).start();

} **catch** (UnknownHostException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

服务端：

**package** cn.com.tcp3;

**import** java.io.DataInputStream;

**import** java.io.DataOutputStream;

**import** java.io.IOException;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** TCPServer {

**public** **static** **void** main(String[] args) {

DataInputStream dis = **null**;

DataOutputStream dos = **null**;

**try** {

ServerSocket server = **new** ServerSocket(7758);

Socket socket = server.accept();

dis = **new** DataInputStream(socket.getInputStream());

dos = **new** DataOutputStream(socket.getOutputStream());

**while**(**true**){

String s1 = dis.readUTF();

**if**(s1.equals("bye")){

**break**;

}

System.*out*.println(s1);

dos.writeUTF(s1+"iii");

dos.flush();

}

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

练习2 ：客户端-服务端-客户端编程。

接受线程：

**package** cn.com.tcp5;

**import** java.io.DataInputStream;

**import** java.io.IOException;

**import** java.net.Socket;

**public** **class** Recive **implements** Runnable{

**private** DataInputStream dis;

**private** **boolean** flag = **true**;

**public** Recive (Socket socket) **throws** IOException{

**try** {

dis = **new** DataInputStream(socket.getInputStream());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

CloseUtil.*closeAll*(dis);

}

}

**public** String recive(){

String inf = "";

**try** {

inf = dis.readUTF();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

CloseUtil.*closeAll*(dis);

}

**return** inf;

}

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

**while**(flag){

System.*out*.println(recive());

}

}

}

发送线程：

**package** cn.com.tcp5;

**import** java.io.BufferedReader;

**import** java.io.DataOutputStream;

**import** java.io.IOException;

**import** java.io.InputStreamReader;

**import** java.net.Socket;

**public** **class** Send **implements** Runnable{

**private** BufferedReader br;

**private** DataOutputStream dos;

**private** **boolean** flag = **true**;

**public** Send(){

}

**public** Send(Socket socket){

br = **new** BufferedReader(**new** InputStreamReader(System.*in*));

**try** {

dos = **new** DataOutputStream(socket.getOutputStream());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

CloseUtil.*closeAll*(dos);

}

}

**public** String getConsole(){

String msg = "";

**try** {

msg = br.readLine();

**return** msg;

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

**return** "";

}

}

**public** **void** send(){

String msg = getConsole();

**try** {

**if**(msg!=**null**&&!msg.equals("")){

dos.writeUTF(msg);

dos.flush();

}

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

flag = **false**;

CloseUtil.*closeAll*(dos);

}

}

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

**while**(flag){

send();

}

}

}

**package** cn.com.tcp5;

**import** java.io.DataInputStream;

**import** java.io.DataOutputStream;

**import** java.io.IOException;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**import** java.util.ArrayList;

**public** **class** TCPServer {

ArrayList<MyChannel> all = **new** ArrayList<TCPServer.MyChannel>();

Socket socket = **null**;

ServerSocket server = **null**;

**public** **static** **void** main(String[] args) {

TCPServer serve = **new** TCPServer();

serve.start1();

}

**public** **void** start1() {

**try** {

server = **new** ServerSocket(7757);

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**while** (**true**) {

**try** {

socket = server.accept();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

MyChannel channel = **new** MyChannel(socket);

all.add(channel);

**new** Thread(channel).start();

}

}

**private** **class** MyChannel **implements** Runnable {

DataInputStream dis = **null**;

DataOutputStream dos = **null**;

**boolean** isRunning = **true**;

MyChannel(Socket socket) {

**try** {

dis = **new** DataInputStream(socket.getInputStream());

dos = **new** DataOutputStream(socket.getOutputStream());

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

isRunning = **false**;

CloseUtil.*closeAll*(dos, dis);

all.remove(**this**);

}

}

**private** String recive() {

String inf = "";

**try** {

inf = dis.readUTF();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

isRunning = **false**;

CloseUtil.*closeAll*(dis);

all.remove(**this**);

}

**return** inf;

}

**private** **void** send(String inf) {

**if** (inf != **null** && !inf.equals("")) {

**try** {

dos.writeUTF(inf);

dos.flush();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

isRunning = **false**;

CloseUtil.*closeAll*(dos);

}

}

}

**private** **void** sendOthers(){

String inf = recive();

**for**(MyChannel others : all){

**if**(others==**this**){

**continue**;

}

others.send(inf);

}

}

@Override

**public** **void** run() {

// **TODO** Auto-generated method stub

**try**{

**while**(isRunning){

sendOthers();

}

} **catch**(Exception e){

e.printStackTrace();

}

**finally**{

CloseUtil.*closeAll*(dis,dos);

**try** {

socket.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

}

**package** cn.com.tcp5;

**import** java.io.IOException;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**public** **class** TCPClient {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**try** {

Socket client = **new** Socket("localhost",7757);

**new** Thread(**new** Send(client)).start();

**new** Thread(**new** Recive(client)).start();

} **catch** (UnknownHostException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

**package** cn.com.tcp5;

**import** java.io.Closeable;

**import** java.io.IOException;

**public** **class** CloseUtil {

**public** **static** **void** closeAll(Closeable ...io ){

**for**(Closeable temps: io){

**if**(temps!=**null**){

**try** {

temps.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

}

**package** cn.com.tcp5;

**import** java.io.Closeable;

**import** java.io.IOException;

**public** **class** CloseUtil {

**public** **static** **void** closeAll(Closeable ...io ){

**for**(Closeable temps: io){

**if**(temps!=**null**){

**try** {

temps.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

}

}