# UDP编程

DatagramSocket与DatagramPacket

建立发送端，接收端。

建立数据包。

调用Socket的发送接收方法。

关闭Socket。

发送端与接收端是两个独立的运行程序。

## UDP传输-发送端思路

1:建立udp的socket服务

2:将要发送的数据封装成数据包

3:通过udp的socket服务,将数据包发送出

4:关闭资源

/\*

\* UDP协议发送数据：

\* A:创建发送端Socket对象

\* B:创建数据，并把数据打包

\* C:调用Socket对象的发送方法发送数据包

\* D:释放资源

\*/

**public** **class** SendDome {

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

// 创建发送端Socket对象

// DatagramSocket()

DatagramSocket ds = **new** DatagramSocket();

// 创建数据，并把数据打包

// DatagramPacket(byte[] buf, int length, InetAddress address, int port)

// 创建数据

**byte**[] bys = "hello,udp,我来了".getBytes();

// 长度

**int** length = bys.length;

// IP地址对象

InetAddress address = InetAddress.*getByName*("192.168.1.2");

// 端口

**int** port = 10086;

DatagramPacket dp = **new** DatagramPacket(bys, length, address, port);

// 调用Socket对象的发送方法发送数据包

// public void send(DatagramPacket p)

ds.send(dp);

// 释放资源

ds.close();

}

}

## UDP传输-接收端思路

1:建立udp的socket服务.

2:通过receive方法接收数据

3:将收到的数据存储到数据包对象中

4:通过数据包对象的功能来完成对接收到数据进行解析.

5:可以对资源进行关闭

/\*

\* UDP协议发送数据：

\* A:创建发送端Socket对象

\* B:创建数据，并把数据打包

\* C:调用Socket对象的发送方法发送数据包

\* D:释放资源

\*/

**public** **class** SendDome {

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

// 创建发送端Socket对象

// DatagramSocket()

DatagramSocket ds = **new** DatagramSocket();

// 创建数据，并把数据打包

// DatagramPacket(byte[] buf, int length, InetAddress address, int port)

// 创建数据

**byte**[] bys = "hello,udp,我来了".getBytes();

// 长度

**int** length = bys.length;

// IP地址对象

InetAddress address = InetAddress.*getByName*("192.168.1.2");

// 端口

**int** port = 10086;

DatagramPacket dp = **new** DatagramPacket(bys, length, address, port);

// 调用Socket对象的发送方法发送数据包

// public void send(DatagramPacket p)

ds.send(dp);

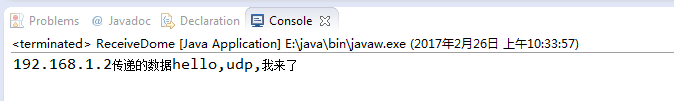
// 释放资源

ds.close();

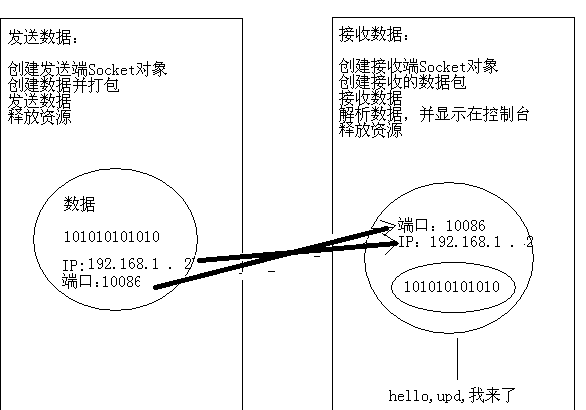
}

}

先运行接受端，在运行发送端



## 运行图解



## UDP案例

从键盘录入数据进行发送，如果输入的是886那么客户端就结束输入数据。

这个时候完全可以把发送端代码发给大家了，我把接收端代码开启，大家就可以实现聊天了，但是，大家都要看我们的屏幕，即使我把接收端发给大家也是一样的，如何改进呢，使用广播地址即可。

最后，把刚才发送和接收程序分别用线程进行封装，完成一个UDP的聊天程序。

**public** **class** SendThread **implements** Runnable {

**private** DatagramSocket ds ;

**public** SendThread(DatagramSocket ds) {

**this**.ds = ds;

}

@Override

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

//封装键盘录入

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

String line = **null**;

**try** {

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

**if** ("886".equals(line)) {

**break**;

}

// 创建数据并打包

**byte**[] bys = line.getBytes();

DatagramPacket dp = **new** DatagramPacket(bys, bys.length,

InetAddress.*getByName*("192.168.1.255"), 10086);

// 发送数据

ds.send(dp);

}

} **catch** (IOException e) {

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

e.printStackTrace();

}**finally**{

// 释放资源

ds.close();

}

}

}

**public** **class** ReceiveThread **implements** Runnable {

**private** DatagramSocket ds ;

**public** ReceiveThread(DatagramSocket ds) {

**this**.ds = ds;

}

@Override

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

**try** {

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

// 创建一个包裹

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

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

// 接收数据

ds.receive(dp);

// 解析数据

String ip = dp.getAddress().getHostAddress();

String s = **new** String(dp.getData(), 0, dp.getLength());

System.*out*.println("from " + ip + " data is : " + s);

}

} **catch** (IOException e) {

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

e.printStackTrace();

}

}

}

**public** **class** ChatRoom {

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

DatagramSocket deSend = **new** DatagramSocket();

DatagramSocket deReceive = **new** DatagramSocket(10086);

SendThread st = **new** SendThread(deSend);

ReceiveThread rt = **new** ReceiveThread(deReceive);

Thread t1 = **new** Thread(st);

Thread t2 = **new** Thread(rt);

t1.start();

t2.start();

}

}