



**课 程 项 目 论 文**

课程名称\_\_\_\_\_高级计算机网络

题目名称\_\_\_\_\_ UDP聊天室 \_ \_

学生学院\_\_\_\_\_ 计算机学院\_\_\_\_\_\_

专业班级\_\_\_\_\_\_网络工程3班\_\_\_

学 号\_\_\_\_\_\_ 3113006541\_\_ \_\_

学生姓名\_\_\_\_\_ 朱淼鑫\_\_\_ \_

指导教师\_\_\_\_\_\_ 何翠红\_\_\_\_\_\_

2016 年11月5日

**目 录**

[1) 摘要 - 1 -](#_Toc466238091)

[2) 关键词 - 1 -](#_Toc466238092)

[3) 引言 - 1 -](#_Toc466238093)

[4) 技术分析与设计 - 1 -](#_Toc466238094)

[5) 系统的实现 - 2 -](#_Toc466238095)

[服务器端： - 3 -](#_Toc466238096)

[客户端： - 4 -](#_Toc466238097)

[6) 系统部署与运行 - 5 -](#_Toc466238098)

[7) 总结 - 8 -](#_Toc466238099)

## 摘要

UDP协议是一种简洁、高效，被广泛使用的协议。本项目通过JAVA的socket编程实现了使用UDP协议的网络通信程序，然后通过所实现的程序进行网络通信试验。

## 关键词

用户数据报协议；Socket套接字；通信；

## 引言

1. 目的

分析总结UDP协议的优缺点,掌握UDP协议面向无链接的通信方式，掌握UDP协议的工作原理及应用方法。

1. 内容和要求

1、实现UDP通信。

2、采用多线程技术，实现多用户同时在线。

3、实现群聊功能，一个人发消息多人可收到。

## 技术分析与设计

1. UDP协议分析：

用户数据报协议（UDP）是 OSI 参考模型中一种无连接的传输层协议，提供面向事务的简单不可靠信息传送服务。UDP 协议适用端口分别运行在同一台设备上的多个应用程序。

UDP协议有如下的特点：

1、UDP传送数据前并不与对方建立连接，即UDP是无连接的，在传输数据前，发送方和接收方相互交换信息使双方同步。

2、UDP不对收到的数据进行排序，在UDP报文的首部中并没有关于数据顺序的信息（如TCP所采用的序号），而且报文不一定按顺序到达的，所以接收端无从排起。

3、UDP对接收到的数据报不发送确认信号，发送端不知道数据是否被正确接收，也不会重发数据。

4、UDP传送数据较TCP快速，系统开销也少。

从以上特点可知，UDP提供的是无连接的、不可靠的数据传送方式，是一种尽力而为的数据交付服务。

1. 基于UDP协议的通信实现：

在采用UDP协议进行的网络通信方式中，通信机制是发送独立的数据报（Datagram）到接受目的地。基于UDP的通信和基于TCP的通信不同，基于UDP的信息传递更快，但不提供可靠的保证。使用UDP传输数据时，用户无法知道数据能否正确主机，也不能确定到达目的地的顺序是否和发送顺序相同。虽然UDP是一种不可靠的协议，但如果需要较快地传输信息， 并能容忍小的错误，可以考虑使用UDP。

基于UDP通信的基本模式如下：

1. 将数据打包（称为数据包），然后将数据包发往目的地。
2. 接受别人发来的数据包，然后查看数据包。

下面是总结的UDP程序步骤：

发送数据包：

1. 使用DatagramSocket()创建一个数据包套接字。
2. 使用DatagramPacket(byte[] buf, int length, InetAddress address, int port)创建要发送的数据包。
3. 使用DatagramSocket类的send（）方法发送数据包。

接受数据包：

1. 使用DatagramSocket(int port)创建数据包套接字，绑定到指定端口。
2. 使用DatagramPacket(byte[] buf, int length)创建字节数组来接受数据包。
3. 使用DatagramSocket类的receive()方法接受UDP包。

为了方便管理进入聊天室的用户，本系统采用一个服务器端和多个客户端的聊天方式来实现。在C/S模式中，数据的分发采用专门的服务器，多个客户端都从此服务器获取数据。这种模式的优点是：数据的一致性容易控制，系统也容易管理。

## 系统的实现



### 服务器端：

ChatServer类：

DatagramSocket DS1 = **null**;

DatagramSocket DS2 = **null**;

MulticastSocket McS = **null**;

InetAddress iaddress = **null**;

**int** mPort = 2333;

**int** uPort1 = 6666;

**int** uPort2 = 8888;

Vector<Client> clients = **new** Vector<Client>();

（1）服务端器端创建了两个数据包套接字，绑定到了两个端口。其中DS1在主程序监听和接收用户群聊的信息，DS2在私聊线程监听和接收用户私聊的信息。

（2）多播套接字McS绑定到端口mPort，并将McS加入到多播组地址（"224.255.10.0"）。在主程序监听和接收到用户群聊的信息后，提取源ip地址，同clients集合中的Client比较，若没有相同的ip,既为新用户，将它加入集合。启动广播线程，用McS套接字将信息广播给多播组的成员。

（3）私聊信息的模式为“\*\*\*@@\*\*\*”，@@前的信息私聊对象的用户名，@@后面是信息。私聊进程接受到信息后，提取出用户名和信息，在集合clients中找到私聊对象ip,即目的地址，分别使用源地址和目的地址创建发送数据包，使用DS2套接字发送数据包给私聊的两个用户。

### 客户端：

ChatClient类：

DatagramSocket DS = **null**;

MulticastSocket McS = **null**;

InetAddress iaddress = **null**;

**int** mPort = 2333;

**int** uPort = 6666;

String ServerIP = "127.0.0.1";

Choice choice = **new** Choice();

TextField tfTxt = **new** TextField(65); //编辑窗口

TextArea taContent = **new** TextArea(20,80);//显示窗口

（1）客户端群聊程序创建了一个数据包套接字DS和一个多播套接字McS，分别绑定到uPort端口和mPort端口。 McS套接字加入到多播组地址（"224.255.10.0"）。

（2）在文本框tfTxt上添加监听动作，一旦点击发送或按Enter键，执行动作，将信息发送到SeverIP即服务器ip的uport端口上。此信息为群聊信息。

（3）接收线程上，使用McS套接字监听和接收多播信息，并将信息显示到文本框taContent上。

（4）choice上添加监听动作，一旦选择“私聊”，打开私聊程序。

PrivateChat类：

DatagramSocket DS = **null**;

**int** uPort = 8888;

String ServerIP = "127.0.0.1";

TextField tfTxt1 = **new** TextField(6);

TextField tfTxt2 = **new** TextField(35); //编辑窗口

TextArea taContent = **new** TextArea(15,60);//显示窗口

1. 客户端私聊程序创建了一个数据包套接字DS，绑定到端口uPort上。
2. 在文本框tfTxt2上添加监听动作，一旦点击发送或按Enter键，执行动作，将tfTxt1上的私聊对象加“@@”，再加tfTxt2上的私聊信息封装到数据包发送到SeverIP即服务器ip的uport端口上。此信息为私聊聊信息。
3. 接收线程上，使用DS套接字监听和接收信息，并将信息显示到文本框taContent上。

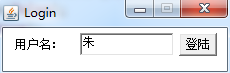
## 系统部署与运行

因为采用的是C/S模式，所以系统部署需要用到至少两台计算机以上，若要群聊，则需要三台计算机以上。

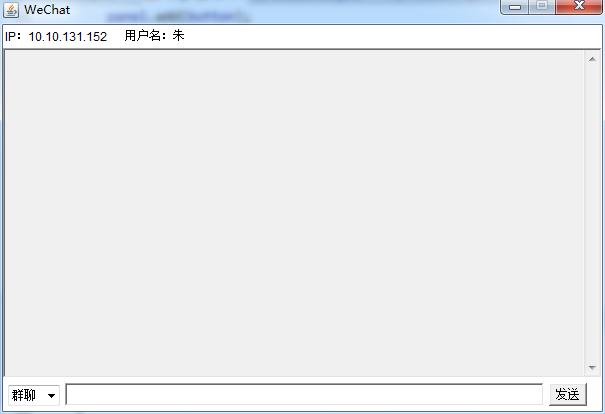
（1）用其中一台计算机运行服务器程序；

（2）其他几台进行测试的计算机，将服务器地址（SeverIP）设置为运行服务器程序的计算机的IP地址;

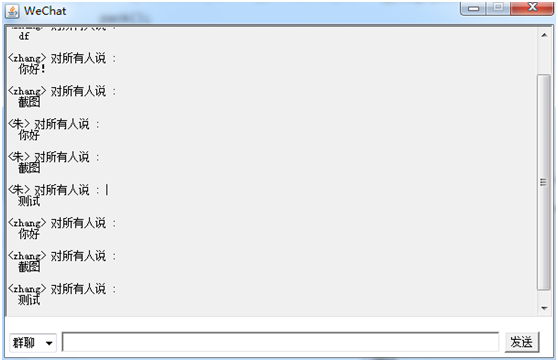
1. 运行Login.java,输入用户名并登陆：



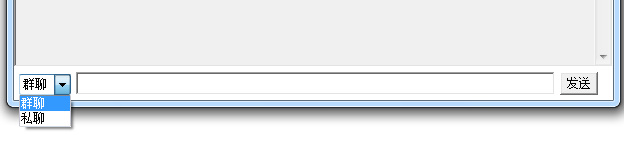
1. 进入群聊界面：



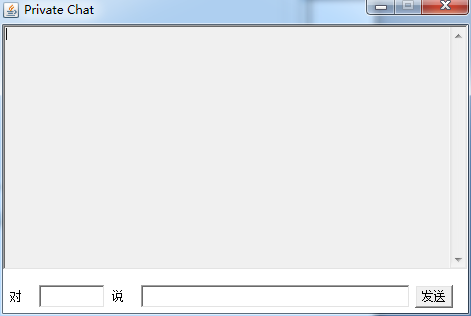
聊天：



1. 选择私聊：



1. 进入私聊界面：



私聊：



## 总结

这次的课程设计虽然原理较为简单，利用java提供的java.net.\*网络类可以很容易实现的UDP聊天功能，但是因为采取的是C/S模式，数据的接受和转发因为要进行源地址和目的地址的转化，所以变得有点复杂。刚开始写程序时，自己都没搞清楚要用哪个套接字接收信息，所以写程序也有乱，后来一点一点写出来，才渐渐理清思路。

期间也遇到不少问题，比如，因为测试要用到多台计算机，比较麻烦；还有在自己的计算机上运行成功，但到其他计算机上就运行不了，无法接通之类的，后来发现是因为360安全卫士和计算机本身的拦火墙的拦截，所以无法发送信息，而不是程序本身的问题。还有因为数据报的编码方式，导致显示乱码。

虽然遇到很多问题，但是都耐心的一一解决了。通过这次课程设计，我对应用层的通信方式，TCP和UDP都有了更加深刻的了解。