Socket 客户端-服务器（C-S）通信实验

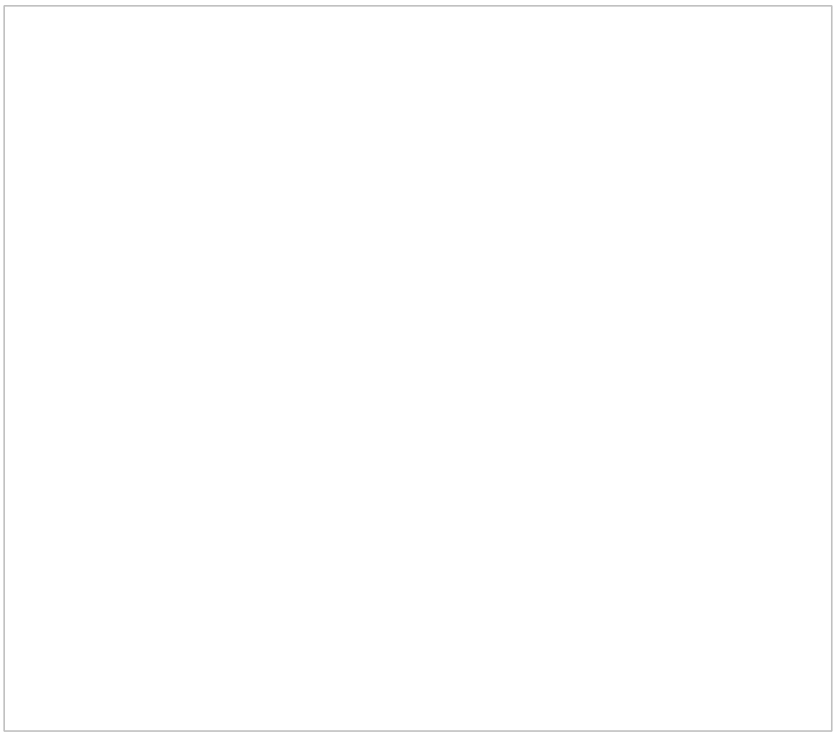
原创

含笑无情 于 2016-04-24 09:19:38 发布 4652 收藏 7
分类专栏： 计算机网络编程 文章标签： socket 通信 网络编程 java

学习计算机网络编程也有一段时间了，对这段时间学习的东西做一个小总结吧，主要是基于socket，实现客户端和服务器的通信，编程用java语言。具体的实验要求如下：

设计程序，分别构建通信的两端:服务器端和客户端应用程序,套接字类型为面向连接的Socket,自己构建双方的应答模式，实现双方的数据的发送和接收（S发给C，C发给S）。

服务端程序能响应单个或任意多个客户端连接请求；服务端能向单个客户发送消息，支持群发消息给所有客户端；
通信的双方具备异常响应功能，包括对方异常退出的处理。如果客户端退出，服务器有响应；反之亦然。
Client-Server通信效果的图片说明：



编程实现的思路：在服务器端，首先要启动一条线程用于监听某个指定端口（比如：8000），并且还要再开一条线程用于接收消息，客户端尝试连接该端口（8000），如果成功连接则会返回一个Socket类的实例对象(Socket socket = serversocket.accept();)，很显然服务器端便应该保存有一个客户列表（比如可以用：ArrayList），使得服务器可以发消息给某个指定的客户端。而客户端在连接服务器之后也应该启动一条线程用于接收消息。当某个客户端进来时，便发送Login消息给服务器，服务器将此消息广播发给当前在线的所有用户，当某个客户端退出时，发送Logout消息给服务器，服务器将此消息广播发给当前在线的所有用户，当客户端之间进行通信时，会发送Talk消息给服务器，服务器再将此消息转发给指定的客户端，也就是所有的通信都是通过服务器进行转发的。

主要的代码实现如下：

服务器端：

```
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.InetAddress;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.UnknownHostException;
import java.util.ArrayList;
import javax.swing.JScrollBar;



public class ChatServer extends javax.swing.JFrame {
    ArrayList<User> list = new ArrayList<User>();
    static String local;
    int port = 8000;
    boolean isNomalexit = false;
    Mythread  jtThread;


    public ChatServer() {
        initComponents();
        this.setTitle("服务器");
        this.setResizable(false);
        this.setLocationRelativeTo(null);
         this.addWindowListener(new WindowAdapter() {
                public void windowClosing(WindowEvent e) {
                    super.windowClosing(e);
                    dispose();

                }
            });
    }

 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) { } //开始监听按钮的事件
```

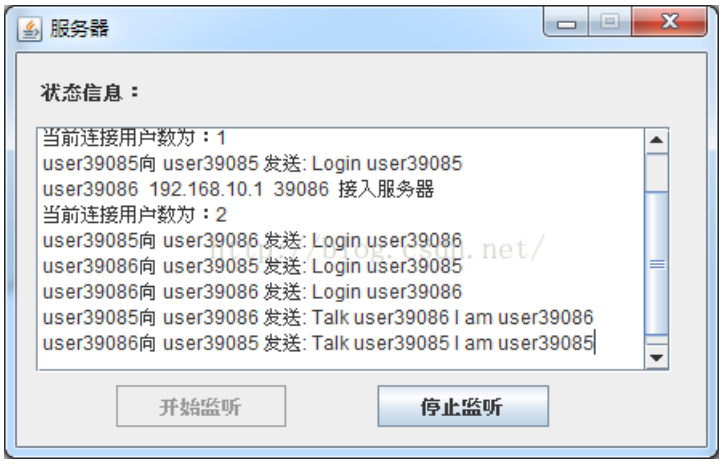
```
 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) { } //停止监听按钮的事件

class Mythread extends Thread{ }                //负责监听端口8000的线程

class ClientThread extends Thread{ }          //负责接收消息的线程

public void sendtoclient(User user,String message){ }     //发送消息message给用户，user为发送者

public void sendtoallclient(User user,String message){ }  //发送消息message给所有用户，user发送者

public void removeuser(User user){ }

 public static void main(String args[]) throws UnknownHostException {
            local = InetAddress.getLocalHost().getHostAddress();
            ChatServer frame = new ChatServer();
            frame.setVisible(true);

    }
```

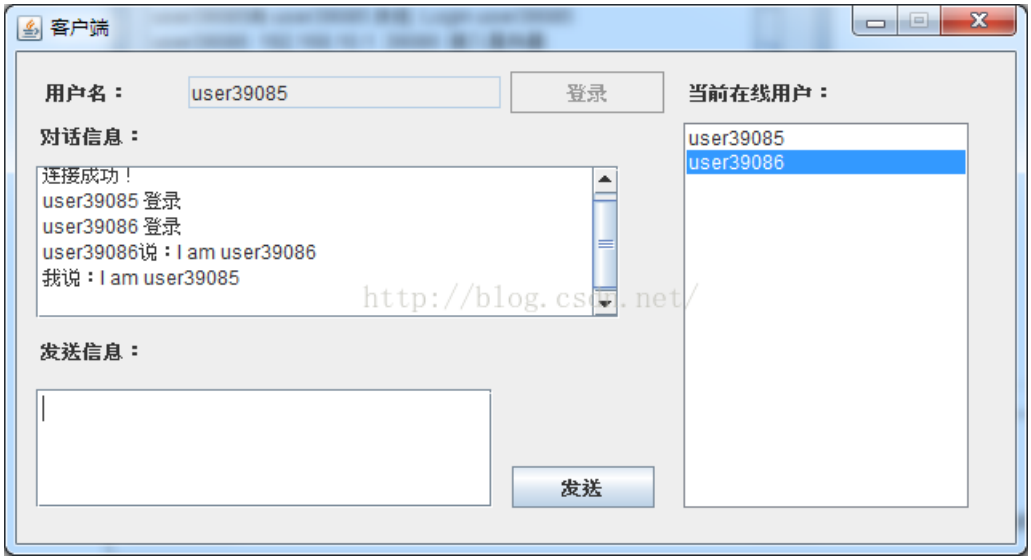
客户端：

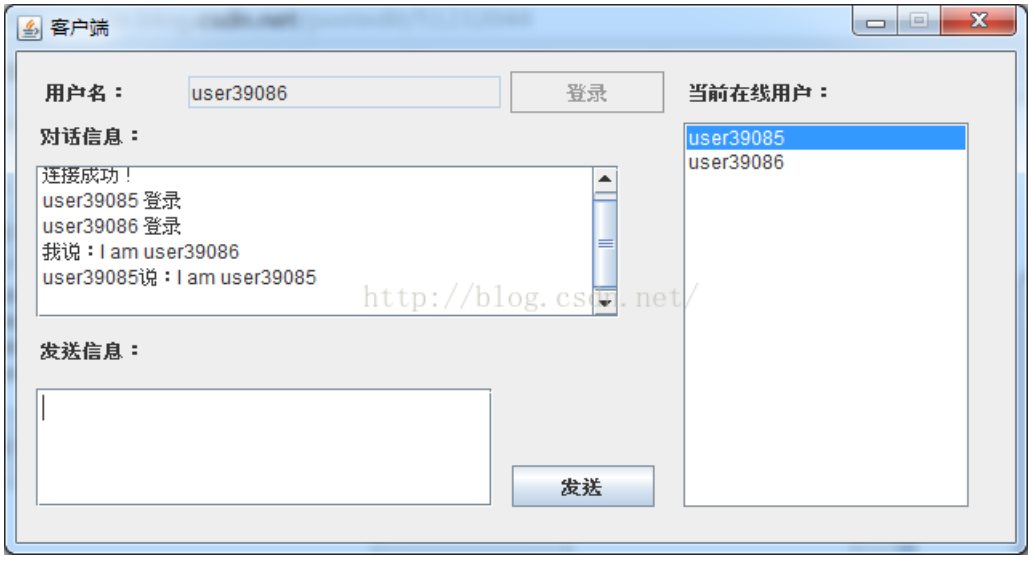
```
import java.awt.event.KeyEvent;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.io.DataInputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.InetAddress;
import java.net.Socket;
import java.util.ArrayList;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
import javax.swing.JScrollBar;

public class ChatClient extends javax.swing.JFrame {
       Socket s;
       boolean isExit = false;
       DataInputStream in;
       PrintStream out;
       ArrayList<String> list =new  ArrayList<String>();
       static String name;
       ReceivemessageThread thread;

    public ChatClient() {
        initComponents();
        this.setTitle("客户端");
        this.setResizable(false);
        this.setLocationRelativeTo(null);
         this.addWindowListener(new WindowAdapter() {
                public void windowClosing(WindowEvent e) {
                  if(s!=null){
                      sendmessage("Logout " + name);
                      try {
                          isExit = true;
                          in.close();
                           out.close();
                           s.close();
                      } catch (IOException ex) {
                          Logger.getLogger(ChatClient.class.getName()).log(Level.SEVERE, null, ex);
                      }
                  }
                }
            });
    }

 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt){ } //登录按钮的事件

 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt){ } //发送按钮的事件
```

```
class ReceivemessageThread extends Thread{ } // 接收消息的线程

public void sendmessage(String message){ }  //发送消息

public void removeuser(String name){ } //移除在线列表上的用户名

public static void main(String args[]) {
        ChatClient frame = new ChatClient();
        frame.setVisible(true);
    }
```

运行演示：







在做完这个程序之后，对于网络编程也算是基本上入门了，后续还会有很多的通信程序，因为这门课的要求就是做一个类似QQ的通信程序，写下这篇博客也算是记录自己的学习进程。