```java
package net108.chat;

import java.io.*;
import net108.util.*;


/**
 * Represents the client of a simple chat system, accepts user input
 * that is sent to the server to be broadcast to any other chat
 * clients currently running.
 */
public class Client extends net108.util.Client
{
    /**
     * Create a client that connects to a server already running on
     * given hostName.
     */
    public Client (String hostName, int port)
    {
        super(hostName, port);
        addInputHandler(new net108.util.PrintInputHandler());
    }

    /**
     * A generic read loop for client.
     */
    public void chatLoop ()
    {
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in))
;
        while (true)
        {
            System.out.print(">> ");
            System.out.flush();
            try
            {
                send(in.readLine());
            }
            catch (IOException e)
            {}
        }
    }

    /**
     * Starts the client up.
     */
    public static void main (String[] args)
    {
        String hostname = Constants.HOST;
        if (args.length > 0)
        {
            hostname = args[0];
        }

        new Client(hostname, Constants.PORT).chatLoop();
    }
}
```

```java
package net108.chat;

import java.util.Random;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import net108.util.*;


/**
 * Simple GUI for Chat/Client
 * @author Robert Duvall
 * @author Owen Astrachan (modified on 3/29/2004)
 */
public class ClientGUI extends JFrame implements InputHandler
{
    private JTextField myInput;
    private JTextArea myInputHistory;
    private JTextArea myOutput;
    private Client myClient;
    private String myName;

    public ClientGUI (String hostName, int port, String clientName)
    {
        myClient = new Client(hostName, port);
        myClient.addInputHandler(this);
        myName = clientName;

        getContentPane().add(makeInputPanel(), BorderLayout.NORTH);
        getContentPane().add(makeHistoryPanel(), BorderLayout.CENTER);
        getContentPane().add(makeOutputPanel(), BorderLayout.SOUTH);

        setTitle("108/Chat "+myName);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        pack();
        show();
    }

    public void handleInput (Object o)
    {
        myOutput.append(o.toString() + "\n");
    }

    private JComponent makeInputPanel ()
    {
        JPanel top = new JPanel();
        top.add(makeTextField(30, new BroadcastInput(this)));
        top.add(makeButton("Send", new BroadcastInput(this)));
        top.add(makeButton("Clear", new ActionListener()
            {
                public void actionPerformed(ActionEvent e)
                {
                    myInput.setText("");
                    myInputHistory.setText("");
                    myOutput.setText("");
                }
            }));
        return top;
    }

    private JComponent makeHistoryPanel ()
    {
        myInputHistory = makeTextArea(10, 40);
        return new JScrollPane(myInputHistory);
    }

    private JComponent makeOutputPanel ()
    {
        myOutput = makeTextArea(10, 40);
```



```java
        return new JScrollPane(myOutput);
    }

    private JButton makeButton (String label, ActionListener action)
    {
        JButton button = new JButton(label);
        button.addActionListener(action);
        return button;
    }

    private JTextField makeTextField (int numCharsWide, ActionListener action)
    {
        myInput = new JTextField(numCharsWide);
        myInput.addActionListener(action);
        return myInput;
    }

    private JTextArea makeTextArea (int rows, int columns)
    {
        JTextArea textArea = new JTextArea(rows, columns);
        textArea.setEditable(false);
        return textArea;
    }


    private class BroadcastInput implements ActionListener
    {
        ClientGUI myTarget;

        public BroadcastInput (ClientGUI target)
        {
            myTarget = target;
        }

        public void actionPerformed(ActionEvent e)
        {
            String message = myTarget.myInput.getText();
            message = myName + ">> " + message;
            myTarget.myInput.setText("");
            myTarget.myClient.send(message);
            myTarget.myInputHistory.append(message + "\n");
        }
    }

    public static String[] names = {
        "Fred", "Wilma", "Barney", "Betty"
    };
    public static int ourCount = 0;

    public static void main (String[] args)
    {
        Random rando = new Random();
        String hostname = Constants.HOST;
        if (args.length > 0)
        {
            hostname = args[0];
        }

        new ClientGUI(hostname, Constants.PORT,
                      names[rando.nextInt(names.length)]);
    }
}
```

```java
package net108.chat;


public interface Constants
{
    int PORT = 12345;
    String HOST = "localhost";
}
```

```java
package net108.chat;


/**
 * Represents
 */
public class Server extends net108.util.Server
{
    /**
     * Start the server on a default port.
     */
    public Server ()
    {
        super(Constants.PORT);
    }

    /**
     * Creates a server that simply listens on a given port and
     * broadcasts any messages it receives to all connected clients.
     */
    public static void main (String[] args)
    {
        net108.util.Debug.turnOn();
        new Server();
    }
}
```