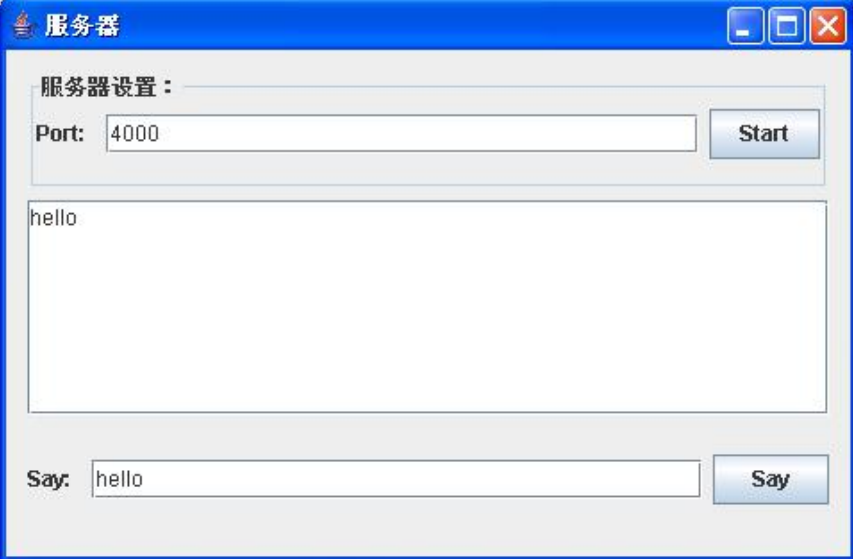


# 《Java 技术》实验报告

实验三：

2020 年 6 月 3 日

学院	软件学院	专业班级	软件工程 18-4 班	姓名	刘铭源		成绩	
课程名称	Java 技术	实验项目名称	基于 GUI 的网络通信设计				指导教师	
教师评语	<div>教师签名：_____ 年 月 日</div>							
<div><b>一、实验目的</b><ol style="list-style-type: none"><li>1. 掌握 Java 中 GUI 程序的编写，包括事件监听机制。</li><li>2. 掌握 Java 的网络通信编程，ServerSocket, Socket 类的使用。</li><li>3. 掌握 Java 中多线程的编程，Thread 类，Runnable 接口的使用。</li><li>4. 掌握用面向对象的方法分析和解决复杂问题。</li></ol><b>二、实验原理</b><p>1 编写程序完成以下功能：</p><ol style="list-style-type: none"><li>1. 设计一个基于 GUI 的客户-服务器的通信应用程序，如图 1，图 2 所示。</li></ol></div> <div></div> <p>图 1 Socket 通信服务器端界面</p>								

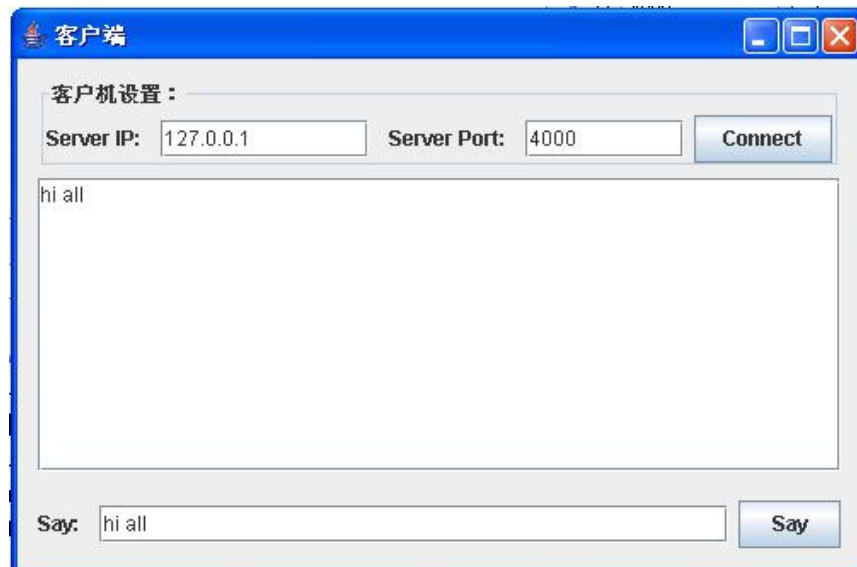


图 2 Socket 通信客户端界面

2. 图 1 为 Socket 通信服务器端界面，点击该界面中的【Start】按钮，启动服务器监听服务（在图 1 界面中间的多行文本区域显示“Server starting...”字样）。图 2 为 Socket 通信客户端界面，点击该界面中的【Connect】按钮与服务器建立链接，并在图 2 所示界面中间的多行文本区域显示“Connect to server...”字样，当服务器端监听到客户端的连接后，在图 1 界面中间的多行文本区域追加一行“Client connected...”字样，并与客户端建立 Socket 连接。

3. 当图 1 所示的服务器端和图 2 所示的客户机端建立 Socket 连接后，编程实现服务端、客户端之间的“单向通信”：在客户端的输入界面发送消息，在服务端接收该消息，并将接收到对方的数据追加显示在文本框中。

### 三、使用硬件、软件环境

Windows10, 内存 8g, 硬盘 1TB, JDK1.8, eclipse

### 四、实验过程、步骤及原始记录(算法、原程序、测试结果，分析等)

客户端：

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.net.Socket;

public class Client {
    private static Socket socket = null;
    public static void main(String[] args){
        JFrame.setDefaultLookAndFeelDecorated(true);
        JFrame frame = new JFrame("客户端");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setResizable(false);
        frame.setBounds(300, 200, 500, 300);
```

```

frame.setLayout(new BorderLayout(10, 0));

JPanel northPanel = new JPanel();
JLabel ipLabel = new JLabel("Server Ip: ");
JTextField ipField = new JTextField(10);
JLabel portLabel = new JLabel("Server Port: ");
JTextField portField = new JTextField(6);
northPanel.setBorder(BorderFactory.createTitledBorder("客户机设置: "));
JButton connectBtn = new JButton("Connect");
northPanel.add(ipLabel);
northPanel.add(ipField);
northPanel.add(portLabel);
northPanel.add(portField);
northPanel.add(connectBtn);

JPanel centerPanel = new JPanel();
JTextArea area = new JTextArea(10, 35);
area.setMargin(new Insets(10, 10, 10, 10));
area.setEnabled(false);
centerPanel.add(area);

JPanel southPanel = new JPanel();
JLabel sayLabel = new JLabel("Say: ");
JTextField sayField = new JTextField(30);
JButton sayBtn = new JButton("Say");
southPanel.add(sayLabel);
southPanel.add(sayField);
southPanel.add(sayBtn);

frame.add(northPanel, BorderLayout.NORTH);
frame.add(centerPanel, BorderLayout.CENTER);
frame.add(southPanel, BorderLayout.SOUTH);

frame.setVisible(true);

connectBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        area.append("Connect to server...\n");
        try {
            socket = new Socket(ipField.getText(),
Integer.parseInt(portField.getText()));
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
});

```

```

        }
        new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    while(true){
                        BufferedReader br = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
                        String res = null;
                        while((res = br.readLine()) != null){
                            area.append(res + "\n");
                        }
                    }
                } catch (IOException ex) {
                    ex.printStackTrace();
                }
            }
        }).start();
    }
});

sayBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try{
            PrintWriter pw = new PrintWriter(socket.getOutputStream());
            pw.println(sayField.getText());
            pw.flush();
            area.append(sayField.getText() + "\n");
        }catch (IOException ex){
            ex.printStackTrace();
        }
    }
});
}

```

#### 服务器端:

```

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;

```

```
public class Server {
    private static ServerSocket serverSocket = null;
    private static Socket socket = null;
    public static void main(String[] args) {
        JFrame.setDefaultLookAndFeelDecorated(true);
        JFrame frame = new JFrame("服务端");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setResizable(false);
        frame.setBounds(300, 200, 500, 300);
        frame.setLayout(new BorderLayout(10, 0));

        JPanel northPanel = new JPanel();
        JLabel portLabel = new JLabel("Port: ");
        JTextField portField = new JTextField(28);
        northPanel.setBorder(BorderFactory.createTitledBorder("服务器设置: "));
        JButton startBtn = new JButton("Start");
        northPanel.add(portLabel);
        northPanel.add(portField);
        northPanel.add(startBtn);

        JPanel centerPanel = new JPanel();
        JTextArea area = new JTextArea(10, 35);
        area.setMargin(new Insets(10, 10, 10, 10));
        area.setEnabled(false);
        centerPanel.add(area);

        JPanel southPanel = new JPanel();
        JLabel sayLabel = new JLabel("Say: ");
        JTextField sayField = new JTextField(30);
        JButton sayBtn = new JButton("Say");
        southPanel.add(sayLabel);
        southPanel.add(sayField);
        southPanel.add(sayBtn);

        frame.add(northPanel, BorderLayout.NORTH);
        frame.add(centerPanel, BorderLayout.CENTER);
        frame.add(southPanel, BorderLayout.SOUTH);

        frame.setVisible(true);

        startBtn.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
```

```

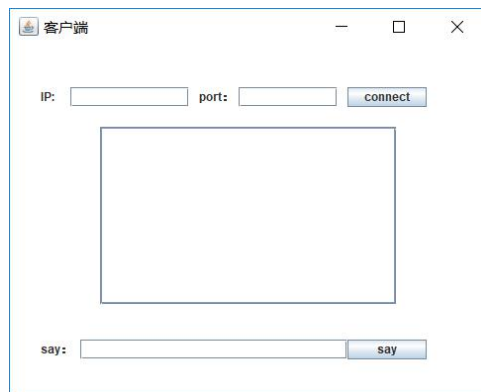
        area.append("Server starting...\n");
        new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    serverSocket = new
ServerSocket(Integer.parseInt(portField.getText()));
                    socket = serverSocket.accept();
                    area.append("Client connected...\n");
                    while(true){
                        BufferedReader br = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
                        String res = null;
                        while((res = br.readLine()) != null){
                            area.append(res + "\n");
                        }
                    } catch (IOException ex) {
                        ex.printStackTrace();
                    }
                }
            }
        }).start();
    }
});

sayBtn.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try{
            PrintWriter pw = new PrintWriter(socket.getOutputStream());
            pw.println(sayField.getText());
            pw.flush();
            area.append(sayField.getText() + "\n");
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }
});
}
}

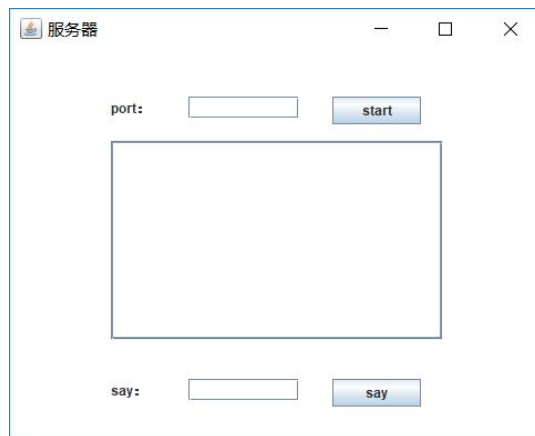
```

程序截图:

客户端:



服务器端：



## 五、实验结论、分析、思考题与心得体会

### 1. 实验心得：

通过此次实验掌握了图形界面的设计，掌握了各种组件的设计如何进行布局，学会了如何布置监听器，通过 Socket 网络通信实现网络通信，掌握了多线程变成方法。

### 2. 双向通信的功能

服务端：

```
import java.awt.BorderLayout;
import java.awt.Dimension;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JSplitPane;
```

```
import javax.swing.JTextArea;

public class ChatFrameServer{

    private PrintWriter pw;
    private JFrame frame;
    private JPanel pane_button;
    private JSplitPane pane_center;

    //显示内容的文本框，输入内容的文本框，发送内容按钮
    private JScrollPane pane_showWindow;
    private JScrollPane pane_inputWindow;
    private JTextArea area_showWindow;
    private JTextArea area_inputWindow;

    private JButton btn_send;

    private Dimension dimension;//用于设置 area_showWindow 可拖拉的大小

    //初始化
    public ChatFrameServer() {
        frame = new JFrame();
        pane_button = new JPanel();
        pane_showWindow = new JScrollPane();
        pane_inputWindow = new JScrollPane();
        area_showWindow = new JTextArea();
        area_inputWindow = new JTextArea();
        pane_center = new JSplitPane(JSplitPane.VERTICAL_SPLIT, false, pane_showWindow,
pane_inputWindow);
        btn_send = new JButton("发送");

        dimension = new Dimension(50, 300);

    }

    //调用方法显示窗口
    public void showFrame(){
        initFrame();
        initChatTextArea();
        initButton();
        btn_send();
        socket();
    }
}
```



```

//主窗体
public void initFrame(){
    frame.setTitle("服务端");
    int width = (int)Toolkit.getDefaultToolkit().getScreenSize().getWidth();
    int height = (int)Toolkit.getDefaultToolkit().getScreenSize().getHeight();
    frame.setBounds(width / 2, height / 2, 400, 450);
    frame.setVisible(true);
}

//内容显示文本框和输入内容文本框
private void initChatTextArea(){
    //取得视图焦点
    pane_showWindow.getViewport().add(area_showWindow);
    pane_inputWindow.getViewport().add(area_inputWindow);
    //将显示文本域设置为不可编辑
    area_showWindow.setEditable(false);
    //设置显示文本域可拖拉的大小
    pane_showWindow.setMinimumSize(dimension);
    frame.add(pane_center, BorderLayout.CENTER);
}

//发送文件，发送内容按钮
public void initButton(){
    pane_bottom.add(btn_send);
    frame.add(pane_bottom, BorderLayout.SOUTH);
}

private void btn_send(){
    btn_send.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
            String info = area_inputWindow.getText();
            area_showWindow.append("服务端: "+info+"\r\n");
            pw.println(info);
            area_inputWindow.setText("");
        }
    });
}

private void socket(){
    ServerSocket ss;
    try {
        ss = new ServerSocket(9988);
    }
}

```

```

        //等待连接 客户端
        Socket s=ss.accept();
        InputStreamReader isr=new InputStreamReader(s.getInputStream());
        BufferedReader br=new BufferedReader(isr);
        //PrintWriter 必须和 socket 有密切的关系
        pw=new PrintWriter(s.getOutputStream(),true);

        //读取从客户端发来的信息
        while(true) {
            //读取从客户端发来的信息
            String info=br.readLine();
            //在文本栏里显示
            area_showWindow.append("客户端:"+info+"\r\n");
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}

public static void main(String[] args) {
    ChatFrameServer chat = new ChatFrameServer();
    chat.showFrame();
}
}

```

客户端:

```

import java.awt.BorderLayout;
import java.awt.Dimension;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;
import java.net.UnknownHostException;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JSplitPane;
import javax.swing.JTextArea;

```

```

public class ChatFrame{
    private PrintWriter pw;
    private JFrame frame;
    private JPanel pane_button;
    private JSplitPane pane_center;

    //显示内容的文本框，输入内容的文本框,发送内容按钮
    private JScrollPane pane_showWindow;
    private JScrollPane pane_inputWindow;
    private JTextArea area_showWindow;
    private JTextArea area_inputWindow;

    private JButton btn_send;

    private Dimension dimension;//用于设置 area_showWindow 可拖拉的大小

    //初始化
    public ChatFrame() {
        frame = new JFrame();
        pane_button = new JPanel();
        pane_showWindow = new JScrollPane();
        pane_inputWindow = new JScrollPane();
        area_showWindow = new JTextArea();
        area_inputWindow = new JTextArea();
        pane_center = new JSplitPane(JSplitPane.VERTICAL_SPLIT, false, pane_showWindow,
pane_inputWindow);
        btn_send = new JButton("发送");

        dimension = new Dimension(50, 300);
    }

    //调用方法显示窗口
    public void showFrame(){
        initFrame();
        initChatTextArea();
        initButton();
        btn_send();
        socket();
    }

    //主窗体
    public void initFrame(){
        frame.setTitle("客户端");
    }

```

```

        int width = (int)Toolkit.getDefaultToolkit().getScreenSize().getWidth();
        int height = (int)Toolkit.getDefaultToolkit().getScreenSize().getHeight();
        frame.setBounds(width / 2, height / 2, 400, 450);
        frame.setVisible(true);
    }

    //内容显示文本框和输入内容文本框
    private void initChatTextArea(){
        //取得视图焦点
        pane_showWindow.getViewport().add(area_showWindow);
        pane_inputWindow.getViewport().add(area_inputWindow);
        //将显示文本域设置为不可编辑
        area_showWindow.setEditable(false);
        //设置显示文本域可拖拉的大小
        pane_showWindow.setMinimumSize(dimension);
        frame.add(pane_center, BorderLayout.CENTER);
    }

    //发送文件，发送内容按钮
    public void initButton(){
        pane_bottom.add(btn_send);
        frame.add(pane_bottom, BorderLayout.SOUTH);
    }

    private void btn_send(){
        btn_send.addActionListener(new ActionListener() {

            @Override
            public void actionPerformed(ActionEvent e) {
                String info = area_inputWindow.getText();
                area_showWindow.append("客户端:  "+info+"\r\n");
                pw.println(info);
                area_inputWindow.setText("");
            }
        });
    }

    private void socket(){
        try {
            Socket s = new Socket("127.0.0.1",9988);
            InputStreamReader isr=new InputStreamReader(s.getInputStream());
            BufferedReader br=new BufferedReader(isr);
            pw=new PrintWriter(s.getOutputStream(),true);
            while(true){

```

```
//不停地读取从服务器端发来的信息
String info=br.readLine();
area_showWindow.append("服务端:  "+info+"\r\n");
    }
} catch (UnknownHostException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
}
public static void main(String[] args) {
    ChatFrame chat = new ChatFrame();
    chat.showFrame();
}
}
```

