DAY 3———一客户一线程通讯

问题 3：如何实现多客户聊天？双机通讯？

3.1 学习内容

一客户一线程通讯，请学习教程的第 2 章 一客户一线程。

3.2 任务清单

① 窗口界面设计。

② 阅读书上程序，写出主要函数的步骤过程。

③ 画出服务器、客户机的软件结构。

④ 在小组内，抽签阐述服务器或客户端的软件结构，并录制视频。

3.3 任务分解

① 详细写出 main 函数的执行步骤，按序号列出每一步完成的工作。

② 特别注意并记录程序的善后工作步骤。

3.4 关键技术

3.4.1 输入/输出流

```
 in=new BufferedReader(new
InputStreamReader(toClientSocket.getInputStream(),"UTF-8"));
```

```
 out=new PrintWriter(new
OutputStreamWriter(toClientSocket.getOutputStream(),"UTF-8"),true);
```

3.4.2 客户端关键代码

```
      //1、向服务器发送消息，并接收服务器的 echo 消息
        private void btnSpeakActionPerformed(java.awt.event.ActionEvent evt) {
            if (clientSocket==null) {
                JOptionPane.showMessageDialog(null, "请先连接到服务器！", "错误
提示", JOptionPane.ERROR_MESSAGE);
                return;
            }
            //获取待发消息
            String outStr=txtInput.getText();
            if (outStr.length()==0) { //待发消息框为空
                JOptionPane.showMessageDialog(null, "请输入发送消息！","提示
",JOptionPane.INFORMATION_MESSAGE);
                return;
            }
            //发送
            out.println(outStr);
            txtInput.setText("");//发送的文字框清空
            try {
                //按照 echo 协议，客户机应立即接收服务器回送消息
                String inStr;
                inStr=in.readLine();
                //收到的 echo 消息加入下面的文本框
                txtArea.append("Echo: "+inStr+"\n");
            } catch (IOException ex) {
```

```
                JOptionPane.showMessageDialog(null,"客户机接收消息错误!","错误
提示",JOptionPane.ERROR_MESSAGE);
            }
        }
        //按下回车键
        private void txtInputActionPerformed(java.awt.event.ActionEvent evt) {
            btnSpeakActionPerformed(evt); //直接调用 btnSpeak 按钮的响应函数即可
        }
        //关闭客户机之前的资源释放工作
        private void formWindowClosing(java.awt.event.WindowEvent evt) {
            try {
                //4. 关闭并销毁网络流
                if (in!=null) in.close();
                if (out!=null) out.close();
                //5. 关闭并销毁套接字
                if (clientSocket!=null) clientSocket.close();
            } catch (IOException ex) {   }
        }
```

3.4.3 服务器端关键代码

3.4.3.1 服务器窗口程序（ServerUI.java ）

```
    //启动服务器
    private void btnStartActionPerformed(java.awt.event.ActionEvent evt) {
            try {
            btnStart.setEnabled(false); //禁用按钮，避免重复启动
            String hostName=txtHostName.getText();//主机名
            int hostPort=Integer.parseInt(txtHostPort.getText());//端口
            //构建服务器的 SocketAddress 格式地址
```

```
            SocketAddress serverAddr=new
InetSocketAddress(InetAddress.getByName(hostName),hostPort);
            listenSocket=new ServerSocket(); //创建侦听套接字
            listenSocket.bind(serverAddr);  //绑定到工作地址
            txtArea.append("服务器开始等待客户机连接...\n");
        } catch (IOException ex) {    }

       //创建一个匿名线程，用于侦听和接受客户机连接，并创建响应客户机的会话线程
        new Thread(new Runnable() {
            @Override
            public void run() {
                try {
                    while (true) { //处理客户机连接
                        toClientSocket=listenSocket.accept();//侦听接受客户机连
接
                        clientCounts++;//客户机数量加 1
                        txtArea.append(toClientSocket.getRemoteSocketAddress()+
" 客户机编号: "+clientCounts+ " 会话开始...\n");
                        //创建客户线程 clientThread，实现一客户一线程
                       Thread clientThread=new
ClientThread(toClientSocket,clientCounts);
                        clientThread.start(); //启动任务线程
                    } //end while
                } catch (IOException ex) {
                    JOptionPane.showMessageDialog(null, ex.getMessage(), "错误提
示", JOptionPane.ERROR_MESSAGE);
                }
            }//end run()
        }).start();
    }
```

```
    //关闭服务器之前
    private void formWindowClosing(java.awt.event.WindowEvent evt) {
        //关闭服务器之前释放套接字
        if (listenSocket!=null) listenSocket=null;
        if (toClientSocket!=null) toClientSocket=null;
    }
```

3.4.3.2 与客户会话程序（ClientThread.java ）

```
public class ClientThread extends Thread {
    private Socket toClientSocket=null;//会话套接字
    private BufferedReader in; //网络输入流
    private PrintWriter out; //网络输出流
    private int clientCounts=0;//在线客户机总数
    public ClientThread(Socket toClientSocket,int clientCounts) { //构造函数
        this.toClientSocket=toClientSocket;
        this.clientCounts=clientCounts;
    }
    @Override
    public void run(){
        try {
            // 创建绑定到套接字 toClientSocket 上的网络输入流与输出流
            in=new BufferedReader(new
InputStreamReader(toClientSocket.getInputStream(),"UTF-8"));
            out=new PrintWriter(new
OutputStreamWriter(toClientSocket.getOutputStream(),"UTF-8"),true);
            //5. 根据服务器协议，在网络流上进行读写操作
            String recvStr;
            while ((recvStr=in.readLine())!=null){ //只要客户机不关闭，则反复等
待和接收客户机消息
                Date date=new Date();
```

```
                DateFormat format=new SimpleDateFormat("yyyy-mm-dd hh:mm:ss");
                String time=format.format(date);
                ServerUI.txtArea.append(toClientSocket.getRemoteSocketAddress()+
" 客户机编号: "+clientCounts+" 消息："+recvStr+" ："+time+"\n"); //解析并显示收
到的消息
                //按照 echo 协议原封不动回送消息
                out.println(toClientSocket.getLocalSocketAddress()+ " 客户机编号:
"+clientCounts+" Echo 消息："+recvStr+" : "+time);
            }//end while
            ServerUI.clientCounts--; //客户机总数减 1
            //远程客户机断开连接，线程释放资源
            if (in!=null) in.close();
            if (out!=null) out.close();
            if (toClientSocket!=null) toClientSocket.close();
        }catch (IOException ex) {}
    } //end run
} //end class
```

3.5 问题讨论

① 如何知道服务器地址？怎样获取和显示本地 IP？