JAVA实验报告

1. 实验要求

实现一个Markdown编辑器，具有如下功能：

能编辑Markdown文档

能在编辑区的左侧看到实时的文档目录

能保存和打开Markdown文档

能输出html

能建立一个网络服务，以供其他编辑器连接

能连接其他编辑器，连接后可编辑对方正在编辑的文档

连接了其他编辑器后，能实时同步反映服务器上的文件在其他编辑器上的修改

加分项（5分）：能实时在编辑区右侧看Markdown渲染后的效果

要求提交源代码和实验报告，如果使用了第三方库，需要给出第三方库的URL。

二．实验原理及过程

本实验基于IntelliJ IDEA Community Edition 2018.2.3 x64的Maven建立。

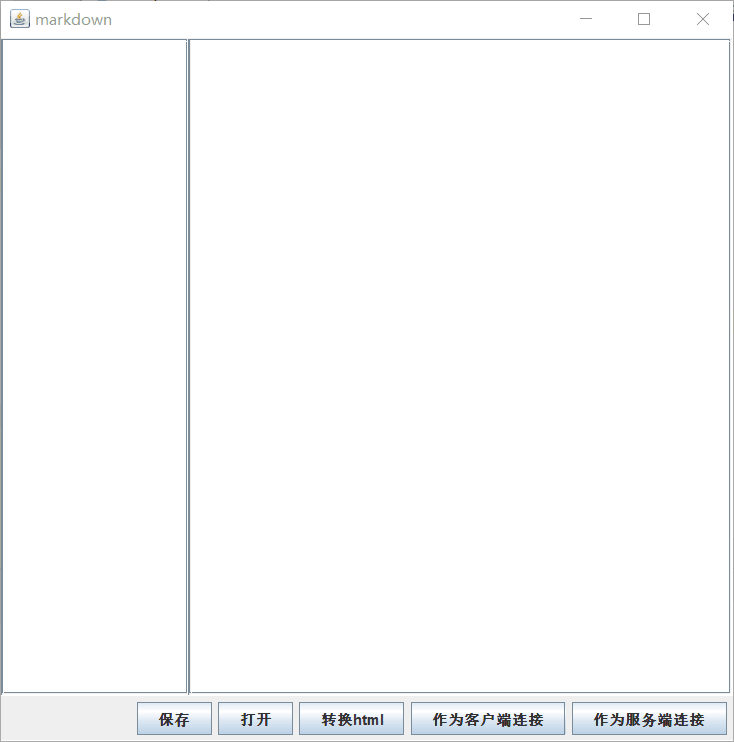
1. 使用JTextArea组件，两个JTextArea分别表示编辑区域和目录区域，目录区域设置为不可编辑。
2. 编辑文档时，每一次改动都遍历文档搜索目录结构，并显示在左侧目录区域。
3. 利用JFilechooser打开和保存文件。
4. 利用flexmark包实现md文件转换html文件,在工程中添加依赖项。

URL: https://github.com/vsch/flexmark-java

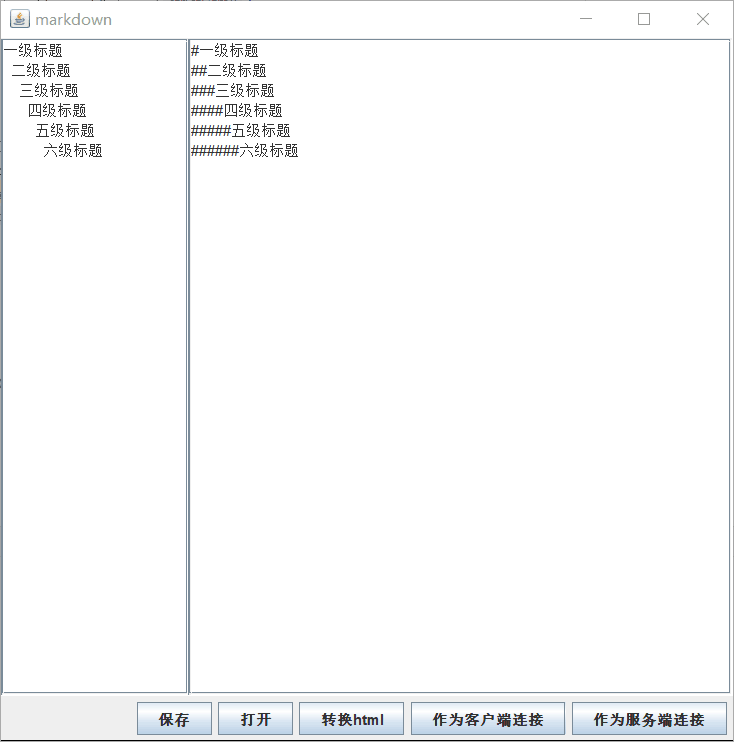
1. 同步在线编辑：利用socket实现两个编辑器互联，其中一个先设置为服务端，然后另一个编辑器根据ip和端口连接服务端。连接成功以后，每次进行修改都会发送文本给另一个编辑器，然后另一个编辑器根据收到的文本也进行修改，实现同步编辑。

三.实验结果

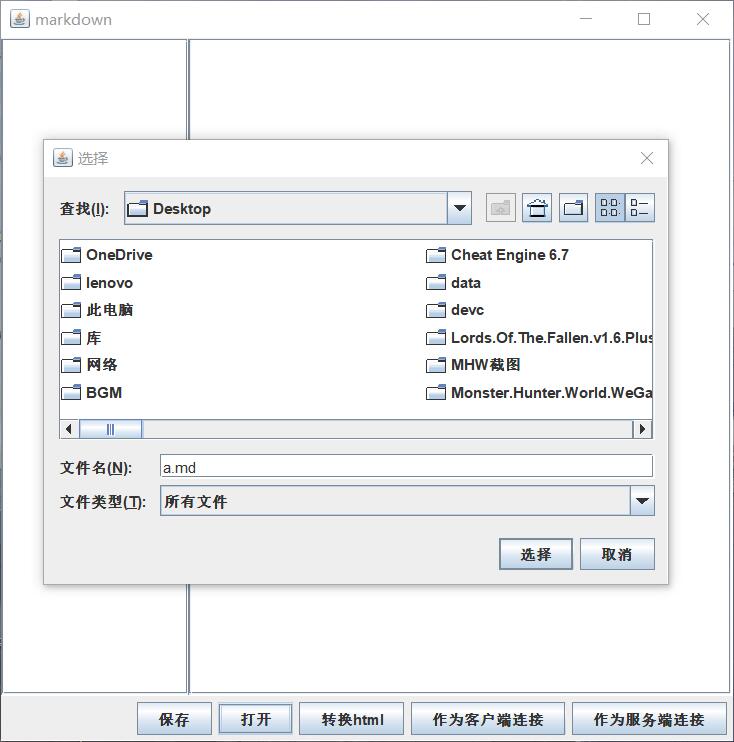
1.图形界面



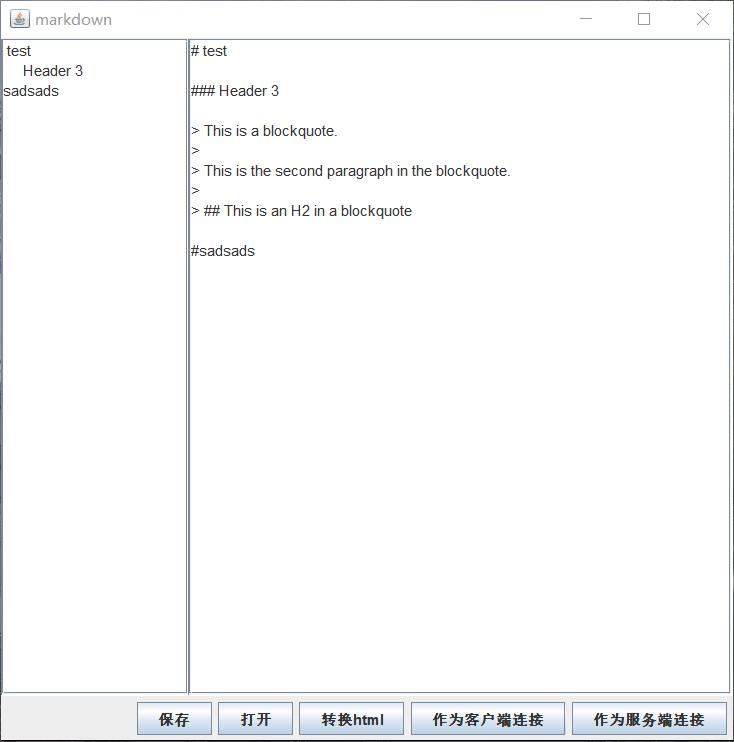
2.左侧目录

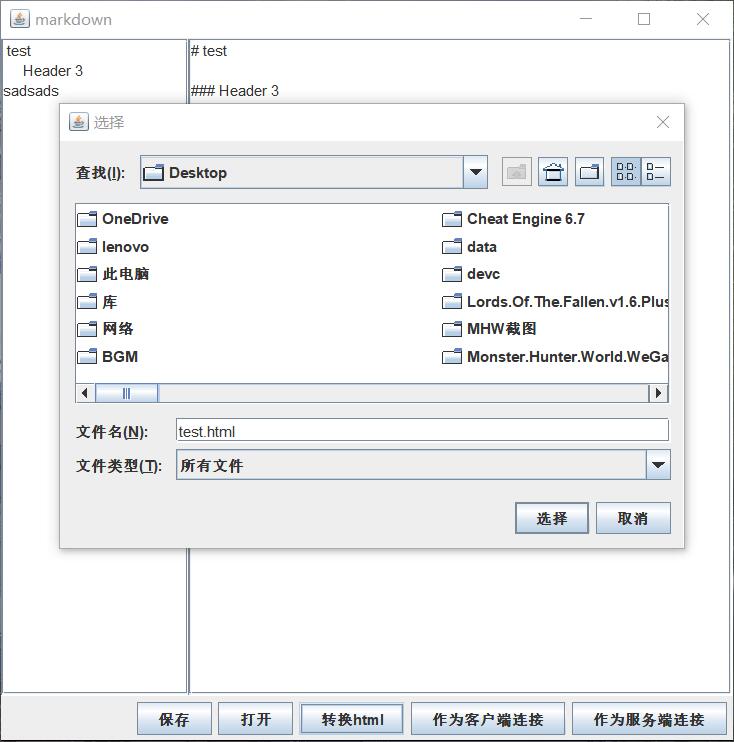


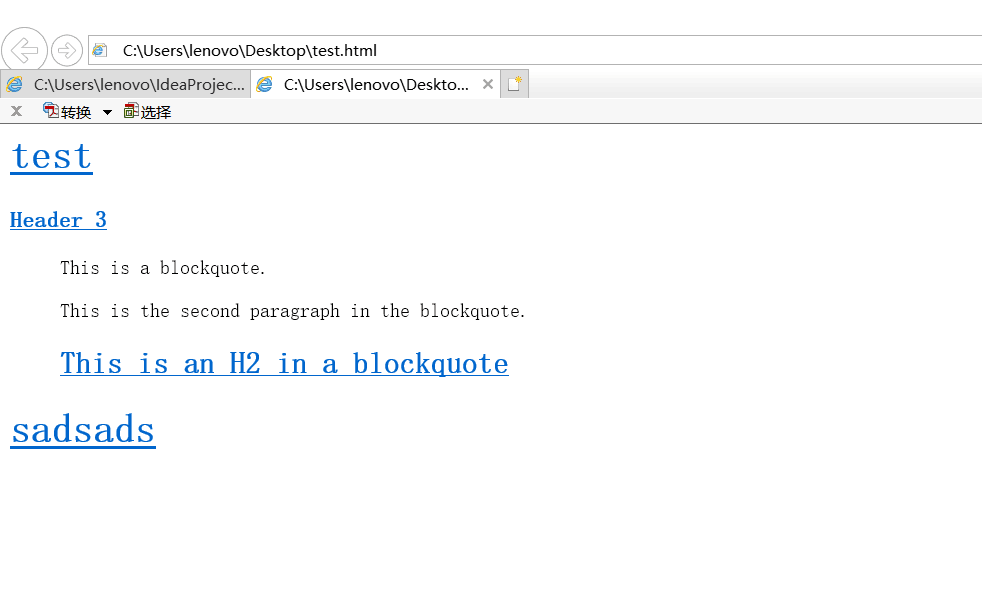
3.保存和打开md文件



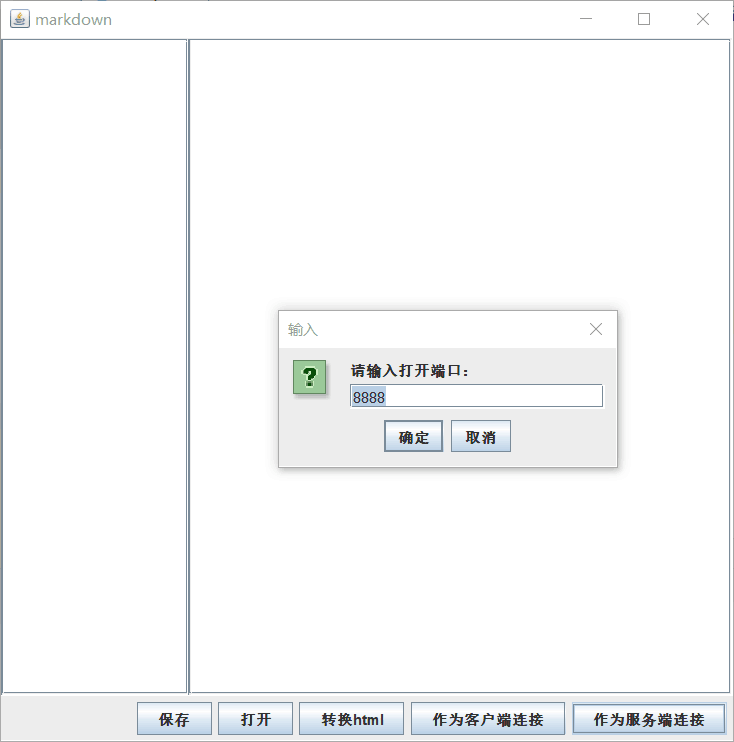
4.转换成html文件

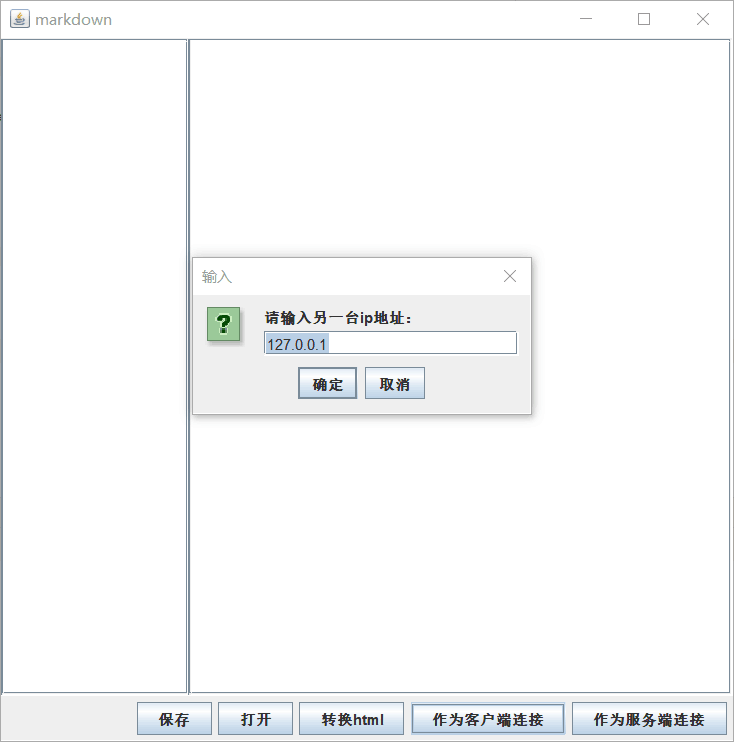


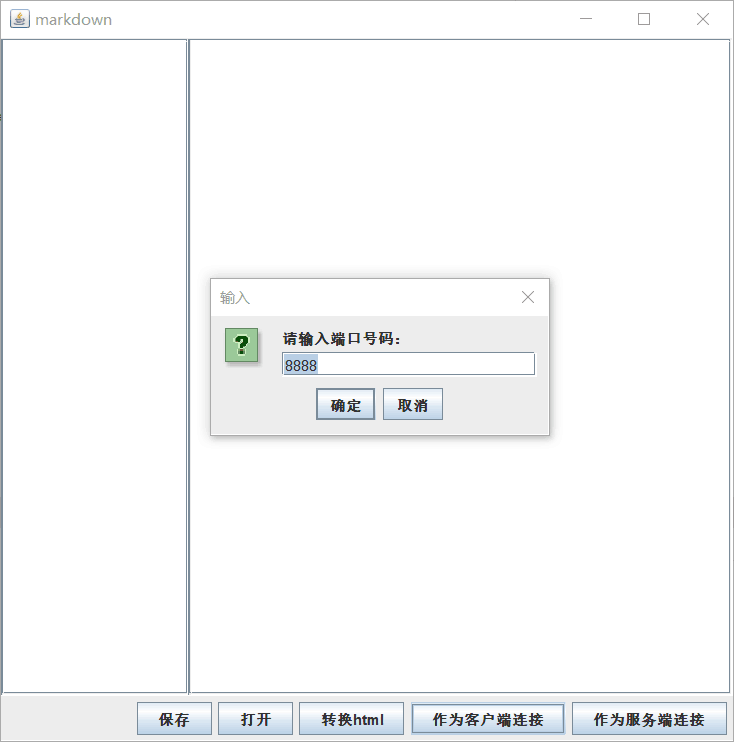


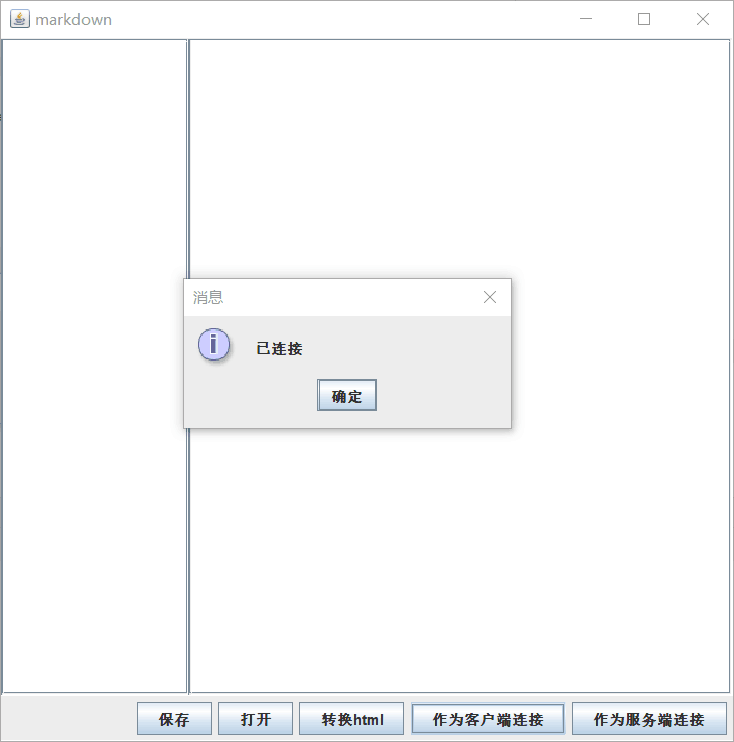


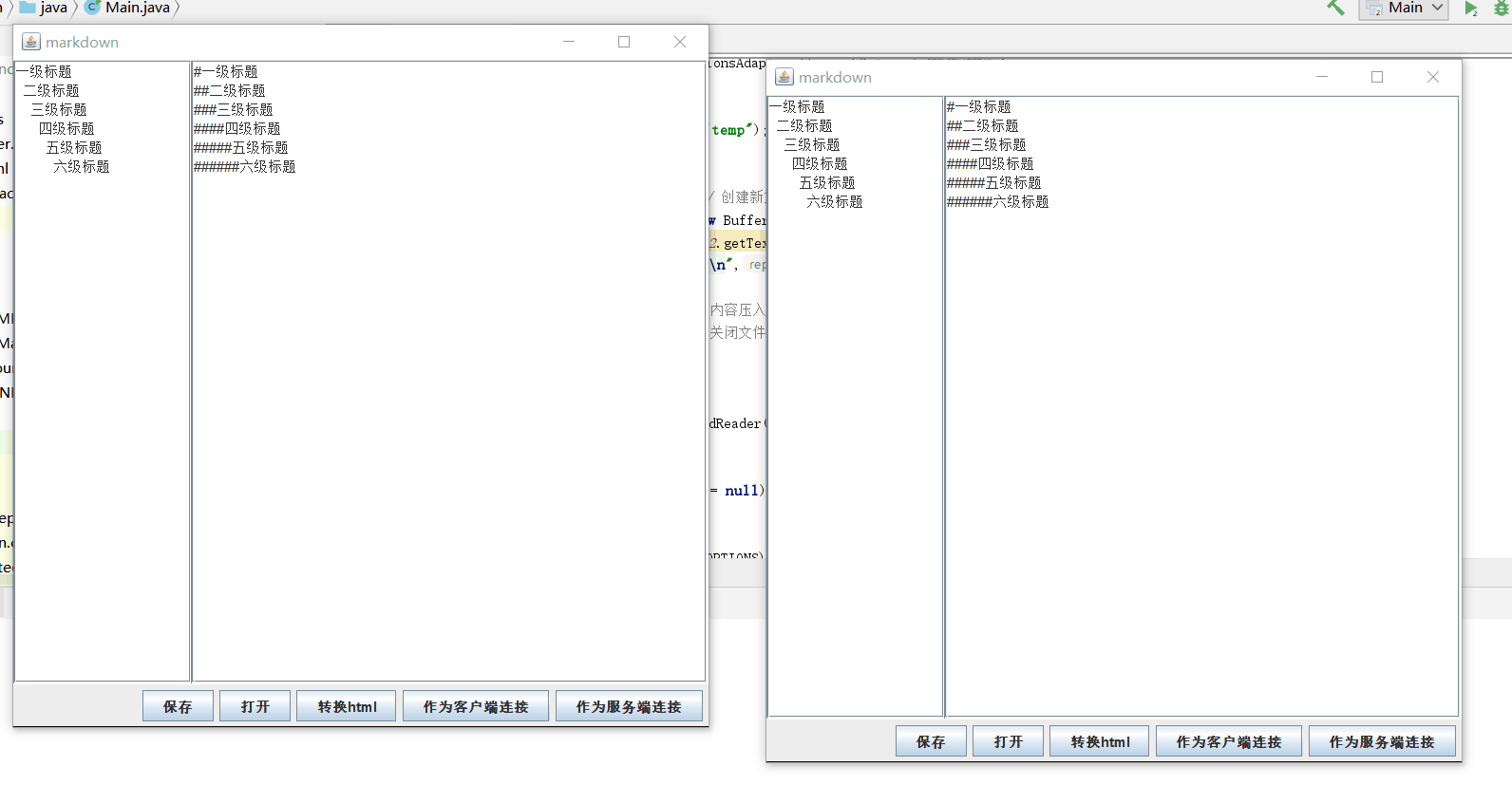
5.协同编辑。先设置服务端端口号，然后再作为客户端根据ip和端口连接。默认值为8888端口，本机ip地址。











四.实验代码

**import** com.vladsch.flexmark.ast.Node;  
**import** com.vladsch.flexmark.html.HtmlRenderer;  
**import** com.vladsch.flexmark.parser.Parser;  
**import** com.vladsch.flexmark.profiles.pegdown.Extensions;  
**import** com.vladsch.flexmark.profiles.pegdown.PegdownOptionsAdapter;  
**import** com.vladsch.flexmark.util.options.DataHolder;  
  
**import** javax.swing.\*;  
**import** javax.swing.event.DocumentEvent;  
**import** javax.swing.event.DocumentListener;  
**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** Main **extends** JFrame{ *//继承JFrame顶层框架  
  
 //定义组件  
 //上部组件* JPanel **jp1**; *//定义面板* JSplitPane **jsp**; *//定义拆分窗格* JTextArea **jta1**; *//定义文本域* JScrollPane **jspane1**; *//定义滚动窗格* **static** JTextArea *jta2*;  
 JScrollPane **jspane2**;  
 *//下部组件* JPanel **jp2**;  
 JButton **jb1**,**jb2**,**jb3**,**jb4**,**jb5**; *//定义按钮* JComboBox **jcb1**; *//定义下拉框* **static boolean** *from\_socket*=**false**;  
 **static boolean** *connected\_as\_client*=**false**;  
 **static boolean** *connected\_as\_server*=**false**;  
 **static** ServerSocket *ss*=**null**;  
 **static** Socket *s*=**null**;  
 **static** BufferedReader *br\_server*=**null**;  
 **static** BufferedReader *br\_client*=**null**;  
 **static** BufferedWriter *bw\_server*=**null**;  
 **static** BufferedWriter *bw\_client*=**null**;  
  
 **public static void** main(String[] args) {  
 Main a=**new** Main(); *//显示界面* }  
 **public** Main() *//构造函数* {  
 **jp1**=**new** JPanel(); *//创建面板* **jta1**=**new** JTextArea(); *//创建多行文本框* **jta1**.setEditable(**false**);  
 **jta1**.setLineWrap(**true**); *//设置多行文本框自动换行* **jspane1**=**new** JScrollPane(**jta1**); *//创建滚动窗格  
 jta2*=**new** JTextArea();  
 *jta2*.setLineWrap(**true**);  
 **jspane2**=**new** JScrollPane(*jta2*);  
 **jsp**=**new** JSplitPane(JSplitPane.***HORIZONTAL\_SPLIT***,**jspane1**,**jspane2**); *//创建拆分窗格* **jsp**.setDividerLocation(150); *//设置拆分窗格分频器初始位置* **jsp**.setDividerSize(1); *//设置分频器大小* **jp2**=**new** JPanel();  
 **jb1**=**new** JButton(**"保存"**); **jb1**.addActionListener(**new** ActionListener() {**public void** actionPerformed(ActionEvent e) { save\_md();}});  
 **jb2**=**new** JButton(**"打开"**);**jb2**.addActionListener(**new** ActionListener() {**public void** actionPerformed(ActionEvent e) { read\_md();}});  
 **jb3**=**new** JButton(**"转换html"**);**jb3**.addActionListener(**new** ActionListener() {**public void** actionPerformed(ActionEvent e) { **try**{generateHtml();}**catch** (Exception e1){System.***out***.println(**"HTML ERROR"**);}}});  
 **jb4**=**new** JButton(**"作为客户端连接"**);**jb4**.addActionListener(**new** ActionListener() {**public void** actionPerformed(ActionEvent e) { connect\_client();}});  
 **jb5**=**new** JButton(**"作为服务端连接"**);**jb5**.addActionListener(**new** ActionListener() {**public void** actionPerformed(ActionEvent e) { connect\_server();}});  
 *//设置布局管理* **jp1**.setLayout(**new** BorderLayout()); *//设置面板布局* **jp2**.setLayout(**new** FlowLayout(FlowLayout.***RIGHT***));  
 *//添加组件* **jp1**.add(**jsp**);**jp2**.add(**jb1**);**jp2**.add(**jb2**);**jp2**.add(**jb3**);**jp2**.add(**jb4**);**jp2**.add(**jb5**);  
 **this**.add(**jp1**,BorderLayout.***CENTER***);  
 **this**.add(**jp2**,BorderLayout.***SOUTH***);  
 *//设置窗体实行* **this**.setTitle(**"markdown"**); *//设置界面标题* **this**.setSize(600, 600); *//设置界面像素* **this**.setLocation(200, 200); *//设置界面初始位置* **this**.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***); *//设置虚拟机和界面一同关闭* **this**.setVisible(**true**); *//设置界面可视化  
 jta2*.getDocument().addDocumentListener(**new** DocumentListener() {  
 **public void** insertUpdate(DocumentEvent e) {  
 **if** (!*from\_socket*){  
 send\_mess(*jta2*.getText());  
 *from\_socket*=**false**;}  
 String[] con=*jta2*.getText().split(**"\\n"**);String res=**""**;  
 **int** i=0;String thisline=**""**;  
 **for** (i=0;i<con.**length**;i++)  
 {  
 thisline=con[i];  
 **int** j=0;**int** number\_sharp=0;  
 **for** (j=0;j<thisline.length();j++){  
 **if** (thisline.charAt(j)==**'#'**){  
 number\_sharp++;  
 }  
 **else** {  
 **break**;  
 }  
 }  
 **if** (number\_sharp>6) number\_sharp=6;  
 String out=**""**;  
 **if** (number\_sharp>=1) {  
 **for** (j=0;j<number\_sharp-1;j++){  
 out=out+**" "**;  
 }  
 res=res+out+thisline.substring(number\_sharp)+**"\n"**;  
 }  
 }  
 **jta1**.setText(res);  
 }  
 **public void** removeUpdate(DocumentEvent e) {  
 **if** (!*from\_socket*){  
 send\_mess(*jta2*.getText());  
 *from\_socket*=**false**;}  
 String[] con=*jta2*.getText().split(**"\\n"**);String res=**""**;  
 **int** i=0;String thisline=**""**;  
 **for** (i=0;i<con.**length**;i++)  
 {  
 thisline=con[i];  
 **int** j=0;**int** number\_sharp=0;  
 **for** (j=0;j<thisline.length();j++){  
 **if** (thisline.charAt(j)==**'#'**){  
 number\_sharp++;  
 }  
 **else** {  
 **break**;  
 }  
 }  
 **if** (number\_sharp>6) number\_sharp=6;  
 String out=**""**;  
 **if** (number\_sharp>=1) {  
 **for** (j=0;j<number\_sharp-1;j++){  
 out=out+**" "**;  
 }  
 res=res+out+thisline.substring(number\_sharp)+**"\n"**;  
 }  
 }  
 **jta1**.setText(res);  
 }  
 **public void** changedUpdate(DocumentEvent e) {  
 }  
 });  
  
 AcceptThread acceptThread = **new** AcceptThread();  
 acceptThread.start();  
 **try** {  
 acceptThread.join();  
 }**catch** (Exception e){e.printStackTrace();}  
 }  
  
 **void** send\_mess(String mess)  
 {  
 **if** (*connected\_as\_client*) {  
 **try**{  
 *bw\_client*.write(mess+**'\n'**+**"\_end\_"**+**'\n'**);  
 *bw\_client*.flush();}**catch** (Exception e){e.printStackTrace();}  
 }  
 **else if** (*connected\_as\_server*){  
 **try**{  
 *bw\_server*.write(mess+**'\n'**+**"\_end\_"**+**'\n'**);  
 *bw\_server*.flush();}**catch** (Exception e){e.printStackTrace();}  
 }  
 }  
  
  
 **public** String generateHtml() **throws** IOException {  
 DataHolder OPTIONS = PegdownOptionsAdapter.*flexmarkOptions*(**true**,  
 Extensions.***ALL*** );  
 File file=**new** File(**"temp"**);  
 **if** (file!=**null**) {  
 **try** {  
 file.createNewFile(); *// 创建新文件* BufferedWriter out = **new** BufferedWriter(**new** FileWriter(file));  
 String o=**new** String(*jta2*.getText());  
 o=o.replaceAll(**"\n"**,**"\r\n"**);  
 out.write(o);  
 out.flush(); *// 把缓存区内容压入文件* out.close(); *// 最后记得关闭文件* }  
 **catch** (Exception e) { }  
 }  
 BufferedReader br = **new** BufferedReader(**new** InputStreamReader(**new** FileInputStream(**"temp"**), **"UTF-8"**));  
 String line = **null**;  
 String mdContent = **""**;  
 **while** ((line = br.readLine()) != **null**) {  
 mdContent += line + **"\r\n"**;  
 }  
 Parser parser = Parser.*builder*(OPTIONS).build();  
 HtmlRenderer renderer = HtmlRenderer.*builder*(OPTIONS).build();  
 Node document = parser.parse(mdContent);  
 JFileChooser jfc=**new** JFileChooser();  
 jfc.setFileSelectionMode(JFileChooser.***FILES\_AND\_DIRECTORIES*** );  
 jfc.showDialog(**new** JLabel(), **"选择"**);  
 File f=jfc.getSelectedFile();  
 FileOutputStream fos1=**new** FileOutputStream(f);  
 OutputStreamWriter dos1=**new** OutputStreamWriter(fos1);  
 dos1.write(renderer.render(document));  
 dos1.close();  
 **return** renderer.render(document);  
 }  
  
 **void** read\_md()  
 {  
 JFileChooser jfc=**new** JFileChooser();  
 jfc.setFileSelectionMode(JFileChooser.***FILES\_AND\_DIRECTORIES*** );  
 jfc.showDialog(**new** JLabel(), **"选择"**);  
 File file=jfc.getSelectedFile();  
 **if** (file!=**null**) {  
 **if** (file.isDirectory()) {  
 System.***out***.println(**"不能是文件夹"**);  
 } **else if** (file.isFile())  
 {  
 **try** {  
 BufferedReader br = **new** BufferedReader(**new** FileReader(file));*//构造一个BufferedReader类来读取文件* String s = **""**;String temp=**""**;  
 **while**((temp = br.readLine())!=**null**){  
 s=s+temp+**'\n'**;  
 }  
 *jta2*.setText(s);  
 br.close();  
 }  
 **catch** (Exception e) { System.***out***.println(**"read error"**); }  
 }  
 }  
 }  
 **void** save\_md()  
 {  
 JFileChooser jfc=**new** JFileChooser();  
 jfc.setFileSelectionMode(JFileChooser.***FILES\_AND\_DIRECTORIES*** );  
 jfc.showDialog(**new** JLabel(), **"选择"**);  
 File file=jfc.getSelectedFile();  
 **if** (file!=**null**) {  
 **try** {  
 file.createNewFile(); *// 创建新文件* BufferedWriter out = **new** BufferedWriter(**new** FileWriter(file));  
 String o=**new** String(*jta2*.getText());  
 o=o.replaceAll(**"\n"**,**"\r\n"**);  
 out.write(o);  
 out.flush(); *// 把缓存区内容压入文件* out.close(); *// 最后记得关闭文件* }  
 **catch** (Exception e) { System.***out***.println(**"save error"**);}  
 }  
 }  
  
 **void** connect\_server(){  
 **if** (*connected\_as\_server*||*connected\_as\_client*){  
 JOptionPane.*showMessageDialog*(**null**,**"已连接"**);  
 **return**;  
 }  
 String port\_name=JOptionPane.*showInputDialog*(**"请输入打开端口：\n"**,8888);  
 **try** {  
 *ss* = **new** ServerSocket(Integer.*parseInt*(port\_name));  
 Socket t = *ss*.accept();  
 System.***out***.println(**"客户端:"**+t.getInetAddress().*getLocalHost*()+**"已连接到服务器"**);  
 *br\_server* = **new** BufferedReader(**new** InputStreamReader(t.getInputStream()));  
 *bw\_server* = **new** BufferedWriter(**new** OutputStreamWriter(t.getOutputStream()));  
 JOptionPane.*showMessageDialog*(**null**,**"已连接"**);  
 *connected\_as\_server*=**true**;  
 }**catch** (Exception e){  
 e.printStackTrace();  
 }  
 }  
 **void** connect\_client(){  
 **if** (*connected\_as\_client*||*connected\_as\_server*){  
 JOptionPane.*showMessageDialog*(**null**,**"已连接"**);  
 **return**;  
 }  
 String ip=JOptionPane.*showInputDialog*(**"请输入另一台ip地址：\n"**,**"127.0.0.1"**);  
 String port\_name=JOptionPane.*showInputDialog*(**"请输入端口号码：\n"**,8888);  
 **try** {  
 *s* = **new** Socket(ip,Integer.*parseInt*(port\_name));  
 InputStream is = *s*.getInputStream();  
 OutputStream os = *s*.getOutputStream();  
  
 *bw\_client* = **new** BufferedWriter(**new** OutputStreamWriter(os));  
 *br\_client* = **new** BufferedReader(**new** InputStreamReader(is));  
 JOptionPane.*showMessageDialog*(**null**,**"已连接"**);  
 *connected\_as\_client*=**true**;  
 }**catch** (Exception e){  
 e.printStackTrace();  
 }  
 }  
  
}  
  
**class** AcceptThread **extends** Thread  
{  
 **public void** run()  
 {  
 **try** {  
 **while** (**true**) {  
 **if** (Main.*connected\_as\_server*) {  
 System.***out***.println(**"test1"**);  
 String s = **""**;String temp=**""**;  
 **while**((temp = Main.*br\_server*.readLine())!=**null**){  
 **if** (temp.equals(**"\_end\_"**)){**break**;}  
 s=s+temp+**'\n'**;  
 }  
 *//System.out.println(m);* Main.*from\_socket*=**true**;  
 Main.*jta2*.setText(s);  
 Main.*from\_socket*=**false**;  
 } **else if** (Main.*connected\_as\_client*) {  
 System.***out***.println(**"test2"**);  
 String s = **""**;String temp=**""**;  
 **while**((temp = Main.*br\_client*.readLine())!=**null**){  
 **if** (temp.equals(**"\_end\_"**)){**break**;}  
 s=s+temp+**'\n'**;  
 }  
 Main.*from\_socket*=**true**;  
 Main.*jta2*.setText(s);  
 Main.*from\_socket*=**false**;  
 }  
  
 *sleep*(100);  
 }  
 }**catch** (Exception e){e.printStackTrace();}  
 }  
}