浙江大学实验报告

一、 实验目的和要求

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

以上功能均已全部实现。

二、 实验内容和原理

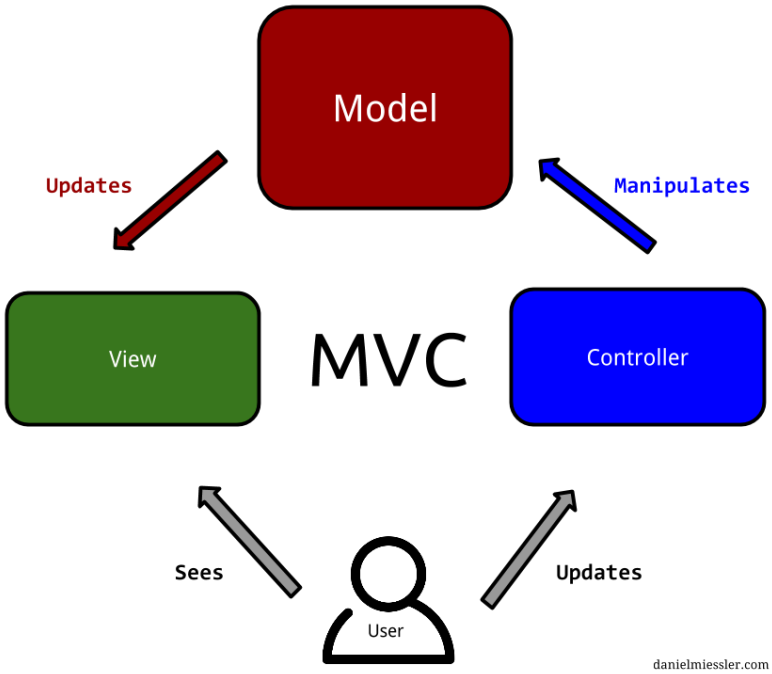
1. 利用所学知识设计出所要求的 GUI，掌握组件的事件响应原理；
2. 熟悉 Java 的功能和操作；
3. 熟悉 Java socket 编程
4. 熟悉 Java 多线程编程

三、 实验内容

1 整体架构——MVC 模式

本程序采用 Model-View-Controller（模型-视图-控制器）模式即 MVC 模式来架构代码。

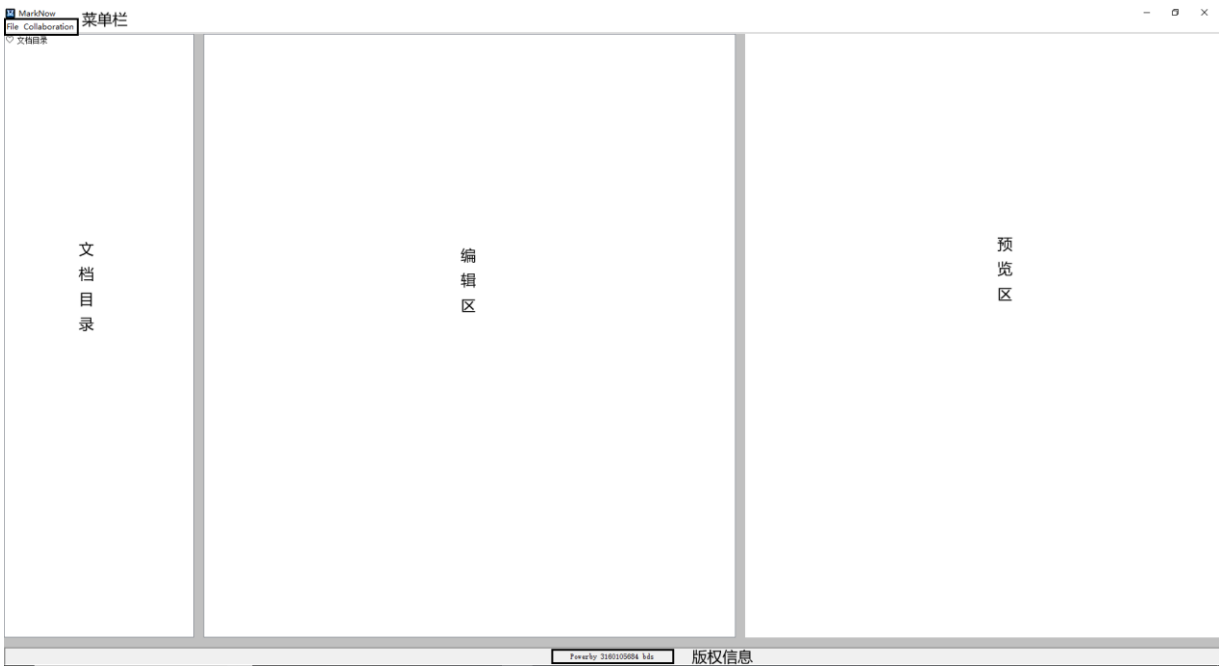
- **Model（模型）** - 模型层是最底下的一层，代表一个存取数据的对象或 JAVA POJO。它也可以带有逻辑，在数据变化时更新控制器。
- **View（视图）** - 视图层是最上面的一层，代表模型包含的数据的可视化。
- **Controller（控制器）** - 控制层是中间一层，作用于模型和视图上。它控制数据流向模型对象，并在数据变化时更新视图。它使视图与模型分离开。



这三层是紧密联系在一起的，但又是互相独立的，每一层内部的变化不影响其他层。每一层都对外提供接口（Interface），供上面一层调用。这样一来，软件就可以实现模块化，修改外观或者变更数据都不用修改其他层，大大方便了维护和升级。

2 UI 设计

采用 Win10 的系统界面设计。包括菜单栏、文档目录、编辑区、预览区和版权信息栏。其中，菜单栏的“File”下拉菜单中包括“Open”、“Save”和“Export”3 项，“Collaboration”下拉菜单中包括“New Server”，“Connect To Server”和“Disconnect”3 项。



3 实现思路

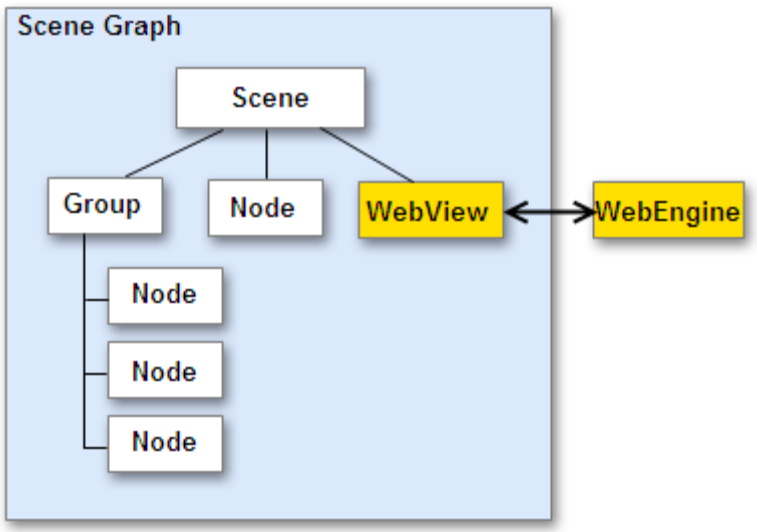
本程序的图形界面是用 Swing 实现，网络服务是用 Socket 实现。代码整体分为 3 个 package，分别是 model, view 和 net。其中，model 和 view 实现了双向绑定，即 Text 类绑定了 view 部分的几个图形面板，view 部分的几个图形面板也绑定了 Text 类。这样只要 view 部分发生变化，对应的 Text 对象就会相应的做出改变；Text 对象发生变化，对应的 view 部分也会相应做出刷新改变。

- **View 模块：**

View 模块包括 CatalogPanel、DisplayPanel、EditPanel、MenuBar 和 MainView 一共 5 个部分。

EditPanel 部分是一个继承 JPanel 的类，里面有一个 TextArea 用来编辑 MarkDown 文件。

DisplayPanel 部分是一个继承 JFXPanel 的类，在 JFXPanel 的内嵌浏览器 WebView 组件来实现将 markdown 的 html 文件显示出来。该内嵌浏览器基于开源 web 浏览器引擎 WebKit。其架构如下：



CatalogPanel 部分是一个继承 JPanel 的类，通过字符串匹配得到文档的各级标题，然后建立 JTree。一个 JTree 对象并没有包含实际的数据；它只是提供了数据的一个视图。像其他非平凡的（ nontrivial ） Swing 组件一样，这种 Jtree 通过查询她的数据模型获得数据。

MenuBar 部分是菜单栏。菜单栏的“File”下拉菜单中包括“Open”、“Save”和“Export”3 项，“Collaboration”下拉菜单中包括“New Server”，“Connect To Server”和“Disconnect”3 项。其中，“Export”是导出 html 文件。“New Server”是新建一个服务器，“Connect To

Server”是连接到服务器，“Disconnect” 是断开与服务器的连接。

MainView 部分继承了 JFrame，将上述的所有模块都添加到 JFrame 里，同时设置诸如图标之类的组件。

- **Model 模块：**

Model 模块包括 CatalogNode、Text 一共 2 个部分。

CatalogNode 部分是和 View 中的 CatalogPanel 模块相关联，自定义了 JTree 节点的数据，包含 3 个字段：level（标题的级别）、lineNO（对应 EditPanel 的行号）、title（标题名称）。toString 方法也重新实现，返回 title。

Text 部分存放了编辑器的 markdown 文本内容，和 view 模块的各部分实现了双向绑定。使用了第三方库 CommonMark 来把 markdown 文件转换为 html 格式。使用了编辑锁 Semaphore 的机制，即当有人在编辑某个文档时，系统会将这个文档锁定，避免其他人同时编辑，这样可以有效的避免冲突产生。同时为了防止出现服务器对文本做出的更新，用户又将这次更新传至服务器导致无限循环这样的状况，使用了 isLocked 这样一个 boolean 值来区分解决用户更新文本和服务器更新文本两种情况。

- **Net 模块：**

Net 模块包括 Server、Client 一共 2 个部分。

Server 部分守听在用户指定的端口，新建一个线程阻塞等待用户的连接。对于每一个连接，Server 都会创建一个线程进行管理。当接收到某个客户端的信息后，将信息广播至所有连接此客户端的用户。

Client 部分实现较为简单，只需创建一个线程来收发数据即可。

- **程序入口：**

因为我使用了 Maven 对此项目进行构建和依赖管理。所以程序入口是在 App.Java 中。具体如下所示：

```
public class App
{
    public static void main( String[] args )
    {
        Text text = new Text();
        @SuppressWarnings("unused")
        Client client = new Client(text);
        @SuppressWarnings("unused")
        MainView mainView = new MainView(text);
```

```
    }
}
```

四、 运行环境

CPU: Intel(R)Core(TM)i5-5200U CPU@2.20GHz
Memory: 8GB(DDR3L 1600MHZ)
Operating System: Windows 10
Compiler: jdk1.8.0
IDE: Eclipse

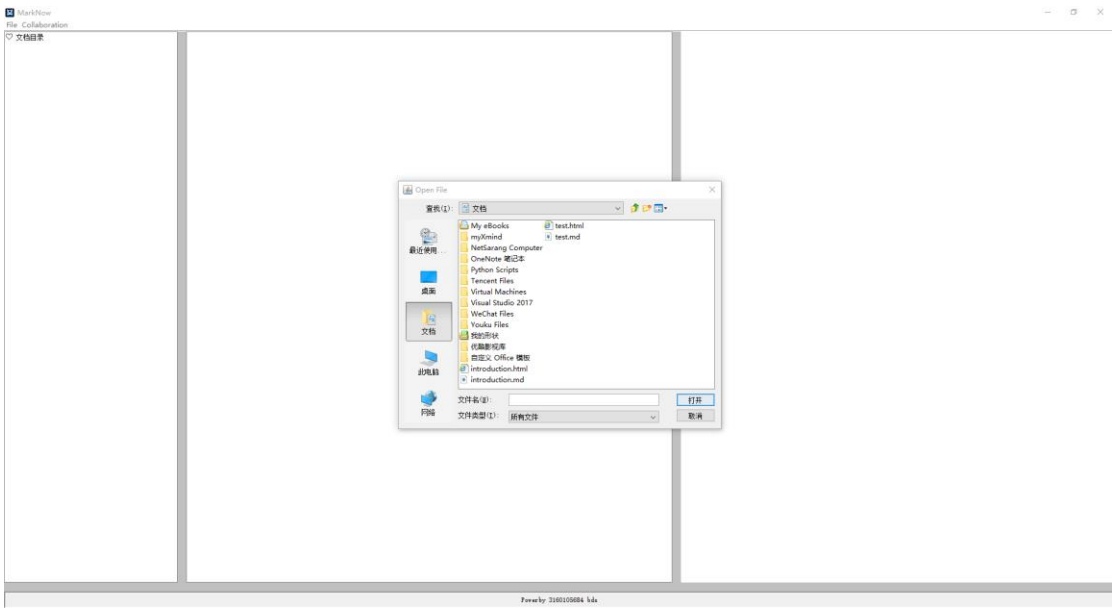
五、 操作方法与实验步骤

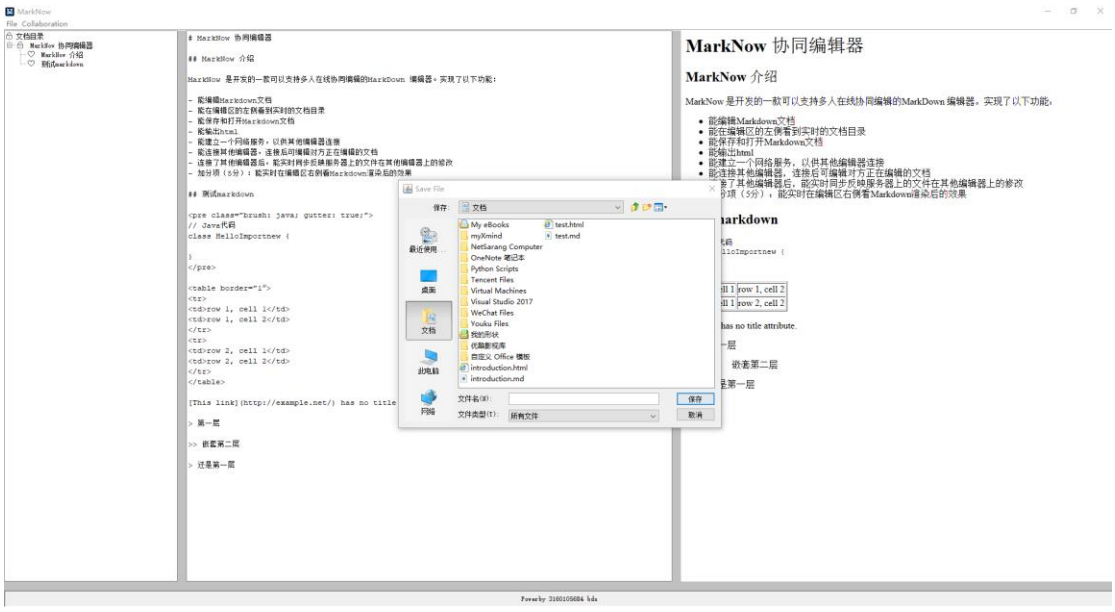
1. 双击 miniCAD 打开程序；
2. 在编辑区中输入 markdown 内容并查看文档目录和相应的实时渲染效果；
3. 点击菜单栏“File”，选择“Open”，可以打开一个 md 文件；
4. 点击菜单栏“File”，选择“Save”，可以保存当前的 md 文件；
5. 点击菜单栏“File”，选择“Export”，可以导出 html 文件；
6. 点击菜单栏“Collaboration”，选择“New Server”，可以新建一个服务端；
7. 点击菜单栏“Collaboration”，选择“Connect To Server”，可以连接到对应的服务端（注意：即使此客户端是新建服务器的客户端，它仍需连接对应的服务器）；
8. 点击菜单栏“Collaboration”，选择“Disconnect”，可以断开与服务端的连接；

六、 实验结果与分析

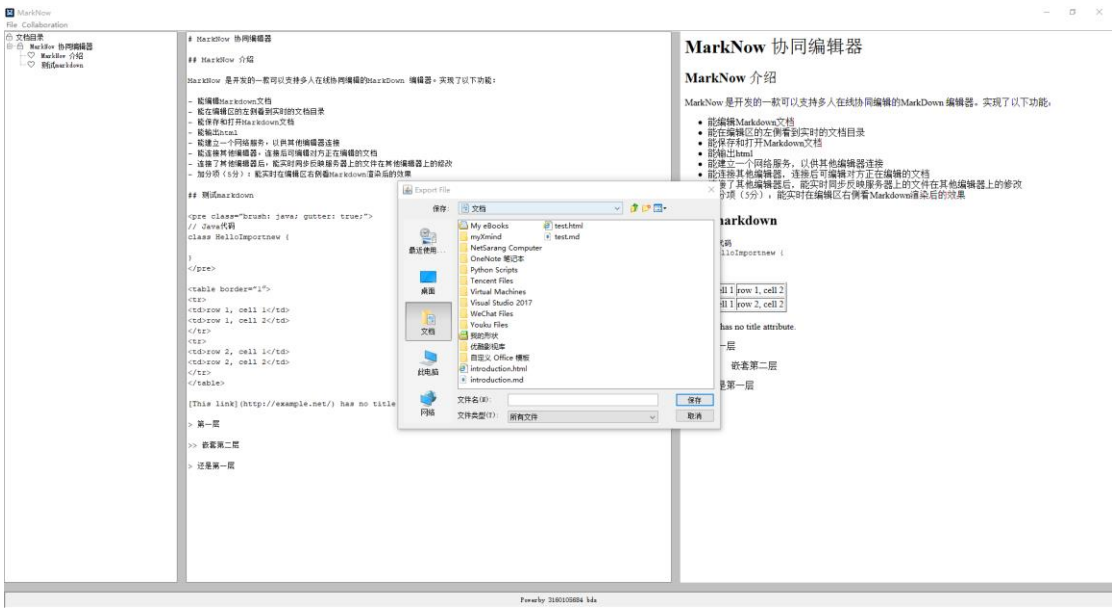
1. 编辑 markdown 文件并实现文档目录、实时渲染等功能。

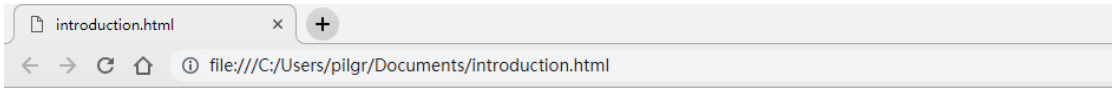
2. 打开、保存 md 文件





3. 导出 html 文件





MarkNow 协同编辑器

MarkNow 介绍

MarkNow 是开发的一款可以支持多人在线协同编辑的MarkDown 编辑器。实现了以下功能：

- 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

测试markdown

```
// Java代码
class HelloImportnew {

}
```

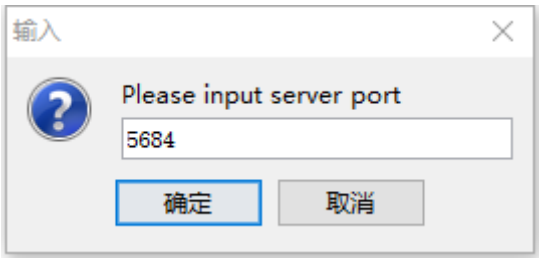
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

This link has no title attribute.

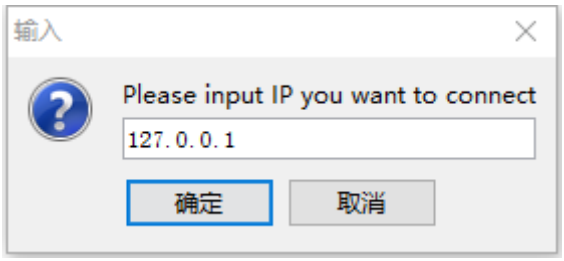
> 第一层
>
> > 嵌套第二层
>
> 还是第一层

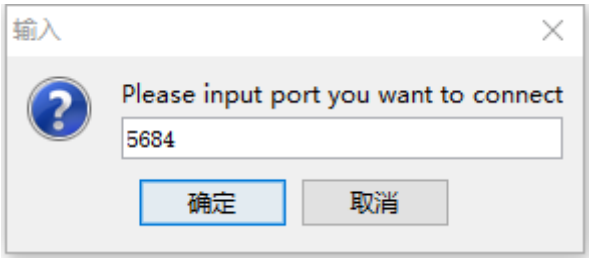
4. 网络服务

新建 server



连接到 server





断开连接

七、 第三方库

- CommonMark https://commonmark.org

八、 讨论、心得

本次实验是通过 MVC 模式设计，整体的架构还是比较清晰的，让我对 Java GUI，socket 编程以及 markdown 等有了更加深入的理解。实验中遇到了一些问题，也通过阅读 Java 文档和网上查阅资料得到了解决。同时我对以下几个小细节也有了更深的印象：

1. Java 组件渲染 html

   刚开始不太懂如何将 html 文件在 Java 中显示，所以最直观的想法就是想在面板里嵌入一个浏览器。所以我选择了使用 JFXPanel 的 WebView。后来发现 JEditorPanel 可以直接渲染 html，而且实现起来也比较简单，整个程序也比较轻。不过囿于时间原因，我就没有改动之前的代码。不过以后在动手写之前还是要搜集足够的信息，不能盲目去写。

2. 实现协同编辑的机制

   协同编辑最大的难点就是解决冲突问题。当时也看了一些资料，目前使用比较广泛的是 Google 的 OT 算法。但期末作业太多，没法花几天时间阅读论文再复现出来，

我最后选择的解决方案是低成本的加编辑锁。不过我倒是对 OT 算法挺感兴趣的，寒假会抽出一点时间将我的协同编辑部分改由 OT 算法实现。