浙江大学实验报告

课程名称：Java 应用技术 实验类型：

实验项目名称：MiniCAD

学生姓名：柏东山 专业：软件工程 学号：3160105684

同组学生姓名：无 指导老师：翁恺

实验地点： 实验日期：2018 年 12 月 13 日

一、 实验目的和要求

实现一个图形绘制软件，具有如下功能和非功能要求：

1. 能用鼠标绘制（各 1 分）：
   - 线段
   - 矩形
   - 椭圆
   - 填充的矩形
   - 填充的椭圆
   - 多点折线
   - 多边形和
   - 文字块；
2. 可以用鼠标选中已经绘制的图形（1 分）；
3. 可以移动选中的图形（1 分）；
4. 可以修改选中的图形的颜色、大小、线条粗细和文字内容（2 分）；
5. 可以删除选中的图形（1 分）；
6. 可以将所绘制的图形保存在文件中（1 分）；
7. 可以将保存的文件中的图形加载到当前的图形中（1 分）；
8. 其他以上未提及的功能（2 分）；
9. 代码和报告质量（3 分）。

二、 实验内容和原理

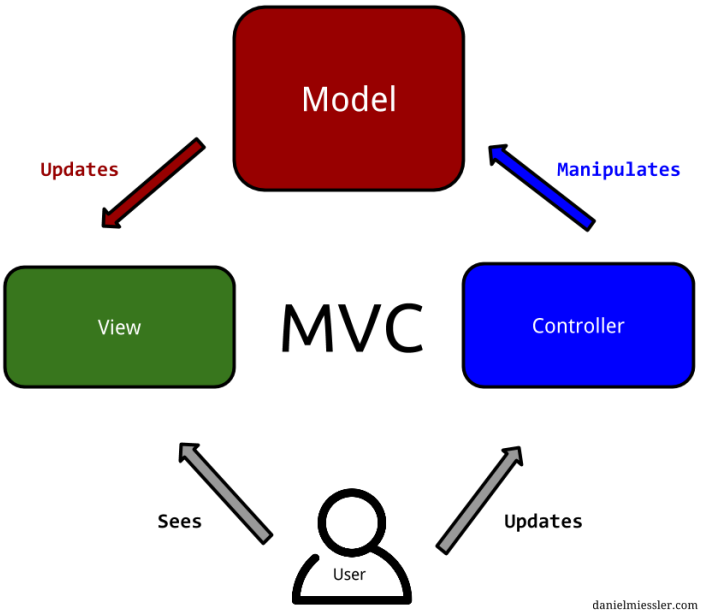
1. 利用所学知识设计出所要求的 GUI，掌握组件的事件响应原理；
2. 熟悉 Java 的功能和操作；
3. 熟悉 MVC 设计模式

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。

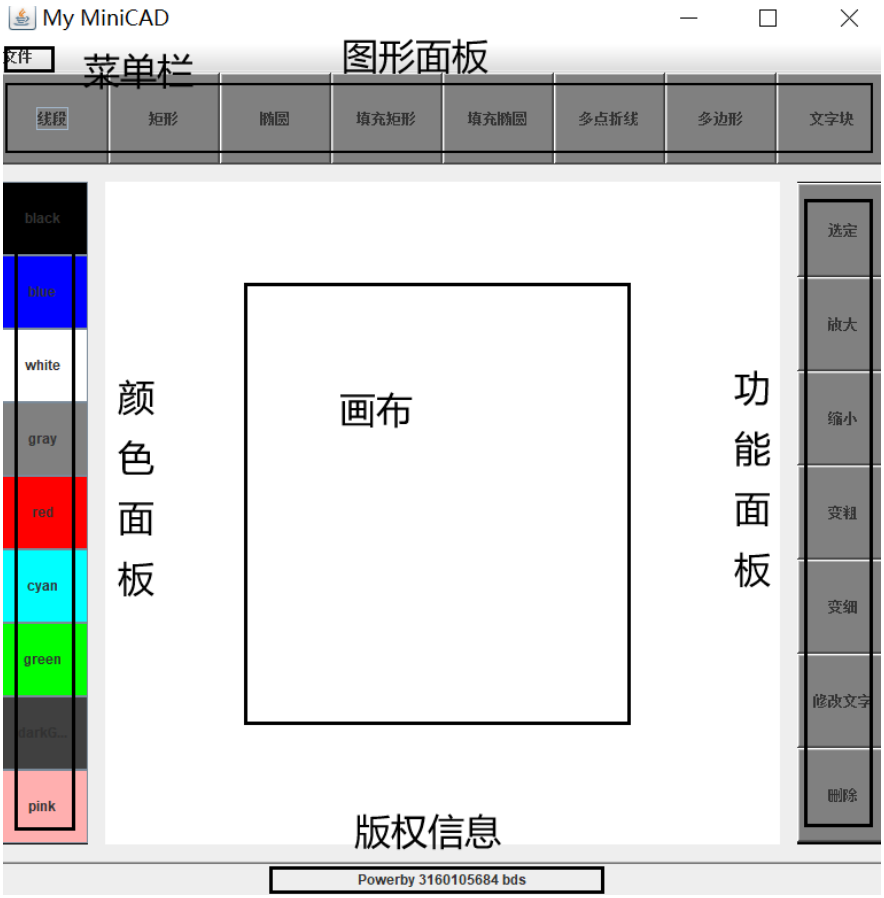
- **Model（模型）** - 模型层是最底下的一层，代表一个存取数据的对象或 JAVA POJO。它也可以带有逻辑，在数据变化时更新控制器。
- **View（视图）** - 视图层是最上面的一层，代表模型包含的数据的可视化。
- **Controller（控制器）** - 控制层是中间一层，作用于模型和视图上。它控制数据流向模型对象，并在数据变化时更新视图。它使视图与模型分离开。



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

采用类似演示视频的界面设计。包括菜单栏、图形面板、功能面板、颜色面板和版权信息栏。其中，菜单栏的下拉菜单中包括“新建”、“打开”和“保存”3 项。



3 实现思路

本程序代码按照 MVC 模式架构，分为 3 个层次，包含 5 个 Java 文件。其中 CADmain.java 中定义了窗口除画布之外的其他组件，同时是整个程序的入口，将 Model 层和 View 层用 Controller 来连接绑定。

```
Controller ctrl = new Controller(model, view);
```

Model 层

Model 类中用 Arraylist 来存储所有的图形。有 getAll、getCurrent、add、setAll、removeAll 共 5 种方法，分别用来获取所有图形、获取当前图形、增加图形、设置所有图形、移除所有图形。

Shape.java 中有父类 Shape 以及其他子类。其中父类实现了 Serializable 接口，这样可以在保存时把对象转换为字节序列，在打开时把字节序列恢复为对象。Shape 类中用一个 Arraylist 来存储图形的顶点。有 2 个抽象方法 draw 和 isSelected 分别用来画图和判断是否被选中。方法 adjustPoint 用来将画图坐标标准化（因为画图函数是取左上到右下的坐标，所以需要将不同相对位置的坐标标准化）。方法 changeSize 用来将图形等比例放大或者缩小。

View 层

View 类继承 JPanel，代表窗口中的画布部分。重写 paint 方法以实现在每次 repaint 时将所有图形绘制在画布上，这些图形是在 controller 调用 updateView 方法时传递给 view 层的。

Controller 层

Controller类中有3个成员变量，分别是model、view、paintlistener和state，方法updateView用来更新View层。paintlistener实现了ActionListener, MouseListener和MouseMotionListener。其中各种按钮由ActionListener监听，画布由MouseListener和MouseMotionListener监听。然后在对应的方法中根据不同的按键来完成不同的功能。

四、 运行环境

CPU: Intel(R)Core(TM)i5-5200U CPU@2.20GHz
Memory: 8GB(DDR3L 1600MHZ)
Operating System: Windows 10
Compiler: Java-SE 1.8
IDE: Eclipse

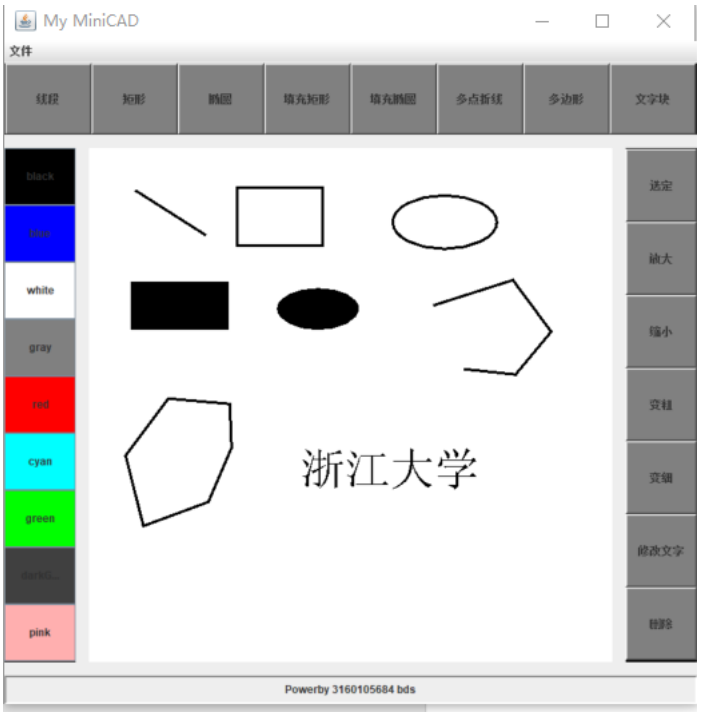
五、 操作方法与实验步骤

1. 双击 miniCAD；
2. 点击“线段”按钮，在画布上拖动绘制线段；
3. 点击“矩形”按钮，在画布上拖动绘制矩形；
4. 点击“椭圆”按钮，在画布上拖动绘制椭圆；
5. 点击“填充矩形”按钮，在画布上拖动绘制填充矩形；
6. 点击“填充椭圆”按钮，在画布上拖动绘制填充椭圆；
7. 点击“多点折线”按钮，在画布上拖动绘制多点折线；
8. 点击“多边形”按钮，在画布上拖动绘制多边形；
9. 点击“文字框”按钮，在弹出的对话框中输入想要填写的文本，并且在画布上拖动绘制文本；
10. 点击“选定”按钮，点击图形，在画布上可以拖动图形；
11. 点击“选定”按钮并选中图形后，点击“放大”按钮，可以看到图形等比例放大；
12. 点击“选定”按钮并选中图形后，点击“缩小”按钮，可以看到图形等比例缩小；
13. 点击“选定”按钮并选中图形后，点击“变粗”按钮，可以看到图形线条变粗；
14. 点击“选定”按钮并选中图形后，点击“变细”按钮，可以看到图形等比例缩小（当线宽缩小为 1 后再点击缩小，缩小效果可能不会太明显）；
15. 点击“选定”按钮并选中文字块后，点击“修改文字”按钮，在弹出的对话框中输入想要修改后的文本，点击确定；
16. 点击“选定”按钮并选中图形后，点击“删除”按钮，图形即被删除；
17. 点击“选定”按钮并选中图形后，点击“不同的颜色按钮，图形颜色做出相应改变；
18. 点击菜单栏“文件”，选择“新建”，可以新建一个 cad 文件；
19. 点击菜单栏“文件”，选择“打开”，可以打开一个 cad 文件；

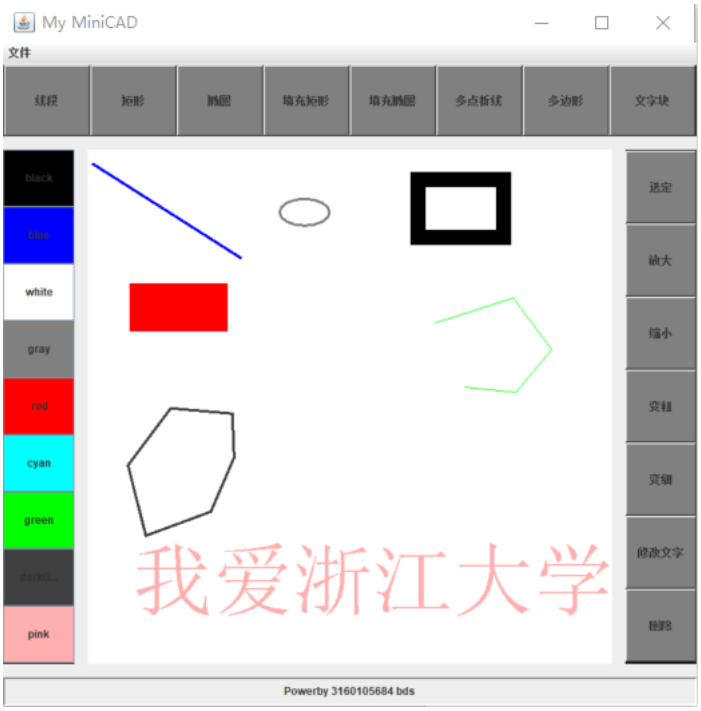
20. 点击菜单栏“文件”，选择“保存”，可以保存当前的 cad 文件；

六、 实验结果与分析

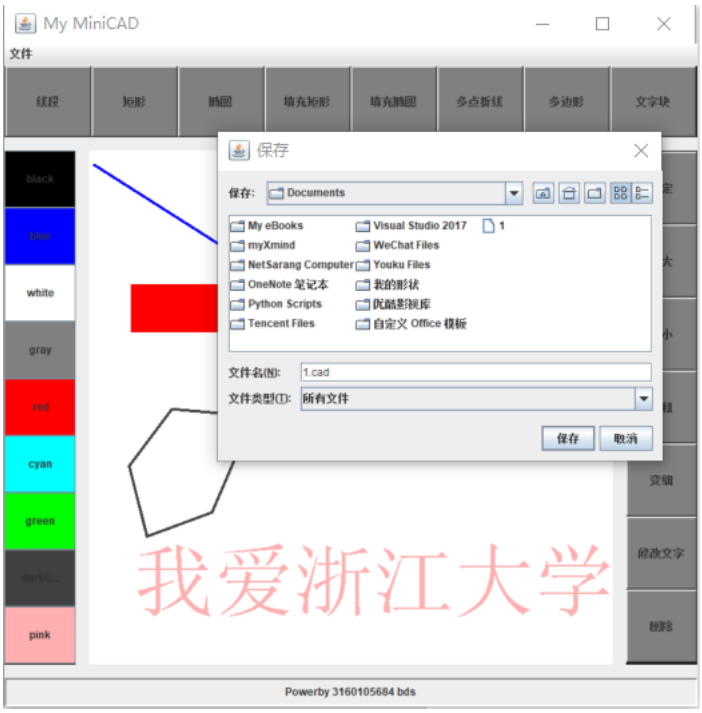
1. 绘制各种图形

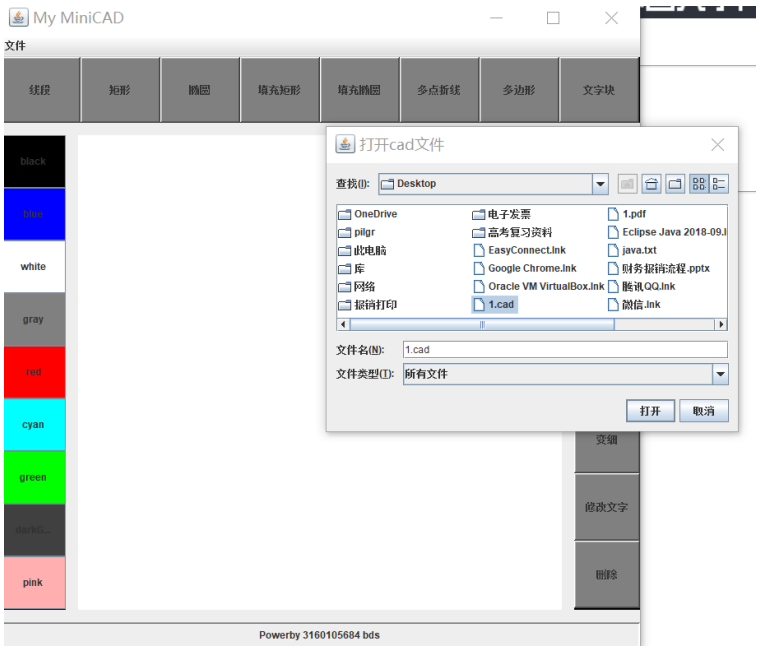


2. 依次完成题目要求的所有功能



3. 新建、打开和保存文件





七、 讨论、心得

本次实验是使用 MVC 模式设计，整体的架构还是比较清晰的，让我对 Java GUI，监听事件响应等概念有了实践中的理解。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 drawString 参数是字体框左下角；

② java.lang.Math.sqrt(x) 用来开根号；

③ 形象地来讲就是 Graphics 是一个画板，paint 是一个画家，而 repaint 是领导，领导告诉画家，每 1/60 秒，先擦掉画板上原来的，再重新画一遍，这就是重绘。重写 paint 的方法，就相当于，你告诉画家，不要画你原来想画的东西，我来告诉你画什么，怎么画。而先 getGraphics 的方法，就相当于，你先把画板抢过来，上面有画家的画，而你自己再修改，所以要保证你的修改用户能看到，你只能和画家一样的速度或者比他更快，不然你画上去，1/60 秒后就被画家擦掉了，但这依然是一种不好的做法。而且 getGraphics 可能会产生返回空指针的问题，所以最好还是重写 paint 方法。

④ 刚开始没想完整就直接上手写了，写到多边形和多点折线后发现前面的图形都可以归一到这两种图形。所以我又把整个 Shape.java 文件重构了一遍，减少了大量的代码冗余。